Optimization of Dynamic Virtual Machine Consolidation in Cloud Computing Data Centers

Alireza Najari[1,*]
Department of Computer Engineering
Ahvaz Branch, Islamic Azad University
Ahvaz, Iran
Department of Computer Engineering
Khouzestan Science and Research Branch Islamic Azad University
Ahvaz, Iran

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

Mohammad Reza Noorimehr[3]
Department of Computer Engineering
Ahvaz Branch, Islamic Azad University
Ahvaz, Iran

***Abstract*—The present study aims at recognizing the problem of dynamic virtual machine (VM) Consolidation using virtualization, live migration of VMs from underloaded and overloaded hosts and switching idle nodes to the sleep mode as a very effective approach for utilizing resources and accessing energy efficient cloud computing data centres. The challenge in the present study is to reduce energy consumption thus guarantee Service Level Agreement (SLA) at its highest level. The proposed algorithm predicts CPU utilization in near future using Time-Series method as well as Simple Exponential Smoothing (SES) technique, and takes appropriate action based on the current and predicted CPU utilization and comparison of their values with the dynamic upper and lower thresholds. The four phases in this algorithm include identification of overloaded hosts, identification of underloaded hosts, selection of VMs for migration and identification of appropriate hosts as the migration destination. The study proposes solutions along with dynamic upper and lower thresholds in regard with the first two phases. By comparing current and predicted CPU utilizations with these thresholds, overloaded and underloaded hosts are accurately identified to let migration happen only from the hosts which are currently as well as in near future overloaded and underloaded. The authors have used Maximum Correlation (MC) VM selection policy in the third phase, and attempted in phase four such that hosts with moderate loads, i.e. not overloaded hosts, liable to overloading and underloaded, are selected as the migration destination. The simulation results from the Clouds framework demonstrate an average reduction of 83.25, 25.23 percent and 61.1 in the number of VM migrations, energy consumption and SLA violations (SLAV), respectively.**

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

I. INTRODUCTION

According to the definition provided by NIST [1] "cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services). It can be rapidly provisioned and released with minimal management effort or service provider interaction". Infrastructure as a Service (IaaS) is among the services provided by cloud computing that offers processing resources to users as services. The clients rent the equipment from infrastructure providers as a service and only pay for the amount of service they really consume [2, 3].

The ever-increasing growth and wide applications of cloud computing, as well as the extensive usage of cloud services in all scopes, have caused a growing trend in energy consumption of cloud computing data centres. Therefore, the operational costs in these centres are intensively increasing due to the electric energy used [4].

According to the reports published by Microsoft [5], the consumed energy used by physical resources can account for 45 percent of the operational costs in a data centre. This amount has multiplied in the last five years [6]. Therefore, to maintain their business and to remain in the market competition, service providers need to minimize energy consumption to cut the excessive operational costs in a way that the integrity and quality of service remain intact [7]. Hence, two challenging tasks in IaaS are management and optimized allocation of resources, to the extent that the success of cloud services heavily relies on this issue.

The present study recognizes the problem of dynamic virtual machine Consolidation using virtualization, live migration of VMs from underloaded and overloaded hosts and switching idle nodes to the sleep mode, as a very effective approach for utilizing resources and accessing energy efficient cloud computing data centres [8-11].

The challenges faced are the consolidation of VMs and their allocation and placement on physical service providers in a way that they minimized energy consumption in the entire data centre and the number of active hosts as well as SLA violation, which is a contract between the clients and the providers. Many studies [11-13] have reported that fully idle

hosts consume as much as 70 percent of the energy used in maximum utilization mode. Therefore, approaches should be taken to minimize the number of underloaded hosts; transfer hosted VMs to other hosts and switch idle hosts to the sleep mode to further decrease energy consumption.

Live VM migration technique transfers a VM from one host to another without any interruptions with the minimum downtime [6, 14-16]. As authors have pointed out in [11, 15], every VM migration process can cause performance degradation, which can roughly be considered 10 percent of CPU utilization. This finding indicates that each VM migration can lead to SLA violation and unnecessary VM migrations may impose extra management costs and consequently extra energy consumption. Therefore, it is necessary to minimize the number of VM migrations and therefore minimize SLA violation and energy consumption as well.

In most research studies conducted in this area [6, 8, 17-21], the necessary decisions are made based on current utilization of hosts and VMs are immediately migrated from hosts which researchers currently identify them as overloaded [8, 15]. The proposed algorithm in the present work attempts to predict CPU utilization using Time-Series Method and SES technique and identifies a VM as overloaded or underloaded based on current and predicted utilization values and their comparison with dynamic upper and lower thresholds. The rest of the paper is structured as follows: it presents Related Work in Section 2 and explains the proposed algorithm in Section 3. Then it provides the results from simulating the proposed algorithm in Section 4, along with their analysis and evaluation. Ultimately, in Section 5, the paper discusses conclusions and suggestions for future works.

II. Related Work

Beloglazov and Buyya [8] proposed Mean Absolute Deviation (MAD) and Interquartile Range (IQR) methods to determine the dynamic upper threshold. In their study, they considered a host as overloaded if the current utilization was greater than the upper threshold. They suggested LR and LRR methods to forecast future loads on the host. This approach recognizes a host as overloaded if its predicted utilization is 100 percent or higher.

To solve the consolidation problem, in addition to VMs energy consumption, authors in [21] also investigated energy consumption in intercommunication networks at data centres. The generated solutions using the genetic algorithm (GA) were significantly better than those of the first-fit decreasing algorithm. However, the computation time in GA was linearly proportional to the number of VMs and hosts.

Gao et al. [18] used Multi-Objective Ant Colony Optimization (MOCO) algorithm for resource allocation with energy efficiency and resource wastage as the two objectives. They used a modification of ant colony algorithm (ACO) in which pheromone updates, definition and accumulation were modified to suit multi-objective problems better. Ultimately, the ACO-based method outperformed GA algorithm.

By continuing the work in [21], authors presented a Hybrid GA (HGA) in [21] for solving the consolidation problem. They used an infeasible solution repairing procedure, in which by gradual resolving of constraint violations it converts an infeasible solution to a feasible one, along with a local optimization procedure which quickly improved the solutions. As compared with GA, HGA yielded more promising results and was able to find local optimums more efficiently in a new search space. However, the workload in HGA increased after implementing the two procedures.

Singh and Shaw [15] employed a load forecast model to determine the necessity of migration and identify appropriate destination hosts. They utilized a dynamic upper threshold and incorporated Time-Series prediction method and Dynamic Exponential Smoothing (DES) and SES techniques. According to their algorithm, a host is considered overloaded if the values of current and predicted CPU utilizations exceed the upper threshold.

In [6], authors presented a novel selection policy called MP, in which they used the dynamic upper and lower thresholds as well as a variable to determine the degree of resource satisfaction. They suggested a new placement policy called MCC, which relocates a migratable VM to a host with minimum correlation to the VM.

Arianyan et al. [17] proposed a holistic resource management procedure and a heuristic intelligent technology method based on multi-criteria decision-making method to determine underloaded hosts for placement of migratable VMs. They presented a multi-criteria method known as Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) by focusing their work on methods for determining the time to consider a host as underloaded and by finding a new location for placement of the VMs selected among underloaded and overloaded hosts.

Joseph et al. [19] introduced a Parallel GA model known as Family GA (FGA) with the aim to generate an optimized mapping between the set of hosts and VMs. This model divides the entire population into a number of families on which it performs genetic operations to overcome the GA limitations. They used a self-adjusting mutation operator to prevent premature convergence of the individuals in the population, which makes the probability of mutation dynamic.

In [22], the authors attempted to solve the consolidation problem by presenting a type of self-adjusting mutation operator as well as considering current and future resource demands and based on the k-Nearest Neighbor (K-NN) regression/prediction model. They proposed the K-NN model in their previous study [23].

III. The Proposed Algorithm

Since the problem of dynamic consolidation of VMs in cloud computing data centres is wide extent, it is broken down into the four following phases [4]:

- Phase 1: Identification of overloaded hosts.
- Phase 2: Identification of underloaded hosts.
- Phase 3: Selection of VMs to migrate from overloaded hosts.

- Phase 4: Determining appropriate destinations for migration.

This study presents algorithms and approaches for each of the phases:

A. Phase 1: Identification of Overloaded Hosts

In this phase, similar to [15], the researchers identified a host as overloaded if it is currently and in near future overloaded; however, unlike [15], they used the Time-Series Prediction method as well as SES techniques to predict CPU utilization in near future. The reader may refer to [24-26] for further information on this topic. Moreover, the study proposed a more optimized, efficient equation for determining dynamic upper threshold. It is noteworthy that the study used DES method for obtaining the best results in [15], therefore, it compared our proposed algorithm with these best results from [15].

```
Algorithm of Phase 1: Identification of Overloaded Hosts
1.  Input: host      Output: migration decision (true/false)
2.  flagP = flagF = false
3.  Find current Utilization of host h
    Utilization = total requested MIPS/h.getTotalMips ()
4.  data [] = h.getUtilizationHistory ()
5.  Calculate MAD and find UpperThreshold using MAD
    UpperThreshold = 1- MAD
    If (Utilization > UpperThreshold) then
        flapP = true
7.  If(data.lenght < 12 and flagP == true) then
        OverUtilizedHosts.add (h)
        Return True
8.  Find future Utilization using SES Technique
    futureUtilization = getHostFutureLoad (data)
9.  If (futureUtilization > UpperThreshold) then
        flagF = true
10. If (flagF == false and flagP == true) then
        currentOverUtilizedHosts.add (h)
11. If (flagF == true and flagP == false) then
        predictedOverUtilizedHosts.add (h)
12. If (flagF == true and flagP == true) then
        overUtilizedHosts.add (h)
        Return true
    Else
        Return false
```

In the first phase of the proposed algorithm, a host is received as input, and depending on the status of the current and predicted CPU utilizations, the researchers categorized it in one of the three lists. Similar to [15], they used the two flags flagF and flagP. True values of flagF and FlagP for a host indicate an overload in near future and at the present, respectively.

The basis of the decision-making in this phase is on the upper threshold. this paper proposed Equation (1) for Calculation of dynamic upper threshold. This Equation is inspired by the method presented in [8] and by employing Median Absolute Deviation (MAD) method [8].

$$\text{UpperThreshold}=1 — \text{MAD} \quad (1)$$

If the current utilization is greater than the upper threshold, the researchers consider the host as overloaded and they set flagP to True (Step 6, Phase 1). [15] and [8] used 10 and 12 data values from the CPU utilization history, respectively, to predict a host overload. Our study considered 12 history values according to our investigations. If this value is lower than 12 and the value of flagP is True, we place the host in the OverUtilizedHosts list, and the algorithm terminates; otherwise, it continues operation (Step 7, Phase 1).

As it was noted earlier, CPU utilization in near future is predicted and calculated using SES method (Step 8, Phase 1). Predicted values higher than the upper threshold mean a host overload in near future will occur; therefore, the researchers set flagF to True (Step 9, Phase 1).

Similar to [15], the study considered three different categories for overloaded hosts. The first category includes currently overloaded hosts (True values for flagP), but are predicted not to remain overloaded in near future (False values for flagF). The researchers added such hosts to the current Over Utilized Hosts list. Since these hosts will not be overload in future and to decrease unnecessary migrations, VMs will not migrate from this category of hosts (Step 10, Phase 1).

The second category includes hosts which are not currently overloaded (False values for flagP), but the study predicts that they will overload in near future (True values for flagF). The researchers will add such hosts to the predicted Over Utilized Hosts list. Since these hosts are not currently overloaded, VMs will not migrate from them (Step 11, Phase 1). Third category includes hosts which are currently and in near future overloaded (True values for both flagP and flagF). The researchers added such hosts to the over Utilized Hosts list, and some of their hosted VMs are selected and migrated to decrease their load (Step 12, Phase 1).

Among the categories mentioned above, the VMs hosted on the third category are certainly considered overloaded, and some of them will migrate from the host to normalize its load.

B. Phase 2: Identification of Underloaded Hosts

In this phase, the following algorithm is presented to identify the underloaded hosts. In the second phase of the proposed algorithm, the researchers received a list of hosts as the input and a list of underloaded hosts is returned. Decision making in this phase is performed based on the lower threshold. If current CPU utilization of the host is below the lower threshold, the host is considered currently underloaded, and similarly, if the predicted CPU utilization of the host in near future is below the lower threshold, the host is known as

underloaded in near future. Inspired by the dynamic upper

```
Algorithm of Phase 2: Identification of Underloaded Hosts
1.  Input: hostList   Output: List of underUtilizedHosts
2.  For each host h in HostList
3.    LowerThreshold = 1
4.    MAD = 0
5.    data[] = h.getUtilizationHistory ()
6.    If (data.length >= 10)
        Calculate MAD and find LowerThreshold using MAD
        LowerThreshold = 0.25 + MAD
      Else
        LowerThreshold = 0.25
7.    Find current Utilization of host h
      Utilization = total requested MIPS/h.getTotalMips ()
8.    Find future Utilization using SES Technique
      futureUtilization = getHostFutureLoad (data)
9.    If (Utilization > 0 and Not (all VMs are migrating
      from host or any VM migrating to host)
10.     If (data.length < 10 and Utilization < LowerThreshold)
          UnderUtilizedHosts.add (h)
          continue
11.     If (futureUtilization < LowerThreshold and Utilization
        < LowerThreshold)
          UnderUtilizedHosts.add (h)
12. End for
```

threshold method in [8] and using the MAD method, the study proposed an optimized equation for determining the dynamic lower threshold. The study requires at least 10 data values from CPU utilization history for predicting an underloaded host.

In this Paper, With 10 or more data values, the dynamic lower threshold is calculated according to (2); otherwise, the lower threshold assumes a constant value of 0.25 (Step 6, Phase 2).

$$\text{LowerThreshold} = 0.25 + \text{MAD} \quad (2)$$

CPU utilization in near future is predicted and calculated using SES method (Step 8, Phase 2). Before identification of an underloaded host, the researchers investigated two conditions. First, CPU utilization of the host should be larger than zero, and second, no VMs should be in the process of migrating from and to the host (Step 9, Phase 2). If they met conditions, then they will do the investigation to find that if the data length of the host utilization history is lower than 10 and if the current CPU utilization of the host is below the lower threshold. If the conditions hold true, given that the sufficient data for prediction of CPU utilization are not available, and the host is currently underloaded, it is added to the list of underloaded hosts and program control flow makes a jump to Step 2 of Phase 2 (Step 10, Phase 2).

If the condition in Step 10 is not met, Step 11 will evaluate current and predicted CPU utilizations of the respective host. If both values are below the lower threshold, the host is considered currently and in near future as underloaded and hence should be added the list of underloaded hosts. The control program flow then jumps to the beginning of the loop to check the conditions for the next host.

C. Phase 3: Selection of VMs to migrate from overloaded hosts

In this phase, unlike [15], in which the proposed minimum utilization (MU) policy of [8] was used, our proposed algorithm employs maximum correlation (MC) policy introduced in [8] due to its superior performance. The main idea behind MC policy was presented by [27]. The basis of this fact is that the more the correlation between the resource consumptions by the running applications on the host, the higher the possibility of overloading. According to this theory, the researchers will select the VMs on a host which have the maximum correlation with other VMs in consumption of processing resources for migration [8].

D. Phase 4: Identification of appropriate destinations for migration

In this phase, to identify the appropriate destinations of migration, the work in [15] is optimized by excluding underloaded hosts from the list of migration destinations. In [15], the researchers exclude only the three categories mentioned above including overloaded and prone to overload hosts from the list of appropriate hosts as destinations of migration, and efforts were made to select underloaded hosts and hosts with moderate loads as the destination of migration. In our study, in addition to excluding the overloaded and/or prone to overload hosts, underloaded hosts were also excluded from the list of appropriate destinations for migration. According to the made decisions, the effort was to select the VMs among those with moderate loads. This way, selection of destination hosts was optimized, the number VM migrations dropped significantly and they prevented from the underloaded machines to remain switched on, which could be turned off to significantly decrease energy consumption in the data centre.

IV. Simulation Results and Assessment of the Proposed Algorithm

This section provides a simulation of the algorithm and its assessment. Then it compares proposed algorithm with MAD-MU algorithm in [8] the proposed algorithm in [15] and the results were analyzed and examined. Clouds framework [28] was used to simulate the proposed algorithm.

A. Experiment Settings

In the present study, a data centre with 800 heterogeneous physical hosts was simulated using Clouds framework. Half of the hosts are HP ProLiant ML110 G4 (Intel Xeon 3040, two cores × 1860 MHz, 4 GB) and the other half are HP ProLiant ML110 G5 (Intel Xeon 3075, two cores × 2660 MHz, 4 GB). The data centre includes 4 types of single-core virtual machines: 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, 0.633 GB. The proposed algorithm aims to improve the proposed algorithms in [8, 15]. Therefore, to be able to carry out a performance comparison, settings similar to those of [8] and [15] were applied to our proposed algorithm.

B. Workload Data

Since most of the reviewed studies used the workload data from the CoMon project, which is a monitoring infrastructure associated with PlanetLab, we also used the same data in our study for assessing the proposed algorithm and for its omparison with its counterpart algorithms. For more realistic results, a CPU utilization dataset was used, the data of which were measured in 5-minute time intervals and were collected from more than thousands of operational VMs in over 500 locations around the world. To carry out a reasonable and appropriate comparison, the researchers used a workload data collected from 10 days in March and April 2011. These data are available in the Clouds framework at the moment.

C. Performance Metrics

Six parameters were used to assess and compare the proposed algorithm with those of other studies. These metrics included the number of VM migrations from overloaded and underloaded hosts, the total energy consumption of physical resources, performance degradation due to VM Migration (PDM) [8], SLA Violation Time per Active Host (SLATAH) [8], which can be defined as the percentage of the period when the host experiences a CPU utilization of 100%, the researchers calculated the combined metric SLAV by multiplying PDM with SLATAH [8] and indicated the duration in which the allocated resources to the host is lower than the required amount, ESV combined metric which is calculated by multiplication of total energy consumption with SLA violation [8] and is used to measure the simultaneous improvements in both metrics and indicates the trade-off between them.

D. Simulation Results

Figs. 1 to 6 demonstrate simulation results of the compared algorithms in for different metrics, and a detailed discussion is presented for each metric as follows. From now on, the study will refer to the proposed MAD-MU algorithm in [8] and the proposed Shaw and Singh Algorithm in [15] as MM and SSA for brevity respectively. Since our proposed algorithm is a modification to optimize SSA, we will call it Optimized SSA, and refer to it as OSSA for brevity.

Authors of [8] implemented MM which is currently in Clouds framework and it employs MAD technique to determine the upper threshold, and MU method to select the VMs for migration. SSA, which depends on MM to select VMs for migration as well as determining the upper threshold, used DES for its best CPU utilization prediction in future. Our proposed algorithm attempts to optimize SSA using the presented methods in Section III.

A comparison between MM, SSA and OSSA is demonstrated in Fig. 1 regarding the number of VM migrations. OSSA achieved 86.83, and 79.65 percent decreases as compared with MM and SSA, respectively. Increased accuracy in calculation of the upper threshold and consequently increased accuracy in identification of overloaded hosts is among the reasons for the significant reduction in the number of migrations in OSSA algorithm as compared with the other two. Therefore, migrations only take place on the VMs which are more accurately identified as overloaded. Another reason for the reductions are the presentation of a new algorithm for identification of underloaded hosts. Through this, underloaded hosts are more accurately identified and the entire hosted VMs are more accurately migrated. As the third reason, by optimizing the procedure of finding appropriate destinations of migration, unnecessary VM migrations to inappropriate hosts are eliminated to a great extent.

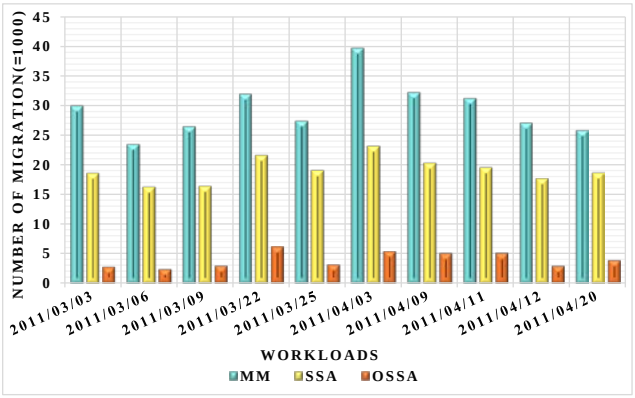

Fig. 1. Comparison of Number of VM Migrations metric against workload

Fig. 2 demonstrates a comparison between MM, SSA and OSSA from the energy consumption perspective. OSSA achieved 32.25 and, 18.2 percent decreases as compared with MM and SSA, respectively.

The main reason for these significant reductions is that OSSA uses a lower threshold for optimized selection of hosts with low utilization levels to prevent energy dissipation by switching them off. Another reason for this improvement is the use of an optimized upper threshold by OSSA which leads to more efficient and effective utilization of processing resources on the hosts by VMs. This improvement gives the opportunity to switch more hosts off to further decrease energy consumption.

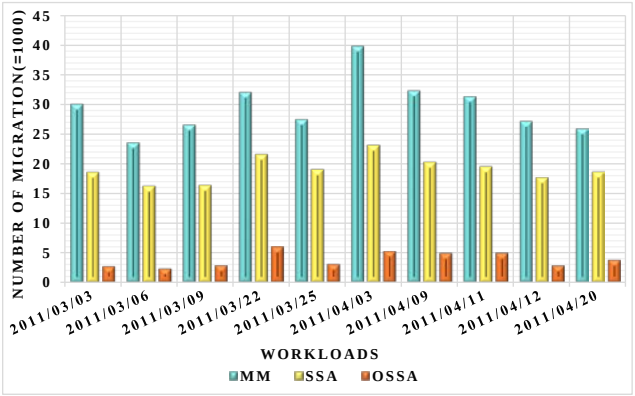

Fig. 2. Comparison of Energy Consumption metric against workload

Fig. 3. Demonstrates a comparison between MM, SSA, and OSSA on PDM metric. OSSA achieved 71.37 and 61.83 percent decreases as compared with MM and SSA,

respectively. The main reason for these significant reductions is the significant decrease in the number of migration in OSSA as compared with the other two algorithms.

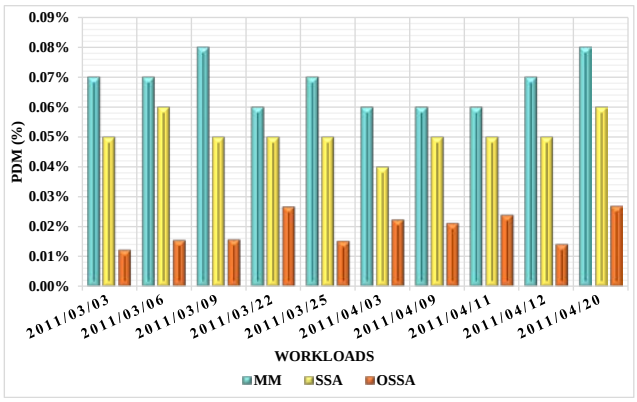

Fig. 3. Comparison of PDM metric against workload

Fig. 4 shows a comparison between MM, SSA and OSSA from the SLATAH metric point of view. As can be seen from this figure, OSSA demonstrated a poor performance in most cases as compared with the other two algorithms. The main reason for the poor performance can be associated with the attempts made by OSSA to achieve maximum host utilizations. However, since SLAV metric is calculated by a multiplication of PDM with SLATAH metrics, poor SLATAH performances may be neglected against the good PDM performances.

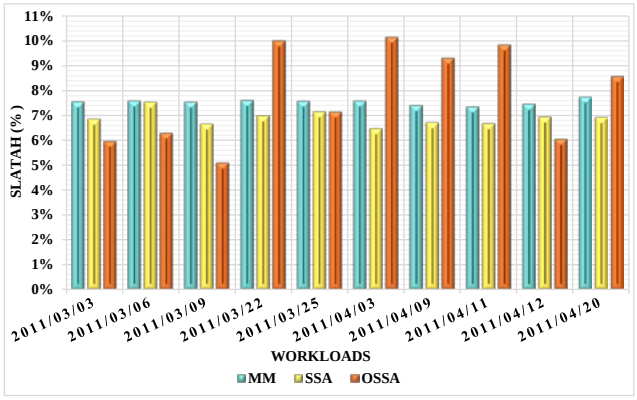

Fig. 4. Comparison of SLATAH metric against workload

A comparison between MM, SSA and OSSA with respect to SLAV metric is given in Fig. 5. OSSA achieved 68.06 and 54.13 percent decreases as compared with MM and SSA. This significant improvement, which is a result of the significant decrease in PDM metric, confirms that the poor performance of SLATAH metric in OSSA could in effect be neglected.

Fig. 6 demonstrates a comparison between MM, SSA and OSSA with respect to the combined ESV metric. OSSA achieved 77.49 and 60.47 percent decreases as compared with MM and SSA. Considering that ESV is calculated from multiplication of the two metrics of energy consumption and SLAV, therefore, the reason for this significant decrease is improvements in both mentioned metrics. There is a good trade-off between the two metrics in OSSA.

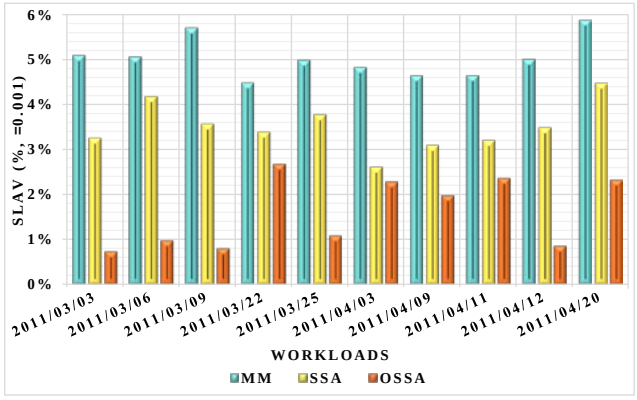

Fig. 5. Comparison of SLA Violation metric against workload

V. Conclusion and Future Works

This study investigated 4 phases of dynamic virtual machine consolidation problem, and for each, presented proper solutions. Also proposed an optimized equation for calculating the dynamic upper threshold and utilized maximum CPU capacity. SLA violation was decreased by eliminating unnecessary migrations, since migrations only took place on actually overloaded hosts. Use of maximum host processing power while maintaining SLA violation in an acceptable level led to increased number of VMs on the hosts, which consequently resulted in better conditions for switching off idle hosts and for decreasing energy consumption.

The study presented an optimized algorithm for identification of underloaded hosts and proposed an equation for calculation of the dynamic lower threshold. Using this threshold, VMs were migrated from underloaded hosts more accurately, allowing them to be switched off. This way, the researchers eliminated unnecessary migrations and decreased SLA violation, and on the other hand, optimized switch offs resulted in decreased energy consumption in the entire data centre.

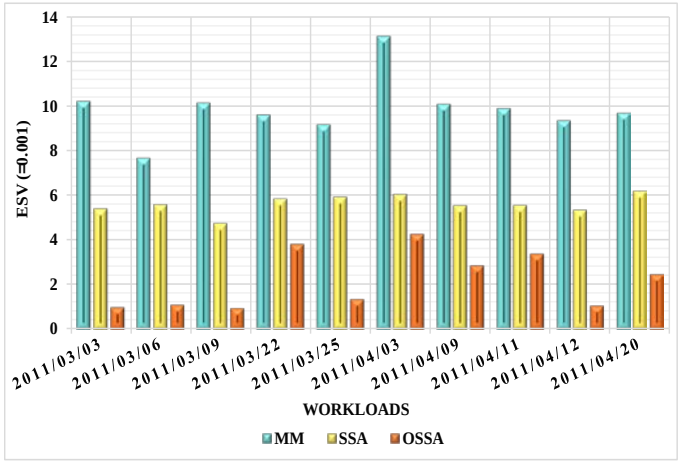

Fig. 6. Comparison of ESV metric against workload

To determine appropriate hosts as the migration destination, all hosts who were considered currently and in

near future overloaded, as well as underloaded hosts, were excluded from the list of migration destinations. The list helped to migrate VMs to destinations of higher quality, and by prevention of unnecessary migrations, SLA violation was decreased. By employing this policy, underloaded hosts were excluded from the list of appropriate destinations of migration, hence preventing VM migrations to this category of hosts. Therefore, opportunities to switch off hosts were protected, leading to further decreases in energy consumption.

OSSA, as compared with MM and SSA, were able to respectively achieve 86.83 and 79.65 percent decreases in the metric of number of migrations. It also can achieve 32.25 and 18.25 percent decreases in energy consumption metric, 71.37 and 61.83 percent decreases in PDM metric, 68.06 and 54.13 percent decreases in SLA violation metric, and 77.49 and 60.47 percent decreases in ESV metric. It also achieved a good trade-off between energy consumption and SLA violation.

It is suggested for future works to further investigate the poor performance of the proposed algorithm in SLATAH metric, since achieving improved SLA metrics leads to increases in the quality of the proposed algorithm. The performance of the proposed algorithm in real infrastructures are yet to known. Therefore, for a real world performance evaluation, use of software packages such as OpenStack are suggested.

REFERENCES

[1] P. Mell, and T. Grance, “The NIST definition of cloud computing,” Communications of the ACM, vol. 53, no. 6, pp. 50, 2010.

[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] Q. Zhang, L. Cheng, and R. Boutaba, “Cloud computing: state-of-the-art and research challenges,” Journal of internet services and applications, vol. 1, no. 1, pp. 7-18, 2010.

[4] X. Zhu, D. Young, B. J. Watson, Z. Wang, J. Rolia, S. Singhal, B. Mckee, C. Hyser, D. Gmach, and R. Gardner, “1000 islands: an integrated approach to resource management for virtualized data centres,” Cluster Computing, vol. 12, no. 1, pp. 45-57, 2009.

[5] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, “The cost of a cloud: research problems in data centre networks,” ACM SIGCOMM computer communication review, vol. 39, no. 1, pp. 68-73, 2008.

[6] 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.

[7] J. Dong, X. Jin, H. Wang, Y. Li, P. Zhang, and S. Cheng, "Energy-saving virtual machine placement in cloud data centres.", Distributed Computing Systems Workshops (ICDCSW), 2013 IEEE 33rd International Conference ,pp. 618-624, 2013.

[8] 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 centres,” Concurrency and Computation: Practice and Experience, vol. 24, no. 13, pp. 1397-1420, 2012.

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

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

[11] A. Beloglazov, and R. Buyya, "Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centres.", International Workshop on Middleware for Grids, Clouds and e-Science , 2010

[12] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer." pp. 13-23, 2007.

[13] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy, and G. Jiang, “Power and performance management of virtualized computing environments via lookahead control,” Cluster computing, vol. 12, no. 1, pp. 1-15, 2009.

[14] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines." pp. 273-286, 2005.

[15] 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 centre,” Computers & Electrical Engineering, vol. 47, pp. 241-254, 2015.

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

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

[18] 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.

[19] 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.

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

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

[22] F. Farahnakian, T. Pahikkala, P. Liljeberg, J. Plosila, and H. Tenhunen, "Utilization Prediction Aware VM Consolidation Approach for Green Cloud Computing." pp. 381-388, 2015.

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

[24] C. Chatfield, Time-series forecasting: CRC Press, 2000.

[25] C. Chatfield, The analysis of time series: an introduction: CRC press, 2016.

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

[27] A. Verma, P. Ahuja, and A. Neogi, "pMapper: Power and Migration Cost Aware Application Placement in Virtualized Systems." pp. 243-264, 2008.

[28] 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.