International Journal of Grid Distribution Computing
Vol. 8, No. 3, (2015), pp.87-96
http://dx.doi.org/10.14257/ijgdc.2015.8.3.09

Modeling Server Load Balance in Cloud Clusters Based on Multi-Objective Particle Swarm Optimization

Cao Lijun[1] and Liu Xiyin[2]

[1,2]*Hebei Normal University of Science & Technology, Qinhuangdao, Hebei Province, China*
misscao6666@163.com, liuxiyin2003@sina.com

Abstract

Load balancing is one of the hotspots in cloud computing research. Typically, the objective of workload balance in cloud environment is to assign tasks to proper virtual machines and physical servers with consideration of computing capability and communication cost. In this work, we propose to leverage PSO based algorithm to dynamically balance the workload of physical cloud servers. The problem is formulated as an optimization of the best solution of task assignment with the objectives of minimizing the average workload of all servers in cloud clusters, the deviation of the workload, and the migration cost between servers. Moreover, in order to avoid the low diversity of particles searching and low convergence speed, we employ a multi-swarm PSO method and introduce communication between swarms by random restructuring. Our extensive experiments show that our modified PSO method is efficient in balancing the workload of cloud servers.

Keywords: *Cloud computing, Load balancing, Particle Swarm Optimization (PSO)*

1. Introduction

Cloud computing [1-2] was first proposed by Google, which refers to a large scale business driven computing diagram, providing infrastructure, computing and storage capabilities by dynamically virtualizing the resource pool through Internet. In cloud computing environment, user needs are ensured through resource provision services based on service level agreement (SLA). One of the significant issues during this process is to dynamically balance the load of cloud servers to improve the resource utilization [3].

Indeed, there exist many algorithms for load balancing. The most traditional method is to utilize the shortest path first algorithm such as Dijkstra to explore the shortest path for forwarding network package [4]. However, if the network traffic suddenly increases a lot, some network links would be idle and lots of data would be lost, which leads to network congestion. Then, some researchers proposed a Multi-Protocol Label Switch (MPLS) technique for load balancing [5]. However, the algorithm is too idealized and the model is too complicated to apply in real world applications. Generally, the load balancing of the whole network is related to bandwidth, the utilization of the network links, network delay, and the ratio of package loss, *etc*. It is a typical NP complete problem, and is difficult to solve using traditional mathematical models [6].

Recent years, inspired by the biological mechanisms, some researchers proposed some heuristic swarm intelligent algorithms such as Genetic Algorithm (GA) [7], Artificial Fish-swarm Algorithm (AFSA) [8], Particle Swarm Optimization (PSO) [9] and Ant Colony Optimization (ACO) [10] to solve the load balancing problem. The characteristics of above algorithms such as fast search and convergence prove the feasibility of being applied to NP complete problems [11-13]. For example, they have already been deployed in applications such as robot path planning [14] and urban traffic management [15].

ISSN: 2005-4262 IJGDC

In this paper, we propose to leverage PSO based algorithm to dynamically balance the workload of cloud cluster servers. The most significant difference in cloud clusters is that there are multiple virtual machines on a single physical cloud server. Therefore, in this paper, we first mathematically formulate the workload of physical servers. Then, the problem of load balancing in cloud clusters is defined as finding the best solution of task assignment with the objectives of minimizing the average workload of all servers in cloud clusters, the deviation of the workload, and the migration cost between servers. Moreover, in order to avoid the low diversity of particles searching and low convergence speed, we employ a multi-swarm PSO method [16] and introduce communication between swarms by random restructuring. Finally, we conduct extensive experiments for evaluation.

The remainder of this paper is organized as follows. Section 2 reviews related work, and Section 3 formulates the problem statement. The proposed algorithm is discussed in Section 4. In section 5, we conduct extensive empirical experiments. Finally, the paper is concluded in Section 6.

2. Related Work

There exist some efforts on workload balancing in cloud computing. For example, Bhadani *et. al.*, [17] proposed the workload balancing strategy based on virtual machines. Liu *et. al.*, [18] built a three-layered structure to implement storage virtualization, and then achieve the balancing through a load balancing virtual storage strategy (LBVS). Based on the analysis of common workload balancing algorithms on clusters, Bo *et. al.*, [19] proposed CBL to solve the workload balancing between cloud servers. However, the proposed algorithm is too complicated. Zhao *et. al.*, [20] proposed a workload balancing method based on real-time migration. Wang *et. al.*, [21] proposed a two-stage scheduling algorithm by combing OLB and LBMM. However, this method is more suitable for static scenarios, but not for dynamic cloud computing environment. Feng *et. al.*, [22] designed a trust-driven model for resource load balancing, and employed a heuristic method to solve. Vesna *et. al.*, [23] applied swarm intelligence algorithm to deal with the load balancing problem of call center. Inspired by the max-min features of workload balancing [24], Zhang *et. al.*, [25] proposed an algorithm based on complicated network theory to approximately achieve the balance of the whole system.

3. Problem Statement

Traditionally, the objective of load balancing is to find the optimal available links to reasonably share the network flow, in order to avoid network congestion as well as improve resource utilization and the Quality of Service (QoS). Specially, in cloud computing environment, there are typically multiple virtual machines (VM) in physical network nodes (*i.e.*, cloud servers). Suppose the physical cloud servers are $P = \{P_1, P_2, ..., P_N\}$, where $P_i (i = 1,2,..., N)$ is the i-th physical server. Let the VMs on P_i be $V_i = \{V_{i1}, V_{i2}, ..., V_{im_i}\}$, where $V_{ij} (i = 1,2,..., N, j = 1,2,..., m_i)$ is the j-th VM, and m_i is the number of VMs on P_i.

Typically, the workload of cloud server is the combination of VMs on it. In this paper, we consider four major factors for load balancing: CPU utilization U_{cpu}, memory utilization U_{mem}, disk utilization U_{dis} and network utilization U_{net}. Let the workload for V_{ij} during time span T is $L(i, j, T)$, and therefore the workload can be represented as:

$$L(i, j, T) = w_1 U_{cpu} + w_2 U_{mem} + w_3 U_{dis} + w_4 U_{net}, \quad (1)$$

where $w_i (i = 1,2,3,4)$ denotes the importance of each factor, and

$w_1 + w_2 + w_3 + w_4 = 1$.

The average workload for VM V_{ij} during a monitored time period is calculated as:

$$\overline{L}(i, j, T) = \frac{1}{T}\sum_{k=1}^{n} L(i, j, k)(t_k - t_{k-1}), \quad (2)$$

where the workload monitoring period T is divided into n small segments, i.e., $T = [(t_1 - t_0), (t_2 - t_1),..., (t_k - t_{k-1}),..., (t_n - t_{n-1})]$, and the workload in each segment remains stable. Therefore, the workload for physical cloud server P_i during T is:

$$L(i, T) = \sum_{j=1}^{m_i} \overline{L}(i, j, T) \quad . \quad (3)$$

Suppose the VM to be allocated is V and its workload is estimated as $L(V)$. If V is assigned to server P_j , the workload of each server $P_i (1 \le i \le N)$ is:

$$L(i, T)' = \begin{cases} L(j, T) + L(V), & i = j; \\ L(i, T), & else \ . \end{cases} \quad (4)$$

where $L(i, T)'$ is the workload after the assignment.

Denote the solution of assigning VMs to P_j as S_j. The workload change of S_j during T is represented as:

$$\sigma(S_j, T) = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (\overline{L}(T)' - L(i, T)')^2} \quad , \quad (5)$$

where $\overline{L}(T)'$ is the average workload of all servers:

$$\overline{L}(T)' = \frac{1}{N}\sum_{i=1}^{N} L(i, T)' \quad . \quad (6)$$

Suppose there are k_0 VMs allocated in the cloud clusters initially, and k_1 VMs to be assigned. Let the complete solution of workload is $S = \{S_{1n_1}, S_{2n_2},..., S_{in_i},..., S_{kn_k}\}$, where S_{in_i} denotes the mapping of i-th VM to server P_{n_i} , $n_i = 1,2,..., N$, and $k = k_0 + k_1$. Define the cost of solution S as the migration cost of virtual machines to achieve load balancing. Suppose the number of VMs to be migrated is M' and the total number of VMs is M . Therefore, the cost is calculated as the ratio as follows:

$$\rho(S) = \frac{M'}{M} \quad . \quad (7)$$

In this study, the problem of load balancing in cloud clusters is to find the best solution S^* with the objectives of minimizing the average workload of all servers in cloud clusters, the deviation of the workload, and the migration cost between servers. Formally, it is described as a multi-objection optimization problem with the objective of:

$$\min \quad \lambda_1 L(T) + \lambda_2 \sigma(S, T) + \lambda_3 \rho(S) \quad , \quad (8)$$

where $\lambda_1 + \lambda_2 + \lambda_3 = 1$.

4. Proposed Model

4.1. PSO Basics

PSO algorithm origins from the simulation of pattern and behavior of bird flock, where each bird is abstracted as a particle. There are two attributes of each particle: location and speed. Suppose there are N_p particles in a D-dimensional space, and for particle i, the location is $x_i = (x_{i1}, x_{i2},..., x_{iD})$, and the speed is $v_i = (v_{i1}, v_{i2},..., v_{iD})$. Particles update their location and speed during the flying. Let the best location of particle i be $p_i = (p_{i1}, p_{i2},..., p_{iD})$, and the overall best location of all particles be $p_g = (p_{g1}, p_{g2},..., p_{gD})$. The update of speed and location is as follows:

$$v_{id}(t+1) = wv_{id}(t) + c_1\gamma_1(p_{id}(t) - x_{id}(t)) + c_2\gamma_2(p_{gd}(t) - x_{gd}(t)), \quad (9)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1), \quad (10)$$

where $i = 1,2,..., N_p, d = 1,2,..., D$, c_1, c_2 are learning factors, γ_1, γ_2 are random numbers between 0 and 1, and w is an inertia weight (typically 0.1~0.9). As proved in [26], w is linearly reduced and contributes to convergence. $v_{id}(t)$ is the speed of particle i at iteration t, and $x_{id}(t)$ is the location of particle i at iteration t.

Equation (9) is the update rule of particle speed, where the first component is the current speed of particle, the second component is the influence of self-recognition, i.e., the distance between the current and the best location of each particle, and the third one is the influence of the whole population, i.e., the distance between the current location of the particle and the overall best location of all particles. Equation (10) is the update rule of particle location, which indicates the location at iteration $t+1$ is the location at iteration t plus the flying distance during one iteration. Figure 1 gives the pseudo code of PSO.

Algorithm 1 PSO algorithm

```
repeat
    initialize particles and their positions, speed
    for each particle i do
        calculate the fitness of i, f(i)
        if f(i) > f*(i) then
            update best location of i, p_i
        end if
    end for
    if the overall best location is better than p_g then
        update overall best location p_g
    end if
    for each particle i do
        for each dimension d do
            v_id(t+1) = W × v_id(t) + c1 × rand() × (p_i d(t) − x_id(t)) + c2 × rand() ×
            (p_g d(t) − x_gd(t))
            x_id(t+1) = x_id(t) + v_id(t+1)
        end for
    end for
    t = t + 1
until t > MAX_INTERATIONS
```

Figure 1. Algorithm Description of PSO

4.2. Modified PSO

However, there exist some limitations of the basic PSO algorithm. For example, when a particle discovers a best location, it moves fast toward that location, which leads to local optimal point. In this case, due to the low diversity of particles searching, it is hard to find the global best point, and the convergence speed and efficiency of the algorithm are dramatically reduced. To avoid this situation, we propose to employ a multi-swarm PSO method and introduce communication between swarms by random restructuring.

Typically, since there exist a set of equivalent compromise solutions, it is hard to select a global optimal solution at each iteration. In this paper, we introduce the Pareto optimum [27]. Unlike single objective optimization problem, the selection of global optimal solution relies on an external storage. Find the best non-dominated solution from the external storage as the global optimum. Then, after the restructuring of the swarms, sort by one random objective, and equally divide the the non-dominated solutions based on the number of swarms. For each swarm, we randomly select a solution within the domain average as the current local optimum of the swarm. The flow chart of modified PSO is shown as in Figure 2.

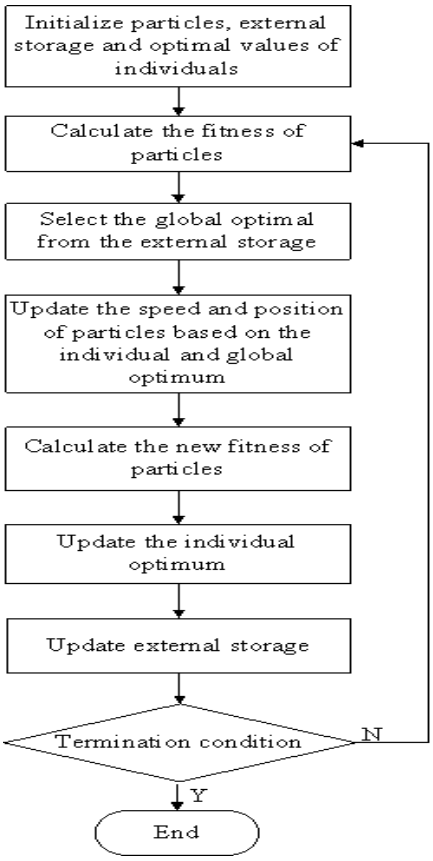

Figure 2. Flow Chart of Modified PSO

5. Experiment

In our study, we use Cloud Sim [28] to simulate the cloud computing environment. We have 8 computers to build the cloud cluster, where 1 is name node and the remaining 7 are data nodes. The configuration of each PC is: Intel Core 2 2.0 GHz, 2G RAM, and 250G hard disk. The settings of PSO are as follows: the number of particles $N_p = 200$, maximum number of iterations is 1000, $c_1 = c_2 = 50$.

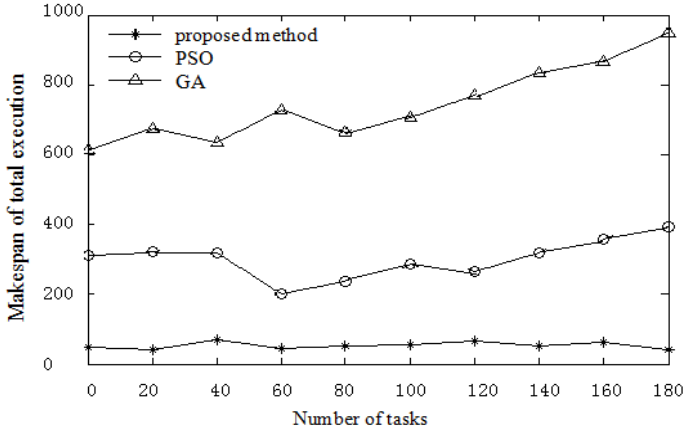

Figure 3. Makespan Comparison of Three Methods

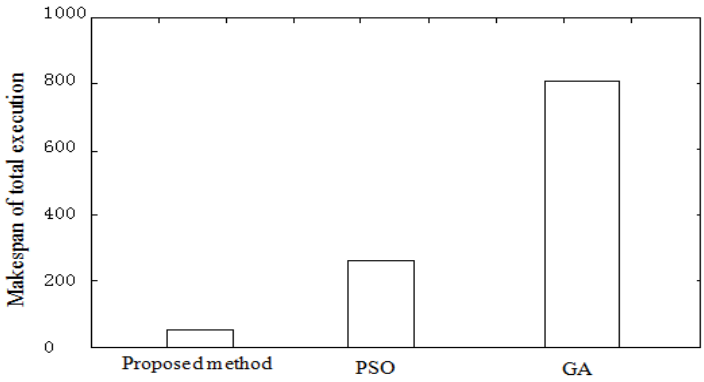

Figure 4. Average Makespan of A Single Task

We compare our method with GA and basic PSO in solving load balancing problem. First of all, the make span comparison of task execution for load balancing using GA, PSO and proposed modified PSO showed in Figure 3. Makespan of task execution represents the task completion of allocated nodes, and the less makespan it takes, the more appropriate the assignment is. From the figure, we have the following observations. (1) As the number of tasks increases, the total makespan increases generally. (2) The makespan of our proposed method remains stable with different numbers of tasks, while the performance of the other two is dramatically reduced for large number of tasks. (3) Our proposed method outperforms others, which means the node assignment solution is more efficient to find the best resource nodes to complete certain tasks with less execution time. Figure 4 shows the average makespan of a single task.

Second, we compare the degree of workload balance of all physical servers for three methods. Figures 5, 6 and 7 show the workloads of all 8 physical servers for GA, PSO, and proposed method respectively. From these figures, we have above observations. First, as the running of tasks, the workload keeps growing for all servers. Second, for the GA based method in Figure 5, the workloads of 8 physical servers are widely scattered, which means the whole system is not very balanced. Third, for PSO based method in Figure 6, the distribution of the 8 lines representing workloads of physical servers is relatively compact compared to Figure 5, which means the degree of workload balance has been improved. Last, for our proposed the method in Figure 7, which has the most compact distribution curves, we can easily see that our method has the best performance in terms of workload balance.

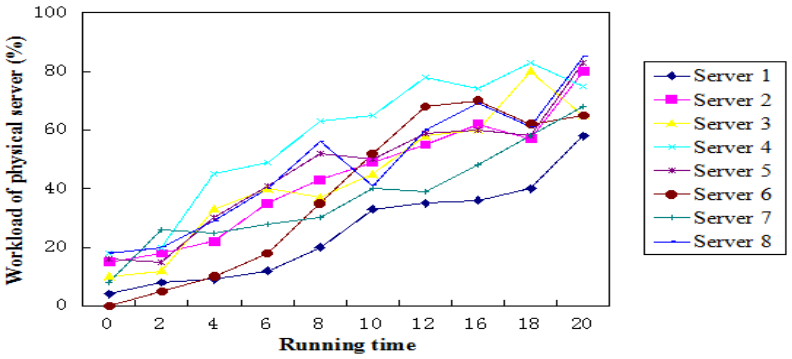

Figure 5. Workload of 8 Physical Servers for Ga Based Method

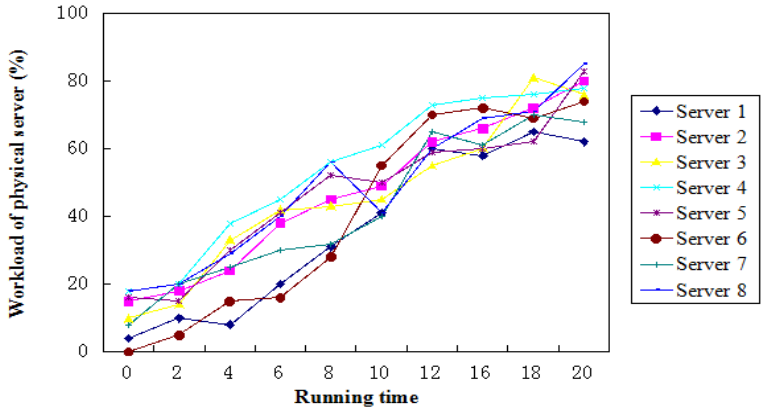

Figure 6. Workload of 8 Physical Servers for Pso Based Method

Third, we compare the average waiting time for three methods in Figure 8. Waiting time refers to the time period between task requesting and actual execution. In PSO based method, waiting time is the time cost of a particle searching for a best node. We can see that as the increasing of the numbers of tasks, the average waiting time grows. However, for our proposed method, the waiting time does not change significantly.

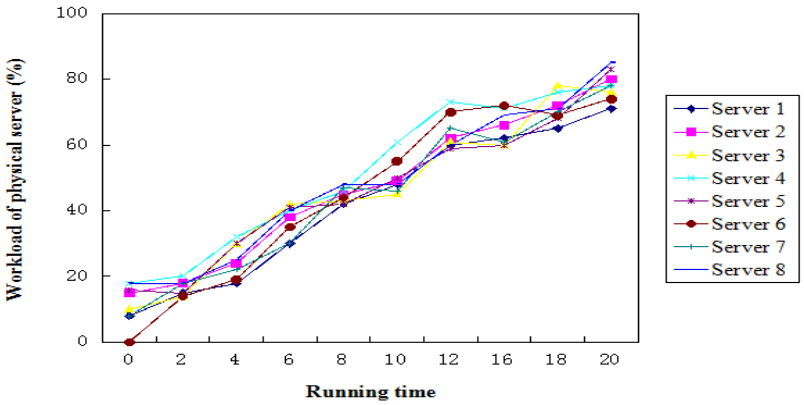

Figure 7. Workload of 8 Physical Servers for Proposed Method

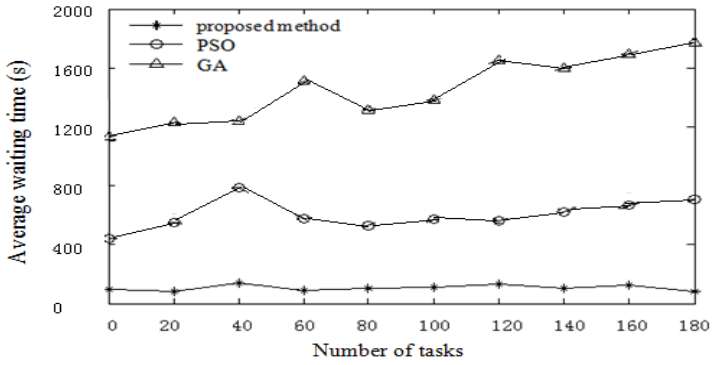

Figure 8. Average Waiting Time for Three Method

6. Conclusion

Focused on the workload balancing problem in cloud clusters, we proposed a modified PSO algorithm based on virtual machines in this paper. Specifically, we aim to minimize the average workload of all servers in cloud clusters and the deviation of the workload. Besides, we also consider the migration cost between servers. Our experiments show that our method outperforms GA and basic PSO method. In future, we would like to consider more characteristic in cloud environment for load balancing, such as elasticity and on-demand cloud services.

Acknowledgements

This work has been financially supported by the Project of Major Reform to the Network Engineering. Programme in Hebei Normal University of Science and Technology, which was assigned by Hebei Education Department in 2012. This work has been also sponsored by Qinhuangdao science and technology research (201101A038\201101A185\2012021A133).

References

[1] M. Armbrust, *et. al.*, "A view of cloud computing", Communications of the ACM, vol. 53, no.4, **(2010)**, pp. 50-58.
[2] P. Mell and T. Grance, "The NIST definition of cloud computing (draft)", NIST special publication vol. 800, no. 145, **(2011)**, pp. 7.
[3] M. Randles, D. Lamb and A. Taleb-Bendiab, "A comparative study into distributed load balancing algorithms for cloud computing", Advanced Information Networking and Applications Workshops (WAINA), 2010 IEEE 24th International Conference on. IEEE, **(2010)**.
[4] H. Dai and R. Han, "A node-centric load balancing algorithm for wireless sensor networks", Global Telecommunications Conference, 2003. GLOBECOM'03. IEEE. Vol. 1. IEEE, **(2003)**.
[5] K. Long, Z. Zhongshan and C. Shiduan, "Load balancing algorithms in MPLS traffic engineering", High Performance Switching and Routing, 2001 IEEE Workshop on. IEEE, **(2001)**.
[6] J. Kleinberg, Y. Rabani and É. Tardos, "Fairness in routing and load balancing", Foundations of Computer Science, 1999. 40th Annual Symposium on. IEEE, **(1999)**.
[7] D. Lawrence, "ed. Handbook of genetic algorithms", New York, Van Nostrand Reinhold, Vol. 115, **(1991)**.
[8] L. Xiao-lei and J. Qian, "Studies on Artificial Fish Swarm Optimization Algorithm based on Decomposition and Coordination Techniques", Journal of Circuits and Systems 1 **(2003)**, pp. 1-6.
[9] J. Kennedy, "Particle swarm optimization", Encyclopedia of Machine Learning, **(2010)**, pp. 760-766, Springer USA.
[10] M. Dorigo and M. Birattari, "Ant colony optimization", Encyclopedia of Machine Learning, **(2010)**, pp. 36-39, Springer USA.
[11] S. Pandey, *et. al.*, "A particle swarm optimization based heuristic for scheduling workflow applications in cloud computing environments", Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on. IEEE, **(2010)**.

[12] I, Kassabalidis, *et. al.*, "Swarm intelligence for routing in communication networks", Global Telecommunications Conference, 2001. GLOBECOM'01. IEEE. Vol. 6. IEEE, **(2001)**.
[13] J. E. Bell and P. R. McMullen, "Ant colony optimization techniques for the vehicle routing problem", Advanced Engineering Informatics, vol.18, no. 1, **(2004)**, pp. 41-48.
[14] M. Saska, *et. al.*, "Robot path planning using particle swarm optimization of Ferguson splines", Emerging Technologies and Factory Automation, 2006. ETFA'06. IEEE Conference on. IEEE, **(2006)**.
[15] J. Dallmeyer, *et. al.*, "Don't go with the ant flow: Ant-inspired traffic routing in urban environments", Seventh International Workshop on Agents in Traffic and Transportation (ATT 2012), **(2012)**, Valencia, Spain.
[16] J. J. Liang and P. N. Suganthan, "Dynamic multi-swarm particle swarm optimizer", Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE. IEEE, **(2005)**.
[17] A. Bhadani and S. Chaudhary, "Performance evaluation of web servers using central load balancing policy over virtual machines on cloud", Proceedings of the Third Annual ACM Bangalore Conference, ACM, **(2010)**.
[18] H. Liu, *et. al.*, "Lbvs, A load balancing strategy for virtual storage", Service Sciences (ICSS), 2010 International Conference on. IEEE, **(2010)**.
[19] Z. Bo, G. Ji and A. Jieqing, "Cloud loading balance algorithm", Information Science and Engineering (ICISE), 2010 2nd International Conference on. IEEE, **(2010)**.
[20] Y. Zhao and W. Huang, "Adaptive distributed load balancing algorithm based on live migration of virtual machines in cloud", INC, IMS and IDC, 2009. NCM'09. Fifth International Joint Conference on. IEEE, **(2009)**.
[21] S. C. Wang *et. al.*, "Towards a load balancing in a three-level cloud computing network" Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on. Vol. 1. IEEE, **(2010)**.
[22] F. Xiaojing and P. Yu, "DPSO resource load balancing in cloud computing [J]", CEA, 2013, vol. 49, no. 6, (2013), pp. 105-108, in Chinese.
[22] V. Sesum-Cavic and E. Kuhn, "Applying swarm intelligence algorithms for dynamic load balancing to a cloud based call center", Self-Adaptive and Self-Organizing Systems (SASO), 2010 4th IEEE International Conference on. IEEE, **(2010)**.
[23] Y. Bejerano, H. Seung-Jae and L. E. Li, "Fairness and load balancing in wireless LANs using association control", Proceedings of the 10th annual international conference on Mobile computing and networking. ACM, **(2004)**.
[24] Z. Zhang and X. Zhang, "A load balancing mechanism based on ant colony and complex network theory in open cloud computing federation", Industrial Mechatronics and Automation (ICIMA), 2010 2nd International Conference on. Vol. 2. IEEE, **(2010)**.
[25] Y. Shi and R. Eberhart, "A modified particle swarm optimizer", Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence, the 1998 IEEE International Conference on, IEEE, **(1998)**.
[26] A. Ben-Tal, "Characterization of Pareto and lexicographic optimal solutions", Multiple Criteria Decision Making Theory and Application, **(1980)**, pp. 1-11, Springer Berlin Heidelberg.
[27] R. N. Calheiros *et. al.*, "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, **(2011)**, pp. 23-50.

Authors

Cao Lijun, She was born in 1971. Now she is an associate professor with master degree in college of mathematics and information science of Hebei Normal University of Science and Technology. Her main research directions are data mining and grid technology.

Liu Xiyin, He is born in 1970. Now he is an associate professor with master degree in college of mechanical and electrical engineering of Hebei Normal University of Science and Technology. His main research directions are data mining and grid technology.