International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 16 (2018) pp. 12917-12922

A Proposed Energy and Performance Aware Cloud Framework for Improving Service Level Agreements (SLAs) in Cloud Datacenters

Ali Abdullah Hamed Al-Mahruqi[1], Vallavaraj Athinarayanana[1], Gordon Morison[2] and Brian G Stewart[3]

[1]*Department of Electrical and Computer Engineering, Caledonian (University) College of Engineering. Muscat, Oman.*

[2]*School of Engineering and Built Environment, Glasgow Caledonian University, Glasgow, G4 0BA, UK.*

[3]*Department of Electronic and Electrical Engineering, University of Strathclyde, Glasgow, G1 1XW, UK.*

Abstract

Physical computer hardware is being replaced with virtual hardware in cloud computing for cost efficient operations. It is expected that by 2020, most of medium and large organizations will migrate to cloud computing for enhanced and sustainable business. However, the trade-off between meeting Service Level Agreements (SLA) and minimizing energy consumption of physical machines in data centers is not fully optimized, thus leaving scope for further improvement. In spite of the fact that cloud providers are using state-of-the-art technologies, cloud clients are still demanding higher and ever increasing Quality of Service (QoS) to satisfy the need of customers. Based on the authors' previous practical work on a high-end server on ESXi 5.5 hypervisor platform, a novel framework for improving computational efficiency, performance and reducing cloud energy consumption is proposed in this paper. The proposed framework integrates with hardware through a server classification process (idle server, under-loaded server, balance server and over-loaded server) and distributes computational loads with a built-in logic to reduce energy consumption. In addition, the framework promises an efficient solution for Virtual Machines (VMs) allocation and optimization that will satisfy SLAs for cloud consumers. In all four server categories, tracking and a recording system is considered for Physical Machine (PMs) and VMs. For effective utilization of the idle server state, a wake-up and sleep mode decider are proposed. The uniqueness of the framework can be validated with its implementation on CloudSim software.

Keywords: Cloud Computing; Framework; Service Level Agreement, Green Cloud Computing; Energy Efficient Cloud Computing; Quality of Service

INTRODUCTION

Cloud computing offers on demand conveyance of infrastructure that provides an opportunity for many organizations to deploy computer applications and other related IT resources from service providers with minimal costs [1]. The resources of cloud computing are managed by service providers (through third parties) who offer Software as a Service (SaaS), Platform as a Service (PaaS), Infrastructure as a Service (IaaS) and other service models [2]. As the population of users of computing power is continually increasing, cloud computing is expected to dominate desktop computing by 2020 [3]. This is a clear indication that cloud-computing resources will play a significant part in the future of strategic computing. Though there are many cloud service providers (CSP) on the market, it has become a major challenge for clients to select an appropriate CSP. Most clients are prone to change a provider if the level of expected satisfaction is not met. Normally most experienced cloud clients measure the QoS provided against the required SLA [4] that have been agreed in order to judge if a CSP is providing good value to their business.

Energy efficiency (EE) and QoS have become important factors within cloud computing environments because of the increasing number of cloud datacenters. In order for datacenters to sustain their services to cloud clients, they are obliged to ensure proper planning of their resources in terms of improved energy consumption and improved client performance. The lack of proper planning in relation to energy efficiency and resource performance can potentially fuel a significant increase of overall operational costs, which can result in failure to retain customers when required QoS and SLA targets are at stake.

In order to address these issues, we propose a cloud computing framework that will benefit data centers by improving energy consumption challenges and SLA requirements. The framework offers an efficient solution for the optimization of VMs allocation to PMs that will satisfy SLAs for cloud consumers and help minimize general energy consumption for cloud providers.

The organization of this paper is as follows. Related work is discussed in section II. Previous experimental work in relation to server utilsation is presented in section III. The proposed framework is elaborated upon in Section IV, while Section V summarises the main conclusions related to the proposed framework.

RELATED WORK

Several researchers [5]-[8] have suggested monitoring cloud service providers in relation to meeting the requirement of SLAs for their consumers. Authors in [9] proposed a framework that could be used to monitor the efficiency of SLAs. The research analyzed SLA parameters such as response time, job execution time, threat limits, runtime data etc to isolate the sources of SLA violations. Based on the prediction of SLA violations, implementation of improvements in an attempt to mitigate the violations was introduced through

adaptive resource allocation by making use the results of SLA violation. The adaptive resource allocation system does provide cloud applications with additional computer resources which will help to reduce the SLA violation. The authors claimed that such a method was able to reduce the occurrence of SLA violations. As per their test results, the research claimed to produce efficient outcomes which reduce the SLA violation occurrence and satisfy cloud clients and cloud providers. The proposed framework works well in private cloud computing rather than any other types of cloud computing such as public and hybrid [2]. Other research performed by [10] proposed a framework algorithm that offered a load balancing technique and QoS improvements among servers. The proposed algorithm was divided into two stages. The first stage used an SLA scheduling algorithm to determine the highest priority tasks to be allocated to the available server. The second stage concentrated on a monitoring algorithm for idle servers which was used to balance the load for each working server. The algorithms were implemented and tested in a cloud simulation environment. The results claimed to have improved response time and effective resource utilization with better load balancing among servers compared to other existing algorithms.

Proposed research by [11] suggested a framework for SLA assurance for both cloud benefactors and consumers. The work proposed metrics (application performance metrics and network performance metrics) to be used in performance improvement for different types of applications, however the framework has still to be evaluated and tested in different server environments to assure QoS.

The above research appears to contribute to QoS improvements that help to reduce SLA violations. However, most of the proposed techniques, algorithms and metrics do not consider different server load categories i.e. server idle, underloaded, balance and overloaded conditions, which are important to balance the trade-off between meeting an appropriate SLA and minimizing the energy consumption of PMs. Additionally most of the above developments have focused primarily on simulations, with no testing on real environments of PMs and VMs. Since previous initial research by the authors in [12] was able to categorise practical server utilization into different load categories with different metrics, we believe the following framework and methodology will be able to tackle issues associated with SLA and help cloud providers to improve further QoS for consumers.

EARLIER EXPERIMENTAL SETUP

In the previous research [12], a test bed was set up to investigate utilization of cloud applications in relation to energy consumption and QoS performance on a real server. A VM consolidation setup was performed based on the ESXi hypervisor on PMs that was used to determine overloaded, underloaded, balanced and idle parameters in a server. The setup included one high-end server, which was consolidated with 17 VMs. Each VM in the high-end server was installed with a different operating system that consisted of servers and clients. The size of Random Access Memory (RAM) and number of Central Processing Unit (CPU) cores were allotted to each VM based on minimum requirements (i.e. 2GB of RAM for a client computer and 4GB for a server, 1 CPU core for a client and 2 CPU cores for a server). V-Center was used for the management of all consolidated VMs, and V-Sphere client was used to connect and access the hypervisor remotely. The test bed architecture and set up are shown in Figures 1 and 2. The framework was set up by using a type 1 “bare metal” hypervisor, whereby the hypervisor is installed on the empty server without any operating system.

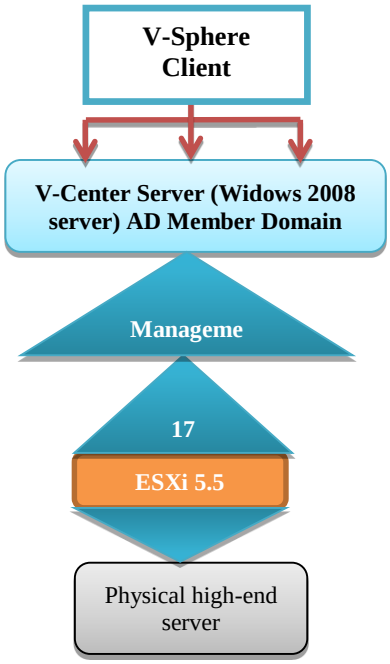

Figure 1. Test bed architecture

(ESXi 5.5, VSphere Client and V-center)

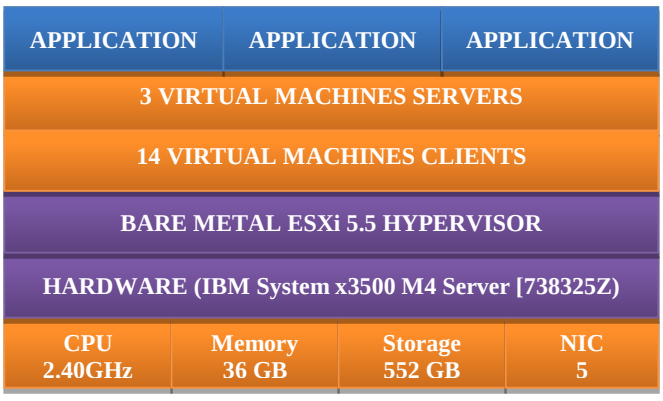

Figure 2. Cloud test bed setup on high-end server.

Testing

A test was performed by checking the utilization of the CPUs and primary memory (RAM which stores information temporality while the computer is in use). Though CPU is considered to be a major contributing factor in energy consumption[13], primary memory is another contributing

factor which can degrade performance and contribute to energy consumption [14][15]. Based on this, the following utilization equations were formulated (adapted from [13]) that include nodes with CPU and memory utilization.

$$G_{Cloud} = G_{PMserver} + VMs_{usage} \quad (1)$$

$$G_{PMServer} = G_{CPU_{usage}} + G_{Memory_{usage}} \quad (2)$$

Equations (1) and (2) checks the usage of CPU and memory while all the VMs are switched ON in a physical machine within the cloud environment. In our experiment, we consider cloud computing to consist of physical machines consolidated with virtual machines. As that is the case, all VMs will compete to utilize CPU and Memory which will help to judge the utilization of energy consumption and performance. G_{Cloud} stand for the global cloud computing, $G_{PMserver}$ stands for Global physical machines server, VMs_{usage}stands for usage of virtual machines, $G_{CPU_{usage}}$ stands for Global usage of CPU and $G_{Memory_{usage}}$ stands for global usage of memory.

Table 1: Load Testing Criteria

Category	Description
Idle servers	All the VMs were switched ON, no load was given to the VMs, and the results were recorded after all the VMs were stable from rebooting
Under loaded servers	All the VMs were switched ON and 3D videos were played on all of the VMs and multiple applications were open on some of the VMs.
Overloaded servers	All the VMs were switched ON and multiple tests were done as follows: Playing 3D video on all of the VMs, disk defragmentation on all VMs, open of multiple applications on all VMs and scanning of all of the VMs by antivirus.
balance servers	All the VMs were switched ON and the test was performed as follows: Playing 3D video on some of the VMs, disk defragmentation on some of the VMs and multiple application were opened on all VMs

The load testing criteria is shown in Table 1. The estimated percentages of server utilization in order to judge when the server will become overloaded, under loaded, balance and idle are shown in Table 2. The parameters below were decided after stressing the server in each category - the test was conducted by sending the load to the VMs for about an hour and results were retrieved and recorded [16]. The parameters may slightly change due the load being given and tested in physical server. However based on the behavior of the server in term of performance, the fixed values were decided for each category. Google datacenters use a strategy of 50% maximum CPU utilization to enable best performance and minimization of VMs migration [17]. This was based on their decision to have zero tolerance on performance in order to satisfy customer needs. This strategy appears to be acceptable since Google can afford to set all their physical machines to 50%. However, 70% - 75% is still good enough as the stability of server performance is convincing [18]. The parameters below are important to the proposed framework as each category is used to improve the framework performance and assist cloud datacenters

Table 2: Stress Parameters

Testing Result		
#	**Category test**	**Percentage**
1	Overloaded	>75%
2	Underloaded Server	15%-40%
3	Balance Server	41%-60%
4	Idle Server	5%-15%

$$\text{Load category} = \begin{cases} PM_{OL(i)} & \text{if } C_i \geq 75\% \\ PM_{UL(i)} & \text{if } 15\% \leq C_i \leq 40\% \\ PM_{BL(i)} & \text{if } 41\% \leq C_i \leq 60\% \\ PM_{IS(i)} & \text{if } 5\% \leq C_i \leq 15\% \end{cases} \quad (3)$$

Equation (3) was formulated after our earlier experiment [16], which is summarized in table 2. $PM_{OL(i)}$ stands for overloaded physical machine load category, $PM_{UL(i)}$ standing underloaded physical machine load category, $PM_{BL(i)}$ stands for Balanced physical machine load category and $PM_{IS(i)}$ stands for Idle physical machine load category.

PROPOSED FRAMEWORK

Figure 3 shows the proposed framework to prevent a server reaching the stage of being overloaded. When utilization is measured between 73%-75% then the server is about to be overloaded as it is approaching the 7% level and should trigger the other categories of servers such idle, underloaded and balance to receive the VMs. At this stage, a special mechanism is proposed to check the availability of resources. The framework will use a *PM and VM Tracking System* to check all server categories for possible migration. If the idle server is found to be the only option to transfer VMs, then the *PM and VM Tracking System* will communicate to the *Wakeup and sleep decider* to trigger the idle server via a *Wakeup Trigger* which enables wake up of the idle sever from sleep mode. If there is no need of utilizing the idle server, then the idle server will remain in sleep mode via a *Sleep Trigger*. If the server is

under loaded or balanced and is found to have sufficient space in the resources for VM migration, then the *PM and VM Tracking System* will use similar communications to trigger the balanced or underloaded server to receive the VM load. If any of the categories is selected, then the *PM and VM Tracking System* will communicate to the *Migrate VMs to available Resources* through the *VMs assigner to PM* in order to transfer the load to any of the selected resources. All the migrations and availability of resources will be stored in a *History Recording System* for referencing and logging all system migrations.

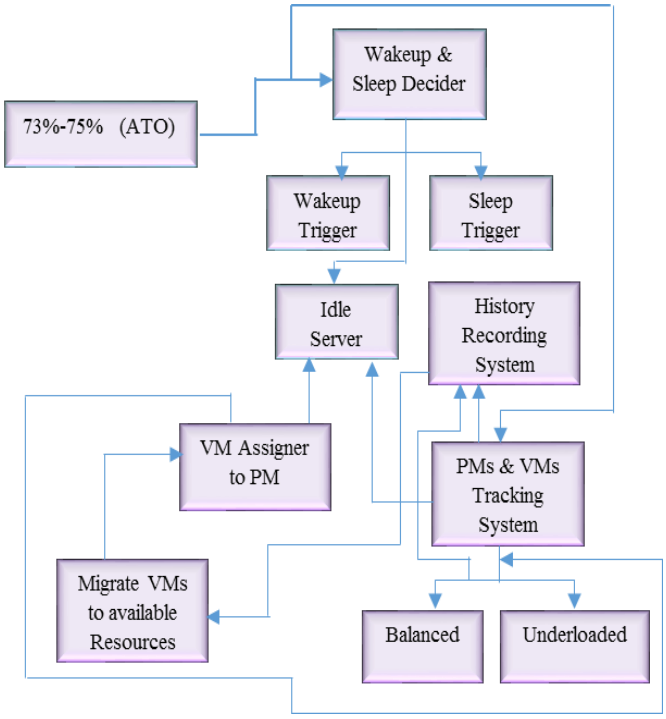

Figure 3. Impending overload.

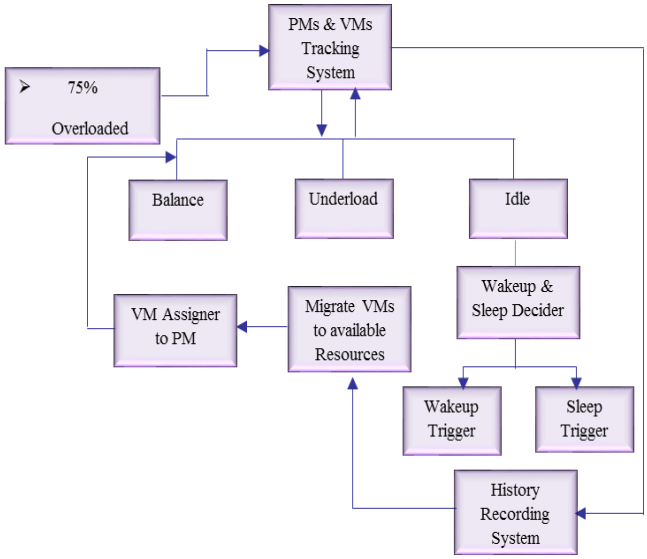

Figure 4. Overloaded Category.

Figure 4 shows part of the framework that operates at the overloaded stage. This is where the server becomes fully overloaded based on the utilization of CPU and memory capacity which will not be able to receive any VMs from other physical machines. It can either transfer some of the VM to an underloaded, balanced or idle server. When the server becomes overloaded, it will use the *PM and VM Tracking System* to track availability of resources of three categories i.e. balanced, underloaded and idle server. If any of the categories are found to have enough space to receive the VMs, then the *PM and VM Tracking System* will communicate to trigger the balanced, underloaded or idle server to receive the load. If the two categories of balanced and underloaded do not possess enough resources, then the idle server will be triggered to wake up from sleep mode.

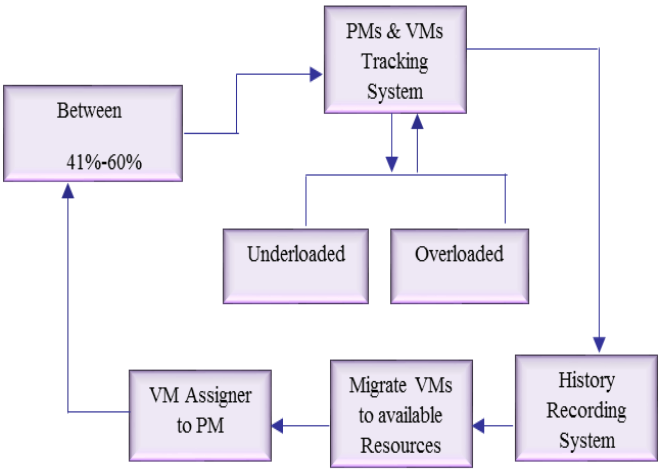

Figure 5. Balanced Category.

Figure 5 shows part of the framework that summarises the balanced server stage. Balanced category is where the server is neither under loaded, nor overloaded or idle. It has a partial workload, and may have some space to accommodate VMs from either overloaded or under loaded servers. The balanced category is an important category that avoids unnecessary usage of the resources in a cloud system. When the server becomes balanced, it will use the *PM and VM Tracking System* to check if there are any servers in the underloaded or overloaded categories. If the underloaded category is available, then the *PM and VM Tracking System* will communicate to the *Migrate VMs to available resources* through the *VMs assigner* to the balanced server and allow the underloaded server to become idle which will in turn be put into sleep mode. If an overloaded server is available, then the *PM and VM Tracking System* will communicate to the *Migrate VMs to available resources* through the *VMs assigner* to the balanced server and allow the overloaded server to operate without performance degradation after migrating the VMs to balanced server.

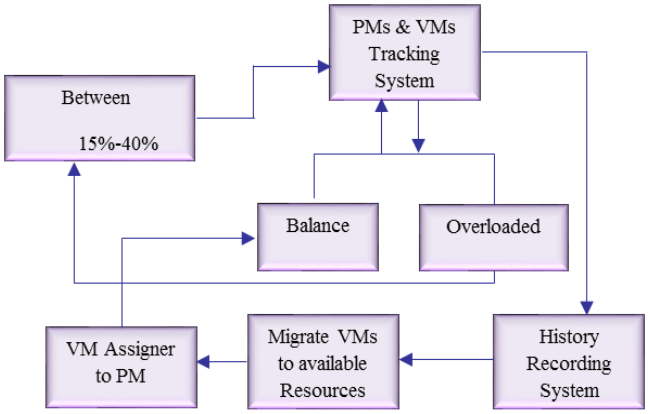

Figure 6. Underloaded Category.

Figure 6 is a part of the framework that indicates the underloaded stage. This is where the server has minimal tasks to perform and is between the balanced categories. (Between 15% - 40 % of server utilization) At this stage, the *PM and VM Tracking System* checks if there is a balanced server to receive the load (VMs) from the underloaded server or if there is overloaded server to transfer the load to the underloaded server. If all the conditions are found to be true, then the *PM and VM Tracking System* will communicate to the *Migrate VMs to available resources* through the *VMs assigner* and transfer the load from the overloaded server to the underloaded server which will enable the overloaded server to become normal. It could also decide to transfer the load from the underloaded server to a balanced server and allow the underloaded server to become idle and be placed into sleep mode.

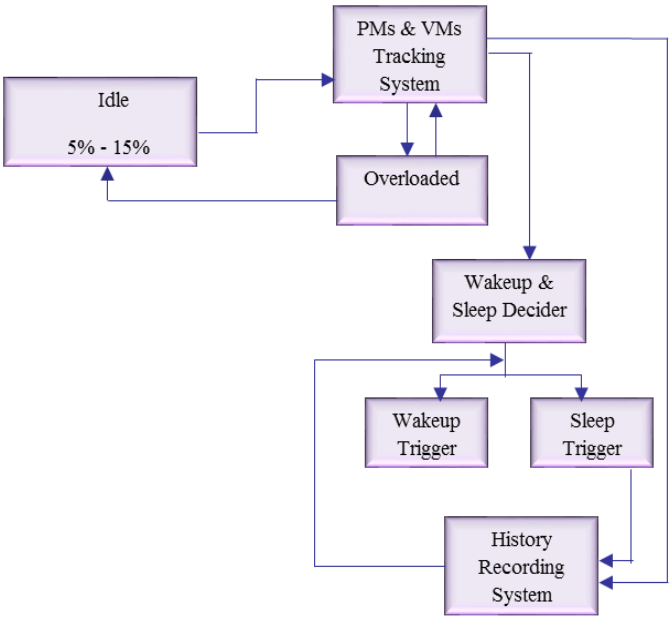

Figure 7. Idle Category.

Figure 7 shows the idle server stage within the proposed framework. This is where the server is completely without any load which means all the VMs are switched ON but no tasks are given to any of the VMs. At this stage, idle server can either receive the load from an overloaded server or be turned into power saving mode. If the server is idle, it will use the *PM and VM Tracking System* to check if the overloaded server is available to send the load and to prevent the overloaded server from becoming further overloaded. If there is no overloaded category, the *PM and VM Tracking System* will communicate to the *Wakeup and sleep decider* to trigger the idle server via the *Sleep Trigger* and put the idle server into sleep mode.

CONCLUSION

A green cloud-computing framework is proposed in this paper meeting the requirements of SLAs to cloud customers. The framework operation is embedded with certain built-in key aspects like; reduced energy consumption, efficient VMs allocation, and optimization. The framework consists of five stages, whereby all the five stages will be used to prevent the server from being overloaded, under-loaded, balance and idle. Special mechanism such as *PM and VM Tracking System* are to be used to identify the category of the server and communicate to *VM Assigner to PM* in order to migrate the VMs to a server that is found to have enough capacity. The entire framework will ensure all servers operate with optimized performance and will prevent SLA violations between cloud datacenters and cloud clients. Future research work will involve implementing the proposed framework and validating its performance in a cloud simulation environment.

REFERENCES

[1] Singha. S, Young. S and HyukParka. J, "A survey on cloud computing security: Issues, threats, and solutions," Journal of Network and Computer Applications., vol. 75, pp. 200-222, November. 2016.

[2] Al-Mahruqi. A, Stewart. B and Hainey. B, "A review of green cloud computing techniques and algorithms," 2nd International Conference on next generation of computing and communication technology., pp. 1-7, 22-23, April 2015.

[3] Owusu. F. and Pattinson. C, "The current state of understanding of the energy efficiency of cloud computing," IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications., pp. 1948–1953, September. 2012.

[4] Mohamed. M, Obinna. A, Sakairi. T, Tata. S, Mandagere. N and Ludwing. H, "The rSLA Framework: Monitoring and Enforcement of Service Level Agreements for Cloud services," IEEE International Conference on Services Computing., pp. 625–632, September. 2016.

[5] Fu. H, Xinjun. M, Wei. W, Lu. L and John. P, "A Cloud-Oriented Services Self-Management Approach Based on Multi-Agent System Technique," IEEE International Conference on Utility and Cloud Computing., pp. 261–268, 2014.

[6] Khan H, Gaik-Yee. C and Fang-Fang. C, "An Adaptive Monitoring Framework for Ensuring Accountability and Quality of Services in Cloud Computing," IEEE International Conference on Information Networking., pp. 249–253, 2016.

[7] Ahmed. T, Salman. M and Neeraj. S. "SLA-based Service Selection for Multi-Cloud Environments," IEEE 1st International Conference on Edge Computing., pp. 65–72, 2017.

[8] Issaret. P, Wachirawit. A, Nasoret. P, Montri. S and Praisan, P. "Auto-scaling microservices on IaaS under SLA with cost-effective Framework," IEEE Tenth International Conference on Advanced Computational Intelligence.; pp. 583–588, 2018.

[9] Anithakumari. S and Chandransekaran. K, "Monitoring and Management of Service Level Agreements in Cloud Computing," IEEE International Conference on Cloud and Autonomic Computing., pp. 204–207, 2015.

[10] Rajeshwari. B and Dakshayini. M, "Optimized Service Level Agreement Based Workload Balancing Strategy for Cloud Environment," IEEE International Advance Computing Conference (IACC)., pp. 160–165, 2015.

[11] Zainelabden. A, Ibrahim. A, Kliazovich. D and Bouvry. P "Service Level Agreement Assurance between Cloud Services Providers and Cloud Customers," 16th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing., pp. 588–591, 2016.

[12] Al-Mahruqi. A, Stewart. B, Hainey. B, Vallavaraj. A, "A Review of Performance and Energy Aware Improvement Methods for Future Green Cloud Computing," International Journal of Computer Applications., (0975 – 8887) Volume 144 – No.11, pp. 18–24, June. 2016.

[13] Aschberger. C and Halbrainer. F, "Energy Efficiency in Cloud Computing.," pp. 1–16, 2013.

[14] Mukherjee. K, "Green Cloud : An Algorithmic Approach," International Journal of Computer Applications., (0975 – 8887) Volume 9– No.9, pp. 1–6, November. 2010.

[15] Liang. Y, Wu. W, Dichen. Di, Zhang. F, Yan. Y, "A resource scheduling algorithm of cloud computing based on energy efficient optimization methods," International Green Computing Conference (IGCC)., pp. 1–6, 2012.

[16] Al-Mahruqi. A, Stewart. B and Hainey. B, "Energy and Performance-Aware Server VMs Consolidation in IaaS Cloud Computing," 2nd International Conference on next generation of computing and communication technology., pp. 1-6, April 22-23, 2015.

[17] Nath. S, Roytman. A, Kansal. A, Govindan. S and Liu. J, "Algorithm Design for Performance Aware VM Consolidation," Miicrosoft Res., pp. 1–32, 2013.

[18] Hsu. C, Chen. S, Lee. C, Chang. H, Lai. K, Li. K and Rong. C, "Energy-Aware Task Consolidation Technique for Cloud Computing," in 2011 Third IEEE International Conference on Coud Computing Technology and Science., pp. 115–121, 2011.