An Approach for Energy Efficient Dynamic Virtual Machine Consolidation in Cloud Environment

Sara Nikzad[1]
Department of Computer Engineering
Isfahan (Khorasgan) Branch
Islamic Azad University
Isfahan, Iran

Seyed EnayatOllah Alavi[2,*]
Department of Computer Engineering
Shahid Chamran University of Ahvaz
Ahvaz, Iran

Mohammad Reza Soltanaghaei[3]
Department of Computer Engineering
Isfahan (Khorasgan) Branch
Islamic Azad University
Isfahan, Iran

***Abstract*—Nowadays, as the use of cloud computing service becomes more extensive and the customers welcome this service, an increasing trend in energy consumption and operational costs of these centers may be seen. To reduce operational costs, the providers should decrease energy consumption to an extent that Service Level Agreement (SLA) maintains at a desirable level. This paper adopts the virtual machine consolidation problem in cloud computing data centers as a solution to achieve this goal, putting forward solutions to make the decision regarding the necessity of migration from hosts and finding appropriate hosts as destinations of migration. Using time-series forecasting method and Double Exponential Smoothing (DES) technique, the proposed algorithm predicts CPU utilization in near future. It also proposes an optimal equation for the dynamic lower threshold. Comparing current and predicted CPU utilization with dynamic upper and lower thresholds, this algorithm identifies and categorizes underloaded and overloaded hosts. According to this categorization, migration then occurs from the hosts that meet the necessary conditions for migration. This paper identifies a certain type of hosts as "troublemaker hosts". Most probably, the process of prediction and decision making regarding the necessity of migration will be disrupted in the case of these hosts. Upon encountering this type of hosts, the algorithm adopts policies to modify them or switch them to sleep mode, thereby preventing the adverse effects caused by their existence. The researchers excluded all overloaded, prone-to-be-overloaded, underloaded, and prone-to-be-underloaded hosts from the list of suitable hosts to find suitable hosts as destinations of migration. An average improvement of 86.2%, 28.4%, and 87.2% respectively for the number of migrations of virtual machines, energy consumption, and SLA violation is among the simulation achievements of this algorithm using Clouds tool.**

Keywords*—Cloud Computing; Service Level Agreement; Energy Consumption; Virtualization; Dynamic Consolidation; Data Center

I. INTRODUCTION

Cloud computing is a model that provides access to infrastructure including a set of configurable computing resources such as servers, storages, applications, services, etc. This model provides them for applicants via available connection infrastructure such as network and the Internet in an easy, rapid and on-demand manner, while taking into account quality of service. IaaS, PaaS, and SaaS are the three major types of cloud computing services [1, 2]. IaaS presents data centre infrastructures, servers and storage spaces as well as hardware policies independently from location and geographical limitations and under computer networks. Instead of purchasing IT infrastructure and getting involved with equipment maintenance and enhancement, organizations fulfill their computational needs using cloud computing on a pay-as-you-go basis [3, 4].

In recent years, in light of the ever increasing expansion of the use of cloud computing services and due to the fact that customers welcomed this service, cloud computing service providers have increased the number and volume of greedy data centres that consume huge amounts of energy [5]. This has incurred enormous operational costs. Quality of service assurance, included in SLA, and is agreed upon between customers and providers, is a necessity for the cloud computing environment. Hence, cloud computing service providers tend to bring about a trade-off between energy and performance and should reduce energy consumption because it would not disrupt or decrease quality of service to reduce operational costs [3].

The major energy loss mainly occurs in hardware infrastructure of cloud computing data centres. Research has shown that although the power consumed by hardware equipment is idle, it is almost equal to that at the peak of consumption. Thus, failure to utilize them in a perfect manner will result in a huge energy loss [6]. In this respect, Forrester research team observed that when a server is idle for 70% of the time, it consumes a power of almost 30% of the consumption peak power [7]; therefore, what mainly accounts for energy loss in cloud computing data centres is the use of equipment while their utilization is at low levels [6]. Virtualization, is the key feature and main basis of cloud computing, making possible the establishment of several VMs on a host as well as the migration of VMs [8].

The optimal consolidation problem of VMs using virtualization technology is an effective approach to achieve energy efficient cloud computing data centres [9-11]. Because it allows VMs on hosts to migrate to other suitable hosts when the work load of hosts is low, and the hosts that have become idle to switch to sleep mode [5].

The study employed live migration of VM to transfer VM without suspension and with minimum downtime. However, any VM migration involves certain performance degradation and consequently potential SLA violation [12]. On the other hand, unnecessary VM migration will lead to extra management costs (such as VM reconfiguration, VM creation

and destruction, etc.), resulting in additional energy consumption [13]. Hence, the researchers avoided unnecessary VM migrations to reduce SLA violation and energy consumption to the extent possible, i.e. they should minimize the number of migrations. This paper addressed VM dynamic consolidation problem in cloud computing data centres as a solution to tackle the mentioned problems.

In some related studies, the authors made decisions just based on current utilization of hosts. On the other hand, in some other related studies, the researchers made decisions just based on upper threshold. However, In this study, Proposed Algorithm makes decision based on dynamic upper and lower threshold as well as current and predicted CPU utilization. Solutions are put forward in this regard that are briefly described as follows:

- Proposing an optimal equation to calculate the dynamic lower threshold and presenting a technique to identify and categorize underloaded hosts.
- Decision making regarding the necessity of migration from hosts using the comparison of current and predicted CPU utilization with dynamic upper and lower thresholds as well as the identification and categorization of overloaded and underloaded hosts.
- Identifying a certain type of hosts called troublemaker hosts and adopting policies to modify them or switch them to sleep mode.
- Presenting a method to find hosts that are appropriate destinations by excluding existing hosts in all of the considered categories.

Further in this paper, section 2 investigates previous research. Section 3 describes the proposed algorithm. Section 4 determines the capability of running the proposed algorithm using Clouds tool. Finally, Section 5 presents the conclusion and looks into future works.

II. Previous Research

In [14], Wu et al. used GA to solve the consolidation problem. They investigated the energy consumed by physical machines and inter-connection networks in data centers. They found out that compared with FFD [15], their technique generates better solutions. However, FFD is faster in calculation compared with their method.

In [11], the authors put forward MU, MMT, MC, and RC policies for the VM selection problem. They proposed MAD and IQR techniques to find the dynamic upper threshold,. In their study, a host is considered to be an overloaded one provided that its current CPU utilization is greater than the dynamic upper threshold.

In [16], Tang and Pan proposed an Hybrid Genetic Algorithm (HGA) to solve the consolidation problem. To rapidly improve solutions, they adopted a local optimization procedure. To gradually work out the violations of conditions in infeasible solutions and convert an infeasible solution to a feasible one, they employed an infeasible solution repairing procedure. They realized that HGA converges faster than GA and also exhibits remarkably better results in terms of performance and utilization.

In [17], the authors divided up the entire population to a number of families and performed genetic operations on these families in parallel in order to generate an optimal mapping between the set of hosts and VMs. Thus, they presented Family GA, which is a model of Parallel GA. They made use of a self-adjusting mutation operator to prevent untimely convergence in the people of the population.

In [18], Farahnakian et al. adopted the regression forecasting technique k-nearest Neighbor, proposed in [19], to forecast resource utilization. They solved the consolidation problem using the current utilization of resources and prediction of resource utilization in future.

In [6], the authors proposed policies to determine underloaded hosts as well as a policy for the placement of migratable VMs, where they used a multi-criteria decision making technique. They also put forward a novel and comprehensive procedure for cloud resource management called Enhanced Optimization (EO) that offers an all-embracing outlook on the resource management procedure.

In [12], Singh and Shaw proposed an algorithm for decision making about the necessity of migration and finding the appropriate destination host using time-series forecasting technique as well as dynamic upper threshold and moving average, SES, and DES techniques. In their algorithm, a host is determined to be overloaded whose current and predicted CPU utilization is greater than the upper threshold.

In [20], the authors made changes to certain parts of the original ant colony algorithm, namely pheromone updating, definition, and aggregation, in a way that it would be fit to be used in multi-objective problems. They adopted the mentioned algorithm to allocate resources to VMs in order to reduce energy consumption and waste of resources. The results indicated a better performance of this algorithm compared with GA in terms of both aspects.

In [13], Fu and Zhou presented two new policies, namely MP and MCC. The first policy functions with the aid of satisfaction from resources and CPU utilization and selects VMs for migration using dynamic upper and lower thresholds. The second policy functions using a correlation coefficient and finds the suitable destination host.

III. The Proposed Algorithm

In light of [21], the authors divided the VM dynamic consolidation problem in cloud computing data centres into the following three parts:

- Part 1: Decision making regarding the necessity of migration from hosts.
- Part 2: VM selection for migration.
- Part 3: Finding suitable destination hosts.

The authors answered the first and the third parts of the consolidation problem using the proposed algorithm. To answer to the VM selection problem, the study adopted the

minimum utilization (MU) policy presented in [11], as in [12]. From among the VMs existing on one host, this policy selects the VM with minimum CPU utilization.

A. Decision Making regarding the Necessity of Migration from Hosts

In the proposed algorithm, to make decision regarding the necessity of migration from hosts, current CPU utilization is not the mere criterion to take action; rather, as in [12], the authors also used the CPU utilization history of the host in several recent periods and forecasted CPU utilization in near future by taking advantage of time-series method and DES technique. [22-24] provided further explanation about the forecasting method. In contrast with [12], where it merely employed dynamic upper threshold, this algorithm also makes use of the dynamic lower threshold. Comparison of forecasted and current CPU utilization with dynamic upper and lower thresholds determined the status of each host. The authors, then categorized hosts and made decisions regarding the necessity of migration using the categorization mentioned above. The following section describes the proposed algorithm.

This algorithm receives a host as input, examines its status, and makes a decision for its migration. As in [12], the method presented in [11] (Step 4) calculated the upper threshold, where the authors considered the value of parameter s_1 to be 2.5 in conformity with [11]. To calculate the dynamic lower threshold, they proposed and adopted equation (1), (Step 6) inspired by the upper threshold method presented in [11] and using median absolute deviation (MAD) method. It is noteworthy that MAD technique [11] also proposed. In (1), they considered the value of s_2 to be 2.5 empirically and through conducting numerous experiments.

$$\text{Lower Threshold} = 0.3 + s_2 \times \text{MAD} \quad (1)$$

The study included a great number of experiments to identify a certain type of hosts that are prone to the emergence of undesirable conditions. These hosts are called "troublemaker hosts" in this paper. To prevent the emergence of undesirable conditions and improve results, Step 6 of the proposed algorithm adopts policies to identify troublemaker hosts given the status of the hosts, and then modifies or removes them.

The first type of troublemaker hosts are those whose MAD is greater than 0.065 (this number is obtained empirically). Considering that MAD shows the strength of the deviation of the host CPU utilization, the experiments demonstrated that when the value of MAD exceeds 0.065, the deviations of CPU utilization increase and algorithm accuracy in predicting CPU utilization decreases.

On the other hand, as MAD increases, the accuracy of the calculation of dynamic upper and lower thresholds decreases, in a way that it may consider a host with a normal load to be overloaded or underloaded and this may lead to unnecessary migrations; or it may consider an overloaded or underloaded host to be a host with normal load, and this may lead to SLA violation in the host mentioned above. It is notable that the bigger the MAD, the smaller the upper threshold. Thus, CPU utilization is not used desirably. On the other hand, as MAD increases, it is more likely for CPU utilization to reach 100% and SLA to violate [11]. In the proposed algorithm, to prevent potential problems from arising regarding this type of troublemaker hosts, it considered the lower threshold to be equal to 0.9. Using this technique, this type of hosts most probably become underloaded and all of the VMs thereon are transferred to suitable hosts. Then the idle hosts will switch to sleep mode.

Proposed Algorithm:
Decision making regarding the necessity of migration

Input: host
Output: migration decision (true/false)

```
1)  flagPO = flagFO = flagPU = flagFU = false
2)  Find current Utilization of host h
    Utilization = total requested MIPS/h.getTotalMips()
3)  data[] = h.getUtilizationHistory()
4)  Find UpperThreshold using MAD
    UpperThreshold = 1- s1× MAD
5)  Find LowerThreshold using MAD
    LowerThreshold = 0.3+ s2× MAD
6)  if (MAD > 0.065) then
        LowerThreshold = 0.9
    else
        if (UpperThreshold < 0.85) then
            UpperThreshold = 0.9
        if (LowerThreshold > 0.35) then
            LowerThreshold = 0.3
7)  if (Utilization > UpperThreshold) then
        flapPO = true
8)  if (Utilization < LowerThreshold) then
        flapPU = true
9)  if (data.lenght < 10 and flagPO == true) then
        OverUtilizedHosts.add(h)
        Return True
10) Find future Utilization using DES Technique
        future_Utilization = getHostFutureLoad(data)
11) if (future_Utilization > UpperThreshold) then
        flagFO = true
12) if (future_Utilization < LowerThreshold) then
        flagFU = true
13) if (flagFO == false and flagPO == true) then
        currentOverUtilizedHosts.add(h)
14) if (flagFO == true and flagPO == false) then
        predictedOverUtilizedHosts.add(h)
15) if (flagFO == true and flagPO == true) then
        overUtilizedHosts.add(h)
16) if (flagFU == false and flagPU == true) then
        currentUnderUtilizedHosts.add(h)
17) if (flagFU == true and flagPU == false) then
        predictedUnderUtilizedHosts.add(h)
18) if (flagFU == true and flagPU == true) then
        undertilizedHosts.add(h)
```

A MAD empirically below 0.065 and various experiments identified the second type of troublemaker hosts. In this case, if

the upper threshold is smaller than 0.85, it is equal to 0.9 because authors determined overloaded hosts incorrectly and used their capacity improperly. On the other hand, if the lower threshold is greater than 0.35, the lower threshold is equal to 0.3 because they determined underloaded hosts incorrectly and unnecessary migrations increase.

Contrary to [12] that uses two flags, the proposed algorithm uses four flags. Each of the flags flagFO, flagPO, flagFU, and flagPU being true respectively shows the host's being potentially overloaded in near future. The host's being overloaded at present, the host's being potentially underloaded in near future, and the host's being underloaded at present. In this algorithm, if current CPU utilization is greater than the upper threshold, flagPO will be true (Step 7). If current CPU utilization is below the lower threshold, flagPU will be true (Step 8).

As in [12], the length of data array is considered to be 10; that is, to forecast a host's being overloaded or underloaded required at least 10 data from CPU utilization history. If fewer than 10 data are available and flagPO is true, that host will enter the list of overloaded hosts and the algorithm ends; otherwise, the algorithm continues and runs the subsequent steps (Step 9). Authors used DES technique (Step 10) to forecast CPU utilization in near future. They compared the resulting value with dynamic upper and lower thresholds and flagFO and flagFU values are then set (Step 11 and 12). Step [12] considered three categories to categorize overloaded hosts. In the proposed algorithm, three other categories will add to the previous three categories to categorize underloaded hosts. Thus, they categorized overloaded and underloaded hosts in 6 different categories. Further, section below describes each category:

- First category: FlagFO is false and flagPO is true. In this case, the host overloads at present; however, the authors forecasted that it will not overload in near future. In these circumstances, they adds the respective host to the list of currently overloaded hosts (currentOverUtilizedHosts). However, the migration of VMs does not take place from this category of hosts since they forecasted that this host will not overload in future and to decrease unnecessary migration (Step 13).
- Second category: FlagFO is true and flagPO is false. In this case, the host does not overload at present; however, the authors forecasted that it will overload in near future. In these circumstances, they added the respective host to the list of hosts that they forecasted to overload in future (predictedOverUtilizedHosts). However, the migration of VMs does not take place from this category of hosts since they will not overload at present (Step 14).
- Third category: FlagFO and flagPO are both true. In this case, the host overloads at present and the forecast is that it will overload also in near future. In these circumstances, the respective host will add to the list of overloaded hosts (overUtilizedHosts). To reduce the load of this category of hosts, they selected and migrated a number of VMs (Step 15).
- Fourth category: FlagFU is false and flagPU is true. In this case, the host underloads at present; however, the forecast is that it will not remain underloaded in near future. In these circumstances, the respective host will add to the list of currently underloaded hosts (currentUnderUtilizedHosts). However, the migration of VMs does not take place from this category of hosts since the forecast is that this host will not underload in future and in order to decrease unnecessary migration (Step 16).
- Fifth category: FlagFU is true and flagPU is false. In this case, the host does not underload at present; however, the forecast is that it will underload in near future. In these circumstances, the respective host will add to the list of hosts that based on forecast, will underload in future (predictedUnderUtilizedHosts). However, the migration of VMs does not take place from this category of hosts since they does not underload at present (Step 17).
- Sixth category: FlagFU and flagPU are both true. In this case, the host underloads at present and the forecast is that it will underload in near future as well. In these circumstances, the respective host will add to the list of underloaded hosts (underUtilizedHosts). To reduce energy consumption, all of the VMs on these hosts will migrate. Afterward, idle hosts will switch to sleep mode (Step 18).

In the above categorization, migration is necessary merely for the hosts in the third and sixth categories. The hosts in the third category will definitely be overloaded. In order for their load to become normal and their CPU utilization to be below the upper threshold, one VM or more should migrate therefrom. The hosts in the sixth category will definitely be underloaded. All of the VMs thereon should migrate to hosts that meet the necessary conditions as migration destinations. If the migration of all of the VMs on the source host succeeds, that host will switch to sleep mode because it becomes idle. Table (1) shows a summary of the above categorization.

TABLE I. CATEGORIZATION OF OVERLOADED AND UNDERLOADED HOSTS

NO.	flagFO	flagPO	flagFU	flagPU	List Name
1	False	True	-	-	currentOverUtilizedHosts
2	True	False	-	-	predictedOverUtilizedHosts
3	True	True	-	-	overUtilizedHosts
4	-	-	False	True	currentUnderUtilizedHosts
5	-	-	True	False	predictedUnderUtilizedHosts
6	-	-	True	True	underUtilizedHosts

As mentioned in [12] and in light of the mechanism of finding underloaded hosts in [12], it is noteworthy that at load peak time, when the utilization of all hosts is at a high level, the hosts that have lower utilizations compared with other hosts will identify as underloaded hosts and the VMs thereon will migrate to other hosts. This may increase the rate of

unnecessary migration. In the proposed algorithm, as stated earlier, this problem will resolve using the dynamic lower threshold and a proper method to find underloaded hosts.

B. Finding Suitable Destination Hosts

To find appropriate destination hosts for migration, the authors eliminates a number of hosts that do not fit to be a destination host from the list of suitable destination hosts. They consider a total of 6 six different categories for overloaded and underloaded hosts in this paper. In [12], they exclude only the first three categories from the mentioned categorization from the list of suitable destination hosts. In addition to the first three categories, they also excluded the second three categories that contain underloaded hosts or are prone to be underloaded from the list of suitable destination hosts in this paper. In fact, this prevents the hosts that are underloaded or prone to become underloaded from remaining on. A considerable reduction in the energy consumption of the data center may be brought about by turning them off. In contrast with [12] that tries to select the destination host from among underloaded hosts and those with normal loads, efforts are made in this paper to select those hosts as destination hosts that have normal loads. With this policy adopted, they optimized the selection of destination hosts. As a result, they eliminated unnecessary migrations and energy consumption decreases substantially.

IV. Integration of Parts of Proposed Algorithm

In the first part of the proposed algorithm, the authors have classified the overloaded hosts, those hosts prone to be overloaded, also underloaded hosts, and those prone to be underloaded into six different categories.

To make the load normal on the overloaded hosts by utilizing MU policy, the authors have selected certain number of virtual machines to migrate from these hosts. They have also added all the virtual machines available on the underloaded hosts, for migration, to the migration list.

In the second part of the proposed algorithm with the aim of preventing from unnecessary migrations of virtual machine, authors have excluded the six identified categories in the first part from destination host list. Therefore, they can create an optimized list of destination hosts.

Finally, the authors have selected a mapping from among virtual machines for migration (from overloaded and underloaded hosts) and made a suitable list of destination hosts. Then, each virtual machine will migrate to a host that has the minimum increasing power after migration of that virtual machine.

V. Performance Analysis

Authors selected Clouds 3.0.3 tool for the simulation of this paper. Further explanation about this tool may be found in [25-27]. To investigate the performance of the proposed algorithm, they compared this algorithm, MAD-MU algorithm presented in [11], and the algorithm presented in [12] in terms of different metrics. They analyzed and examined the results obtained from the comparison of these algorithms.

MAD-MU algorithm, henceforth called MM in this paper for brevity, is implemented in Cloudsim tool by the authors of [11]. To select a VM for migration, this algorithm makes use of MU policy and takes advantage of MAD technique to calculate dynamic upper threshold. The algorithm presented in [12], henceforth called MMD (MAD-MU-DES) in this paper for brevity, functions similarly to MM upon selecting VM and calculating dynamic upper threshold. It makes use of DES technique for the best results obtained to predict host CPU utilization in future. Since the algorithm proposed in this paper strives to optimize MMD, it is henceforth called OMMD (Optimized MMD) for brevity.

A. Performance Metrics

This paper adopted 6 metrics to compare the proposed algorithm with MM and MMD algorithms, namely energy consumption, the number of VM migrations, PDM, SLATAH, SLAV, and ESV.

PDM metric demonstrates performance degradation due to VM migration. SLATAH metric shows the percentage of time that the host has a CPU utilization of 100%. SLAV metric specifies in what percentage of time the resources allocated to the host are less than the resources demanded by that host and it determines the rate of SLA violation. ESV metric is obtained by multiplying energy consumption by SLAV. It indicates the simultaneous improvement of these two metrics and reveals a trade-off between them. section [11] includes further explanations about each of these metric.

B. Experiment Settings

Since the algorithm presented in this paper attempts to improve the performance of MM and MMD, the authors employed experiment settings similar to these algorithms, which are available in [11, 12]. They simulated a data centre including 800 heterogeneous physical hosts. Half of these hosts are of type HP ProLiant ML110 G4 (Intel Xeon 3040, 2 cores × 1860 MHz, 4 GB) and the other half is of type HP ProLiant ML110 G5 (Intel Xeon 3075, 2 cores × 2660 MHz, 4 GB). This data centre has several types of VMs including High-CPU Medium Instance (2500 MIPS, 0.85 GB), Extra Large Instance (2000 MIPS, 3.75 GB), Small Instance (1000 MIPS, 1.7 GB), and Micro Instance (500 MIPS, 613 MB).

C. Workload Data

Today, research projects that require the work load of real data centres for simulation make use of the data pertaining to a 10-day workload from CoMon project [28], which is a monitoring infrastructure for PlanetLab and collected in March and April 2011. This data comprises CPU utilization data collected at 5-minute intervals from over thousands of operational VMs relating to service providers in more than 500 locations around the world. They will embed as defaults in Clouds simulator. This paper adopts the same data to evaluate the performance of the proposed algorithm and compare it with MM and MMD.

D. Simulation Results

This section will compare the proposed algorithm with MM and MMD algorithms according to the mentioned metrics and using the data of 10 workdays. Fig. 1 to Fig. 6 depict the comparison results.

Fig. 1 shows the comparison of the performances of MM, MMD, and OMMD in terms of the number of migrations metric. On average, OMMD has achieved 89.16% and 83.25% decrease respectively in comparison with MM and MMD.

In OMMD, the number of migrations has considerably decreased using the method presented for finding underloaded hosts and adding 3 new categories for categorizing this sort of hosts and also by eliminating unnecessary migrations from the hosts that are not really underloaded. Another reason why OMMD shows improved results regarding this metric is the identification of troublemaker hosts and adopting policies to modify or eliminate them.

By modifying the aforesaid hosts, the accuracy of identifying overloaded and underloaded hosts increases. This fact causes the number of migrations stemming from the existence of this type of hosts to decrease. On the other hand, in OMMD, in addition to overloaded hosts and those that are prone to be overloaded, the authors excluded underloaded hosts and those that are prone to be underloaded from the list of destination hosts. Consequently, they select the destination hosts with higher accuracy and quality. Thus, this will prevent the repeated migration of VMs as a result of migration to inappropriate destinations.

Fig. 2 shows the comparison of the performance of MM, MMD, and OMMD with respect to the energy consumption metric. On average, OMMD has achieved 35.09% and 21.63% decrease respectively in comparison with MM and MMD. What mainly accounts for the reduced energy consumption is the fact that OMMD obtains underloaded hosts optimally and with greater accuracy in comparison with the other two algorithms. As a result, it prevents energy loss in data centers to a great extent by turning off hosts where their utilization is at a low level.

On the other hand, since the authors selected the destination hosts more accurately in OMMD, this will prevent from the migration of VMs to hosts that are underloaded or prone to be underloaded. Thus they provided more appropriate conditions to switch to sleep mode. In addition to the above, the policies adopted to manage the problems of troublemaker hosts exerted favorable effects on the quality of selecting underloaded and overloaded hosts and, hence, reduced energy consumption.

Fig. 3 exhibits a comparison of the performance of the three mentioned algorithms with regard to PDM metric. On average, OMMD has achieved 90.65% and 87.54% decrease respectively in comparison with MM and MMD. What mainly accounts for this remarkable improvement is the substantial decrease in the number of migrations in OMM

Fig. 4 depicts a comparison of the performance of MM, MMD, and OMMD with respect to SLATAH metric. In comparison with MM and MMD, OMMD has shown a poorer performance in the majority of cases. Efforts made to maximize the utilization of the hosts perhaps have caused this. Since SLAV metric is the multiplication of PDM and SLATAH metrics, in light of the remarkable results of PDM metric, somewhat poor results regarding SLATAH metric is negligible in the proposed algorithm. This may be clearly seen upon investigating and analyzing the figure pertaining to SLAV metric.

Fig. 5 shows a comparison of the performance of the three mentioned algorithms with regard to SLAV metric. On average, OMMD has achieved 89.46% and 84.86% decrease respectively in comparison with MM and MMD. What mainly accounts for this substantial improvement is the improvement of PDM metric.

Fig. 6 shows a comparison of the performance of MM, MMD, and OMMD with respect to ESV metric. On average, OMMD has achieved 93.17% and 88% decrease respectively in comparison with MM and MMD. The reason behind this considerable decrease is the decreases in energy consumption and SLA violation rate. As a matter of fact, these results suggest that there has been a successful trade-off in this paper between these two metrics.

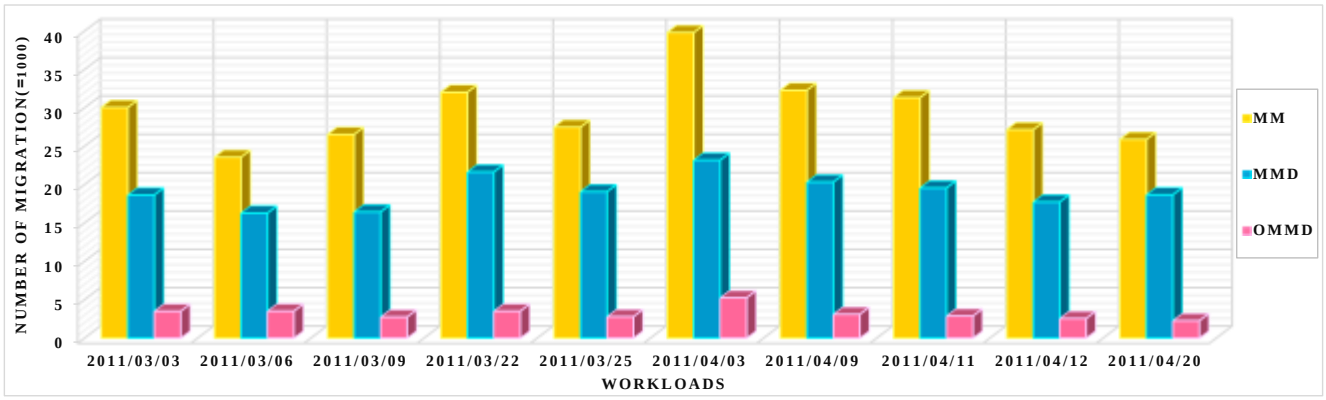

Fig. 1. Comparison of algorithms with regard to number of migration for 10 workdays

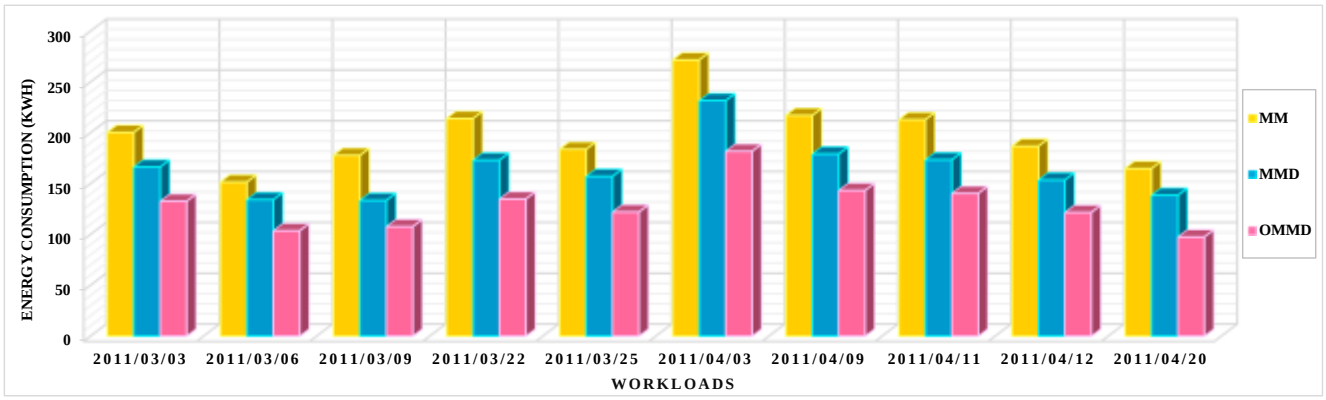

Fig. 2. Comparison of algorithms with regard to energy consumption for 10 workdays

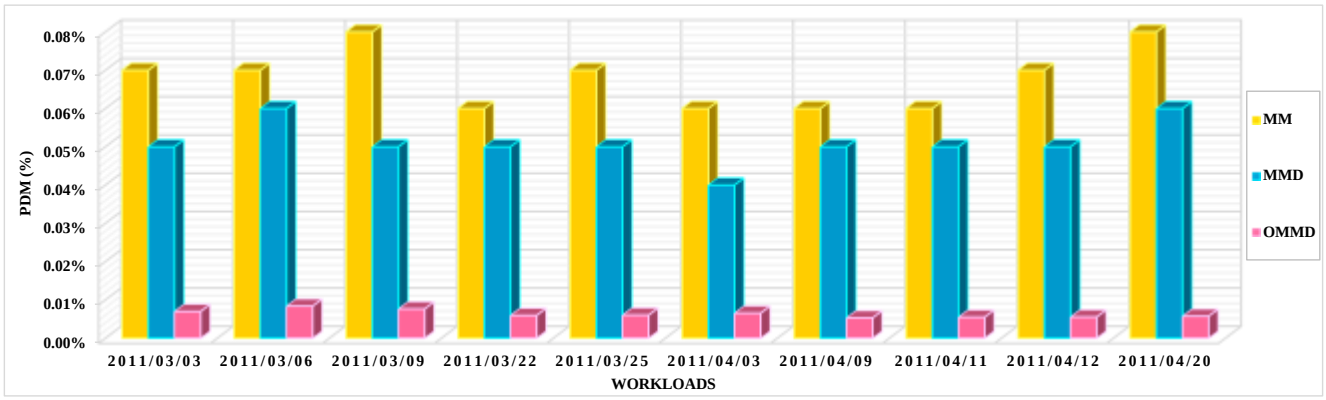

Fig. 3. Comparison of algorithms with regard to PDM for 10 workdays

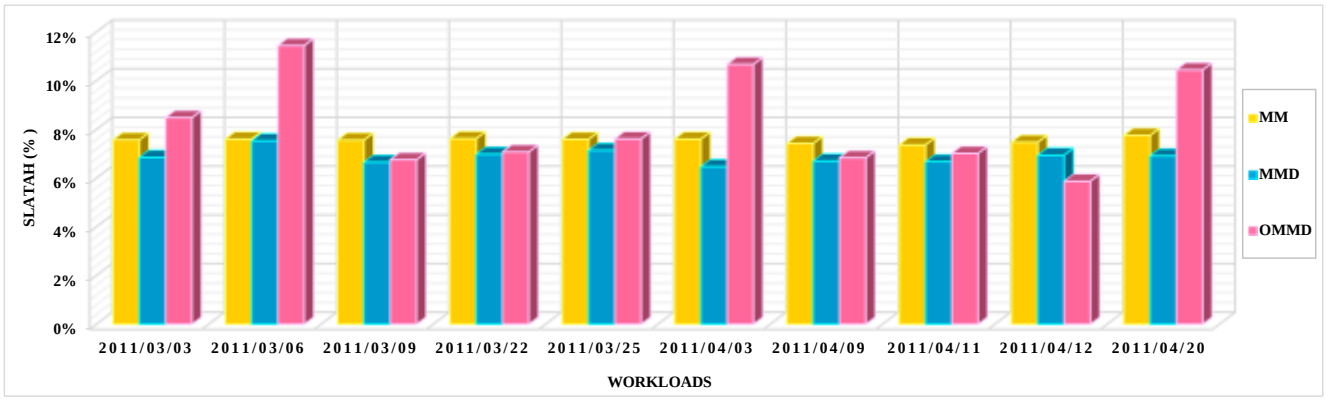

Fig. 4. Comparison of algorithms with regard to SLATAH for 10 workdays

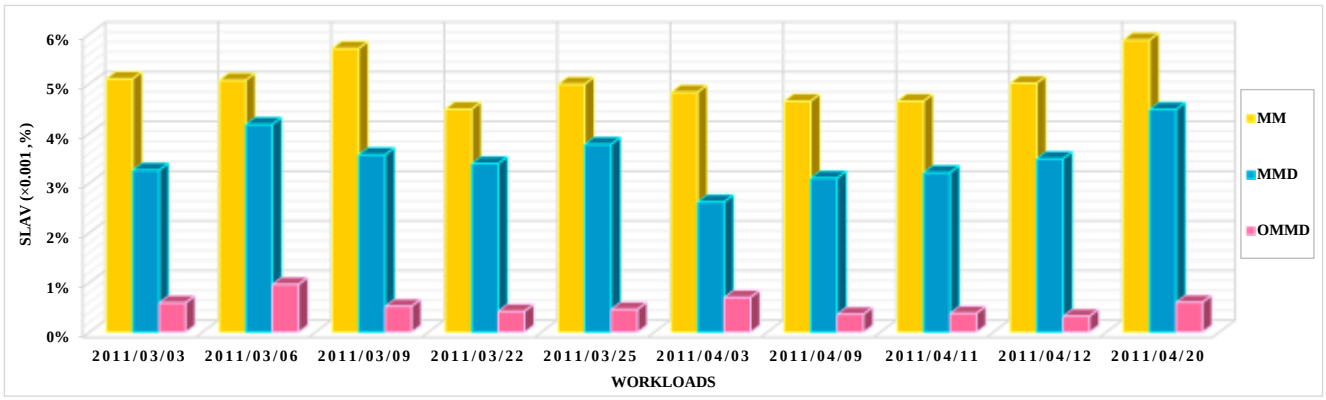

Fig. 5. Comparison of algorithms with regard to SLAV for 10 workdays

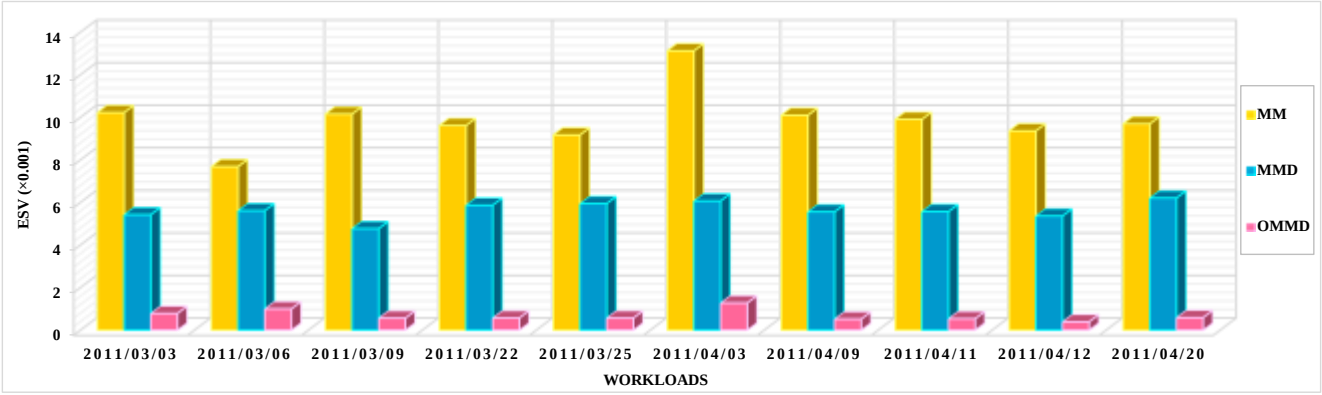

Fig. 6. Comparison of algorithms with regard to ESV for 10 workdays

VI. Conclusion and Future Research Opportunities

The major concern of cloud computing data centres is the decrease in energy consumption and, consequently, reduced operational costs and increased profitability of these centres.

In OMMD, the VM dynamic consolidation problem in cloud computing data centers was brought into spotlight as a solution to battle this problem. In this respect, the authors have provided solutions to make decision about the necessity of migration from hosts and finding suitable destination hosts.

To make decision about the necessity of migration, they have compared current and predicted CPU utilization with dynamic upper and lower thresholds. Thereby, they have identified and categorized underloaded and overloaded hosts.

According to the categorization and the identity of each category, migration took place from the hosts that met necessary conditions for migration. The number of migrations and, as a result, SLA violation rate decreased remarkably using the method proposed for calculating the dynamic lower threshold and finding underloaded hosts and adding 3 new categories to categorize these hosts and also by eliminating unnecessary migrations from hosts that are not really underloaded.

On the other hand, as the accuracy in identifying underloaded hosts increased and by turning them off, they prevented from energy loss in the data center to a considerable extent.

To encounter and prevent the disruptions and adverse effects stemming from the existence of troublemaker hosts, OMMD adopted the policy of modifying them or switching them to sleep mode given the status of those hosts. Thus, the accuracy of identifying overloaded and underloaded hosts increased. This fact had a substantial effect on reduced number of migrations, SLA violation rate, and energy consumption.

OMMD managed to establish a proper trade-off between energy consumption and SLA violation. The results of comparing OMMD with MM and MMD are as follows: respectively 89.16% and 83.25% improvement in the number of migration metric, respectively 35.09% and 21.63% improvement in the energy consumption metric, respectively 90.65% and 87.54% improvement in PDM metric, respectively 89.46% and 84.86% improvement in SLAV metric, and respectively 93.17% and 88% improvement in ESV metric.

Proposed future works:

- OMMD have adopted MU technique for the VM selection problem and no new algorithm was put forward for that. Therefore, the authors recommended that this technique be optimized or a new method be adopted to improve the results even more.
- Given that the improvement of SLA metric can substantially affect the quality improvement of the results of the proposed algorithm, efforts should be made in future studies to alleviate the defect of the SLATAH metric.
- Even though OMMD exhibited remarkable results in the simulation environment, the effect of this algorithm in a real cloud infrastructure is not clearly obvious. Hence, in order to evaluate the performance of the proposed algorithm, it can develop in a real cloud environment such as OpenStack, which is a free open-source software, for future works.
- In addition to physical hosts energy consumption, energy consumption can be Investigate, examine and take into consideration in the communication infrastructures.

References

[1] A. Beloglazov, J. Abawajy, and R. Buyya, “Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing,” Future generation computer systems, vol. 28, no. 5, pp. 755-768, 2012.

[2] R. Jeyarani, N. Nagaveni, and R. V. Ram, “Design and implementation of adaptive power-aware virtual machine provisioner (APA-VMP) using swarm intelligence,” Future Generation Computer Systems, vol. 28, no. 5, pp. 811-821, 2012.

[3] A. Beloglazov, and R. Buyya, "Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers.", MGC '10 Proceedings of the 8th International Workshop on Middleware for Grids, Clouds and e-Science, article no. 4, 2010.

[4] R. Buyya, C. S. Yeo, and S. Venugopal, "Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities." pp. 5-13, 2008.

[5] Y. Gao, H. Guan, Z. Qi, T. Song, F. Huan, and L. Liu, “Service level agreement based energy-efficient resource management in cloud data centers,” Computers & Electrical Engineering, vol. 40, no. 5, pp. 1621-1633, 2014.

[6] E. Arianyan, H. Taheri, and S. Sharifian, “Novel energy and SLA efficient resource management heuristics for consolidation of virtual machines in cloud data centers,” Computers & Electrical Engineering, vol. 47, pp. 222-240, 2015.

[7] M. Poess, and R. O. Nambiar, “Energy cost, the key challenge of today's data centers: a power consumption analysis of TPC-C results,” Proceedings of the VLDB Endowment, vol. 1, no. 2, pp. 1229-1240, 2008.

[8] A. Horri, M. S. Mozafari, and G. Dastghaibyfard, “Novel resource allocation algorithms to performance and energy efficiency in cloud computing,” The Journal of Supercomputing, vol. 69, no. 3, pp. 1445-1461, 2014.

[9] S. Esfandiarpoor, A. Pahlavan, and M. Goudarzi, “Structure-aware online virtual machine consolidation for datacenter energy improvement in cloud computing,” Computers & Electrical Engineering, vol. 42, pp. 74-89, 2015.

[10] A. Beloglazov, and R. Buyya, “Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints,” IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 7, pp. 1366-1379, 2013.

[11] A. Beloglazov, and R. Buyya, “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, vol. 24, no. 13, pp. 1397-1420, 2012.

[12] S. B. Shaw, and A. K. Singh, “Use of proactive and reactive hotspot detection technique to reduce the number of virtual machine migration and energy consumption in cloud data center,” Computers & Electrical Engineering, vol. 47, pp. 241-254, 2015.

[13] X. Fu, and C. Zhou, “Virtual machine selection and placement for dynamic consolidation in Cloud computing environment,” Frontiers of Computer Science, vol. 9, no. 2, pp. 322-330, 2015.

[14] G. Wu, M. Tang, Y.-C. Tian, and W. Li, “Energy-Efficient Virtual Machine Placement in Data Centers by Genetic Algorithm,” vol. 7665, pp. 315-323, 2012.

[15] J. Xu, and J. A. Fortes, "Multi-Objective Virtual Machine Placement in Virtualized Data Center Environments." pp. 179-188.

[16] M. Tang, and S. Pan, “A Hybrid Genetic Algorithm for the Energy-Efficient Virtual Machine Placement Problem in Data Centers,” Neural Processing Letters, vol. 41, no. 2, pp. 211-221, 2014.

[17] C. T. Joseph, K. Chandrasekaran, and R. Cyriac, “A Novel Family Genetic Approach for Virtual Machine Allocation,” Procedia Computer Science, vol. 46, pp. 558-565, 2015.

[18] F. Farahnakian, T. Pahikkala, P. Liljeberg, J. Plosila, and H. Tenhunen, "Utilization Prediction Aware VM Consolidation Approach for Green Cloud Computing." pp. 381-388, 2010.

[19] F. Farahnakian, P. Liljeberg, and J. Plosila, "LiRCUP: Linear Regression Based CPU Usage Prediction Algorithm for Live Migration of Virtual Machines in Data Centers." pp. 357-364, 2013.

[20] Y. Gao, H. Guan, Z. Qi, Y. Hou, and L. Liu, “A multi-objective ant colony system algorithm for virtual machine placement in cloud computing,” Journal of Computer and System Sciences, vol. 79, no. 8, pp. 1230-1242, 2013.

[21] S. B. Shaw, and A. Singh, "A survey on scheduling and load balancing techniques in cloud computing environment." pp. 87-95, 2014.

[22] M. Natrella, “NIST/SEMATECH e-handbook of statistical methods”, 2010.

[23] [C. Chatfield, The analysis of time series. Boca Raton, FL: Chapman & Hall/CRC, 2016.

[24] C. Chatfield, *Time-series forecasting*. Boca Raton: Chapman & Hall/CRC, 2001.

[25] R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities." pp. 1-11, 2009.

[26] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, “CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms,” Softw. Pract. Exper, vol. 41, pp. 23-50, 2011.

[27] R. N. Calheiros, R. Ranjan, C. A. De Rose, and R. Buyya, “CloudSim: A Novel Framework for Modeling and Simulation of Cloud Computing Infrastructures and Services,” arXiv preprint arXiv:0903.2525, 2009.

[28] K. Park, and V. S. Pai, “CoMon: a mostly-scalable monitoring system for PlanetLab,” SIGOPS Oper. Syst. Rev., vol. 40, no. 1, pp. 65-74, 2006.