Cloud data centers energy-saving scheduling algorithm based on CPU frequency scaling

Yongchao Xiang[1], Zhen Liu[1], Guoqiang Zhang[2]
[1]School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China
[2]School of Computer Science and Technology, Nanjing Normal University, Nanjing, China

Abstract

The high energy consumption in cloud data centers has become an urgent problem. The scale and architecture of cloud data centers are growing increasingly immense and complex in recent years, which bring more severe challenges on the energy consumption management. This paper proposes new approaches for virtual machines (VMs) placement based on CPU frequency scaling. In the stage of initial VM placement, we propose a multi-objective optimization approach based on a heuristic ant colony algorithm, which can satisfy energy saving as well as service-level agreement (SLA). In the stage of dynamic management, by using autoregressive prediction and CPU frequency scaling, the proposed approach can adjust the CPU utilization while reducing the VM migration times and the migration cost. The experiments results show that the energy saving algorithms based on CPU frequency scaling are much better than the traditional best fit descending and first fit descending methods in saving energy and satisfying SLA.

Keywords: cloud datacenter, energy consumption, service level agreement, CPU frequency scaling, VM migration

1 Introduction

Resources adjusting in cloud data centers are always divided into two stages: initial placement stage and dynamic management stage. Initial placement stage is made up of a map of virtual machines (VMs) to physical machines (PMs) according to the requirements of resources, the SLA requirements set by users, and the available resources on PMs. The VMs placement in this stage has long-term effects, which places an important role in efficient use of resources and energy saving in data centers [1]. Dynamic management refers to that system should make quick decisions to adjust resources to match with the changes of workload during applications running time.

Traditional servers in data centers run at a fixed CPU frequency; however, the frequency may not be the fittest frequency to run this application. In addition, in dynamic management stage, traditional energy saving plans always migrate VMs to reduce workloads on hotspots (with heavy workload) or switch off coldspots (with light workload), making balance among each node to save energy. Kirk et al. [2] proposed an approach that migrates all VMs out on coldspots to save energy and migrating some VMs out on hotspots to reduce resources utilization and clear away hotspots by predicting resources utilizations on each node. Xu and Fortes [3] propose the prediction of the trends for the same process by monitoring resources utilization. However, there are two problems of wrong predictions and the costs of VMs migrations in the above approaches [1].

This paper proposes scheduling algorithm for energy saving in data centers based on CPU frequency scaling (SCFS). This algorithm aims to save energy and meet service-level agreement (SLA). In initial placement stage, a heuristic ant colony approach is proposed, which searches the solution space and try to find the best VMs placement solution. Compared with the traditional placements, SCFS can choose lower CPU frequencies, which leads to low-energy consumption. In dynamic management stage, if CPU utilization exceeds a pre-set threshold, SCFS can increase CPU frequency to make CPU utilization return to the normal range. In this way, some VM migration can be avoided and the whole VMs migration times can be reduced. Therefore, compared with the traditional approaches relying on VMs migration to adjust resources, SCFS has more complementary methods to optimize the data center resource management.

The rest of this paper is organized as follows. The VMs initial placement part of SCFS is discussed in Section 2, which is followed by the discussion of the dynamic management part of SCFS in Section 3. Section 4 is the experiments results between SCFS and traditional approaches. Conclusion is given in Section 5.

2 The initial placement part of SCFS

VMs initial placement can be considered as a variant packing problem: placing n VMs on m PMs, considering each PM has r frequencies and can select different CPU frequencies according to its workload. Then the solution space is $r^m m^n$. SCFS solves the conflicts among multiple objectives by using hierarchical sequence method [4], meeting SLA, energy saving and resources balancing in turn. In this way, the optimum solution of VMs placement can be searched out from $r^m m^n$ solution space.

2.1 Energy consumption

$$f_{\text{energy}} = c + k * f^3 * u_{\text{cpu}} \tag{1}$$

f_{energy} indicates the instantaneous power of PMs, parameter c in formula (1) indicates energy consumption of PM in idle status, f indicates instantaneous frequency, u_{cpu} indicates CPU utilization, k is the coefficient and indicates that dynamic energy consumption of CPU is proportional to the cubic of frequency and utilization [5].

The unbalance of resources utilizations will easily lead to a waste of resources. For instance, in a PM with multi-dimensional resources, only if all the resources are satisfied with VM requirements, the resources can be allocated to the VM. Therefore, we define resources balancing degree as follows.

2.2 Resources balancing degree

$$f_{\text{balance}} = \left(1 - \frac{u_{\text{cpu}}}{u}\right)^2 + \left(1 - \frac{u_{\text{mem}}}{u}\right)^2 + \left(1 - \frac{u_{\text{bw}}}{u}\right)^2 \qquad (2)$$
$$\left(u = \frac{u_{\text{cpu}} + u_{\text{mem}} + u_{\text{bw}}}{3}\right)$$

u_{mem}, u_{bw} indicate utilizations of memory and bandwidth on PMs, respectively. According to the formula, the value of f_{balance} would be correspondingly small if the three kinds of resource utilizations are balancing.

Based on Formulas (1) and (2), the objective function of VMs in initial placement can be represented as:

$$\min F = a * \sum_{i=0}^{n-1} f_{\text{energy}i} + b * \sum_{i=0}^{n-1} f_{\text{balance}\,i} \qquad (3)$$
$$s.t.\, u_{\text{cpu}} \le u_{\text{sla}}, u_{\text{mem}} \le u_{\text{sla}}, u_{\text{bw}} \le u_{\text{sla}}$$

$f_{\text{energy}\,i}$ indicates the energy consumption of PM i, $\text{f}_{\text{balance}i}$ indicates the resources balancing degree of PM i. a and b are weights which stand for the degree of importance on energy consumption and resources balancing. Objective SLA is implemented by constraining resources utilizations in f_{energy} and f_{balance}, u_{sla} indicates the upper limit of resources utilizations on PMs.

Function F is a multi-objective optimization problem. We consider each objective, respectively, by using hierarchical sequence method. More specifically, we set SLA, energy consumption, and resources balancing as the first, second, and third objective, respectively. We then propose a heuristic ant colony optimization algorithm to form the optical VMs placement by considering SLA energy consumption and resources balancing, respectively.

3 The dynamic management part of SCFS

After the initial placement, resources utilizations on PM would change due to the workload of applications change during running time. Therefore, it is needed to

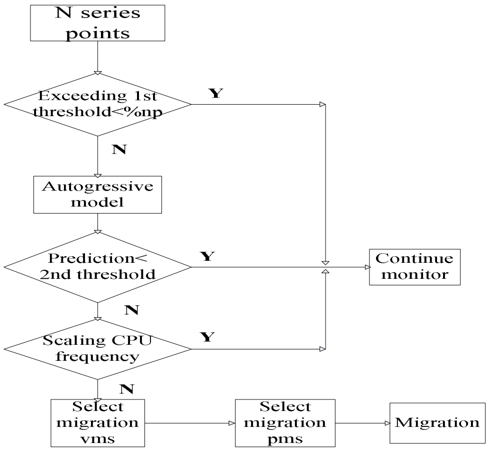

Figure 1: Resources monitoring and management flow chart.

monitor resources utilizations and adjust resources according to the changes of workload. There is a resources monitoring and management flowchart.

As shown in Figure 1, if there is more than a certain percent sequence point whose resources utilizations exceeding the first threshold, then analysis time series using autoregressive method[9] and predict the resources utilizations of next time point. If the prediction exceeds the second threshold, then attempts to increase CPU frequency. If it does not work, migrates VMs.

3.1 Adjusting CPU frequency response to workload changes

Since CPU is the most active resources element, so when prediction result exceeds the second threshold and only CPU resource utilization exceeds threshold, we can adjust CPU frequency instead of migrating VMs.

CPU utilization on PMs violating threshold can be classified into two situations: CPU utilization is more than the high threshold set by SLA, and CPU utilization is less than the low threshold, in this situation PM consumes plenty of static energy while do not run corresponding workload. We attempt to increase CPU frequency to return CPU utilization to normal level for the first situation. We decrease CPU frequency down to make the energy consumption match with the workload on PMs for the second situation.

3.2 The selections of migrate VMs and destination PMs

(1) The selection of migrate VMs

We can only migrate VMs for the situations that adjusting CPU frequency does not work or there are other kind of resources violating thresholds. Since it spends plenty of costs on network to migrate VMs [6], we migrate VMs with less memory priority to ensure a smaller migration cost. Migrate VMs out in descending order of (abnormal resource

utilization)/(memory utilization) until resource utilization returning to normal levels. This migrates as little as possible memory to return resources utilizations to normal level.
We migrate all VMs out on PM with low resources utilizations and then switch off the PM to reduce the static energy consumed by PM.

(2) The selections of destination PMs
Appropriate PMs need to be selected to accept the migrating VMs. It is unrealistic to migrate VMs between a pair of random source PM and destination PM in data centers with tens and thousands PMs today [7].We need to divide data centers into some sub-regions and limit the scope of migrations in the sub-regions. For example, we can select a pair of PMs in the same access network or rack as source node and destination node of a migration. The location relationship in SAN between the source PM and destination PM is also needed to be considered in migrations of VMs considering SAN has become the mainstream of storage in data centers [7].

Based on the principle of large package first load, sort all migrating VMs as their comprehensive utilizations of resources $u(u=p*u_{\mathrm{mem}}+q*u_{\mathrm{cpu}}+r*u_{\mathrm{bw}},p>q>r)$ in descending order and then select destination PMs for them in the above sequence. The coefficients p,q,r in the formula represent the weights of memory, CPU, and bandwidth. $p>q>r$ indicates memory is the most difficult resource to obtain, followed by CPU, the last is bandwidth.

Four principles need to be considered when selecting destination PMs: the first is avoiding new migrations. That is to say migrations would not cause resources utilizations cross thresholds, or migrations would not cause new migrations. The second is balancing resources. The migration VMs and destination PMs can be complementary in kinds of resources. For example, CPU resource and memory resource would be complementary if migrating the memory-intensive VM to the CPU-intensive PM. The third is reducing energy consumption. We can select the PM with the least energy increasing after accepted the migration VMs by taking the energy consumption model and the instantaneous CPU frequency of each PM account. The last is forbidding two-way migrations. That is to say a PM would not accept VMs while migrate VMs out. Because PMs migrate VMs in and out at the same time may lead resources deadlock.

4 Algorithm performance analysis and results of the experiments

This section evaluates the performance of energy-saving scheduling algorithm based on CPU modulation by experiments. It is divided into two parts. First, we compare the different performance among SCFS, first fit descending (FFD), best fit descending (BFD), single-objective method SCFS-balance, and SCFS-energy in the initial placement stage; Second, we compare different comprehensive performance among SCFS, Xu method, single-objective SCFS-balance, and SCFS-energy in the initial placement and dynamic management stage.

FFD: VMs placement algorithm in initial stage. The VMs are arranged in descending order by comprehensive resources utilizations u(comprehensive utilization of resources $u=p*u_{\text{mem}}+q*u_{\text{cpu}}+r*u_{\text{bw}}$, $p>q>r$) and then VM is placed to the first PM which can accommodate it in turn. This placement algorithm tends to select PMs with more available resources. It can avoid the situation of SLA violation because of lack of resources.

BFD: VMs placement algorithm in initial stage. The VMs are arranged in ascending order by comprehensive resources utilizations u. Take turns to select the closet full PM which still can accommodate this VM as the destination PM. The main idea of the algorithm is to prevent wasting resources and keeping the running PMs as less as possible.

The Xu method: Xu proposes a comprehensive management program including VMs initial placement algorithm and dynamic management algorithm in Refs. [3, 8].Initial placement algorithm will get the optimum solution with the multi-objectives of resources utilizations, energy consumption, and temperature by heuristic genetic algorithm; dynamic management algorithm predicts resources utilizations by using autoregressive method, monitoring the upward trends of resources utilizations by two-level thresholds and then determines whether to make a migration.

SCFS-balance and SCFS-energy: these two methods are SCFS with single goal in resource balance and low energy consumption, respectively.

4.1 Experimental environment

Our experiments run on a cluster containing 21 nodes with OpenStack as its cloud infrastructure platform. We configure one IBM System x3650 server as the control node and the other 20 Lenovo R510G7 servers as the compute nodes. All nodes are connected by a Gigabit Ethernet.

We simulate CPU intensive applications by calculating the value of π with Monte Carlo algorithm in experiments, CPU utilization fluctuates according to the sleeping time changing and simulate the I/O intensive applications by running processes reading and writing files, the I/O utilization fluctuates with the amount of data reading and writing. The characteristics of each algorithm are compared in different stages and under types of applications through three experiments. The first two experiments compare the effects of running I/O intensive applications and CPU intensive applications between each algorithm in initial placement stage, respectively. The resources utilizations of VMs are set constant to prevent resources utilizations violating the thresholds in dynamic management stage. The third experiment compares the integrated effects among each method. Each method creates a VMs initial placement solution and places VMs to PMs according to the solution. During the dynamic management stage, the resource utilizations of VMs are fluctuated with the workload changing. We compare and evaluate the effort of each method.

4.2 Experiments results

The following will explore the advantages, disadvantages, and scopes of applications by probing the inherent characteristics and the experimental results of these algorithms.

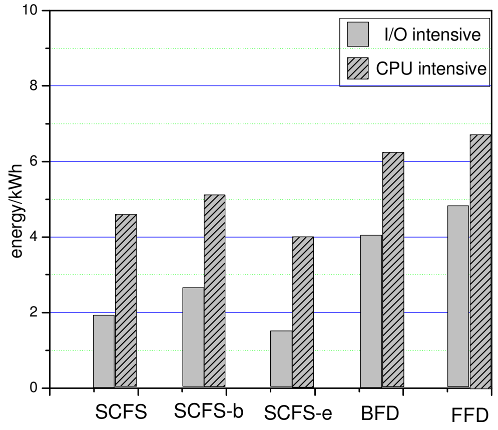

Figure 2: Energy consumption comparison among each algorithm.

(1) Initial placement of VMs

The experiment system consists of 21 PMs which hosting 60 VMs. The first two experiments run I/O intensive applications and CPU intensive applications with constant resources utilizations. We place VMs to PMs according to the initial placement solution of SCFS, SCFS-balance (SCFS-b), SCFS-energy (SCFS-e), BFD, and FFD, respectively, and evaluate for 2 h. The final results of energy consumption are shown in Figure 2.

As shown in Figure 2, no matter running the I/O intensive or CPU intensive applications, SCFS, SCFS-b, SCFS-e which scaling the CPU frequency make significantly less energy consumption than BFD and FFD with fixed CPU frequency. SCFS-e costs the least energy consumption among SCFS, SCFS-e, and SCFS-b because it only considers energy consumption. The second one is SCFS which considers energy consumption and resources balancing together. The last one is SCFS-b considering resources balancing only. Keeping PM resource balance can effectively cope with the workload changes that help PM not to waste idle resources in case of bucket effect when the workload fluctuates fiercely. In this experiment, the workload is unchanging, which means resource utilizations are constant. In Figure 2, BFD and FFD are the algorithms with fixed CPU frequency. BFD prefers to fill up PMs. It takes up at least 15\% of the PMs less than FFD. Its total energy consumption is less than FFD after switching off those idle PMs.

More specifically in Figure 2, there are some differences between the results of the first experiment (running I/O intensive applications) and the second experiment (running CPU intensive applications). First, CPU intensive application will cause a high CPU frequency on PM. Moreover, CPU energy consumption is one of the main aspects of physical energy consumption. Therefore, energy consumption of the second experiment increases at least 41% than that of the first group of experiment (as shown in Figure 2). Second, high CPU frequency of the second experiment will reduce the scaling range of the working frequency of CPU. As the result, the effect of energy consumption of SCFS, SCFS-b, SCFS-e in the first experiment is not as good as the result of the second experiment. While on the other hand, it cannot create so many idle PMs according to BFD due to the high CPU resource utilization in the second experiment. Therefore, the good performance of BFD compared to FFD is decreased in the second experiment.

(2) Two-phase experiment
The third experiment compares the effects of the initial placement and dynamic management among SCFS, SCFS-e, SCFS-b, and Xu. The experiment system consists of 21 PMs, 60 VMs. 30 VMs execute the CPU intensive applications and the remaining 30 VMs execute I/O applications. The resource utilization of these two kinds of application changes according to the sine and cosine fluctuation pattern, respectively (as shown in Figure 3).

In Figure 3, sine function curve represents the CPU utilization of CPU intensive applications. Cosine function curve represents the I/O utilization of I/O intensive applications. These two applications used to simulate the applications in practical data center environment.

In Figure 3, I/O intensive application begins with its lowest value in time 0, while the CPU intensive applications begin with a normal value. I/O applications mainly cost memory and disk resources, and CPU applications mainly cost CPU resources. Each algorithm generates an initial placement solution. SCFS placement solution considers the resource balancing degree among each dimension of resources. About 50% of PMs are allocated with 1 I/O CPU intensive VMs in one PM, and the rest PMs are allocated with 2 I/O intensive VMs and 1 CPU intensive VM in one PM. The PMs with more CPU intensive

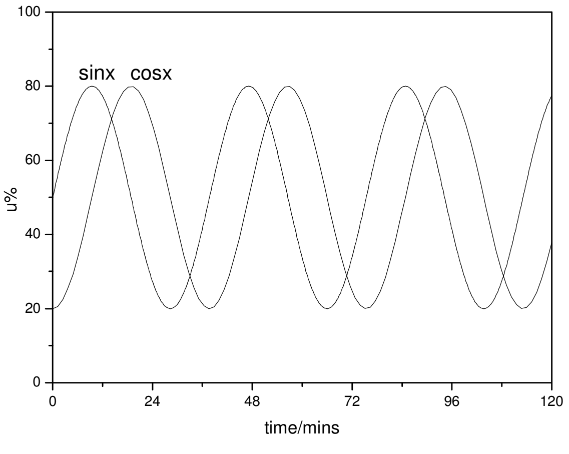

Figure 3: Periodic changes of the resources utilizations of the applications.

VMs have a higher CPU frequency than the PMs with more I/O intensive VMs. SCFS-b generates a similar solution; however all of the PMs have a high CPU frequency. SCFS-e generates a solution with 1 or 2 CPU intensive VMs and n I/O intensive VMs in one PM. The algorithm does not consider the resource balancing degree, the value of *n* is quite different on different PMs. The algorithm of Xu considers the temperature of PM. Therefore, it keeps each PM CPU utilization consistency because the heat is mainly produced by CPU. Xu generates a solution with 50\% PMs allocated by 2 CPU intensive VMs, and 50% PMs allocated by 3 I/O intensive VMs and 1 CPU intensive VM.

With the above initial placement solution, each algorithm shows different effects in dynamic management stage as shown in Figure 4. In Figure 4, by using CPU frequency scaling, SCFS, SCFS-b, and SCFS-e produce significantly less energy consumption and SLA violations than Xu which dose not used CPU frequency scaling. In Xu's solution, I/O-intensive VMs are too concentrated on some PMs which causes a lack of resources when applications fluctuates. It can only rely on migrating VMs to ease the lack of resources on some PMs.

Among the three algorithms of scaling CPU frequency, SCFS-e prefers to pursuit the lowest energy consumption, but resources are not balancing and the CPU frequency is lower, which is easy to violate the low resources threshold. And SCFS-e does not care resources balancing degree when selecting the migration destination PMs, which will lead to the migration again, thus greatly increases the total energy consumption. The energy consumption and SLA violation are 57% and 115% higher than SCFS. SCFS-b mainly pursues resources balancing and does not care energy consumption, which cause its CPU frequency staying at a high level (compared to SCFS); therefore the energy consumption is higher than SCFS. But the resource utilizations rarely violate the thresholds and the SLA violations are comparatively lower in SCFS-b. Its energy consumption and SLA violation are 145% and 90% of that of SCFS.

We can see that the great decreasing of resources utilization violating thresholds and VM placement states becoming similar to VMs initial placement solution of SCFS in the later time of SCFS-e and Xu experiments, which indicates SCFS-e and Xu adjust VMs placement adapting to the regular changing applications. The difference between them is the time of adjusting. SCFS-e has a shorter adjusting time. Because in the initial placement solution of SCFS-e, I/O intensive VMs are placed unbalanced. When I/O applications reach its peak value, I/O intensive VM is migrated to the PMs with less I/O intensive VMs, then balancing solution is formed soon (before I/O application reaches its peak value second time).The initial placement of Xu has a problem of I/O applications over-concentrated in one PM. And it also considers temperature balancing, which limits I/O intensive VMs' migration to PMs with high CPU utilization and less I/O intensive VMs. There are significant reductions of thresholds violation until I/O applications reach its peak value the fourth time in Xu. It can be concluded that the optimal dynamic solutions have the ability to deal with regular changing applications. However, the initial placement solutions which better fit for the applications and the dynamic management algorithm with a more integrated objectives can make a quicker adjusting.

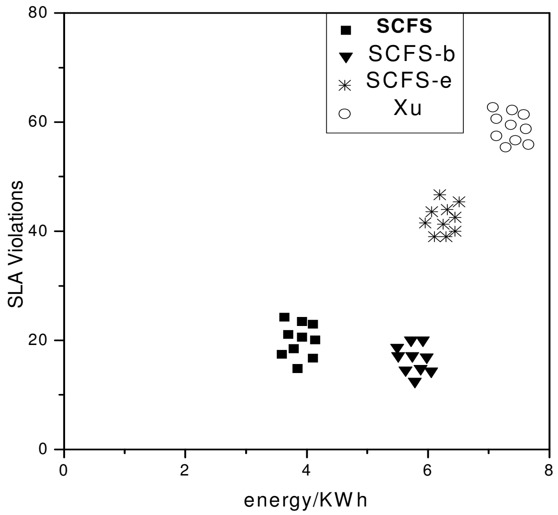

Figure 4: Energy consumption and SLA violations in dynamic management stage.

In practical cloud data centers, PMs accommodate VMs with different applications, which may keep changing in its resources utilizations. Our experiments have tried to simulate the practical environment of the cloud data centers. However, it is harder to define an actual application between I/O intensive or CPU intensive. The demands for each dimension of resources sometimes alternate, which adds the difficulty for resources utilizations prediction. Moreover, applications in practical environment may not fluctuate as sine or cosine curves. There are two approaches to solve this problem. One is to improve the learning and self-adaptive abilities of our algorithm in dynamic stage. The other is to find out changing rules of applications or classifications from immense applications in data centers by data mining.

5 Conclusions

We propose cloud data centers energy-saving scheduling algorithm based on SCFS in this paper. SCFS consists of two parts, acting on the stage of VMs initial placement and dynamic management. The first part searches out an optimal solution of VMs placement considering both energy consumption and SLA by heuristic ant colony optimization algorithm. System places VMs on corresponding PMs according to optimal solution. The second part predicts resources utilizations of next time on PMs by using autoregressive method on history resource utilizations. Then SCFS will decide whether adjust CPU frequency or migrate VMs to optimize resource utilizations. The experiments on the OpenStack cloud management platform show that SCFS has the ability of adapting to various types of applications. SCFS shows significant advantages in energy consumption and SLA satisfaction compared to the traditional VMs placement algorithms such as BFD, FFD, CPU frequency scaling method with single-objective—SCFS-e, SCFS-b, and integrated solution Xu.

Here, we consider I/O intensive applications consume constant energy when running. Practically, I/O intensive applications experience consumption energy changing during running time. And network intensive applications also occupy increasingly important places in data centers. So we will focus on these two points in our future research.

Acknowledgment

This work was supported by the National Natural Science Funds of China under Grants 61202429 and 61100178.The authors would like to thank Shake Chen—an active engineer of OpenStack open source community for helpful discussions and advises.

References

[1] Zhang Wei, Song Ying, Ruan Li, et al., Resource management in Internet-oriented data centers. *Journal of Software*, **23(2)**, pp. 179–199, 2012.

[2] B. Kirk, K. Gautam, K. Andrzej, Application performance management in virtualized server environments. In: *10th IEEE/IFIP Symposium of Network Operations and Management*, 2006, pp. 373–381.

[3] J. Xu, J. Fortes, A multi-objective approach to virtual machines management in datacenters. In: *Proceedings of the 8th ACM International Conference on Autonomic computing. ACM*, pp. 225–234, 2011.

[4] Li Chunming, *Optimization Method*. Southeast University Press, Nan Jing, pp. 147–153, 2009.

[5] Song Jie, Li Tiantian, Yan Zhen xing, et al., Energy-efficiency model and measuring approach for cloud computing. *Journal of Software*, **13(2)**, pp. 200–213, 2012.

[6] C. Clark, I. Pratt, A. Warfield, Live migration of VMs. In: *Proceedings of the 2nd Conference on Symp. on Networked Systems Design\& Implementation*. USENIX Association, Berkeley, 2005.

[7] http://www.valleytalk.org./2011/07/06/ Network Technology in Cloud Computing Data Centers.

[8] J. Xu, J. Fortes, Multi-objective virtual machine placement in virtualized data center environments. In: *Proceedings of the 2010 IEEE/ACM International Conference on Green Computing and Communications Computing*, ACM, pp. 179–188, 2010.

[9] Wang Yan, *Application Time Series Analysis*. China Renmin University Press, Bei Jing, pp. 167–174, 2005.