# Priority-Aware Virtual Machine Selection Algorithm in Dynamic Consolidation

Hanan A. Nadeem[1], Mai A. Fadel[3]
Computer Science Department
Faculty of Computing & Information Technology
King Abdulaziz University, Jeddah,
Saudi Arabia

Hanan Elazhary[2]
Computer Science Department
Faculty of Computing & Information Technology
University of Jeddah, Jeddah, Saudi Arabia
Computers & Systems Department
Electronics Research Institute, Cairo, Egypt

***Abstract*—In the past few years, many researchers attempted to tackle the problem of decreasing energy consumption in cloud data centers. One of the widely adopted techniques for this purpose is dynamic Virtual Machine (VM) consolidation. Consolidation moves VMs between hosts to decrease energy consumption. However, it has a negative impact on performance leading to Service Level Agreement (SLA) violations. Accordingly, selecting which VM to migrate from one host to another is a challenging task since it can affect performance. Researchers came up with several solutions and policies for efficient VM selection. In this paper, we exploit the fact that many tasks and users may tolerate some performance degradation which means, the tasks running on the VMs can be of different priorities. Accordingly, we propose augmenting consolidation with the priority concept, where low priority tasks are always selected first for migration. Towards this goal, we modified the popular Minimum Migration Time VM selection algorithm using the priority concept. The efficiency of the proposed algorithm is confirmed through extensive simulations using CloudSim toolkit and a real workload. The results show that priority awareness has a positive impact on decreasing energy consumption as well as maximizing SLA obligation.**

***Keywords*—Cloud computing; energy efficiency; service level agreement; VM consolidation; VM selection**

## I. Introduction

Virtual Machine (VM) consolidation is a useful technique for enhancing the utilization of the resources of cloud data center and reducing their energy consumption by leveraging the virtualization technology [1]. Virtualization provides the ability to create more than one VM instance on a single Physical Machine (PM) host. Accordingly, it permits more than one application to be allocated on a single PM in order to enhance the overall resource utilization and reduce the overall energy consumption of the data center.

Virtualization also allows live migration of VMs. Dynamic VM consolidation adopts live migration to minimize the number of PMs to which VMs are allocated. This is achieved by shutting down an underutilized host for more energy conservation and migrating its VMs to other PMs. However, this may lead to Service Level Agreement (SLA) violations. SLAs are established between cloud service providers and users to specify the required Quality of Service (QoS). After provisioning QoS, it should be monitored to ensure it is maintained throughout the service duration. QoS provisioning and monitoring are two classical problems in computer science [2] [3]. Unfortunately, when a VM migrates, its primary memory has to be transferred to the destination PM. Unfortunately, during this process, the requested CPU cannot be provided since the VM will be in a transition state which causes performance degradation and leads to SLA violation. Accordingly, enhancement of energy consumption and performance is a trade-off problem. Thus, dynamic VM consolidation techniques need to be designed with ultimate care such that not only power consumption is reduced, but also the requested QoS defined through SLAs is maintained. By carefully choosing which VMs to migrate when needed, consolidation can maintain more obligation of SLAs.

Dynamic VM consolidation is typically broken down into separate sub-problems [4]:

*1)* Host Overload detection: determining if a host is viewed as an overloaded one calling for a decision to choose one or more VMs to be migrated from this host.

*2)* Host Underload detection: determining if a host is viewed as an underloaded one calling for migrating all VMs allocated to this host to another, and switching the host to the low power mode.

*3)* VM Placement: finding a suitable destination host for allocating the migrated VMs from the overloaded and underloaded hosts.

*4)* VM Selection: a decision of which VMs should migrate from an overloaded host.

Simplifying the VM consolidation technique by diving it into four sub-problems and providing a separate algorithm for each one has the advantage of isolated examination and analysis of each algorithm to find a better approach. This work focuses on enhancement of the VM selection sub-problem.

Our proposed technique is based on the observation that some people might tolerate some performance degradation in services provided by a cloud and accept some SLA violations for cost savings while others cannot. For example, latency insensitive applications can tolerate some delay. From this prospect, we propose a novel approach in which we classify cloud user's tasks into two priority classes and deal with them in two different ways as follows:

- The users with high priority tasks should get a maximum obligation of their SLAs, and the cloud service providers should accept some energy consumption.
- The users with low priority tasks encounter reduced cost at the expense of accepting some SLA violations, while the cloud service providers gain more energy savings.

In other words, we treat users and their tasks differently and attempt to balance energy and performance as much as possible by considering priority when selecting a VM to migrate. It is worth noting that the cloud users' tasks priority will be assigned by the cloud provider as requested by the cloud users.

One of the most popular and effective VM selection algorithms in literature is the Minimum Migration Time (MMT) which picks the VMs with the minimum time required for migration. In this work, we propose priority-aware MMT algorithm to reduce the energy consumption while providing more SLA obligation for users with high priorities. The rest of the paper is organized as follows: Section II discusses related work, Section 3 describes the proposed VM selection algorithm, Section 4 describes the experimental settings and results; and finally, the conclusion will be in Section 5.

## II. Related Work

VM selection and VM placement algorithms both comprise a challenging task of choosing a VM for migration and a preferable host for placement respectively. Several algorithms have been proposed in the literature for these purposes. Beloglazov et al. [4] proposed three VM selection algorithms; Random Selection (RS), MMT, and Maximum Correlation (MC). RS randomly chooses any VM for migration without any rules. The idea of MMT migration is to give preference to the VMs that require the minimum time for the whole migration process. Additionally, the VM with the maximum correlation coefficient relative to the other peer VMs on the same PM is the one selected for migration. The correlation is a parameter representing the effect of the VM on overloading the host. Moreover, the authors proposed Power Aware Best Fit Decreasing (PABFD) placement algorithm as a modification of the conventional Best Fit Decreasing (BFD) algorithm. The PABFD algorithm allocates each VM to the host that causes the least increase of power consumption due to placement.

Fu and Zhou [5] proposed a novel VM selection algorithm called Meets Performance (MP). This algorithm finds the host's utilization deviation over the host overload threshold and compares it with the utilization of VMs on the host. It then selects the VM, whose migration results in shifting the utilization of the host nearer to the upper threshold. This is to reduce the number of migrations needed. Furthermore, the authors proposed a novel VM placement algorithm called Minimum Correlation Coefficient (MCC). This coefficient is used to describe the intense of correlation between the selected VM for migration and the destination host. The higher the correlation, the higher the effect on the performance of the destination host. The algorithm selects a VM with the minimum correlation with the target host to avoid degrading the performance of the other VMs allocated to it.

Rahimi et al. [6] proposed a VM placement algorithm based on priority routing. The main idea is to classify VMs based on their resource utilization, and classify hosts based on their resource availability, then give priority to the resources where CPU has the higher priority compared to the RAM, while the bandwidth has the lowest priority. After that, VMs are placed on the host with the most similar categories by creating a routing path table and considering resource priority. It is worth noting that this idea of priority is totally different from our proposed priority concept of tasks and users.

Farahnakian et al. [7] optimized VM placement by adopting Ant Colony Optimization (ACO) technique and proposed the Ant Colony System-based VM Placement Optimization (ACS-PO) algorithm. The proposed approach uses artificial ants in order to consolidate VMs and allocate them to the smallest number of active hosts based on the present requirements of the resources. What is interesting about those ants is that they work concurrently to develop VM migration plans based on a defined objective function.

Monil and Rahman [8] proposed a fuzzy VM selection algorithm. The fuzzy technique is an approach for tackling intelligent decision-making problems. The authors recognized that there are different VM selection algorithms in the literature offering different advantages; and generated a method which can aggregate the advantages of all of them in a single fuzzy logic tool. The input to the fuzzy tool is MMT and MC discussed above and the output is a VM selected for migration.

As discussed above and to the best of our knowledge, none of the algorithms on the literature classifies the cloud user's tasks based on their priorities. In this paper, such classification is exploited for delivering a better balance between energy consumption and performance in cloud data centers. Specifically, we modify the MMT algorithm using this priority concept as explained in the followings section.

## III. Proposed Algorithm

As noted above, cloud service providers can satisfy their requirements of decreasing the cost of energy consumption while optimizing their resource utilization by using dynamic VM consolidation. Typically, the dynamic VM consolidation process sets up a threshold called the utilization threshold. It then monitors all active hosts' utilization ensuring that none of them exceeds this threshold. Whenever such a case is detected, some VM have to be offloaded from the corresponding source host and migrated to another destination host to avoid performance degradation on the source.

As mentioned before, dynamic VM consolidation is typically broken down into separate sub-problems [4]:

*1)* Host overload Detection: Local Regression Robust (LRR) [4] is one of the most efficient and widely used algorithms to set the utilization threshold and keep the summation of the utilization of all VMs bellow it. If the CPU utilization exceeds the set threshold, the consolidation process invokes the VM selection and VM placement algorithms to take an action.

*2)* Host Underload Detection: Host with minimum CPU utilization algorithm [4] is one of the popular and successful

algorithms for this purpose. The idea is to find the host with the minimum CPU utilization compared to the other hosts. This host is recognized as the underloaded host, and all VMs on it have to be migrated with attention to the destination hosts after placing the VMs to avoid making them overloaded.

*3)* VM Placement: PABFD [4] discussed above is the most widely known and one of the most effective VM placement algorithms. It allocates each VM to the host which causes the minimum increase of power consumption due to this allocation.

*4)* VM Selection is a decision of selecting which VM has to be migrated. This is where our paper contributes. Our optimization is based on the priority concept where, as discussed earlier, latency insensitive applications that tolerate delay and users who may accept some performance degradation for price savings are given lower priorities as discussed below.

```
Priority-based Minimum Migration Time Selection algorithm
1  Input: overloadedHost   OutPut: VM selected for migration
2  foreach vm in overloadedHost do
3      if vm utilized by low priority
4          lowPriorityList.add(vm)
5  foreach vm in lowPriorityList do
6      minMigTime ← MIN
7      selectedVm ← NULL
8      if vm.migrationTime() < minMigTime then
9          minMigTime ← vm.migrationTime()
10         selectedVm ← vm
11 if selectedVm ≠ NULL
12 return selectedVm
```

Fig. 1. Priority-aware MMT VM selection algorithm.

When the total requirements of CPU performance by the VMs exceed the available CPU capacity of the PM, the host is recognized as an overloaded host; the overloaded host may cause an increase in response time and a decrease in throughput. In this case, the cloud users do not get the expected QoS, and some VMs must be migrated from this host to decrease the CPU utilization of the host. Since live VM migration also has a negative impact on performance, low priority tasks will be the ones chosen for migration since those tasks accept some performance degradation due to migration and tolerate some SLA violation. On the other hand, the high priority tasks will be kept in the host saved from performance degradation due to migration.

As shown in Figure 1, we adopt the efficient most well-known MMT [4] VM selection algorithm and modify it using the priority concept. We make the selection decision in two phases. In the first phase, we select all the low priority tasks from an overloaded host and prepare a list for the second phase. The second phase selects from the low priority list the VM with the minimum time required for its migration in comparison to the other peer VMs allocated to the host. The time required for migration is defined [4] as the ratio between the RAM amount used by the VM and the available network bandwidth. After migrating the selected VM, the process is iteratively repeated as long as the host is still overloaded.

## IV. Experiments and Results

Since the system of interest is Infrastructure as a Service (IaaS), which is a cloud environment intended to give the users a view of infinite computing resources, it is clear that we need to experiment with and evaluate the proposed VM selection algorithm on a large-scale virtualized data center infrastructure. However, conducting such an experiment in a real environment as a repeatable experiment is very difficult. Thus, simulations were chosen to evaluate the performance of the proposed VM consolidation technique and preserve the repeatability of experiments. CloudSim toolkit [9] is the simulation platform selected because it is a popular framework for simulating cloud computing settings. It can be used for modeling virtualized settings; supporting on-demand resource provisioning management. A recent extension to CloudSim allows energy aware simulations and supports energy-efficient strategies. This is in addition to allowing the simulation of service-oriented applications with dynamic workloads.

### A. Experimental Settings

CPUs with dual-core are adequate for evaluating resource management algorithms intended to run on multi-core CPU architectures. In fact, it is essential to simulate a large number of servers to assess the efficiency of the VM consolidation mechanism. Thus, selecting less powerful CPUs for the simulations is beneficial because fewer workloads will overload a server [4]. To evaluate the efficiency of the proposed algorithm, we simulated a datacenter containing 800 heterogeneous PMs with two configurations:

- HP ProLiant ML110 G4 (Intel Xeon 3040, dual-core 1860 MHz, 4 GB, 1 Gbps).
- HP ProLiant ML110 G5 (Intel Xeon 3075, dual-core 2660 MHz, 4 GB, 1 Gbps).

The characteristics of the VM instances are of types identical to those of Amazon EC2 instances except that all VMs are single core. This is because the workload data employed in the simulation comes from the single core:

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

### B. Performance Metrics

Different performance metrics are used for evaluating the proposed VM selection algorithm. We adopt the same metrics proposed and elaborated by Beloglazov and Buyya [4]:

- Energy consumption in Kwh
- SLATAH (SLA violation Time per Active Host)
- PDM (Performance Degradation due to Migrations)
- SLAV rate (SLA Violation rate), which is the product of SLATAH and PDM

- Number of VMs migrated
- ESV, which is the product of energy consumption and SLAV rate

### C. Workload Data

To validate the proposed VM selection algorithm with more applicable simulations, a real workload from a CoMon system which is a monitoring infrastructure for PlanetLab [10] was used. This CPU utilization data is collected from more than thousand VMs from servers distributed over five hundred locations around the world every five minutes. Data is created from ten days randomly chosen during the months of March and April, 2011. The median value is calculated over the ten days and used with each performance metric. The basic features of this data are presented in Table 1.

TABLE I. WORKLOAD CPU UTILIZATION STATISTICS

| Date | Number of VMs | Mean | St. Dev | Quartile 1 | Median | Quartile 3 |
|---|---|---|---|---|---|---|
| 03/03/2011 | 1052 | 12.31 % | 17.09 % | 2% | 6% | 15% |
| 06/03/2011 | 898 | 11.44 % | 16.83 % | 2% | 5% | 13% |
| 09/03/2011 | 1061 | 10.70 % | 15.57 % | 2% | 4% | 13% |
| 22/03/2011 | 1516 | 9.26% | 12.78 % | 25 | 5% | 12% |
| 25/03/2011 | 1078 | 10.56 % | 14.14 % | 2% | 6% | 14% |
| 03/04/2011 | 1463 | 12.39 % | 16.55 % | 2% | 6% | 17% |
| 09/04/2011 | 1358 | 11.12 % | 15.09 % | 2% | 6% | 15% |
| 11/04/2011 | 1233 | 11.56 % | 15.07 % | 2% | 6% | 16% |
| 12/04/2011 | 1054 | 11.54 % | 15.15 % | 2% | 6% | 16% |
| 20/04/2011 | 1033 | 10.43 % | 15.21 % | 2% | 4% | 12% |

### D. Experimental Results

Using the PlanetLab workload data, we compare the original MMT algorithm with the priority awareness optimization. Figure 2 shows the energy consumption due to consolidation in Kwh. The results show that the priority-aware MMT decreases the energy consumption by 13%. Figure 3 shows the percentage SLATAH; the priority-aware MMT decreases the SLATAH metric by 42%. Figure 4 describes the performance degradation due to migration; choosing the low priority tasks that have minimum time to complete migration can provide 37% decrease in degradation. Figure 5 describes the overall SLA violation delivered by the consolidation technique; priority-aware MMT can provide 21% reduction of SLA violation. Figure 6 shows the energy consumption and SLAV rate; the rate is decreased by 31% when using the priority-aware MMT. Finally, Figure 7 shows the number of VMs migrated due to consolidation; they are reduced by 40% in case of the priority-aware MMT. Since live VM migration results in an overhead on the system, the better consolidation mechanism is the one which requires fewer migrations.

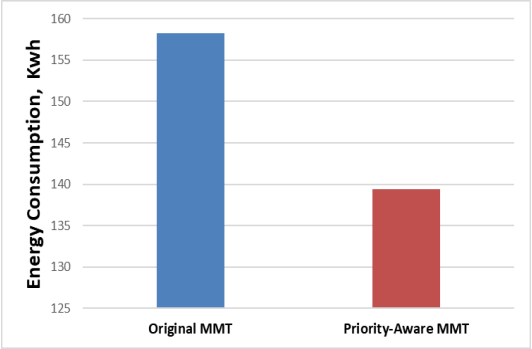


Fig. 2. Energy consumption.

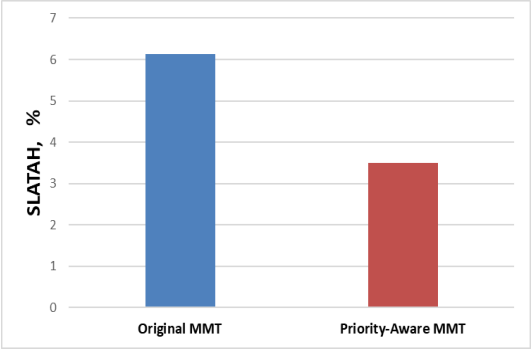


Fig. 3. SLA Time per Active Host.

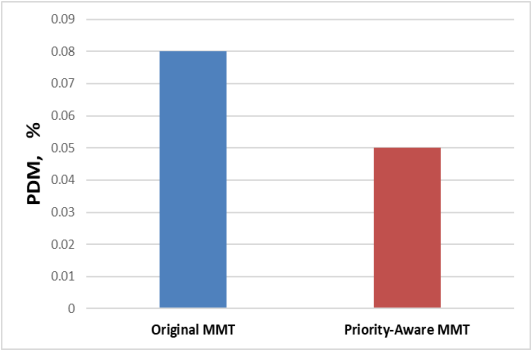


Fig. 4. Pefrormance degradation due to migration.

Based on the results above, it is clear that priority awareness has a considerable positive effect on all performance metrics. In other words, the proposed priority-aware VM selection algorithm is an efficient optimization in comparison to MMT algorithm regarding all metrics.

## V. CONCLUSION AND FUTURE WORK

In this paper, we proposed a novel priority-aware VM selection algorithm, which takes into consideration the priorities of tasks. This algorithm is original since, to the best of our knowledge, it is the first to exploit the priorities of cloud tasks and users. We selected the widely-used and successful MMT VM selection algorithm and showed that modifying it using priority-awareness has a positive effect on energy consumption and on all performance metrics.

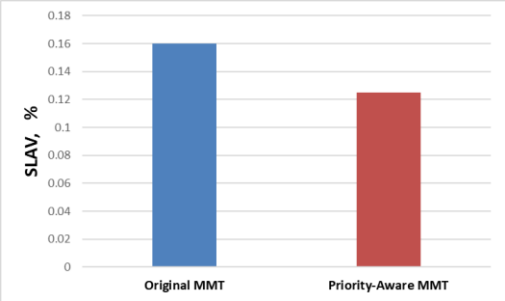


Fig. 5. Overall SLA violation.

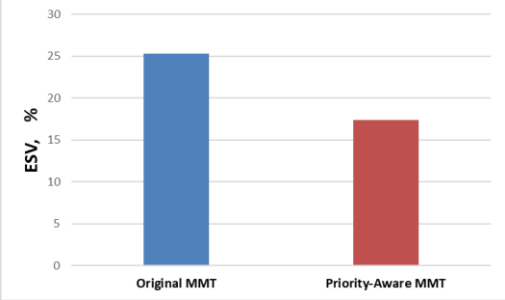


Fig. 6. Energy-SLA violation ratio.

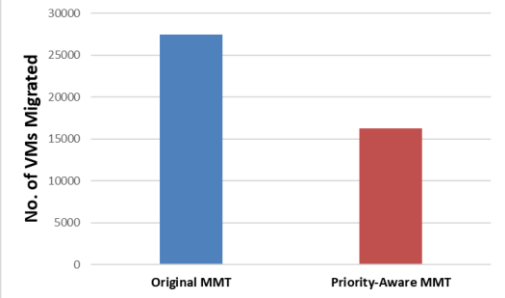


Fig. 7. Number of VMs migrated.

As future work, we intend to apply the proposed priority concept and experiment with different other VM selection algorithms to ensure the reliability of the proposed approach and further assess its effectiveness.

## REFERENCES

[1] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. "Xen and the art of virtualization," in Proc. the 19th ACM symposium on Operating Systems Principles, Bolton Landing, NY, USA, 2003, pp.164-177.

[2] H. Elazhary and S. Gokhale. “An integrated approach for QoS provisioning and monitoring,” in Proc. IASTED PDCN, Austria, 2004.

[3] H. Elazhary and S. Gokhale. “Integrating path computation and precomputation for quality-of-service provisioning,” in Proc. ISCC’04, Alexandria, Egypt, 2004.

[4] A. Beloglazov and R. Buyya. "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers," Concurrency and Computation: Practice & Experience, vol. 24, no.13, pp. 1397-1420, 2012.

[5] X. Fu and C. Zhou, " Virtual machine selection and placement for dynamic consolidation in Cloud computing environment", Frontiers of Computer Science Springer Link , vol. 9, no. 2, pp 322–330, 2015.

[6] A. Rahimi, L. Khanli, and S. Pashazadeh, "Energy efficient virtual machine placement algorithm with balanced resource utilization based on priority of resources," ComEngApp Journal, vol. 4, no. 2, 2015.

[7] F. Farahnakian, A. Ashraf, P. Liljeberg, T. Pahikkala, J. Plosila, I. Porres, and H. Tenhunen. " Energy-Aware Dynamic VM Consolidation in Cloud Data Centers Using Ant Colony System" in Proc. 2014 IEEE 7th International Conference on Cloud Computing, Anchorage, AK, USA, 2014.

[8] M. Rahman and R. Monil. "VM consolidation approach based on heuristics, fuzzy logic, and migration control", Journal of Cloud Computing, vol. 5, no. 8, 2016.

[9] R. Calheiros, R. Ranjan, A. Beloglazov, C. De Rose, and R. Buyya. "CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," Software: Practice and Experience, vol. 41, no. 1, pp. 23-50, 2011.

[10] K. Park and V. Pai. "CoMon: A mostly-scalable monitoring system for PlanetLab," ACM SIGOPS Operating Systems Review, vol. 40, no. 1, pp.65-74, 2006.