Load Balancing and Server Consolidation for Energy Management in Cloud Data Center

Rajkumar Sharma[1,*], and *Bharati Sinha*[2]
[1,2]Department of Computer Engineering, National Institute of Technology, Kurukshetra

Abstract. Users worldwide can access utility-oriented computing services through cloud computing. It enables the pay-as-you-go model to host well-known consumer, educational, and business applications. However, cloud data centers encounter significant issues in effectively managing resources, ensuring optimal performance, and reducing energy usage as the demand for cloud computing services is increasing rapidly. A possible strategy to reduce energy consumption and operating expenses is the energy-efficient management of virtual machines (VMs) and server consolidation. In this paper, a VM migration and server consolidation algorithm is proposed, which is energy efficient and SLA aware. The proposed approach is evaluated using the CloudSim simulation kit and PlanetLab workload. The experiment result shows that the proposed approach has reduced energy consumption and SLA violation.

1 Introduction

Cloud computing provides on-demand computing resources, storage, software application, and other computing services to users around the world. It enables hosting of popular commercial, scientific and consumer applications based on a pay-as-you-go model. Physical resources are owned and managed by cloud service provider (CSP) and customer can use these without any concern of managing it. CSPs offer a variety of services like software as a service (SaaS), infrastructure as a service (IaaS), and platform as a service (PaaS). Cloud computing offers many advantages like flexibility, scalability, and cost savings. However, cloud computing has many disadvantages like dependency on internet, data security, energy consumption, and SLA violation. According to Koomey and Jonathan [1] 1.5% of total uses of electricity in 2010 was contributed to data center, and the percentage had risen to 3% by 2016. There are many ways for energy sustenance in cloud like: virtualization [2] which run multiple virtual machine on a single physical machine, energy efficient hardware [3,4,5] which includes DVFS enabled CPU, multi-speed disk, and flash memory, and data center design [6] which consists of efficient cooling system, and power distribution unit.

One of the most important reason for high energy consumption is the mismanagement of resources [7]. Load balancing and server consolidation [8] are two of the most widely used technique to increasing resource utilization. Techniques of load balancing is used to migrate VM from one host to another host, so that resource utilization can increase. While, In case of server consolidation, VMs are migrated to minimize number of active host, so that some host can be turn off and conserve energy. However, active server consolidation might also adversely affect system performance, since it increases SLA violation. Hence, server consolidation mechanism should be implemented in such a way that SLA violation is within permissible limits, while reducing energy consumption. Live VM [9] migration is used in the proposed approach to increase resource utilization through load balancing and server consolidation.

* Corresponding author: raj45400@gmail.com

The main contribution of the paper are as follows.

- Algorithm design for overload host detection using mean and standard deviation of VM allocated to host, and inter quartile range method
- VM selection methodology considering migration time and utilization of VM
- Under-load host detection algorithm using host utilization history.
- Proof of concept using PlanetLab workload in CloudSim and energy sustenance comparison of proposed method with other standard algorithms.

The rest of the paper is divided as follows. Section 2 covers related work. Section 3 presents proposed approach. Section 4 demonstrate performance analysis, and the conclusion is covered in Section 5.

2 Related work

Beloglazov and Buyya [10] proposed different methods for overload host detection like, median absolute deviation, inter quartile range method, and linear regression based approach. The author also proposed different VM selection policy like minimum migration time, random choice policy and maximum correlation policy. For VM placement, the author proposed power aware best fit decreasing method, in which VM are placed on that host for which increment in energy consumption is minimum. In host under-load detection, all VM of the host having minimum utilization is migrated to another host and source host is put in idle state. This step is repeated for all the host, which are not declared as overloaded.

A consolidation approach with two preset CPU utilization levels was proposed by Beloglazov et al. [11]. Some VMs will be chosen and moved to different hosts if CPU utilization rises higher than the upper threshold value or CPU utilization falls lower than the lower threshold value. To reduce power usage while minimizing SLA violations, the authors performed an experiment to find the appropriate set of higher and lower criteria. The authors also recommended applying three VM selection policies. The first policy is Minimization of Migration (MM). The MM strategy consolidates the fewest possible virtual machines to other hosts. The second approach is Random Choice (RC) policy. The RC policy chooses VM randomly. The third approach is the Highest Potential Growth policy (HPG). The VMs chosen by the Highest Potential Growth policy are those with the lowest CPU utilization value compared to their total necessary CPU.

Pinheiro et al. [12] did one of the first studies to apply power management at the data center level. They suggested a technique for a computer system that supports several web applications to conserve electricity. The method involves using fewer computers to handle as much work as feasible and turning off unused ones. This approach aims to balance performance and power savings, as this method can affect how well the applications operate. The system determines which computers to turn on or off based on how much work is being done at any given time.

Xiong Fu et al. [13] introduced new VM selection and VM placement policy. In the first step of VM selection, host's utilization deviation over upper threshold (*'dev'*) is calculate, which is difference of upper threshold and utilization of the host. In the next step all the VM of the host are sorted in descending order of their utilization. If first VM's utilization is higher than *'dev'*, the VM is selected to be migrated, and if VM's utilization is lower than *'dev',* then SLA is calculated for all VM, and VM having minimum SLA is migrated first. In VM allocation policy, the author try to reduce impact of migration on other VMs, so a migrating VM is placed on a host with a minimal correlation coefficient.

Taheri and Zamanifar [14] proposed an approach in which they try to solve problem of incomplete migration by diving VM migration from overload host and VM migration from under-load host in two different phase. In first phase, only VM from the overloaded host are migrated and in the second phase VM from the under-loaded host are migrated.

Li Lianpeng et al. [15] proposed use of robust simple linear regression (RobustSLR) to predict host's future CPU utilization and marking host as overloaded and for under-load host detection, the author uses interquartile range method.

3 Proposed work

Efficient VM migration and consolidation can reduce consumption of energy significantly and also it can reduce SLA violation. The proposed method involves four steps i.e. detecting overload host, VM

selection, detecting under load host and VM allocation. Fig. 1 illustrates a flowchart diagram of the proposed method.

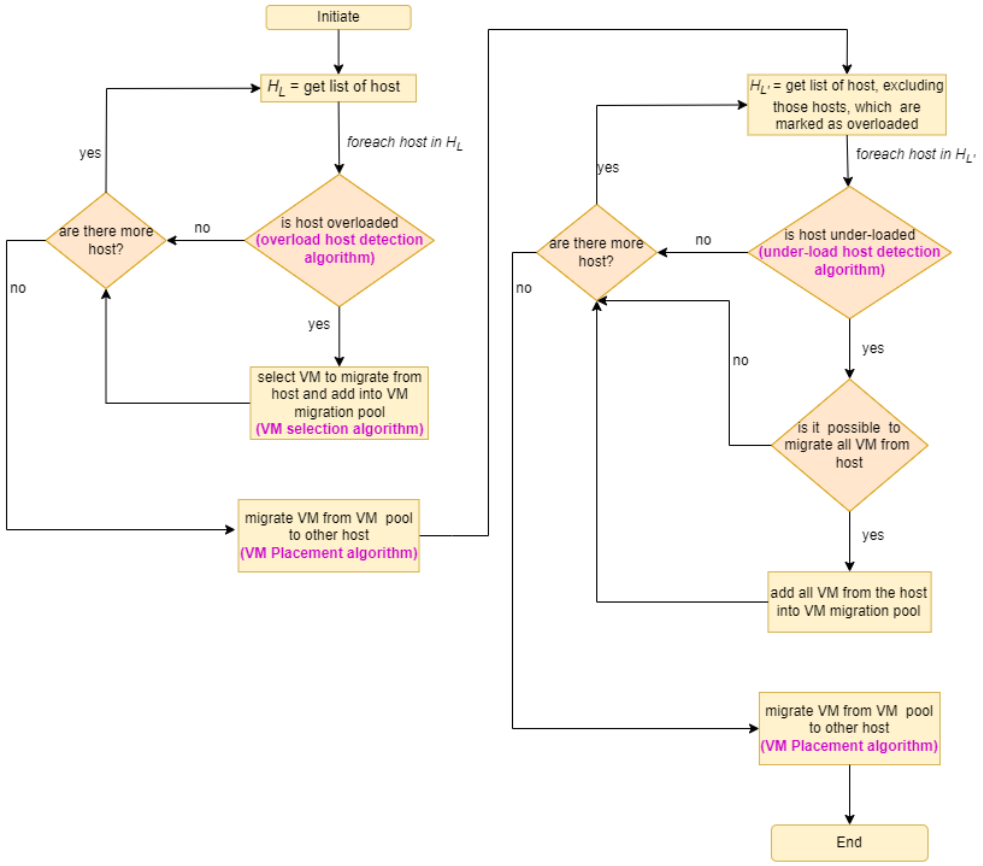

Fig. 1 Flowchart diagram of proposed approach.

3.1 Overload host detection

The host is marked as overloaded if its CPU utilization exceeds the upper threshold. In the proposed approach, utilization (∅) of the host is considered by taking the sum of the mean and the standard deviation value of the CPU utilization history of all VMs running on the host. The method to calculate utilization (∅) is given in eq. 1. To set the upper threshold, an interquartile range (IQR) [10] based approach is being used. IQR uses host historical data to set the upper threshold. In the IQR method, historical data of host utilization is collected and then arrange in sorted order. After that, the difference between the first and third quartiles of data is calculated, known as IQR. The upper threshold is set according to eq. 2. If calculated utilization (∅) using eq. 1 is greater than the calculated upper threshold, then host is marked as overloaded. Algorithm 1 shows the algorithm for overload host detection.

$$\emptyset = \frac{\sum_{i=1}^{n}(meanUtilizationHistory(V_i)+stdutilizationHistory(V_i))}{H_m} \quad (1)$$

$$upperThreshold = (1 - s.IQR) \quad (2)$$

$$IQR = Q_3 - Q_1 \quad (3)$$

In the eq. 1 V_i is i^{th} VM and H_m is total MIPS of host. In the eq. 2 *'s'* is the safety parameter [10].

Algorithm 1: Host Overload Detection
Input: host Output: status of host
1. V_L = getListOfVM(host); 2. calculate Ø using eq. 1; 3. calculate upperThreshold using eq. 2; 4. return Ø > upperThreshold;

3.2 VM selection

After a host is detected as overloaded, VMs are iteratively selected from the host until host return to normal state. VM having minimum '*mark*' is selected first. '*Mark*' can be calculated from eq. 4. '*Mark*' value is directly proportional to migration time and inversely proportional to utilization. It implies that VM having minimum migration time and maximum utilization will be selected first. Algorithm 2 shows the VM selection algorithm.

$$mark = \frac{migration\ time}{utlization} \tag{4}$$

Algorithm 2: VM selection
Input: overload host Output: list of VM to migrate
1. V_L = getListOfVM(host); 2. foreach $V_i \in V_L$ do 3. migrationTime = V_i getMigrationTime(); 4.. utilization = V_i getUtilization(); 5. mark = migrationTime / utilization ; 6. map.put(V_i, mark); 7. end foreach 8. sortAscendingBasedOnValue(map); 9. while (host is overloaded) 10. vmToMigrate.add(map.get(0)); 11. map.remove(0); 12. end while 13. return vmToMigrate;

3.3 VM Placement

The VM placement is the process of finding a suitable host for the VMs. In the presented work, the power aware best fit decreasing method (PABFD) [11] is used to find a suitable host. In the PABFD approach, VMs are sorted in decreasing order of their CPU utilization. Then VM is allocated to the host for which the power difference (power before VM allocation to the host and power after VM allocation to the host) is minimum.

3.4 Under-load host detection

The under-load host detection is the process of finding a host with very less CPU utilization and migrating all VMs of the host to another host, so that the source host can be switched off and energy can be conserved. To detect if a host is under-load, the current utilization of the host is compared with the lower threshold. If the current utilization is less than the lower threshold, then the host is marked as under-loaded. The lower threshold value is set dynamically. The lower threshold value is calculated considering the host utilization history. The mean value of the CPU utilization history of all the host is

calculated, and then their mean value is set as lower threshold. Only the latest historical data is considered to calculate the mean value of the host utilization history. Only the ten latest CPU utilization history is considered in the proposed approach. The method to calculate the lower threshold value is given in eq. 5. Algorithm 3 shows the under-load host detection algorithm.

$$lower\ threshold = \sum_{i=1}^{n} \frac{\sum_{j=1}^{10} \frac{hu_{ij}}{10}}{n} \quad (5)$$

In the eq. 5 hu_{ij} is j^{th} utilization of i^{th} host, and n is the total number of the host.

Algorithm 3: Under-load host detection
Input: host list Output: list of under-load host
1. foreach $H_i \in$ host list 2. mean = getMeanHostUtilizationHistory(H_i); 3. HU.add(mean); 4. end foreach 5. lowerThreshold = getMean(HU); 6. foreach $H_i \in$ host list 7. if (H_i.getCurrentHostUtilization) < lowerThreshold) then 8. underloadHostList.add(H_i); 9. end if 10. end foreach 11. return underloadHostList

4 Experimental Evaluation

In this section, the setup and environment for executing the proposed approach are discussed. The result comparison and analysis between the proposed approach and the existing algorithms are also discussed.

4.1 Experiment Setup

CloudSim toolkit [16] is used to run the experiment. PlanetLab [17] data is being used for real world workloads. There is one large data centre consisting of total 800 hosts of two type. One is HP ProLiant G4 and other is HP ProLiant G5.There are total four types of VM according to their characteristic. VMs characteristic are listed in table 1.

Table 1. VM Characteristic

VM Type	RAM (GB)	CPU (MIPS)
High-CPU medium instance	0.85	2500
Extra-large instance	3.75	2000
Small instance	1.7	1000
Micro instance	0.613	500

4.2 Experiment Result

The proposed method (PM) is compared with standard algorithms like IqrMc, IqrMmt, LrMc, LrMmt, and ThrMc. The proposed method (PM) is executed for ten workload data provided by PlanetLab. The

Graph is plotted based on average of result of all ten workloads. The algorithm is being compared on two most important parameter energy consumption and SLA violation.

4.2.1 Energy Consumption:

Energy consumption is considered as a crucial parameter for doing comparison between algorithms. High energy consumption increases global carbon emissions, and it also increases operational cost, which is the main concern of cloud service provider. The proposed method is compared with the standard algorithms, and the result is presented in Fig. 2. From the presented result, it is observed that energy consumption is less than other standard algorithms. It is observed that, on average, energy consumption has reduced by 18% compared to other standard algorithms.

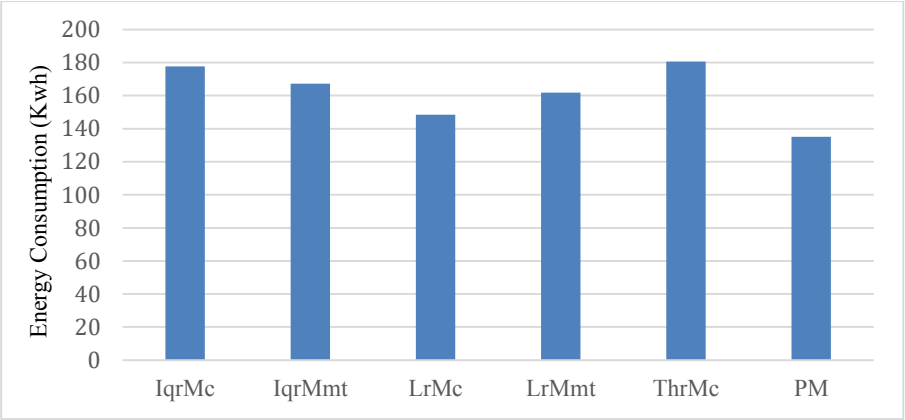

Fig. 2. Energy Consumption

4.2.2 SLA violation:

SLA stands for service level agreement. It is an agreement between a service provider and consumer to provide quality of service (QoS). If there is an increment in SLA violation, then QoS degrades, and also service provider may have to pay penalty. In Fig.3, the result of the SLA violation is presented. The proposed method has less SLA violation than other standard algorithms.

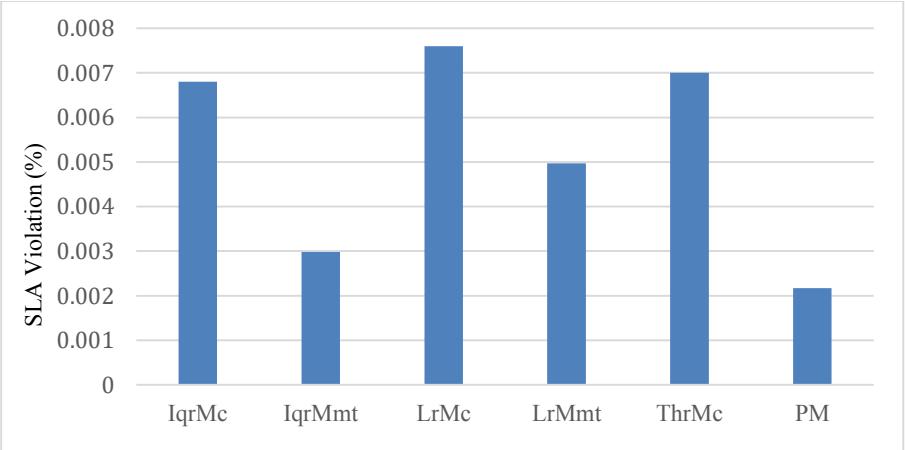

Fig. 3. SLA Violation

4.2.3 Performance Metrics:

Performance metrics is used to evaluate and thoroughly assess an algorithm's performance. Performance metrics for proposed work is defined in eq. 6. Performance metrics is optimized when energy consumption and SLA violations (ECSV) both are minimized. It can be observed from Fig. 4 that proposed method has given better result than other standard algorithms.

$$ECSV = energy\ consumption * SLA\ violation \tag{6}$$

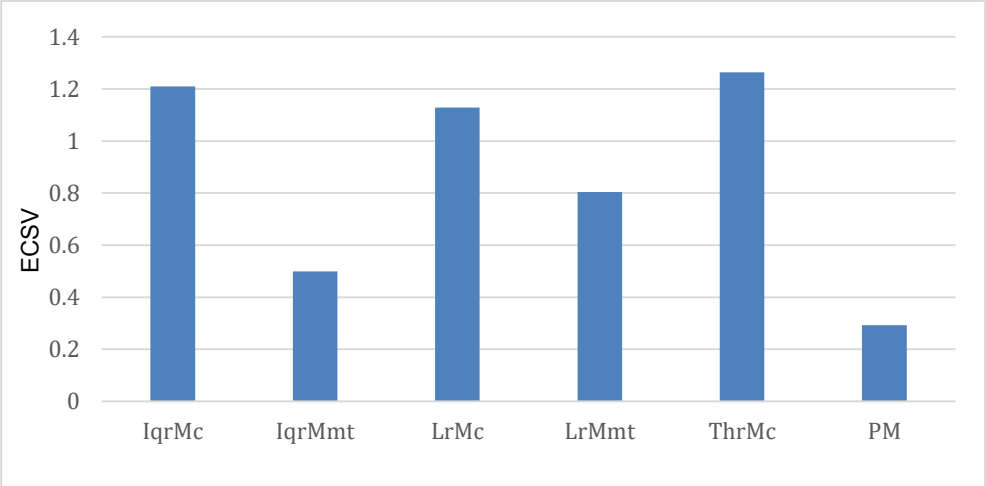

Fig. 4. Performance Metrics

5 Conclusions

This paper addresses issues of VM migration and server consolidation in the data center. In this paper, an algorithms has been proposed for overload/under-load host detection. We use hosts and VMs utilization history to make decision about host status. Also, an algorithm for VM selection policy has been proposed, which consider migration time and utilization, for VM selection. In this paper, we have primarily focused on decreasing energy consumption while maintaining SLA violation, through proper resource utilization. To evaluate the effectiveness of proposed methods, the CloudSim simulator is being used and PlanetLab workload is being used for real world data. The experiment result shows that the proposed method outperforms other standard algorithms. Energy consumption by the proposed method has decreased, showing an average 18% reduction in energy consumption, and SLA violation has also been reduced. In the future work, to detect overload/under-load host, we can also consider other parameter like RAM, storage, bandwidth, etc., instead of considering only CPU utilization.

References

1. Koomey, J. (2011). Growth in data center electricity use 2005 to 2010. *A report by Analytical Press, completed at the request of The New York Times*, ***9***(2011), 161.
2. Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems*, ***25***(6), 599-616.

3. Baliga, J., Ayre, R. W., Hinton, K., & Tucker, R. S. (2010). Green cloud computing: Balancing energy in processing, storage, and transport. *Proceedings of the IEEE*, ***99***(1), 149-167.
4. You, X., Li, Y., Zheng, M., Zhu, C., & Yu, L. (2017). A survey and taxonomy of energy efficiency relevant surveys in cloud-related environments. *IEEE Access*, ***5***, 14066-14078.
5. Mishra, S. K., Khan, M. A., Sahoo, S., & Sahoo, B. (2019). Allocation of energy-efficient task in cloud using DVFS. *International Journal of Computational Science and Engineering*, ***18***(2), 154-163.
6. Katal, A., Dahiya, S., & Choudhury, T. (2022). Energy efficiency in cloud computing data centers: a survey on software technologies. *Cluster Computing*, 1-31
7. LA Barroso and U. Holzle, "The case of energy-proportional computing", in Computer, vol. **40**, 2007, pp. 33-37.
8. Chaurasia, N., Kumar, M., Chaudhry, R., & Verma, O. P. (2021). Comprehensive survey on energy-aware server consolidation techniques in cloud computing. *The Journal of Supercomputing*, *77*, 11682-11737.
9. Clark, C., Fraser, K., Hand, S., Hansen, J. G., Jul, E., Limpach, C., ... & Warfield, A. (2005, May). Live migration of virtual machines. In *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume **2*** (pp. 273-286).
10. Beloglazov, A., & Buyya, R. (2012). Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. *Concurrency and Computation: Practice and Experience*, ***24***(13), 1397-1420.
11. Beloglazov, A., Abawajy, J., & Buyya, R. (2012). Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Future generation computer systems*, ***28***(5), 755-768.
12. Pinheiro, E., Bianchini, R., Carrera, E. V., & Heath, T. (2001). *Load balancing and unbalancing for power and performance in cluster-based systems*. Rutgers University.
13. Fu, X., & Zhou, C. (2015). Virtual machine selection and placement for dynamic consolidation in Cloud computing environment. *Frontiers of Computer Science*, ***9***, 322-330.
14. Taheri, M. M., & Zamanifar, K. (2011, December). 2-phase optimization method for energy aware scheduling of virtual machines in cloud data centers. In *2011 International Conference for Internet Technology and Secured Transactions* (pp. 525-530). IEEE.
15. Li, L., Dong, J., Zuo, D., & Wu, J. (2019). SLA-aware and energy-efficient VM consolidation in cloud data centers using robust linear regression prediction model. *IEEE Access*, *7*, 9490-9500.
16. Buyya, R., & Murshed, M. (2002). Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing. *Concurrency and computation: practice and experience*, ***14***(13-15), 1175-1220.
17. Chun, B., Culler, D., Roscoe, T., Bavier, A., Peterson, L., Wawrzoniak, M., & Bowman, M. (2003). Planetlab: an overlay testbed for broad-coverage services. *ACM SIGCOMM Computer Communication Review*, ***33***(3), 3-12.