Research Article

The Reduce energy consumption in cloud computing data centres by optimizing virtual machines

Marziyeh Bahrami[a], AblofazTarghi Haghighat[b], Majid Gholipour[c]

PhD Student in Software Engineering, Faculty of Electrical and Computer Engineering, Qazvin Islamic Azad University, Qazvin
Assistant Professor and Faculty of Electrical and Computer Engineering, Qazvin Islamic Azad University, Qazvin
Assistant Professor and Faculty of Electrical and Computer Engineering, Qazvin Islamic Azad University, Qazvin

Abstract: In this paper, strategies for deciding on the need to migrate and find suitable hosts as the destination of migration are presented. The proposed algorithm, using time series prediction method and double smooth development technique (DES), predicts the processor efficiency in the future and also proposes the optimal relationship for the dynamic low threshold. The algorithm identifies and categorizes under-Loadandover-load hosts by comparing current and predicted CPU productivity with dynamic high and low thresholds, and based on this classification, migration takes place from hosts that qualify for migration. This article identifies a type of host as a troublesome host that most likely disrupts the predictive and decision-making process. In the face of these types of hosts, the algorithm implements policies to modify or put them to sleep. To find suitable hosts as an immigration destination, all over-load, over-load, low-load, and under-Loadhosts are removed from the list of suitable destinations. Improvements of 86.2%, 28.4% and 87.2% on average and in the criteria of the number of migrations of virtual machines, energy consumption and the rate of SLA violations, respectively, are the achievements of this article.

Keywords: Cloud computing, virtual machine, live migration, dynamic integration, service level agreement, energy consumption

1. Introduction

Cloud computing is a model that provides access to infrastructure including a set of configurable computing resources such as servers, storage, applications, services, etc. through existing communication platforms such as the network and the Internet, in an easy, fast, on-demand and compliant manner. Provides quality service to applicants. IaaS, PaaS and SaaS are the three main types of cloud computing services [1, 2]. At IaaS, data **centre** infrastructure, servers, storage space, and hardware policies are typically provided, independent of location and geographic constraints, and under computer networks, and organizations are involved in maintaining and upgrading equipment instead of purchasing IT infrastructure. Are provided using cloud computing based on their needs [3, 4].

In recent years, due to the increasing use of cloud computing services and following the welcome of customers to these services, cloud computing service providers have increased the number and volume of data **centres** greedy to consume more energy [5], this issue is very costly. Heavy operations have followed. The service quality assurance set out in the SLA, which is regulated between customers and providers, is essential for the cloud computing environment; Therefore, cloud service providers tend to strike a balance between energy and efficiency, and in order to reduce operating costs, they must reduce energy consumption to the extent that it does not disrupt or reduce the quality of service [3].

Most of the energy loss occurs in the hardware infrastructure of cloud computing data **centres**. Research has shown that the power consumption of hardware equipment when idle is about the same as their power consumption at peak times, and not fully utilizing them, will lead to a lot of energy loss [6]. Indeed, the Forrester research group has stated that when a server is idle 70% of its time, it uses up to 30% of its peak power [7]; Therefore, in cloud computing data **centres**, the use of equipment while their productivity is low is the main reason for energy loss [6].

Virtualization is a key feature and mainstay of cloud computing that allows multiple virtual machines to be deployed on a single host, as well as the migration of virtual machines [8]. The problem of optimally integrating virtual machines using virtualization technology is an effective approach to achieving energy efficient cloud computing data **centres** [9-11], as it allows virtual machines to be used when host performance is low. The existing ones should be migrated to other suitable hosts and the hosted ones should be put to sleep [5].

Live virtual machine migration is used to transfer a virtual machine without suspension and with a minimum downtime; However, any migration of the virtual machine will be accompanied by some reduction in the degree of efficiency and consequent possible violation of the SLA [12]; On the other hand, unnecessary migration of the virtual machine will lead to additional management costs (such as reconfiguration of the virtual machine, creation and destruction of the virtual machine, etc.) which will lead to additional energy consumption [13]; Therefore, in order to reduce SLA violations and reduce energy consumption, unnecessary virtual machine migrations should be avoided as much as possible and the number of migrations should be minimized. This article focuses on the issue of dynamic integration of virtual machines in cloud computing data **centres** as a solution to address these issues and problems, and in this regard, the following solutions are briefly presented:

• Propose efficient relationships to calculate dynamic low thresholds and provide a way to find low-end hosts and categorize them.

Decide on the need to migrate hosts using a comparison of predicted and current processor productivity with dynamic high and low thresholds, and identify and categorize over-load and **under-Load**hosts.

• Identify a type of host as a troublesome host and implement policies to modify or put them to sleep.

• Provide a way to find suitable destination hosts by removing hosts in all categories.

In the continuation of this paper, the previous works are examined in section 2, in section 3, the proposed algorithm is described. Section 4 demonstrates the feasibility of the proposed algorithm using the Cloudsim tool, and finally Section 5 examines the conclusions and future work.

2.Significance of The Study

Nowadays, the extension of computational powers using external resources has become common. In order to improve performance of applications running inside a computer, an architecture that allows computers to utilize computational resources to the maximum extent and minimizes the resource consumption is necessary. Thus, an architecture based on virtual machine migration is expected to improve performance of virtual machines by adapting the virtual machine onto a computing resource that is the most comfortable for the computational work running on the virtual machine. In order to adapt virtual machines onto available computing resources, a methodology for selecting a proper hypervisor with necessary computing resources. [1]

becomes necessary. As was mentioned previously, several resources that affect computing performance are CPU, memory, and device I/O. There are various I/O devices in a computer, but the two prominent examples are disk I/O and network I/O. By observing the utilization of these resources and migrating virtual machines to a hypervisor that satisfies resource requirements, the performance of virtual machines is expected to improve. The architecture utilizes live migration of virtual machines for dynamically changing the hosting hypervisor to an appropriate one. Hypervisors are computers that host virtual machines, and since they are also computers running hypervisor OSes, each hypervisor would have different computing resources. Moreover, since a single hypervisor can host multiple virtual machines, a hypervisor with the greatest computing resources may not always be the most effective hypervisor to run the virtual machine on. Thus, in order to introduce the architecture proposed in the thesis, a method for evaluating available computing resources is necessary. In addition to evaluating computing resources, the cost of performing live migrations must also be considered. Performing live migration of virtual machines will transfer memory contents of the virtual machine to a remote hypervisor, consuming resources and network bandwidth. The migration procedure typically takes a few seconds up to a few minutes on modern PC-class hardware, depending on available network bandwidth. Even if other hypervisors have better computing resources, if the cost of migration is too high, sometimes it may be better not to migrate the virtual machine. Therefore, a strategy is necessary for maximizing the use of computing resources available to virtual machines.[1]

3.Review of Related Studies

In [14], the authors used the standard genetic algorithm (GA) to solve the integration problem, and examined the energy consumed by physical machines and interconnection networks in data centres and found that their methods were better than FFD [15]. But in FFD, the computation time is shorter than their method.

The authors in [11] presented the policies of MU, MMT, MC and RC for the issue of choosing a virtual machine. They proposed MAD and IQR methods to find the dynamic high threshold. In their work, a host is considered to be a productive host if its current processor efficiency is greater than the dynamic high threshold.

The authors in [16] proposed a hybrid genetic algorithm (HGA) to solve the integration problem, in which to rapidly improve solutions, from a local optimization procedure and to gradually eliminate the violations of the conditions in the impossible solutions and turn an impossible solution into a Possible solution, they used a correction procedure of impossible solutions. They found that HGA converged faster than GA, and showed significantly better results in terms of efficiency and productivity.

The authors in [17], in order to create an optimal mapping between a set of hosts and virtual machines, divided the whole community into a number of families and performed genetic operations on these families in parallel, thus the family genetic algorithm, which is a model of parallel genetic algorithm. ; Presented. They used a self-regulating mutation operator to prevent untimely convergence of individuals in the community.

In [18], the authors used the k-nearest neighbour regression prediction method proposed in [19] to predict resource productivity. They solved the integration problem by using current resource productivity and predicting future productivity.

The authors in [6] proposed policies for determining low-cost hosts as well as policies for the placement of migratory virtual machines in which they used a multi-criteria decision-making method. They also introduced a new and comprehensive resource management routine for cloud resources called EO, which provides a comprehensive overview of resource management routines.

Using time series prediction and moving average techniques, simple smooth and double smooth development, and using high dynamic thresholds, the authors proposed an algorithm for deciding whether to migrate and find suitable destination hosts. In their algorithm, it is recognized as a high-performance host whose current and predicted CPU efficiency is greater than the upper threshold.

In [20], the authors changed the ant colony algorithm to the main algorithm in the sections of pheromone upgrade, pheromone definition, and pheromone aggregation, so that it is suitable for multi-purpose use and for allocating resources to virtual machines with the goal of reducing consumption. They used energy and reduced resource waste. The results show that this algorithm performs better than GA in both respects.

The authors in [13] introduced two new policies, MP and MCC. The first policy operates using the degree of satisfaction with resources and CPU efficiency, and selects virtual machines for migration using dynamic upper and lower thresholds. The second policy operates using the correlation coefficient and finds the appropriate destination host.

4.Proposed algorithm

According to [21], the problem of dynamic integration of virtual machines in cloud computing data centers is divided into the following three parts:

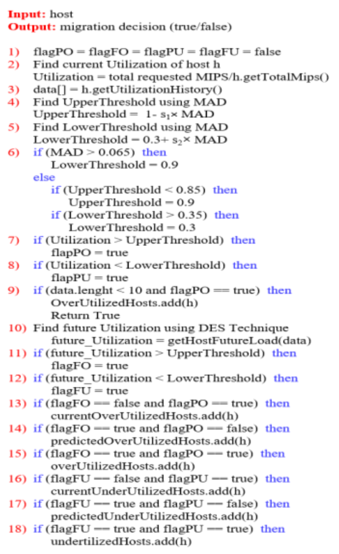
• Decide on the need to migrate from the hosts
• Choose a virtual machine for migration
• Find the right destination hosts

The first and third parts of the integration problem are answered using the proposed algorithm. To address the virtual machine selection problem, as in [12], the minimum productivity (MU) policy presented in [11] is used. This policy selects the virtual machine with the least CPU efficiency from the virtual machines on a host. The following is a description of how the proposed algorithm responds to the first and third sections of the virtual machine integration problem.

4-1- Deciding on the need to migrate from the hosts

In the proposed algorithm, in order to decide on the need to migrate from the hosts, the current processor efficiency is not the sole criterion; Similarly [12], the history of CPU productivity has been used in recent times and by using the time series method and DES technique, CPU productivity is predicted in the near future. Further explanation of the prediction method is given in [22-24]. Unlike [12], which uses only the dynamic upper threshold limit, this algorithm also uses the dynamic low threshold limit. By comparing the predicted and current processor productivity with the dynamic upper and lower thresholds, the status of each host is determined and the hosts are categorized, using this classification to decide whether migration is needed. In the following, the proposed algorithm is given.

Fig.1. Proposed algorithm: Decide on the need to migrate from under-load hosts to over-load hosts



```
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)
```

This algorithm receives a host as input and decides to migrate the host by examining its status. As in [12], the upper threshold is calculated by the method presented in [11] (Step 4), which according to [11], the value of the parameter S_1 is considered equal to 2.5. To calculate the dynamic low threshold, inspired by the high threshold method presented in [11] and using the absolute mean deviation (MAD) method, Equation (1) has been proposed and used (Step 5). It should be noted that the MAD method is also suggested in [11]. In relation (1), the value of S_2 has been considered experimentally and by performing several experiments, equal to the value of 2.5.

LowerThreshold = 0.3 + s_2 × MAD(1)

Numerous experiments have identified a type of host that is prone to adverse conditions. In this article, such hosts are called troublesome hosts. Step 6 of the proposed algorithm, to prevent the occurrence of adverse conditions and improve the results, according to the status of the hosts, by applying policies, identifies the troublesome hosts, modifies or dissolves them.

The first type of troublesome hosts are hosts whose MAD is greater than 0.065 (this number has been obtained experimentally). Due to the fact that MAD indicates the power of the deviation of the host processor, during the experiments, it was found that when the value of MAD exceeds 0.065, the deviations of the processor increase and the accuracy of the algorithm in predicting the efficiency of the processor decreases. On the other hand, with increasing MAD, the accuracy of calculating the dynamic upper and lower thresholds decreases; In such a way that a host with a normal load, high or low load may be considered and lead to unnecessary migrations; Or a high or low load host is considered to be a normal load host, causing the SLA to breach. It should be noted that the larger the MAD, the smaller the upper threshold, and thus the CPU efficiency is not used optimally. 11]. In the proposed algorithm to prevent possible problems in this type of troublesome host, the low threshold is set to 0.9. With this technique, these types of hosts are most likely to be overloaded, and all the virtual machines on them are transferred to the appropriate hosts, then the idle hosts are put to sleep.

During various experiments and experimentally, the second type of troublesome hosts with MAD less than 0.065 were identified. In this case, if the upper threshold is less than 0.85, the high threshold is considered to be 0.9 due to incorrect detection of high-performance hosts and improper use of their capacity. On the other hand, if the lower threshold is greater than 0.35, the lower threshold is considered to be 0.3 due to misdiagnosis of sparse hosts and increased unnecessary migration.

Unlike [12] which uses two flags, in the proposed algorithm, four flags are used. The fact that each of the flags of flagFO, flagPO, flagFU and flagPU is true indicates the probability that the host will be overloaded in the near future, that the host will be overloaded at present, that the host will probably be overloaded in the near future, and that the host will be underloaded at present. In this algorithm, flagPO becomes True (CPU 7) if the current processor output is higher than the threshold (Step 7). If the current CPU output is below the low threshold, flagPU becomes True (Step 8).

As in [12], the length of the data array is assumed to be 10, meaning that at least 10 pieces of CPU productivity history data are required to predict whether a host is high or low; If this value is less than 10 and flagPO is equal to True, that host will be included in the OverUtilizedHosts list and the algorithm will terminate; Otherwise, the algorithm continues and takes its next steps (Step 9).

Using the DES method, CPU efficiency is predicted in the near future (step 10) and by comparing the value with the dynamic upper and lower thresholds, the flagFO and flagFU values are adjusted (steps 11 and 12). In [12], in order to classify prolific hosts, 3 categories are considered; In the proposed algorithm, in addition to the previous 3 categories, three other categories have been added to categorize low-frequency hosts, and thus, high-frequency and low-frequency hosts are classified into 6 different categories. The following is a description of each category:

The first category: flagFO is equal to False and flagPO is equal to True; In this case, the host is already productive, but it is predicted that it will not be productive in the near future. In this case, the desired host is added to the list of currently overloaded hosts (currentOverUtilizedHosts), but because it is predicted that this host will not be fruitful in the future and in order to reduce unnecessary migration, virtual machines will not migrate from this host (step 13).

The second category: flagFO is equal to True and flagPO is equal to False; In this case, the host is not fertile at the moment but it is predicted that it will be fertile in the near future. In this case, the target host is added to the list of predicted over-utilized hosts (predictedOverUtilizedHosts), but virtual machines do not migrate from these hosts because they are not currently loaded (Step 14).

Third category: flagFO and flagPO are both equal to True; In this case, the host is already productive and is expected to be productive in the near future. In this case, the host is added to the list of overUtilized hosts (overUtilizedHosts) and in order to reduce the load of these hosts, a number of virtual machines on them are selected and migrated (Step 15).

Fourth Category: flagFU is False and flagPU is True; In this case, the host is currently low, but it is predicted that it will not be low in the near future. In this case, the desired host is added to the list of current hosts (currentUnderUtilizedHosts), but because it is predicted that this machine will not be used in the future and in order to reduce unnecessary migration, virtual machines will not migrate from this host (step 16).

Fifth category: flagFU is equal to True and flagPU is equal to False; In this case, the host is not currently low, but it is predicted that it will be low in the near future. In this case, the desired host is added to the list of predicted underused hosts in the future (predictedUnderUtilizedHosts), but virtual machines do not migrate from these hosts because they are not currently scarce (Step 17).

Sixth category: flagFU and flagPU are both True; In this case, the host is currently low and it is predicted that it will remain low in the near future. The host conditions are then added to the list of underUtilized hosts, and to reduce power consumption, all the virtual machines on this host are migrated from them, then the idle hosts are put to sleep (step 18).

In the categories performed, migration is required only for hosts in the third and sixth categories. Hosts in the third category are considered to be highly productive, and in order to normalize their load and place their CPU efficiency below the upper limit, one or more virtual machines are migrated from them. Hosts in the sixth category are definitely considered to be sparse, and all virtual machines on them are migrated to hosts that qualify as migrating destinations, and if all virtual machines located on the source host succeed, The host is put to sleep due to unemployment.

It should be noted that, as noted in [12], due to the mechanism for finding sparse hosts in [12], it is possible that at peak times, when the productivity of all hosts is at a high level, the hosts that Hosts, on the other hand, are less productive, have been identified as lesser hosts, and virtual machines have been migrated to other hosts; And this increases the number of unnecessary migrations. In the proposed algorithm, as mentioned earlier, this problem is solved by using a dynamic low threshold and a suitable method for finding under-Loadhosts.

4-2- Finding suitable destination hosts

To find suitable destination hosts for migration, a number of hosts that are not suitable for the migration destination will be removed from the list of suitable hosts as the migration destination. In this article, 6 different categories are considered for high and low load hosts. In [12], only the first three categories of the mentioned category are removed from the list of suitable destination hosts. In this article, in addition to the first three categories, the second three categories that contain sparse or prone to sparse hosts are also removed from the list of suitable destination hosts. They can have a significant impact on reducing data centre power consumption and prevent . Unlike [12], which tries to choose the destination of migration from low-load hosts and hosts with normal load, in this article, we try to select only the hosts as migration destination that have normal load. With the adopted policy, the choice of destination hosts has been optimized and as a result, unnecessary migrations have been eliminated and the amount of energy consumption has been significantly reduced.

5. Performance analysis

In this paper, Cloudsim 3.0.3 tool has been selected as the simulation platform. Further explanation of this tool is given in [25-27]. In order to evaluate the performance of the proposed algorithm, comparisons between this algorithm and the MAD-MU algorithm presented in [11] and the proposed algorithm in [12] have been performed in terms of different criteria and the results of the comparison of these algorithms have been analyzed.
The MAD-MU algorithm, hereinafter referred to as MM in this article, has been implemented in the Cloudsim tool by the authors [11]. This algorithm uses the MU policy to select the virtual machine for migration and the MAD method to calculate the dynamic high threshold. The algorithm presented in [12], hereinafter referred to as MMD (MAD-MU-DES), is similar to MM for selecting a virtual machine and calculating the dynamic high threshold, and in the best results, Uses the DES method to predict future CPU productivity. Since the algorithm proposed in this article tries to optimize MMD; Henceforth, it is abbreviated as OMMD (Optimized MMD).

5-1- Performance criteria

In this paper, three criteria of number of virtual machine migrations, energy consumption and SLA violation rate have been used to compare the proposed algorithm with MM and MMD algorithms. The SLA Violation Criterion determines what percentage of the time the resources allocated to the host were less than the resources requested by that host. Further explanation of each of these criteria is given in [11].

5-2- Test settings

Whereas the algorithm presented in this paper seeks to improve the performance of MM and MMD; It therefore uses the experimental settings similar to those of the algorithms found in [11, 12]. A data center consists of 800 physically heterogeneous hosts, half of which are HP ProLiant ML110 G4 (Intel Xeon 3040, 2 cores × 1860 MHz, 4 GB) and the other half are HP ProLiant ML110 G5 (Intel Xeon 3075, 2 cores). 2660 MHz, 4 GB), simulated. This data center has several types of virtual machines as follows:

- High-CPU Medium Instance (2500 MIPS, 0.85 GB)
- Extra Large Instance (2000 MIPS, 3.75 GB)
- Small Instance (1000 MIPS, 1.7 GB)
- Micro Instance (500 MIPS, 613 MB)

5-3- Workload data

Today, research projects that require real-time data center workloads to simulate use 10-day workload data from the CoMon project [28], a monitoring infrastructure for PlanetLab, collected in March and April 2011. This data includes CPU productivity data collected at intervals of 5 minutes from thousands of operating virtual machines from servers in more than 500 locations around the world and placed by default in the Cloudsim emulator. In this paper, these data have been used to evaluate the performance of the proposed algorithm and compare it with MM and MMD.

5-3- Simulation results

In this section, evaluation and comparison between the proposed algorithm and MM and MMD algorithms, based on the criteria mentioned earlier and using 10 working days data, has been done. Figures (1) to (3) illustrate the comparison results. Figure (1) shows a comparison of the performance of MM, MMD and OMMD in the migration number criterion. OMMD compared to MM and MMD, on average, decreased by 89.16 and 83.25 percent, respectively.

In OMMD, the number of migrations has been significantly reduced by using the proposed method to find sparse hosts and adding 3 new categories to categorize such hosts, as well as removing unnecessary migrations from non-sparse hosts. Another reason for the improvement of OMMD in this criterion is the identification of troublesome hosts and the implementation of policies to modify or dissolve them. On the other hand, in OMMD, in addition to high-performance and low-load hosts, low-load and low-load hosts are also removed from the list of destination hosts, resulting in more careful and quality selection of destination hosts. Is.

A comparison of the performance of MM, MMD and OMMD in the measure of energy consumption is shown in Figure (2). OMMD compared to MM and MMD, on average, decreased by 35.09% and 21.63%, respectively. The main reason for the reduction in energy consumption is that OMMD, compared to the other two algorithms, achieves low-efficiency machines more accurately and optimally, and as a result, by turning off the hosts in a state where their productivity is at a low level, energy loss in Data center is largely preventable. On the other hand, because OMMDs select destination hosts more carefully, they prevent the migration of virtual machines to hosts that are sparse and prone to overload, thus providing better conditions for them to fall asleep. In addition to the above, the policies adopted to manage the problems of troublesome hosts have had a positive effect on the quality of the selection of low-load and over-load hosts and thus reduce energy consumption.

Figure (3) shows a comparison of the performance of the three algorithms in the SLA breach measure. OMMD compared to MM and MMD, on average, achieved a decrease of 89.46 and 84.86, respectively. The reason for this is the significant reduction in the number of migrations in the proposed algorithm compared to the existing algorithms.

Figure2: Comparison of algorithms based on the number of migrations by workload

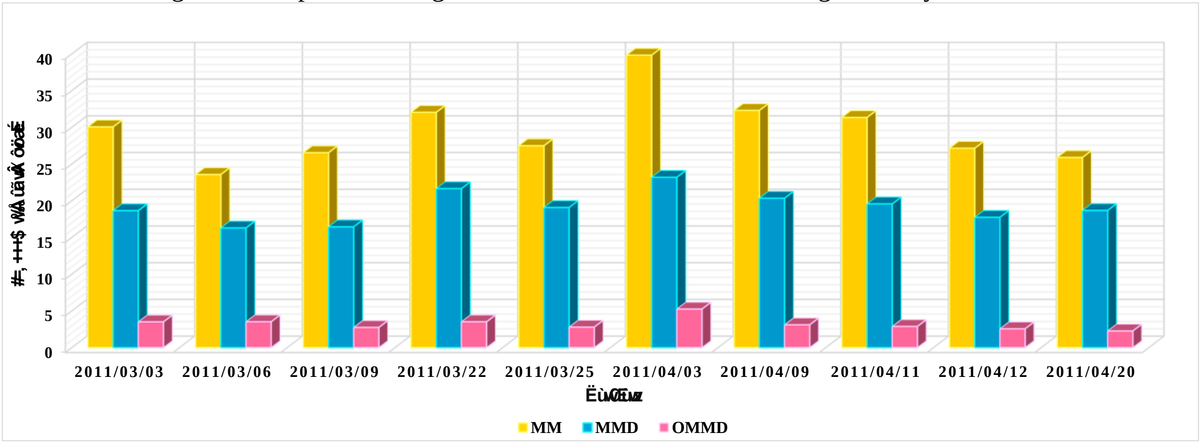


Figure 3: Comparison of algorithms based on energy consumption criteria by workload

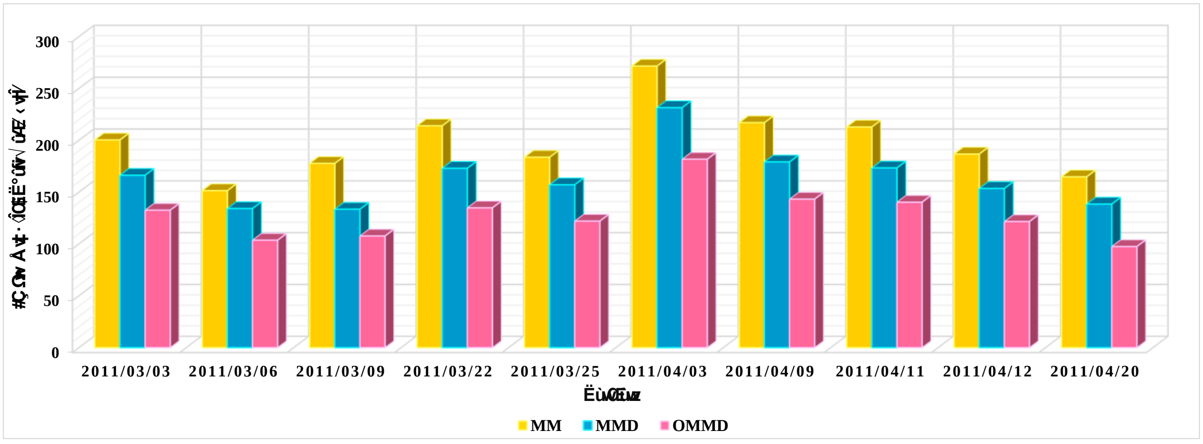
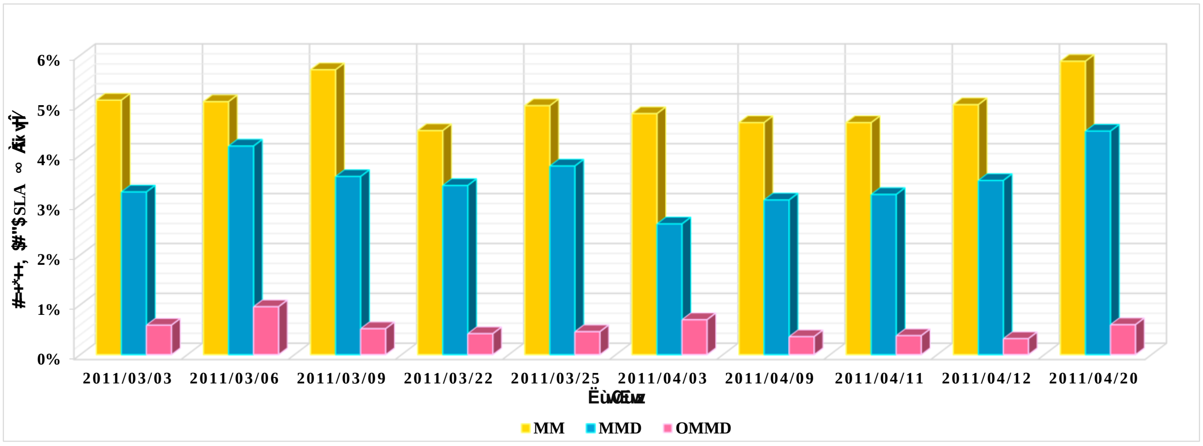


Figure 4: Comparison of algorithms based on SLA breach criteria by workload



8.Conclusionand future work

The main concern of cloud computing data centres is to reduce energy consumption and consequently reduce operating costs and increase the profitability of such centres. OMMD focused on the issue of dynamic integration of virtual machines in cloud computing data centres as a solution to this problem, and in this regard, solutions were proposed to decide on the need to migrate from hosts and find suitable hosts as migration destinations.

To decide on the need for migration, the predicted and current processor productivity was compared with the dynamic upper and lower thresholds, thus identifying and categorizing low-load and over-load hosts. Based on the classification and the nature of each category, the migration took place from hosts that meet the requirements for migration. Using the proposed method to calculate the dynamic low threshold and find sparse hosts and add 3 new categories to categorize such hosts, as well as removing unnecessary migrations from non-sparse hosts, the number of migrations and consequently the extent of SLA breaches to a considerable extent. Reduced. On the other hand, by increasing the accuracy of detecting sparse hosts and turning them off, energy loss in the data centre was largely avoided.

The OMMD has adopted a policy of modifying or dormant hosts, depending on the condition of those hosts, to address and prevent the disturbances and adverse effects of troublesome hosts, thus increasing the accuracy of detecting high and low loads. This had a significant impact on reducing the number of migrants, the rate of SLA violations and energy consumption. In OMMD, a good balance was struck between the criteria of energy consumption and SLA violation. From the results of comparing OMMD with MM and MMD, respectively, an improvement of 89.16 and 83.25% in terms of the number of migrations, an improvement of 35.09 and 21.63% in terms of energy consumption and an improvement of 89.46 and 84.86% in terms of The amount of violation is SLA.

In OMMD, the MU method was used to select virtual machines, and no new algorithm was proposed; Therefore, it is suggested that in order to improve the results as much as possible, this method should be optimized or a new method should be adopted.

Although OMMD showed significant results in the simulated environment; But the impact of this algorithm on the real cloud infrastructure is not exactly clear; Therefore, for future work, in order to evaluate the performance of the proposed algorithm, it can be done in a real cloud environment such as OpenStack, which is a free open source software; Developed.

References

[1] Beloglazov A, Abawajy J, Buyya R. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future generation computer systems. 2012;28(5):755-68.

[2] Jeyarani R, Nagaveni N, Ram RV. Design and implementation of adaptive power-aware virtual machine provisioner (APA-VMP) using swarm intelligence. Future Generation Computer Systems. 2012;28(5):811-21.

[3] Beloglazov A, Buyya R, editors. Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers. Proceedings of the 8th International Workshop on Middleware for Grids, Clouds and e-Science; 2010: ACM.

[4] Buyya R, Yeo CS, Venugopal S, editors. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. High Performance Computing and Communications, 2008 HPCC'08 10th IEEE International Conference on; 2008: Ieee.

[5] Gao Y, Guan H, Qi Z, Song T, Huan F, Liu L. Service level agreement based energy-efficient resource management in cloud data centers. Computers & Electrical Engineering. 2014;40(5):1621-33.

[6] Arianyan E, Taheri H, Sharifian S. Novel energy and SLA efficient resource management heuristics for consolidation of virtual machines in cloud data centers. Computers & Electrical Engineering. 2015;47:222-40.

[7] Poess M, Nambiar RO. Energy cost, the key challenge of today's data centers: a power consumption analysis of TPC-C results. Proceedings of the VLDB Endowment. 2008;1(2):1229-40.

[8] Horri A, Mozafari MS, Dastghaibyfard G. Novel resource allocation algorithms to performance and energy efficiency in cloud computing. The Journal of Supercomputing. 2014;69(3):1445-61.

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

[10] Beloglazov A, Buyya R. 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. 2013;24(7):1366-79.

[11] Beloglazov A, Buyya R. 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. 2012;24(13):1397-420.

[12] Shaw SB, Singh AK. 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. 2015;47:241-54.

[13] Fu X, Zhou C. Virtual machine selection and placement for dynamic consolidation in Cloud computing environment. Frontiers of Computer Science. 2015;9(2):322-30.

[14] Wu G, Tang M, Tian Y-C, Li W. Energy-Efficient Virtual Machine Placement in Data Centers by Genetic Algorithm. 2012;7665:315-23.

[15] Xu J, Fortes JA, editors. Multi-Objective Virtual Machine Placement in Virtualized Data Center Environments. Proceedings of the 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing; 2010: IEEE Computer Society.

[16] Tang M, Pan S. A Hybrid Genetic Algorithm for the Energy-Efficient Virtual Machine Placement Problem in Data Centers. Neural Processing Letters. 2014;41(2):211-21.

[17] Joseph CT, Chandrasekaran K, Cyriac R. A Novel Family Genetic Approach for Virtual Machine Allocation. Procedia Computer Science. 2015;46:558-65.

[18] Farahnakian F, Pahikkala T, Liljeberg P, Plosila J, Tenhunen H, editors. Utilization Prediction Aware VM Consolidation Approach for Green Cloud Computing. Proceedings of the 2015 IEEE 8th International Conference on Cloud Computing; 2015: IEEE Computer Society.

[19] Farahnakian F, Liljeberg P, Plosila J, editors. LiRCUP: Linear Regression Based CPU Usage Prediction Algorithm for Live Migration of Virtual Machines in Data Centers. Proceedings of the 2013 39th Euromicro Conference on Software Engineering and Advanced Applications; 2013: IEEE Computer Society.

[20] Gao Y, Guan H, Qi Z, Hou Y, Liu L. A multi-objective ant colony system algorithm for virtual machine placement in cloud computing. Journal of Computer and System Sciences. 2013;79(8):1230-42.

[21] Shaw SB, Singh A, editors. A survey on scheduling and load balancing techniques in cloud computing environment. Computer and Communication Technology (ICCCT), 2014 International Conference on; 2014: IEEE.
[22] He L. Exponential Smoothing. 18/05/2015. Available from: http://en.wikipedia.org/wiki/Exponential_smoothing.
[23] Chatfield C. The analysis of time series: an introduction: CRC press; 2016.
[24] Chatfield C. Time-series forecasting: CRC Press; 2000.
[25] Buyya R, Ranjan R, Calheiros RN, editors. Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities. High Performance Computing & Simulation, 2009 HPCS'09 International Conference on; 2009: IEEE.
[26] Calheiros RN, Ranjan R, Beloglazov A, De Rose CA, Buyya R. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. SoftwPractExper. 2011;41:23-50.
[27] Calheiros RN, Ranjan R, De Rose CA, Buyya R. CloudSim: A Novel Framework for Modeling and Simulation of Cloud Computing Infrastructures and Services. arXiv preprint arXiv:09032525. 2009.
[28] C. Chou, Y. Chen, D. Milojicic, N. Reddy & P. Gratz. (2019). Optimizing post-copy live migration with system-level checkpoint using fabric-attached memory. IEEE/ACM Workshop on Memory Centric HighPerformance Computing (MCHPC), Available at: https://doi.org/10.1109/MCHPC49590.2019 .00010.
[29] Osama Alrajeha, Matthew Forshawb& Nigel Thomasb. (2021). Using virtual machine live migration in trace-driven energy-aware simulation of highthroughput computing systems. Sustainable Computing: Informatics and Systems, 29, Available at https://doi.org/10.1016/j.suscom.2020.1004 68