On Makespan, Migrations, and QoS Workloads’ Execution Times in High Speed Data Centers

Daniel LAGO*, Edmundo MADEIRA, and Deep MEDHI†, Nonmembers

SUMMARY With the growth of cloud-based services, cloud data centers are experiencing large growth. A key component in a cloud data center is the network technology deployed. In particular, Ethernet technology, commonly deployed in cloud data centers, is already envisioned for 10 Tbps Ethernet. In this paper, we study and analyze the makespan, workload execution times, and virtual machine migrations as the network speed increases. In particular, we consider homogeneous and heterogeneous data centers, virtual machine scheduling algorithms, and workload scheduling algorithms. Results obtained from our study indicate that the increase in a network’s speed reduces makespan and workload execution times, while aiding in the increase of the number of virtual machine migrations. We further observed that the number of migrations’ behaviors in relation to the speed of the networks also depends on the employed virtual machines scheduling algorithm.

key words: high speed networks, cloud computing, scheduling algorithm, quality of service, VM migration

1. Introduction

With the increase in the use of cloud computing services, data centers that support cloud services are experiencing massive growth. Cloud computing services use processing, memory, and storage capabilities from servers at data centers accessed via the Internet, as a utility. This can be done at the level of infrastructure, platform, development, and service [1]. It often uses virtualized resources [2]; among these resources, the virtual machines (VMs) are popular, which are virtual instances of computers accessible through the cloud. A physical machine (PM) may, for example, host a number of VMs, while these VMs can be migrated between the PMs [3].

Another important aspect in cloud data centers is that Ethernet is the predominant network technology of choice for interconnecting resources [4]. In a span of less than three decades, the data rate of Ethernet [5] has seen a remarkable increase since it was originally designed, evolving from 1 Mbps (802.3e, 1987) [6] to 100 Gbps (802.3bm, 2015) [7]. If we make a relationship between Ethernet’s speed increase and time, it is easy to note that the rate has grown over 350,000% per year. More recently, IEEE announced that the Ethernet rate standardization at 1 Tbps in 2015 to 10 Tbps by 2020 [8].

The fast expansion of cloud computing services brings concerns about its required power consumption. In previous work ([9] and [10]), scheduling algorithms of VMs were analyzed and workloads’ behaviors studied in order to minimize the energy consumption. It was found that the increased speed of networks can reduce energy consumption [11], since with faster networks, VM migration becomes faster, allowing to turn off the PMs that were not processing the VMs earlier.

Considering small and medium DCs operating in a cloud, in this work we advance the studies and seek to answer the following questions: How is the makespan affected by the increased speed of the networks? How do scheduling algorithms behave in relation to the makespan on DCs connected by high speed networks? Do high speed networks impact the number of VM migrations? Considering the scheduling algorithms of VMs and workloads, while working with priority-based quality of service, how is the execution time of these workloads affected by the networks operating at a higher speed? The goal of this paper is to answer these questions by analyzing different scenarios with respect to the makespan, migrations, and workloads’ execution times, and also contributing to implement and verify the interaction of some VMs’ scheduling algorithms with priority based QoS workloads’ scheduling algorithms.

This paper is organized as follows: in Sect. 2, we present related work. Basic concepts for understanding this work are presented in Sect. 3. In Sect. 4, we discuss the simulations done to make this work possible and their analysis. The conclusions are presented in Sect. 5.

2. Related Work

There are a number of research challenges in cloud computing [12]. Here, we present related works whose objectives are related to analyzing energy consumption, the number of VM migrations, and makespan or quality of service in the scheduling process of VMs and workloads.

Liu et al. [13] studied the live migration of VMs in DCs, aiming to model the performance and energy consumption. It was found that migration costs may vary significantly for different workloads due to the variety of VMs’ configurations and workload characteristics. After thoroughly analyzing the key parameters that affect the migration cost from theory to practice, they constructed two application-oblivious models for the cost prediction by us-

 Manuscript revised July 30, 2015.
 †The authors are with the University of Campinas, Brazil.
 ††The authors are with the University of Missouri-Kansas City, USA.
a) E-mail: lago@ic.unicamp.br
b) E-mail: edmundo@ic.unicamp.br
c) E-mail: dmedhi@umkc.edu
DOI: 10.1587/transcom.E98.B.2099
ing their knowledge about the workloads at the hypervisor, which should estimate the VMs’ live migration cost in terms of both performance and energy. It was found that model-guided decisions can significantly reduce the migration cost by more than 72.9% at an energy savings of 73.6%.

Stage and Setzer [14] worked with control and optimization of migration algorithms and migration-related network bandwidth consumption. They introduced network topology-aware scheduling models for the VMs’ live migrations, proposing a scheme for classifying the VMs based on their workload characteristics.

Meng et al. [15] proposed use of traffic-aware VM placement to improve the network scalability. By optimizing the placement of the VMs on the PMs, traffic patterns among the VMs can be better aligned with the communication distance between them; e.g., VMs with large mutual bandwidth usage are assigned to PMs in close proximity. A VM placement optimization problem was formulated that falls into the class of quadratic assignment problems, which is known to be NP-hard.

Piao et al. [16] proposed a VM placement and migration approach to minimizing the data transfer time consumption. Results obtained from simulation studies suggested that the proposed approach was effective in optimizing the data transfer between the VM and data, thus helping optimize the overall application performance.

Wood et al. [17] studied and presented black-box and gray-box Strategies for the VM’s migration. They presented Sandpiper, a system that automated the task of monitoring and detecting hotspots, determining a new mapping of physical to virtual resources and initiating the necessary migrations. Sandpiper implemented a black-box approach that was fully OS- and application-agnostic and a gray-box approach that exploited OS- and application-level statistics. Results showed that Sandpiper was able to resolve single server hotspots within 20 seconds and scaled well to larger DC environments.

Beloglazov and Buyya [18] presented results of a simulation-driven evaluation of heuristics for dynamic re-allocation of VMs using live migration according to current requirements for CPU performance. The results showed that the proposed technique brought substantial energy savings, while ensuring reliable QoS.

Verma et al. [19] investigated the design, implementation, and evaluation of a power-aware application placement controller in the context of an environment with heterogeneous virtualized server clusters. They presented the pMapper architecture and placement algorithms to solve the practical formulation of the problem of minimizing power subject to a fixed performance requirement and also presented comprehensive theoretical and experimental evidence to establish the efficacy of pMapper.

Kumar et al. [20] analyzed surveys scheduling techniques in cloud computing scenarios, by taking into consideration factors such as energy efficiency, cost minimization, and maximization of resource utilization. They cited the most efficient scheduling technique available for a particular set of user needs by comparing various techniques and the problems they address.

Agrawal and Rao [21] proposed a new formulation and showed that energy-aware scheduling is a generalization of the minimum makespan scheduling problem. They proposed three different algorithms for energy-aware scheduling, and their work gives insight into the time-energy trade-offs in scheduling.

In Table 1, we summarize a brief comparison of related works’ topics. The column migrations refers to the migration of the VMs; the time-related column refers to concern for the tasks execution time, whether individual (workloads) or the overall time (makespan); network speed refers to the concern of the networks’ speeds; QoS refers to the quality of service; and het. DCs refer to the heterogeneity of the DCs operating in the cloud. While other works have considered some of the factors, we consider all the factors comprehensively in this work. Specifically, we analyze scheduling algorithms to verify how they behave with higher networks’ speeds, with respect to makespan, migration, and the execution times of workloads.

### 3. Virtual Machine Migration and QoS-Based Workload Scheduling

The VM technology has recently emerged as a basic component of DCs and clusters, mainly due to its workloads insulation capabilities, consolidation and migration [22]. Generally, these features allow us to serve multiple users in a secure, efficient, and flexible way. As a result, the virtualized infrastructures are considered key components to drive the emerging paradigm of cloud computing [23].

Power-aware VM scheduling algorithms employ the migration feature in order to reduce DC energy consumption, usually seeking to consolidate a large number of VMs on the most energy efficient PMs, allowing unused PMs to shut down. We chose three representatives VMs scheduling algorithms: Best Resource Selection (BRS), Minimum Power Diff (MPD), and Lago Allocator (LA) [9]. BRS works to select the server that has the processor with the highest ratio of MIPS/mips. MPD aims to send the VMs to the host with the smallest difference between power consumed before and after the VM allocation. LA is based on four tie breaking rules: best host energy efficiency, lowest estimated host power consumption after VM allocation, highest CPU

<table>
<thead>
<tr>
<th>Work</th>
<th>Migrations</th>
<th>Time-Related</th>
<th>Network Speed</th>
<th>QoS</th>
<th>Het. DCs</th>
</tr>
</thead>
<tbody>
<tr>
<td>[13]</td>
<td>●</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[14]</td>
<td>●</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[15]</td>
<td>●</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[16]</td>
<td>●</td>
<td>●</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[17]</td>
<td>●</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[18]</td>
<td>●</td>
<td>●</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[19]</td>
<td>●</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[20]</td>
<td>●</td>
<td>●</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[21]</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td></td>
<td>●</td>
</tr>
<tr>
<td>[This Work]</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
</tbody>
</table>
For each VM’s scheduling algorithm, we consider four power-aware configurations that allow VM migrations: PT, PTQ, PDT, and PDTQ. In these acronyms, P means that unused servers will be turned off when not processing; D stands for a linear model of dynamic voltage and frequency scaling where unused servers consume 70% (based on [24]) of their maximum power, and when operating at full capacity the consumption is 100%; and T means enabled migration of virtual loads, trying to keep the servers’ processing entities working with, at most, 80% of their maximum capacity, where servers’ loads are checked every 5 seconds. These configurations have been selected due to the fact that they perform migrations, which is the most affected aspect by increasing the networks’ speed of DCs.

As we also aimed to assess the workloads’ execution times involving quality of service, we considered an ordinary workloads’ submission algorithm, and a priority based QoS workloads’ submission algorithm [10] to run with each one of the selected VMs’ scheduling algorithms and for each power-aware configuration.

In a cloud operating DC, one of the PMs must perform the function of a broker, which deals with events like resource requests, VM creation, workload return, and finalization of workloads and VMs. Just as a broker performs the allocation of PMs for VMs, it can also take care of the allocation of the VMs for the workloads’ submission. Among the various possible techniques for the workloads’ submission, we consider an ordinary allocation where a newly admitted workload is sent to the first available VM. In addition, we developed the priority-based QoS workloads’ allocation [10], where each admitted workload enters with a priority; loads with higher priorities are sent to the VMs with more processing power and tend to be completed earlier.

### 4. Simulation Studies

We conducted a series of simulations to comprehensively study VM migration and QoS-based workload scheduling with network speed, and the impact on makespan, migration, and execution times. For this, we extended CloudSim [25], a well-proven simulator. Specifically, we extended the simulator by implementing the BRS and LA VMs’ scheduling algorithms, the priority based QoS workloads’ scheduling algorithm, and the interaction between these algorithms with CloudSim algorithms.

We performed 30 simulations for each VM’s scheduling algorithm, with all possible power-aware configurations, and for each selected DC bandwidth: 1 Gbps, 10 Gbps, 100 Gbps, 1 Tbps, and 10 Tbps for each DC size. At the end of the process, we calculated the average (AV) and the 95% confidence interval (CI) values of the collected data from the simulations. For all AV measurements presented, the metric used is seconds for makespan and workloads’ execution times.

Simulations were performed with each of the combinations of the possible configurations. We use the acronyms presented in each referred section to describe the algorithms and their settings.

In the following subsections, we first present details on configurations, followed by a comprehensive discussion on results.

#### 4.1 Data Center Configuration

Recall that a DC is a set of PMs, interconnected by a network, available to receive VMs and workloads. We consider workloads to be submitted as cloudlets to the VMs. The number of VMs that the DC receives is six times the number of PMs, and each VM will receive one cloudlet. These numbers were found through experimentation, and were well suited due to the fact that they were sufficiently large to show the behavior of the DCs under high processing loads, but not too big as to derail the simulations performed. For example, a DC with 100 PMs will receive 600 VMs and therefore, 600 cloudlets. Also, our model of load distribution is the bag-of-tasks single-shot, where the entire load to be processed is submitted at the beginning of the simulation, then the DC handles all workloads, the results are returned to a broker, and the DC is turned off. This interval from the beginning, when loads are received, until the end, when the DC is turned off, is called the makespan.

The common configurations for the PMs, in homogeneous and heterogeneous DCs, used in our studies are as follows:

- **RAM** → Each PM has 8 GB of RAM;
- **Disk Space** → Each PM has 1 TB of disk space;
- **DC’s PMs Architecture** → It is defined that all PMs have a x64 based architecture;
- **Operational System** → The chosen OS for the DC’s servers is GNU/Linux;
- **Processing Entities** → Each PM has a processor that can have multiple cores (specified in the section of the DC in question);
- **VM Monitor** → The VM monitor for DC’s PMs is Xen;
- **VM MIPS** → The VMs in the DC have processing capacities of 1000 MIPS, 2000 MIPS, and 3000 MIPS, in a round-robin fashion. For example, in a simulation with 20 VMs, there will be 7 VMs with 1000 MIPS, 7 VMs with 2000 MIPS, and 6 VMs with 3000 MIPS;
- **Cloudlets Size** → The cloudlets in the DC have 30,000, 120,000, 480,000, 1,920,000, 7,680,000 millions of instructions in a round-robin manner. These sizes were chosen to give a generalized processing variability. For example, a cloudlet with 30,000 million instructions in a VM of 3000 MIPS consumes approximately 10 seconds to be processed, representing a relatively light processing, while a cloudlet of 7,680,000 million instructions on an 1000 MIPS VM consumes about 2 hours, representing a relatively heavy processing.
- **Number of Processing Entities for each Cloudlet** → Each cloudlet uses one processing entity;
- **Cloudlets Priority** → Each cloudlet submitted to the
DC has a priority from the following list, in a round-robin fashion: high, normal, low. In other words, 50% of the workloads receive normal priority, while 25% of the cloudlets have a high priority, and 25% of the cloudlets have a low priority;

- Cloudlets File Size → Each cloudlet submitted to the DC, before processing, has 300 bytes (CloudSim standard model);
- Cloudlets Output File Size → Each cloudlet has 300 bytes after processing (using the CloudSim standard model);
- VM Image Size → Each VM has 2.5 GB of image size;
- VM Bandwidth → Each VM receives 8.33% of its total PM bandwidth.

We avoided the use of scarce or very limited hardware, since it could lead to problems in simulations towards the lose of validity by focusing more on the lack of resources than the variation in studied parameters achieved with high speed networks. With respect to the processing capacity of selected processors, we considered for the purpose of simulation, that each transaction of workloads can be completed in one clock cycle; in other words, there is a 1:1 ratio between frequency (MHz) and MIPS.

4.2 Homogeneous and Heterogenous Data Center Configuration

For simulations in homogeneous DCs, the following settings are adopted:

- PM’s MIPS → Each PM in a homogeneous DC has a processor based on Intel Xeon X5667; in other words, a processor with 4 cores operating at 3.06 GHz and therefore, according to simulation considerations, 3,066 MIPS per core;
- Power Consumption of PMs → The power consumption for each PM in a homogeneous DC is 250 watts;
- VM RAM → Each VM have 1 GB RAM.

For simulations in heterogeneous DCs, the following settings were adopted:

- PM’s MIPS → Each PM in a heterogeneous DC has one processor, in a round-robin manner, based in the following models: Intel Xeon E5603, Intel Xeon E7520, Intel Xeon X5667, AMD Opteron 6204, AMD Opteron 6220; having, respectively, 4 cores with 1,600 MIPS each, 4 cores with 1,860 MIPS each, 4 cores with 3,066 MIPS each, 4 cores with 3,300 MIPS each, and 8 cores with 3,000 MIPS each;
- Power Consumption of PMs → Each PM in a heterogeneous DC is 10 PMs @ 1 Gbps. The criteria for selection of these processors was to create a significant disparity in the processing capacity between the PMs, but considering the DCs that use today’s processors.

4.3 Results: Makespan

We now report and analyze the results for the makespan time for the cases we studied. We present in Figs. 1 to 6 the makespan for the VMs scheduling algorithms BRS, MPD, and LA, under the PT, PDT, PTQ, and PDTQ configurations for homogeneous DCs and two different networks speeds, 1 Gbps and 100 Gbps. Similarly, for heterogenous DCs, the results are reported in Figs. 7 to 12.

We observe that the increase in the speed of the networks leads to the reduction in the makespan of the VMs. This can be explained by the fact that the higher speeds of networks are capable of migrating running VMs faster than at lower speeds, thus, causing the potential processing gap during the live migration to be reduced; as a result, there is greater efficiency for the VMs to process their workloads. Moreover, if one considered that live migration takes no processing performance from the VMs, considering the bag-of-tasks scheduling process, networks with higher speed enable the accelerated dispatch of the VMs and the tasks by the bro-

<table>
<thead>
<tr>
<th>Table 2</th>
<th>PM’s configuration in a heterogeneous DC.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server</td>
<td>$b_1$, $b_2$, $b_3$, $b_4$, $b_5$, $b_6$</td>
</tr>
<tr>
<td>MIPS*Cores</td>
<td>1600<em>4, 1860</em>4, 3066<em>4, 3300</em>4, 3000<em>8, 1600</em>4</td>
</tr>
<tr>
<td>Power (watts)</td>
<td>200, 250, 300, 350, 200, 250</td>
</tr>
</tbody>
</table>

Fig. 1 Makespan — Homogeneous DC — 10 PMs @ 1 Gbps.

Fig. 2 Makespan — Homogeneous DC — 10 PMs @ 100 Gbps.
An important point to note here is that the increase in the networks’ speeds benefits some scheduling algorithms more than others. Analyzing the structure of scheduling algorithms, we note that it is a common strategy on energy-aware algorithms to reduce the number of migrations, trying to place VMs on PMs where they are likely to remain longer without the need to migrate. This occurs as shown in Sect. 4.5. Therefore, if the increase in the speed of the networks impacts the rate of migration, and some algorithms...
intentionally tend to make a lower rate of migration, it is expected that they benefited less than the algorithms that perform a large number of migrations.

We next consider the impact on the makespan as a function of the increase in network speed from 1 Gbps to 10 Tbps; see Figs 13 and 14 for homogeneous and heterogeneous DCs, respectively, for all studied algorithms with PDT configuration for data centers with 100 PMs. We found that the reduction in the makespan has an overall behavior of exponential decay as a function of the speed of the network. Notably, there is a bandwidth bound after which the increase in the network’s speed does not reduce the makespan any further. This occurs for both homogeneous and heterogeneous DCs. This behavior pattern also found to be true for all other configurations and all other DCs’ sizes; for brevity, the graphs are not included.

Considering homogeneous DCs and the 95% confidence interval, we can make the following statements for all studied scenarios (algorithms and settings): in DCs with 10 PMs, there was a speedup in the makespan ranging from 3% to 7% when the network’s speed increased from 1 Gbps to 10 Gbps; and from 2% to 4% by a 10 Gbps to 100 Gbps increase, and no significant improvement observed after this point. DCs with 100 PMs had speedups ranging from 4% to 7% by increasing the network speed from 1 Gbps to 10 Gbps, and from 2% to 3% for the 10 Gbps to 100 Gbps enhancement, and there was no significant improvement after this point; 1000 PMs DCs had a speedup from 4% to 8% when the network’s speed goes up from 1 Gbps to 10 Gbps and from 2% to 3% when the speed changed from 10 Gbps to 100 Gbps, also having no significant improvement after this point.

Similarly, considering heterogeneous DCs, also with a 95% confidence interval, we can state the following for all studied scenarios (algorithms and settings): in DCs of 10 PMs, there was a speedup in the makespan ranging from 0% to 8% when the network’s speed increased from 1 Gbps to 10 Gbps, and except for the LA, there was an improvement in the makespan from 0% to 4% when the network’s speed increased from 10 Gbps to 100 Gbps, with no significant improvement for all other cases. In the case of DCs of 100 PMs, from the increase of the network speed from 1 Gbps to 10 Gbps, we observed an improvement ranging from 5% to 8%. Except for the LA, a speed increase from 10 Gbps to 100 Gbps brought an improvement from 2% to 4%, and there was no significant improvements for all other scenarios. For the DCs with 1000 PMs we observed an improvement of 4% to 7% by increasing the speed from 1 Gbps to 10 Gbps, and from 0% to 4% by increasing from 10 Gbps to 100 Gbps, without any significant improvements after this point.

4.4 Results: Cloudlets Execution Times

We next consider the impact on the execution time. We present the cloudlets execution times for the VMs scheduling algorithms BRS, MPD, and LA under the PT, PDT, PTQ, and PDTQ configurations for homogeneous DCs in Figs. 15 to 20 for two different network speeds, 1 Gbps and 100 Gbps. Recall that there are 3 cloudlets priorities: low, normal and high.

We observe that in the case of the workloads, there was a reduction in the times with the faster network’s speed, as
was the case with the makespan. One possible interpretation of this is that because the live migrations reduce the performance of the VMs that process workloads, and these present increased processing times. The value adopted performance degradation seen was 10% (CloudSim standard), i.e., loads during migration are processed with 90% of its VM’s capacity. We note that this impact on individual loads is large enough to also manifest in the global makespan. This reinforces the idea that the makespan can be improved due to a faster migration that allows less time for the live migration process of VMs and workloads, as it reduces the processing time of these loads, whose consolidation enables a reduction in the makespan.

We present in Figs. 21 to 26 the cloudlets execution times for the VMs scheduling algorithms BRS, MPD, and LA, under the PT, PDT, PTQ, and PDTQ configurations for heterogeneous DCs for network speeds 1 Gbps and 100 Gbps. As we can see, there is a notable reduction of execution times when the network's speed increases from 1 Gbps to 10 Tbps.

We next consider the impact on the execution time as a function of the network’s speed. In Fig. 27, we plot workloads’ times depending on the networks’ speed for homogeneous DCs, and Fig. 28 for heterogeneous DCs; in both
cases, we used all studied algorithms with PT configuration for DCs with 100 PMs.

Similar to the exponential decay behavior pattern for the makespan, we found the workloads’ execution times to follow the exponential decay behavior. In other words, the increase in network speed also reduced the execution times of the workloads, but there was an upper limit after which we observed no further reduction. This behavior pattern is the same for homogeneous and heterogeneous DCs for all algorithms with different settings.

4.5 Results: Number of Migrations

Finally, we consider the impact on the number of VM migrations. We present the number of VM migrations for the VMs scheduling algorithms BRS, MPD, and LA under the PT, PDT, PTQ, and PDTQ configurations for homogeneous DCs in Figs. 29 to 34 for network speeds, 1 Gbps and 100 Gbps. For a small number of PMs, the number of migrations decreases as the network speed increases. However, for a large number of PMs, this change is somewhat dependent on the scheduling algorithms of choice.

For heterogeneous DCs, the number of VM migrations for the VMs scheduling algorithms BRS, MPD, and LA under the PT, PDT, PTQ, and PDTQ configurations are shown
in Figs. 35 to 40. We observed that an intriguing and unexpected behavior occurred with respect to the number of VM migrations as a function of the network’s speed. Unlike the makespan and the runtime of cloudlets, we did not observe an exponential decay here. When we analyzed the network’s impact on the number of migrations, we observed that by increasing the speed of the networks, we noted an increase in the number of migrations until a certain upper limit, after which there was a sharp decline and stabilization. We exemplify sampled points from the number of migrations ac-
according to the bandwidth for homogeneous DCs in Fig. 41 and for heterogeneous DCs in Fig. 42. In both cases, we used all studied VMs’ scheduling algorithms, with PT configuration and 100 PMs. We cannot conclude that the higher or lower number of migrations is necessarily good or bad, since they do not necessarily impact the power consumption, or the makespan, or the execution time of individual loads. This behavior pattern is similar for homogeneous and heterogeneous DCs, and all algorithms with all settings. For all algorithms with all the presented configurations, the peak rate of migration occurred between 1 Gbps and 100 Gbps, indicating that the studied scenarios up to 100 Gbps rates did not increase the rate of migration, in contrast, kept this level stable.

Our initial thought was that the reason for this unusual behavior in the number of migrations was due to the dependency on the scheduling algorithm. To understand this, consider, for example, three PMs, PM1, PM2, and PM3, with high, medium and low energy efficiencies, respectively (i.e., PM1 is the most energy efficient machine), and each of these PMs have VMs VM1, VM2, and VM3, respectively allocated in PM1, PM2, and PM3. Consider also that such VMs’ processing workloads, and VM1 finishes first, followed by VM2, and finally VM3. We believe that it is possible to do live migrations, even where VMs during migration can be processed (although it may experience a performance degradation).

Consider first a high migration time. Suppose that VM1 finalizes, then VM2 will be migrated from PM2 to PM1 and next VM3 migrates from PM3 to PM2. If during the migration of VM3 to PM2, VM2 completes its work, then VM3 may not be immediately migrated to PM1 (the most efficient in energy PM) since it is still migrating to PM2. Only after the migration of VM3 to PM2 completes, VM3 can be migrated to PM1, making this remaining time of the migration to PM2 rendering the system inefficient in energy, makespan and execution time of workloads, since
it will need to keep the PM3 and PM2 turned on when VM3 should be being migrated to PM1. This migration can also lead to degradation in performance in processing VM3. More than that, this long consumed migrations implies fewer migrations.

Consider next, for the same previous scenario, the migration times are low, or have high speed networks in DCs. Suppose that after completion of VM1, VM2 will be migrated from PM2 to PM1 and followed by VM3 from PM3 to PM2. The VM3 migration time for PM1 in this case will be substantially reduced, since the high speed networks will make migrating of VM3 to PM2 to occur at an accelerated rate. In other words, high speed networks favor faster migrations, which imply a faster availability of the PMs to the VMs.

Networks with high speeds increase the availability of the PMs for the VMs, but they also reduce the migration time, which enable the VMs to stay longer on the PMs and consume less time for live migration. This makes it possible for a better handling workloads for the VMs. Now, the following question remains to be answered: for the evaluated scenarios, why does the number of migrations drastically reduce and stabilize after a certain point?

Two hypotheses seem plausible for this question: (I) due to the greater availability of PMs, the scheduling algorithm may be making optimized allocations; or (II) the processing time of the VMs, by not realizing the improvement in the performance of the migration time, improves so much that it causes both the workloads and the VMs to be terminated early with a greater number of unnecessary migration of VMs. To answer this question, we implemented the continued migration algorithm (round-robin – RR) and considered a 10% performance degradation in migration of VMs (standard value adopted in CloudSim). This algorithm aims to choose PMs for VMs in a circular fashion, and as a result remain migrating VMs throughout their lifetime. If the number of migrations continue to rise, this indicates that the hypothesis (I) is correct; otherwise, it is an indication that the hypothesis (II) is correct. Figure 43 presents the behavior of the number of migrations as a function of network speed plotted for the RR-PT algorithm for homogeneous DCs. As we can see, there are increasing numbers of VM migrations, without decay, and stabilization at the end. This does not indicate a decay in the number of VMs processing loads, suggesting that hypothesis (II) is not true.

That is, we have an indication that the change in the number of VM migrations is much more dependent on the algorithm used in the VMs’ scheduling process than the PMs’ processing availability for VMs called out by a faster migration of these VMs.

5. Conclusion and Future Work

In this paper, we present and analyze the impact of the data center network’s speed in DCs operating in the cloud with respect to the number of migrations, makespan, and execution times of individual workloads. For this study, we enhanced CloudSim by modifying the VMs’ scheduling algorithms BRS and MPD, in order to not work with the workloads’ ordinary scheduling algorithm, but also with the priority QoS based scheduling employed in the enhanced LA [10].

We found that for all evaluated scenarios, increased bandwidth reduces the makespan and the workloads’ execution times with an exponential decay behavior as the network speed increases. We also found that the live migration mechanism usually involves a processing performance degradation, but this degradation affects the makespan less than the used scheduling algorithm.

We also found that the rate of migration varies according to the employed scheduling algorithm, but the general behavior is an increase in the number of migrations until it hits a peak, usually between 1 Gbps and 100 Gbps, followed by a sharp fall until this number reaches a stable point. However, we noticed that the number of migrations has little impact on the important factors such as energy consumption or processing times.

In this work, the scheduling algorithm was not bandwidth-aware; a possible future research direction would be to consider bandwidth-aware scheduling algorithms. Secondly, although we highlighted the impact of the network’s speed, we do not consider the actual data center topology. Thus, a second possible direction would be to develop topology-aware schemes. We intend to pursue both these directions.

Acknowledgement

We thank CAPES, CNPq, and FAPESP for grants that supports this work. Deep Medhi was supported by Brazil’s Science Without Borders Program, funded through CNPq.

References


Daniel G. Lago is an Assistant Professor at the Federal Center for Technological Education of Minas Gerais (CEFET-MG), Brazil. He received his MSc in Computer Science from UNICAMP in 2012. He is currently a doctoral student in Institute of Computing of the UNICAMP.

Edmundo R. M. Madeira is a Full Professor at the University of Campinas (UNICAMP), Brazil. He received his Ph.D. in Electrical Engineering from UNICAMP in 1991. He has published over 150 papers in national and international conferences and journals. He was the General Chair of the 7th Latin American Network Operation and Management Symposium (LANOMS’11), and he was a Technical Program Co-chair of the IEEE LatinCloud’12. He is a member of the editorial board of Journal of Network and Systems Management (JNSM), Springer. His research interests include network management, future Internet and cloud computing.

Deep Medhi is a Curators’ Professor in the Department of Computer Science & Electrical Engineering at the University of Missouri-Kansas City, USA, and a honorary professor in the Department of Computer Science & Engineering at the Indian Institute of Technology-Guwahati, India. He obtained B.Sc. in Mathematics from Cotton College, Gauhati University, India, M.Sc. in Mathematics from the University of Delhi, India, and his M.S. and Ph.D. in Computer Sciences from the University of Wisconsin-Madison, USA. Prior to joining UMKC in 1989, he was a member of the technical staff at AT&T Bell Laboratories. He is the Editor-in-Chief of Springer’s Journal of Network and Systems Management, and is on the editorial board of IEEE/ACM Transactions on Networking, IEEE Transactions on Network and Service Management, and IEEE Communications Surveys & Tutorials. He has published over 125 papers, and is co-author of the books, Routing, Flow, and Capacity Design in Communication and Computer Networks (2004) and Network Routing: Algorithms, Protocols, and Architectures (2007), both published by Morgan Kaufmann Publishers/Elsevier Science.