PERFORMANCE ANALYSIS OF AN ENERGY EFFICIENT VIRTUAL MACHINE CONSOLIDATION ALGORITHM IN CLOUD COMPUTING

Ms Jayshri Damodar Pagare
Research Scholar, Sant Gadge Baba Amravati University, Amravati, India

Dr. Nitin A Koli
Head, Computer centre, Sant Gadge Baba Amravati University, Amravati, India

ABSTRACT

VM consolidation has been shown as a one of the promising technique for saving energy costs of a data center, it is an effective way to improve the utilization of resources and energy efficiency in cloud data centers. In this paper, for cloud environments, a VMs consolidation approach is proposed as energy efficiency is becoming a very important issue in Cloud Computing environments and can be achieved using virtual machine consolidation. Proposed algorithms have been implemented and evaluated using CloudSim simulator. Simulation results show improvement in energy consumption which leads to energy efficiency.

Keywords: Energy Efficiency, Data Center, Cloud Computing, Virtual Machine Consolidation.

1. INTRODUCTION

The use of virtualization allows providers to run multiple workloads from different customers on the same computing infrastructure which makes Cloud Computing as a promising approach to improve resources utilization. With adoption of virtualization services, cloud computing platforms are becoming more popular. However, Cloud providers rely on large and power-consuming data centers. The wide adoption of virtualization and cloud computing has added another layer of complexity to enabling an energy-efficient use of computing power in large-scale settings[1]. Energy consumption has always been a major concern in the design and cost of datacenters.

Energy-efficient Cloud environments need to deal not only with energy consumption but also with increasing demand, high QoS expectations and sustainable services [2]. However, a large
A number of cloud servers consume massive energy and produce huge pollution. The Smart2020 analysis shows that cloud-based computing data centers and the telecommunication network will generate emission about 7% and 5% each year in 2002 and 2020, respectively. [3]

Power consumption is one of the most critical problems in data centers. One effective way to reduce power consumption is to consolidate the hosting workloads and shut down physical machines which become idle after consolidation. [4]

The solution to this issue is to migrate and consolidate VMs in data centers. Although it can greatly reduce energy consumption, it will result in SLA violation. Obviously, we cannot eliminate the SLA violation by this way, but we can reduce SLA violation as much as possible while reducing energy consumption, which is called energy-consumption trade-off and also the focus of this article. VM consolidation can be loosely classified into dynamic, semi-static and static. Dynamic consolidation requires a runtime placement manager to migrate VM automatically according to workload variations or sudden burst. On the other hand, in static consolidation, once a VM is placed on a physical server, it will not get migrated for a long time period, such as months or years. Finally, semi-static consolidation migrates VMs in a hourly or daily basis according to the statistics of historical workload measurements. [5]

The main focus of this work is to introduce an efficient SLA-aware algorithm (i.e., to avoid SLA violation as much as possible). The proposed algorithms consider the trade-off between energy consumption and performance. The most important contributions of this paper are as follows:

- Proposing an efficient algorithm which finds and decides overloaded host with SLA violation
- Proposing an efficient algorithm for finding underutilized hosts.
- Combine two proposed algorithms as virtual machine consolidation algorithm to get better results in both power consumption and SLA violation.

The overload detection finds overload host and get the status of Overload host whether it result in SLA Violation or not. If there is no SLA violation then no migration of virtual machines required which saves power required to migrate VMs. But if there is SLA violation then place all the VMs from this host to other hosts until the host becomes Underload.

The underload detection algorithm finds the host with the minimum utilization compared to the other hosts, and tries to place all the VMs from this host to other hosts keeping them not overloaded. If this can be done, the VMs are set for migration to the determined destination hosts, and the source host is switched to the sleep mode once all the migrations have been completed. If all the VMs from the source host cannot be allocated, the host is kept active. This algorithm is repeated for all hosts that have not been considered as being overloaded.

The paper is organized as follows. In Sect. II the related work is discussed. In Sect. III consolidation algorithm is introduced. In Sects. IV methodology and metrics are discussed. In Sect V the experimental setup and their results are discussed and in Sect. VI the system model is introduced that has been used in our work. Finally Sect. VII concludes the paper.

2. RELATED WORK

The first work in large-scale virtualized data centers to manage power efficiently, in which power management has been applied in the context of virtualized data centers, has been done by Nathuji and Schwan [6]. The authors have proposed an architecture of a data center’s resource management system where the resource management is split into local and global managers. At the local level the system leverages the guest OS’s power management strategies. The global manager gets the information on the current resource allocation from the local managers and applies its policy to decide whether the VM placement needs to be adapted. Local manager coordinates power management methods of VMs in each host because the authors assumed that VM guests have a
power aware OS. Global manager monitors the performance of multiple hosts and selects the appropriate host for requested VM migration.

The authors [7] have proposed a method for energy efficient resource allocation, in the context of a networked cloud environment. The method employs dynamic server consolidation by periodic VM migration.

H. Goudarzi et al. [8] presented an approach that first creates multiple copies of VMs and then uses dynamic programming and local search to place these copies on the physical servers.

In this the authors [9] present two novel approaches for virtual machines (VMs) placement consolidation, which aim to maximize the placed VMs on a host and minimize the number of hosts used on a cloud computing environment. The first proposed approach is based on the Knapsack problem and the second one is based on an Evolutionary Computation heuristic.

The work in [10] aims to proactively prevent VM migration for semi-static VM consolidation by proposing a deadline driven VM placement strategy based on the awareness of the server turn-off time and job execution time using a real HPC cluster trace as well as a set of synthetic generated workloads.

Beloglazov et al. [11] use “live-migration” and VM consolidation, but only focus on the QoS of such an approach even in heterogeneous infrastructure containing heterogeneous VMs. An attempt is made to solve both the traditional bin packing problem as well as the “intermediate state” data center optimization problem and four heuristics are proposed for choosing which VM to migrate. Anton et al. [12] have defined an architectural framework and principles for energy-aware Cloud computing and has developed algorithms for energy-aware mapping of VMs, they have used the concept of dynamic consolidation of VM resource partitions.

The fixed threshold in [12] for vm consolidation is not suitable for virtual environment with variable workloads. Therefore, they [13] illustrate that VM consolidation for variability of workloads. Then they propose novel adaptive heuristics for dynamic consolidation of VMs. Results show that the allocation and selection algorithms save energy consumption. However, the SLA violation and energy consumption produced by the framework has a scope of improvement.

The framework selects VMs from an Overload host until the host becomes Underload. If the Overload host does not generate SLA violation, then the migration will be in vain and result in high energy consumption. Therefore, we require a appropriate method to decide the VM selection in this step. By considering above problem, we propose another VM consolidation algorithm in CloudSim, [14] which performs better.

3. ENERGY EFFICIENT CONSOLIDATION ALGORITHM

Following flowchart shows the flow of proposed virtual machine consolidation algorithm which is a combination of two proposed algorithm mentioned above with existing algorithm in [13] for finding new VMs placements.
The proposed virtual machine consolidation is shown in Algorithm 1 has following

**Step I:** First, the algorithm ee the list of hosts and by applying the proposed overload detection algorithm checks

Whether a host is overloaded. If the host is overloaded, the algorithm applies the VM selection policy to select VMs that need to be migrated from the host. If the Overload host does not generate SLA violation, then the migration will result in higher energy consumption. Therefore, we need a method to decide the status of Overload host whether it result in SLA Violation or not.
Step II: Once the list of VMs to be migrated from the overloaded hosts is built, the VM placement algorithm in [13] is invoked to find a new placement for the VMs to be migrated.

Step III: Finding underloaded hosts and a placement of the VMs from these hosts.

The consolidation algorithm returns the combined migration map that contains the information on the new VM placement of the VM selected to be migrated from both overloaded and underloaded hosts. The complexity of the algorithm is 2N, where N is the number of hosts.

Algorithm 1: VM Consolidation Algorithm

1. Input: hostList Output: migrationMap
2. vmsToMigrate = NULL
3. for each host in hostList do
   4. if isHostOverloaded (host) then
      5. vmsToMigrate = vmsToMigrate + getVmsToMigrateFromOverloadedHost (host)
      6. migrationMap.add(getNewVmPlacement(vmsToMigrate))
      7. vmsToMigrate.clear()
   8. for each host in hostList do
      9. if isHostUnderloaded (host) then
         10. vmsToMigrate = vmsToMigrate + getVmsToMigrateFromUnderloadedHost (host)
         11. migrationMap.add(getNewVmPlacement(vmsToMigrate))
   12. return migrationMap

3.1 Overloaded host detection with status of SLA Violation

For step 4 in algorithm 1 above we have proposed an algorithm that finds and decides the status of overloaded host with SLA Violation as given in algorithm 2. Here we have used the parameters OverSLAV for the overloaded host with SLA violation, OverNSLAV for the overloaded with no SLA violation, Over for overloaded host, Under for underloaded host, Idle for the Idle host, Saturated for the host which does not send or receive any migration. According to [13], when the total request utilization of the VMs exceeds the allocated utilization of them on host Hi, Hi will generate SLA violation. If they are equal, it will be assumed to generate no SLA violation. The total allocated utilization of the VMs on host Hi can never exceed the maximum utilization of the host. It means that if the request utilization of the VMs on Hi exceeds the maximum utilization, the host will definitely generate SLA violation.

Once it has been decided that a host is overloaded and got its status, the next step is to select particular VMs to migrate from this host. Here we have used four policies for VM selection. The policies are applied iteratively. After a selection of a VM to migrate, the host is checked again for being overloaded. If it is still considered as being overloaded, the VM selection policy is applied again to select another VM to migrate from the host. This is repeated until the host is considered as being not overloaded. The complexity of the algorithm is nm^2, where n is the number of host and m is the number of VMs that is to be migrated.
3.2 VM placement

Here for step 6 and 11 in algorithm 1 we have used an existing PABFD algorithm in[13] for finding new virtual machine placement.

3.3 Underloaded host detection

For step 9 in algorithm 1 above we have proposed an algorithm that find underloaded host which is given in algorithm 3 below. First find the CPU utilization of each host then sort in decreasing order so as to find the minimum utilization host as underloaded host to migrate all VMs from this host to other host by applying VM placement algorithm in[13] without overloading the other host. The complexity of the algorithm is nm, where n is the number of host and m is the number of VMs that is to be migrated.
The advantage of proposed algorithm is that it reduces no of migrations and hence reduces SLA violation and energy consumption, second a host with least number of VMs has a better chance to be switched to sleep mode in comparison with a host with more VMs.

4 METHODOLOGY AND METRICS

Energy efficiency metrics focus on advancing energy efficiency in data centers and computing ecosystems.

4.1 The overload decision algorithm

For ease, the algorithm can be abbreviated to ODA, which intent to decide a host Overloaded or not. Five ODAs have been implemented in CloudSim, i.e. Interquartile Range (IQR), Static Theshold (THR), Local Regression (LR), Robust Local Regression (LRR) and Median Absolute Deviation (MAD).

4.2 The VM selection algorithm

For ease, the algorithm can be abbreviated to VMSA, which aims to select VMs from Over hosts and prevent them from being over. Four VMSAs have been implemented in CloudSim, i.e. Minimum Migration Time (MMT), Minimum Utilization (MU), Random Selection (RS) and Maximum Correlation (MC).

4.3 SLA Violation Metrics

Meeting QoS requirements is very important for Cloud computing. QoS requirements are commonly formalized as SLAs, which can be determined in terms of characteristics such as minimum throughput, maximum response time or minimum bandwidth and so on. These characteristics are workload or application dependent. However, the algorithm framework belongs to IaaS layer in Cloud computing and should be workload independent. Therefore, we use those SLA-related metrics defined in [13] to evaluate the proposed algorithm in our experiments. Simultaneously, we also use some other metrics including energy consumption, migrations and execution time.

(a) SLATAH (SLA violation Time per Active Host): The percentage of time, during which active hosts have experienced the CPU utilization of 100 %. When a host experiences 100 % utilization, it will not be able to allocate enough CPU to the VMs on it, so it will generate SLA violation. The SLATAH can be calculated using Eq. (1), where \( T_{si} \) is the SLA violation time and \( T_{ai} \) is the active time for \( Hi \).
(b) SLAV (SLA Violation) and ESV (Energy and SLA Violation): The SLATAH is used to evaluate host-level SLA violation due to host overloading. And the PDM is used to evaluate VM-level SLA violation due to the VM migration. Since the two metrics are independent, a combined metric is needed to evaluate the two SLA violations. As a result, the SLAV is proposed to evaluate the two SLA violations, which can be calculated using Eq. (2). Since the Energy and the SLAV are two main metrics, the ESV is proposed to combine the two metrics, which can be calculated using Eq. (3).

\[
SLAV = SLATAH \cdot PDM
\]  

\[
ESV = SLAV \cdot Energy
\]  

5 PERFORMANCE EVALUATION

As the targeted system is an IaaS in cloud computing, it’s better to evaluate the proposed algorithm on a large-scale virtualized data center infrastructure. However, it’s very difficult to conduct repeatable large-scale experiments on a real infrastructure, which is required to evaluate and compare the proposed algorithm. Therefore, to ensure the repeatability of the experiments, simulations have been chosen as a suitable way to evaluate the performance of the proposed algorithm.

For our experiments, the CloudSim toolkit [14] has been chosen as a simulation framework. The toolkit has been developed by the Cloud Computing and Distributed Systems (CLOUDS) Laboratory, University of Melbourne. It supports both system and behavior modeling of cloud system components such as data centers, virtual machines (VMs) and resource provisioning policies. Currently, it supports modeling and simulation of Cloud computing environments consisting of both single and inter-networked clouds (federation of clouds), and also supports energy-efficient management of datacenter resources. Apart from the energy consumption modeling and accounting, the ability to simulate service applications with dynamic workloads has been incorporated.

5.1 Experimental Setup

We have simulated a data center that comprises 800 heterogeneous physical nodes, half of which are HP ProLiant ML110 G4 servers, and the other half consists of HP ProLiant ML110 G5 servers.

The characteristics of the servers and data on their power consumption are given in 6.1 next. The frequency of the servers’ CPUs are mapped onto MIPS ratings: 1860 MIPS each core of the HP ProLiant ML110 G4 server, and 2660 MIPS each core of the HP ProLiant ML110 G5 server. Each server is modeled to have 1 GB/s network bandwidth. The characteristics of the VM types correspond to Amazon EC2 instance types with the only exception that all the VMs are single-core, which is explained by the fact that the workload data used for the simulations come from single-core VMs (Section 6.1). For the same reason the amount of RAM is divided by the number of cores for each VM type: High-CPU Medium Instance (2500 MIPS, 0.85 GB); Extra Large Instance (2000MIPS, 3.75 GB); Small Instance (1000 MIPS, 1.7 GB); and Micro Instance (500 MIPS, 613 MB).

Initially the VMs are allocated according to the resource requirements defined by the VM types. However, during the lifetime, VMs utilize less resources according to the workload data, creating opportunities for dynamic consolidation.
5.2 Workload Data

In order to make the results more convincing, we need to use workload data obtained from real system. For our experiments, we have used data provided as a part of CoMon project, a monitoring infrastructure for PlanetLab [15], and it is about CPU utilization by more than a thousand VMs from servers located at more than 500 places around the world. The interval of utilization measurements for the data is 5 minutes. There are 10-day CPU utilization records during March and April 2011 in the data. Table II from the literature [13], gives brief analysis about the workload.

5.3 Result Analysis

In the experiment, we use four-type ODAs (THR, LR, LRR and MAD) and four-type VMSAs (MC, MU, MMT, RS). Therefore, there are 16-type combinations policies for the ODA and the VMSA. According to the simulation results from reference [13], we set a suitable constant parameter for each ODA, and they are 0.8 for THR, 1.2 for LR, 1.2 for LRR and 2.5 for MAD respectively. The combination are THR_MC_0.8, THR_MU_0.8, THR_MMT_0.8, THR_RS_0.8, LR_MC_1.2, LR_MU_1.2, LR_MMT_1.2, LR_RS_1.2, LRR_MC_1.2, LRR_MU_1.2, LRR_MMT_1.2, LRR_RS_1.2, MAD_MC_2.5, MAD_MU_2.5, MAD_MMT_2.5, MAD_RS_2.5.

In fig 2, we use the 10-day workload to evaluate the proposed algorithm for the 16-type combination policies. In each subfigure of the figure, each candlestick represents results from the 10-day workload. For better comparison, we use the average of the results as an evaluation value.

1) The SLATAH evaluation

The minimum and the maximum evaluation values for the Origin are 5.5% and 7.87% respectively in fig 2.1(a). For the proposed algorithm, the values are 3.29% and 5.57% respectively as in fig 2.1(b). Compared with the Origin, the proposed algorithm has 40.18%~58.19% decrease for the 16-type combinations for the SLATAH respectively.

2) The Energy evaluation: In fig 2.2(a), for the Origin, the minimum and maximum evaluation values are 141.933 kWh and 171.444 kWh respectively. In fig 2.2(b) for the proposed algorithm, the values are 125.03 kWh and 128.29 kWh respectively. Compared with the Origin, the proposed algorithm has 11.9%~27.07% decrease for energy consumption for the 16-type combinations respectively.
3) **The SLAV evaluation:** In fig 2.3(a), for the Origin, the minimum and maximum evaluation values are 0.707% and 1.379% respectively. In fig 2.3(b) for the proposed algorithm, the values are 0.137% and 0.598% respectively. Compared with the Origin, the proposed algorithm has 80.62%~90% decrease for the 16-type combinations for the SLAV respectively.

4) **The ESV evaluation:** In fig 2.4(a), for the Origin, the minimum and maximum evaluation values are 10.06% and 23.25% respectively. In fig 2.4(b) for the proposed algorithm, the values are 2.87% and 6.98% respectively. Compared with the Origin, the proposed algorithm has 72.92%~87.65% decrease for the 16-type combinations for the ESV respectively.

In a nutshell, the proposed consolidation algorithm outperform the origin for the all above metrics.
6 SYSTEM MODEL

6.1 CPU architecture and power model

In this paper, it is assumed that each CPU has \( c \) cores and each core has \( m \) MIPS, so total MIPS of a CPU is \( c \times m \).

The servers used in this paper are: HP ProLiant ML110 G4, HP ProLiant ML110 G5. Tables I and II show the configuration and power consumption, respectively.

### Table I configuration of servers

<table>
<thead>
<tr>
<th>Server</th>
<th>CPU model</th>
<th>Cores</th>
<th>Frequency (MHz)</th>
<th>RAM (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HP ProLiant G4</td>
<td>Intel Xeon 3040</td>
<td>2</td>
<td>1,860</td>
<td>4</td>
</tr>
<tr>
<td>HP ProLiant G5</td>
<td>Intel Xeon 3075</td>
<td>2</td>
<td>2,660</td>
<td>4</td>
</tr>
</tbody>
</table>

The reason why we have not chosen servers with more cores is that it is important to simulate a large number of servers to evaluate the effect of VM consolidation. Thus, simulating less powerful CPUs is advantageous, as less workload is required to overload a server. Nevertheless, dual-core CPUs are sufficient to evaluate resource management algorithms designed for multi-core CPU architectures.

### Table II Power consumption of server at different load levels in watts

<table>
<thead>
<tr>
<th>Server</th>
<th>0%</th>
<th>10%</th>
<th>20%</th>
<th>30%</th>
<th>40%</th>
<th>50%</th>
<th>60%</th>
<th>70%</th>
<th>80%</th>
<th>90%</th>
<th>100%</th>
</tr>
</thead>
<tbody>
<tr>
<td>HP ProLiant G4</td>
<td>86</td>
<td>89.4</td>
<td>92.6</td>
<td>96</td>
<td>99.5</td>
<td>102</td>
<td>106</td>
<td>108</td>
<td>112</td>
<td>114</td>
<td>117</td>
</tr>
<tr>
<td>HP ProLiant G5</td>
<td>93.7</td>
<td>97</td>
<td>101</td>
<td>105</td>
<td>110</td>
<td>116</td>
<td>121</td>
<td>125</td>
<td>129</td>
<td>133</td>
<td>135</td>
</tr>
</tbody>
</table>

7 CONCLUSION

To obtain quick ROI (Return On Investment), Cloud providers should reduce energy consumption as much as possible while keeping a low-level SLA violation in data center, which is also called energy-performance tradeoff. Using energy-efficient resource management policies will lead to increase in their revenues. This can be done by consolidating VMs and switching idle servers to sleep modes. However, improper consolidation may lead to SLA violation. In this paper, we have proposed an energy-efficient algorithm for VM consolidation which can reduce energy consumption and at the same time the SLA violation. The simulations show that the proposed algorithms significantly reduce number of VM migration, SLA violation in comparison with current algorithms [13].

We have evaluated proposed algorithm and the Origin framework through simulation on large-scale experiments driven by workload traces collected from more than a thousand PlanetLab VMs, and the results show that our proposed algorithm gets a better energy-performance. It guarantees 11.9–27.07% decrease in energy consumption, 80.62–90% decrease in SLA violation.
8 REFERENCES


