# Server Consolidation with Migration Control for Virtualized Data Centers

Tiago C. Ferreto[1], Marco A. S. Netto,
Rodrigo N. Calheiros, and César A. F. De Rose

*Pontifical Catholic University of Rio Grande do Sul (PUCRS)*
*Faculty of Informatics*
*Porto Alegre, Brazil*

**Abstract**

Virtualization has become a key technology for simplifying service management and reducing energy costs in data centers. One of the challenges faced by data centers is to decide when, how, and which virtual machines (VMs) have to be consolidated into a single physical server. Server consolidation involves VM migration, which has a direct impact on service response time. Most of existing solutions for server consolidation rely on eager migrations, which try to minimize the number of physical servers running VMs. These solutions generate unnecessary migrations due to unpredictable workloads that require VM resizing. This paper proposes an LP formulation and heuristics to control VM migration, which prioritize virtual machines with steady capacity. We performed experiments using TU-Berlin and Google data center workloads to compare our migration control strategy against existing eager-migration-based solutions. We observed that avoiding migration of VMs with steady capacity reduces the number of migrations with minimal penalty in the number of physical servers.

*Keywords:*
Server Consolidation, Virtualization, Data Centers, Migration Control

[1]Corresponding author: tiago.ferreto@pucrs.br

## 1. Introduction

Virtualization has become a fundamental asset in several areas of Computer Science. Despite being an old concept (initially used by IBM 370 mainframes [1]), virtualization assists current organizations in dealing with problems such as unpredicted demand of computing resources, high management and energy costs, and security. As a consequence, there has been an increasing number of commercial and open-source software products (e.g. Xen [2], VMWare [3], OpenVZ [4], and Microsoft Hyper-V [5]) and investments of major hardware companies to provide better support for virtualization in their products (e.g. Intel VT [6] and AMD Virtualization [7]).

One of the benefits of virtualization is the possibility of gathering several virtual machines (VMs) into a single physical server. This process, known as *server consolidation*, is used by data centers to increase resource utilization and reduce electric power consumption costs. Server consolidation is particularly important when user workloads are unpredictable and need to be revisited periodically. Whenever a user demand changes, VMs can be resized and migrated to other physical servers if necessary.

Our research hypothesis is that a more conservative approach can be used to provide steadier performance guarantees. Our proposal extends current server consolidation solutions by including constraints to define that virtual machines with steady usage are not migrated and virtual machines with variable capacity can be migrated to reduce the number of required physical servers. We call this new approach as *dynamic consolidation with migration control*. Thus, the contributions of this paper are (i) an LP formulation and heuristics to control VM migration, which prioritize virtual machines with steady capacity; and (ii) an extensive evaluation based on TU-Berlin and Google data center workloads that compares our approach with existing ones and corroborates our research hypothesis.

The obtained results are encouraging, since they show that a more conservative migration approach can reduce the number of migrations with minimal penalty in the number of physical servers compared to eager-migration-based solutions [8, 9, 10, 11, 12, 13, 14, 15]. This research thus has a direct impact on service response times and cost saving in data centers.

## 2. Related work

There are several research groups in both academia and industry working on server consolidation [8, 9, 10, 11, 12, 13, 14, 15]. This section presents studies and systems from some of these groups.

Khanna *et al.* [8] proposed a dynamic management algorithm, which is triggered when a physical server becomes overloaded or underloaded. The main goals of their algorithm are to: i) guarantee that SLAs are not violated (SLAs are specified considering mainly response time and throughput); ii) minimize migration cost; iii) optimize the residual capacity of the system; and iv) minimize the number of physical servers used.

Another project that considers SLAs comes from Bobroff *et al.* [14], who proposed and evaluated a dynamic server consolidation algorithm to reduce the amount of required capacity and the rate of SLA violations. The algorithm uses historical data to forecast future demand and relies on periodic executions to minimize the number of physical servers to support the virtual machines.

Speitkamp and Bichler [9, 10] described linear programming formulations for the static and dynamic server consolidation problems. They also designed extension constraints for limiting the number of virtual machines in a physical server, guaranteeing that some virtual machines are assigned to different physical servers, mapping virtual machines to a specific set of physical servers that contain some unique attribute, and limiting the total number of migrations for dynamic consolidation. In addition, they proposed an LP-relaxation based heuristic for minimizing the cost of solving the linear programming formulations.

Mehta and Neogi [11] introduced the ReCon tool, which aims at recommending dynamic server consolidation in multi-cluster data centers. ReCon considers static and dynamic costs of physical servers, the costs of VM migration, and the historical resource consumption data from the existing environment in order to provide an optimal dynamic plan of VMs to physical server mapping over time. Similarly, Verma *et al.* [13] developed the pMapper architecture and a set of server consolidation algorithms for heterogeneous virtualized resources. The algorithms take into account power and migration costs and the performance benefit when consolidating applications into physical servers.

Wood *et al.* [12] developed the Sandpiper system for monitoring and detecting hotspots, and remapping/reconfiguring VMs whenever necessary. In order to choose which VMs to migrate, Sandpiper sorts them using a volume-to-size ratio (VSR), which is a metric based on CPU, network, and memory loads. Sandpiper tries to migrate the most loaded VM from an overloaded physical server to one with sufficient spare capacity.

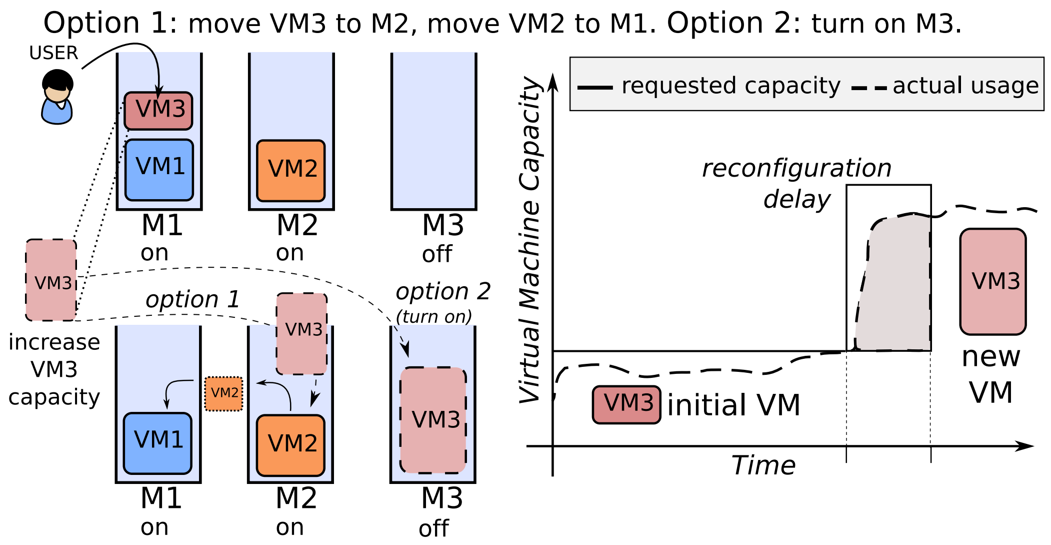


Figure 1: Migration options: tradeoff between number of migrations (option 1) and number of physical servers (option 2).

Different from the existing approaches for server consolidation, we consider the fact that some VMs have steady load, and hence, users of these VMs should experience such stability. Besides, by keeping those VMs on their original physical servers, the number of migrations is minimized with minor penalty on the number of physical servers.

## 3. Server consolidation with migration control

Server consolidation aims at minimizing the number of physical servers required to host a group of virtual machines. This problem can be mapped to the multidimensional bin-packing problem [16]. In this problem, the goal is to map several items, where each item represents a tuple containing its dimensions, into the smallest number of bins as possible. In our case, we consider each virtual machine as an item and the dimensions as its capacities, and the goal is to minimize the number of physical servers that must be used to place all virtual machines, respecting physical servers capacities. This problem is NP-complete and is usually solved through linear programming (LP) or heuristics.

The consolidation process can be performed in a single step using the peak load demands of each workload to configure virtual machine capacities, or reevaluating periodically the workload demand in each virtual machine and performing the required configuration changes. The first approach is known as *static consolidation* since the virtual machines stay in the same physical servers during their whole lifetime. The utilization of the peak load demand ensures that the virtual machine does not overload, however it can also lead to idleness since the workloads can present variable demand patterns. The second approach is known as *dynamic consolidation* and usually results in better consolidation, since it dynamically changes virtual machine capacities according to the current workload demands. However, it may require migrating virtual machines between physical servers in order to: (i) pull out physical servers from an overloaded state when the sum of virtual machines capacities mapped to a physical server becomes higher than its capacity; (ii) or to turn off a physical server when the virtual machines mapped to it can be moved to other physical servers.

The dynamic consolidation problem involves a tradeoff between migrating virtual machines and reducing the number of physical servers to host VMs (Figure 1). Current techniques for virtual machine migration [17] enable the transference of virtual machines between physical servers with negligible downtime, maintaining sessions active and hence being imperceptible for users. However, it usually results in a degradation of virtual machine's performance [18]. This performance penalty can be considered admissible when the virtual machine capacity is being increased, since it will eventually result in a better performance after the migration, or when the capacity is being decreased as the performance may not affect the current workload demand. Nevertheless, workloads with steady capacity should not suffer from these performance variations since they only harm the performance on the workload execution due to the migration cost.

In the following sections, we present a linear programming formulation and heuristics to the static and dynamic consolidation problems, and show the modifications required to include a constraint to control the migration of virtual machines with steady capacity requirements. We define the resulting problem as *dynamic consolidation with migration control*.

## *3.1. Linear programming formulation*

The input parameters and variables used in the linear programming formulation are presented in Table 1. Sets $P$, $V$, and $R$ represent respectively physical servers, virtual machines, and resources such as CPU, memory, and

Table 1: Parameters and variables for the server consolidation problem.

| **Parameters** | |
|---|---|
| $P$ | set of physical servers |
| $V$ | set of virtual machines |
| $R$ | set of resources (cpu, memory, network) |
| $u_{vr} \in \mathbb{R}$ | utilization for virtual machine $v \in V$ of resource $r \in R$ |
| $c_{pr} \in \mathbb{R}$ | capacity for physical server $p \in P$ of resource $r \in R$ |
| **Variables** | |
| $y_p \in \{0, 1\}$ | equals to 1 if physical server $p \in P$ is used, 0 otherwise |
| $x_{pv} \in \{0, 1\}$ | equals to 1 if virtual machine $v \in V$ is mapped to physical server $p \in P$, 0 otherwise |

network. Arrays $u$ and $c$ represent the virtual machine demands and physical server capacities for each resource available. After solving the problem, variables $y$ and $x$ provide the servers used and the mappings of each virtual machine to a physical server.

The LP formulation is presented in Figure 2. The objective function aims at minimizing the number of required physical servers. The constraints guarantee that each virtual machine is mapped to a single physical server and the virtual machine demands allocated in each physical server do not overload its capacity. The LP formulation can be used both for the static and dynamic consolidation approaches. In the static consolidation, array $u$ contains the peak demand of each virtual machine, whereas for dynamic consolidation approach, the problem is solved in several time steps (consolidation steps). In each step, array $u$ contains the virtual machine demand at that specific consolidation time.

When dynamic consolidation is used, each consolidation step may result in a different mapping of virtual machines to physical servers, which requires the migration of these virtual machines to different physical servers. The number of required servers can change after each consolidation step as well. In this case, idle servers are turned off or put in a low energy mode in order to reduce the power consumption. When a physical server is required again, it is turned on.

In order to avoid penalizing virtual machines with steady demands, i.e. keeping them in the same physical servers without migrating, we added pa-

$$\min \sum_{p \in P} y_p$$

$$\begin{aligned} \sum_{p \in P} x_{pv} &= 1 && \forall v \in V \\ \sum_{v \in V} u_{vr} x_{pv} &<= c_{pr} y_p && \forall p \in P, \forall r \in R \end{aligned}$$

Figure 2: LP formulation for the server consolidation problem.

Table 2: New parameters included in the dynamic consolidation with migration control problem.

| **Parameters** | |
|---|---|
| $m_v \in \{0,1\}$ | equals to 0 if there is no change in virtual machine $v$'s demand, 1 otherwise |
| $x'_{pv} \in \{0,1\}$ | mapping of the previous consolidation step, equals to 1 if virtual machine $v \in V$ is mapped to physical server $p \in P$, 0 otherwise |

rameters and a constraint to the original problem. The resulting problem is defined as *dynamic consolidation with migration control.* The parameters included are presented in Table 2. Parameter $m$ differentiates the virtual machines that change or not change their capacity in the current consolidation step, and parameter $x'$ provides the previous mapping of virtual machines to physical servers. The constraint included in the LP formulation is presented in Figure 3. If a virtual machine $v$ requires changing its capacity, then $m_v$ is equal to 1, and the difference between the current and the previous mapping is in the interval $[-1, 1]$. Otherwise, the virtual machine $v$ has a stationary capacity and $m_v$ is equal to 0. Therefore, the difference between current and previous mappings must be equal to 0, i.e. arrays $x_{pv}$ and $x'_{pv}$ for virtual machine $v$ is the same for all physical servers $p \in P$.

This mapping process is executed for each consolidation step and saved to be used in the next mapping ($x'$ parameter). Considering that at the beginning no virtual machine is mapped, the additional constraint and parameters are used only after the completion of first mapping.

$$-\mathrm{m}_v \leq (x_{pv} - x'_{pv}) \leq \mathrm{m}_v \quad \forall p \in P, \forall v \in V$$

Figure 3: Additional constraint for the dynamic consolidation with migration control problem.

After executing the LP problem with the new constraint, variables $y$ and $x$ provide the servers used and the mappings of each virtual machine to physical server, but now guaranteeing that virtual machines with steady capacity remain in the same physical servers.

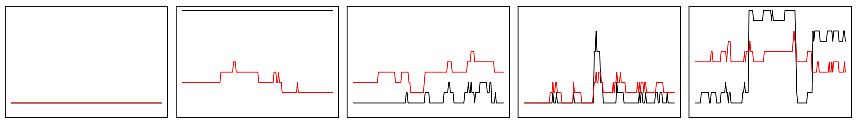
Table 3: Details of TU-Berlin workload groups.

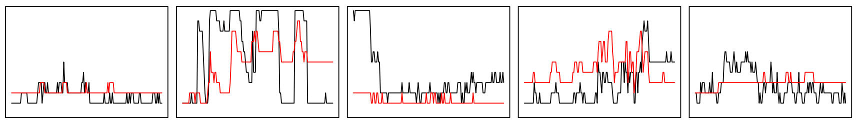
| | **Number of traces** | **Avg. variability index** | **Avg. CPU utilization (%)** | **Avg. memory utilization (%)** |
|---|---|---|---|---|
| **Group 1** | 43 | 0.17 | 25.2 % | 28.36 % |
| **Group 2** | 61 | 0.41 | 32.37 % | 39.39 % |
| **Group 3** | 36 | 0.63 | 47.28 % | 48.67 % |

## *3.2. Heuristics*

The multidimensional bin-packing problem can also be solved using heuristics. Although they do not guarantee an optimal solution, the required time to obtain a feasible solution is much shorter than LP, specially when dealing with large scenarios. We performed minor modifications in some heuristics commonly used for this problem in order to guarantee that in each consolidation step, virtual machines with steady demand are not migrated. The heuristics used are: first-fit decreasing (FFD), best-fit decreasing (BFD), worst-fit decreasing (WFD), and almost worst-fit decreasing (AWFD) [16].

In the original version of all heuristics, virtual machine demands are sorted in decreasing order. Each virtual machine demand is a tuple consisting of its resources (CPU, memory, and network) and is sorted respecting a lexicographic order. After that, the mapping of each virtual machine is performed according to the heuristic definition. In the FFD heuristic, the virtual machine is mapped to the first physical server with available capacity. In the BFD heuristic, the virtual machine is mapped to the physical server


(a) Group 1: Low Variability


(b) Group 2: Medium Variability

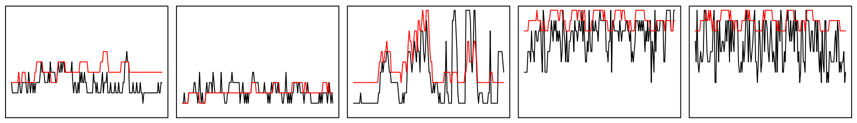

(c) Group 3: High Variability

Figure 4: Example of TU-Berlin workloads from steady to variable CPU and memory loads.

that leaves the least left over space after the mapping between all available physical servers. In the WFD heuristic, the virtual machine is mapped to the physical server that leaves the largest left over space after the mapping between all available physical servers. In the AWFD heuristic, the virtual machine is mapped to the second best physical server that leaves the largest left over space after the mapping between all available physical servers.

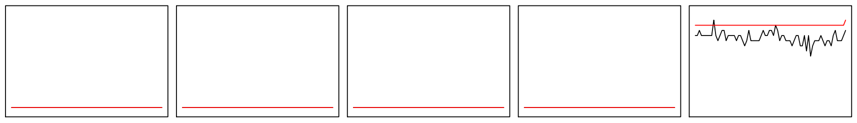
In the static consolidation, the mapping process is performed only once using virtual machine peak demands. In the dynamic consolidation, the algorithm is repeated periodically for each consolidation step using virtual machine demands at that particular moment. In the dynamic consolidation with migration control, the heuristics are modified through the inclusion of a preceding phase. This phase performs the following steps:

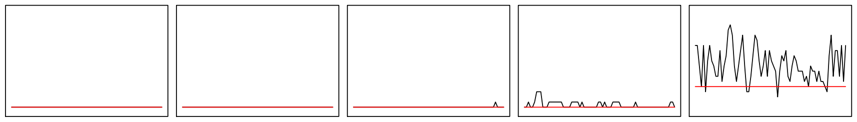
1. maps the virtual machines that do not change their resource demands to the previously mapped physical servers;
2. sorts physical servers increasingly according to their capacities respecting a lexicographic order.

After this phase, the virtual machines remaining to be mapped are those with some variation in their capacities, and they are mapped using the original heuristics.

Table 4: Details of Google workload groups.

| | Number of traces | Avg. variability index | Avg. CPU utilization (%) | Avg. memory utilization (%) |
|---|---|---|---|---|
| **Group 1** | 264 | 0.02 | 6.08 % | 5.79 % |
| **Group 2** | 184 | 0.16 | 9.48 % | 11.72 % |
| **Group 3** | 125 | 0.22 | 9.53 % | 19.03 % |


(a) Group 1: Low Variability


(b) Group 2: Medium Variability

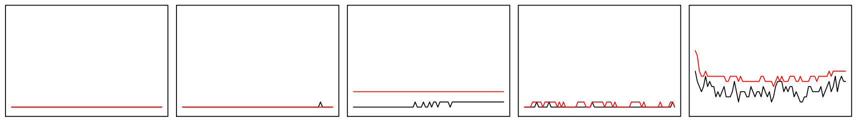

(c) Group 3: High Variability

Figure 5: Example of Google workloads from steady to variable CPU and memory loads.

## 4. Evaluation

The evaluation of the LP and heuristic-based solutions for the dynamic consolidation with migration control problem was performed using two different workloads. The first workload is composed of traces from servers of the Technical University of Berlin (TU-Berlin), which are normally used by researchers and students to execute computational experiments. The second workload is composed of traces from Google servers. Both workloads present periodic samples of CPU and memory utilization for servers (TU-Berlin workload) or jobs (Google workload). In order to evaluate the solutions with more workloads with different characteristics, each workload was divided in three groups containing a subset of the traces from the original workload. The goal was to obtain workloads with different variability characteristics. In order to distinguish each workload regarding its variability, we defined the *Variability index* metric. The variability index of a server trace is the ratio of consolidation steps where a capacity change in the virtual machine is required during its execution. The variability index varies from 0, when the virtual machine keeps with the same capacity during the whole execution, till 1, when the virtual machine capacity changes in every consolidation step. Tables 3 and 4 present the groups derived from each workload with the number of traces, average variability index considering all traces, and average CPU and memory utilization.

**Metrics:** For each workload group the metrics measured are the average number of physical servers required to process the workload and the average number of migrations required in all consolidation steps with respective standard deviations. In each experiment the following analysis are performed: (i) evaluation of the impact of using a more conservative approach (dynamic consolidation with migration control) in the additional number of required physical servers and average number of migrations required per time step, (ii) comparison of the solutions obtained with the LP and heuristics-based solutions, and (iii) verification of the impact in the solutions obtained with workloads with different degrees of variability.

**Baseline for comparison:** In order to evaluate the benefits of our approach, we used consolidation strategies that do not differentiate between virtual machines with steady and variable demand, which is common on various existing solutions [8, 9, 10, 11, 12, 13, 14, 15]. For all approaches a heuristic and an LP-based version were implemented. The LP-based version was implemented to generate more optimized results, since it performs its search through solution paths that heuristics usually do not reach. The approaches implemented were:

- **Static consolidation:** This common approach uses the peak utilization of each server trace and uses this value to perform the consolidation. In this case, no migrations are performed since all demand changes fit in the pre-allocated virtual machine capacity.

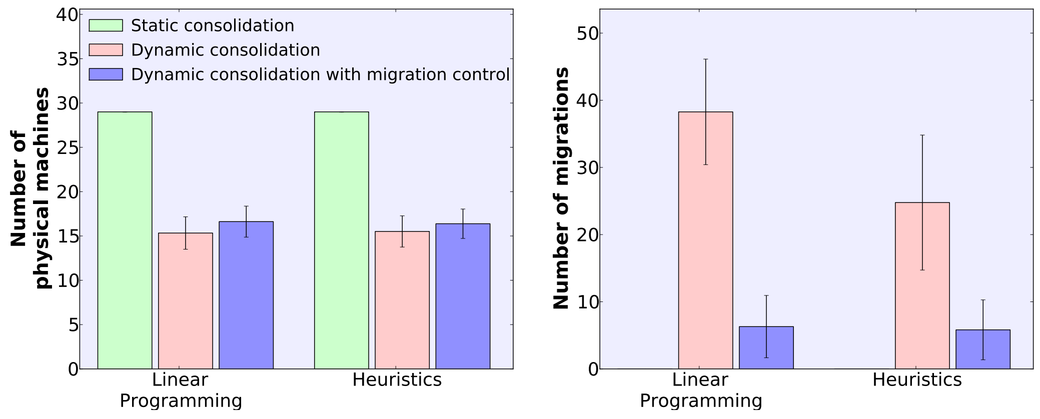
- **Dynamic consolidation:** The dynamic consolidation approach is also well known and consolidates virtual machines periodically based on current demands. The sum of the virtual machines demands can get higher than the physical server capacity and, therefore, migrations might be required.

- **Dynamic consolidation with migration control:** The dynamic consolidation with migration control is proposed in this work and aims at guaranteeing more stability for virtual machines with steady demands. Similarly to the dynamic consolidation approach, it also periodically consolidates the virtual machines in order to reduce the number of required physical servers, but ensures that virtual machines with steady demand do not migrate.

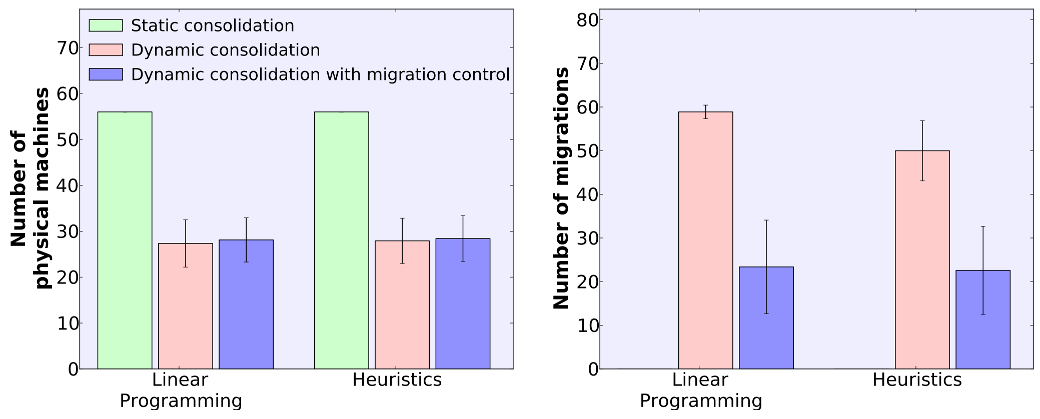
The experiments were performed using LP and heuristic-based versions of the static consolidation, dynamic consolidation, and dynamic consolidation with dynamic control problems resulting in a total of 90 experiments. The static consolidation implementation uses the peak demand along all consolidation steps for each server trace. The LP formulation was implemented using the Zimpl language [19] and executed using the SCIP [20] solver. The heuristics were implemented using the Python language. All experiments were performed on a Intel Core 2 Duo processor with 2.4 GHz and 4 GBytes of memory. The physical infrastructure simulated in the experiments is composed of an unlimited number of physical servers, each one with CPU and memory capacities of 100.

## *4.1. TU-Berlin workload*

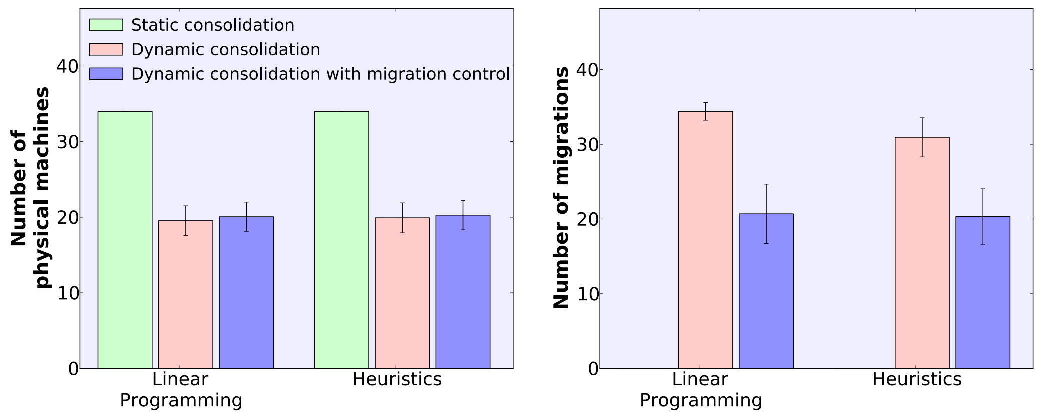
The TU-Berlin workload is composed of 140 server traces, each one containing samples of CPU and memory utilization per hour in a week period, totalizing 168 samples per trace. The workload was divided in 3 groups according to the variability index of each trace. Details about each group are presented in Table 3. Group 1 presents the lower variability, containing 43 traces with variability index ranging from 0 to 0.3. Group 2 is the larger group with 61 traces with variability index ranging from 0.3 to 0.5. Group 3 contains traces with higher variability (variability index from 0.5 to 1.0) and is the smallest group with 36 traces. Each group also presents different values of average CPU and memory utilization. Group 1 presents the lowest average CPU and memory utilization, whereas Group 3 presents the highest averages of utilization. Figure 4 shows a subset of the traces of each group ordered according to its variability index. In each graph, the black line represents the CPU utilization and the red line represents the memory utilization in percentages.



(a) Group 1



(b) Group 2



(c) Group 3

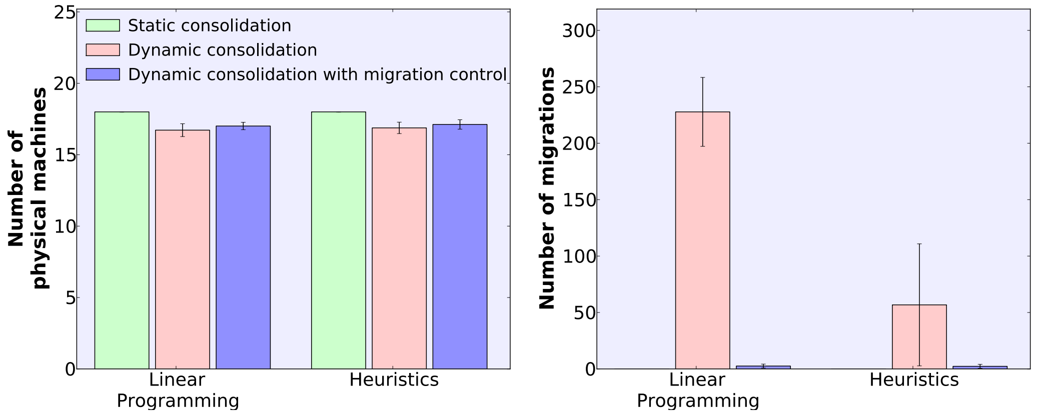
Figure 6: Number of physical servers and migrations as a function of the migration policy for TU-Berlin workloads.

Figure 6 presents the results obtained with TU-Berlin workload groups. The error bars on the lower graphs present the average number of required physical servers for each consolidation strategy (static, dynamic and dynamic with migration control) for each group with corresponding standard deviations. The error bars on the upper graphs present the average number of required migrations per consolidation step corresponding to the algorithms of the lower graphs. The results presented in the graphs for heuristics are based on the best solution found among all heuristics (FFD, BFD, WFD, and AWFD), which minimizes the average number of required physical servers.

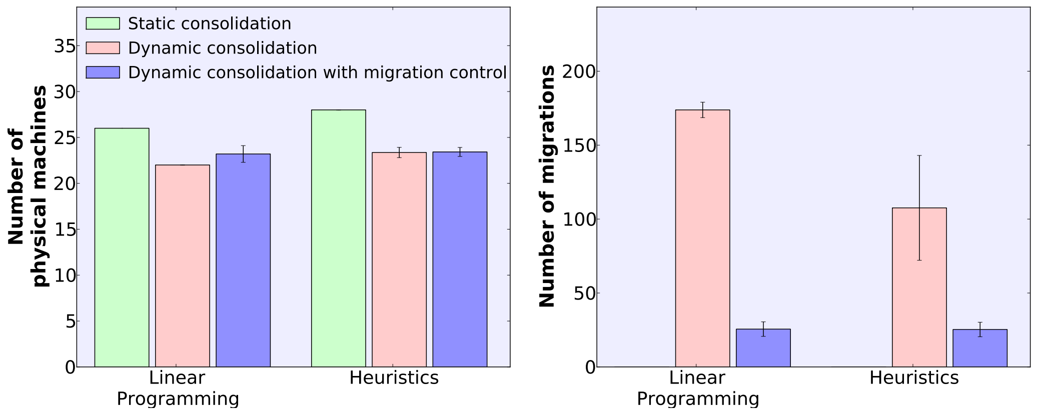
The results obtained show that the average number of migrations for dynamic consolidation with migration control are considerably smaller in comparison to dynamic consolidation. Based on the results obtained with LP, the relation between the average number of migrations between the two approaches is directly related to the variability index. A higher variability index results also in a higher amount of eligible migrations (in the dynamic consolidation with migration control), therefore the average number of migrations for both approaches tend to become similar in workloads with higher variability. However, the effect on the number of physical servers is minimal (approximately 1 additional machine in all groups). Both dynamic consolidation and dynamic consolidation with migration control approaches enable the utilization of a smaller number of physical servers in comparison to static consolidation, even though they require several migrations.

In the dynamic consolidation approach, in each time step approximately 90% of the virtual machines are migrated using the LP approach. In the dynamic consolidation with migration control this value is limited by the average variability index of the workload. The LP and heuristic approaches tend to produce more similar results between themselves in the dynamic consolidation with migration control than with dynamic consolidation.

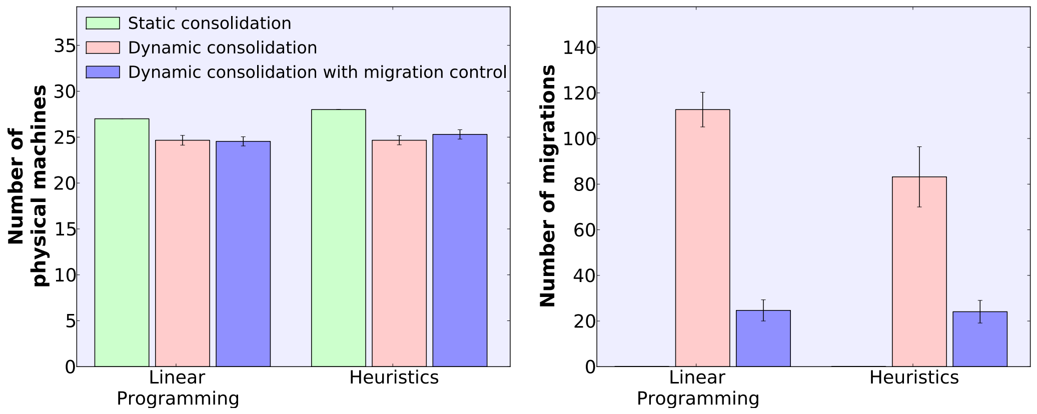
Besides, the solutions with LP also tend to be obtained faster using dynamic consolidation with migration control compared to dynamic consolidation. In the dynamic consolidation approach, the solution of the LP problem for all time steps was obtained within approximately 12 hours for each workload group. In the migration control approach, the total time was approximately 5 minutes for Group 1, 5 hours for Group 2, and 6 hours for Group 3. Therefore, the inclusion of our conservative constraint, despite of increasing the LP problem with more variables and constraints, allows a quicker solution than the original LP formulation. As expected, the time required by the heuristics to compute the mapping is much smaller than the time required



(a) Group 1



(b) Group 2



(c) Group 3

Figure 7: Number of physical servers and migrations as a function of the migration policy for Google workloads.

by LP, approximately five seconds.

## *4.2. Google workload*

The Google workload[2] is composed of job traces from Google's clusters. The workload comprise 9218 jobs, each job with one or more tasks, and contains samples of CPU cores and memory utilization per task during a period of approximately 7 hours sampled in 5-minute intervals. In order to create different workload groups, the tasks of each job were aggregated summing their CPU cores and memory utilization in each time step. The jobs that did not present samples in all time steps were filtered out resulting in a total of 264 jobs. A linear transformation was performed in the workload in order to result in a minimum utilization of CPU and memory of 5% and maximum utilization of 90%. Group 1 contains all resulting jobs and presents the lowest average variability index and utilization of CPU and memory. Group 2 was created by removing 15% of the jobs with lower aggregate CPU and memory utilization and 15% of the jobs with higher aggregate CPU and memory utilization from Group 1. The same process was used to create Group 3 from the jobs of Group 2. Group 3 presents the highest average variability index and utilization of CPU and memory in comparison to the other groups. Details about each group are presented in Table 4. Figure 5 shows a subset of the traces of each group ordered according to their variability index. In each graph, the black line represents the CPU utilization and the red line represents the memory utilization in percentages. The groups contain more server traces than TU-Berlin workload groups, however they also present lower variability and average utilization of resources.

Figure 7 presents the results obtained with the Google workload groups. Despite the differences between the workloads, the results obtained were quite similar. Due to the lower variability, the difference in the average number of migrations between dynamic consolidation and dynamic consolidation with migration control is higher. The LP and heuristic-based implementations also tend to present more similar results between themselves in the dynamic consolidation with migration control than in the dynamic consolidation approach. Even when using a workload with extremely low variability (Group 1 - variability index of 2%), the number of additional physical servers required is still insignificant. Due also to the low variability and low utilization of re-

[2]Google Workloads: `http://code.google.com/p/googleclusterdata/`

sources, the difference between static consolidation and the other approaches is quite small.

## 5. Conclusions

Server consolidation allows data centers to increase resource utilization and reduce electric power consumption by gathering multiple virtual machines into physical servers. The consolidation process may involve resizing of virtual machines and their migration between physical servers. This process may then reduce the Quality-of-Service that users perceive from the environment; and in worst scenarios violate SLAs. SLA (Service Level Agreement) is a formal contract between a consumer and a provider which defines the quality of the service that should be delivered to the consumer, and the penalty to the provider if it is not respected.

In order to minimize the migration problem in virtualized data centers, this paper presented an LP formulation and heuristics to control VM migration. The main difference from existing solutions on server consolidation is that our heuristics prioritize virtual machines with steady capacity. This is possible by including constraints to define that virtual machines with steady usage are not migrated and virtual machines with variable capacity can be migrated to reduce the number of required physical servers. We evaluated the heuristics using both static and dynamic consolidation. Based on TU-Berlin and Google workloads, our main finding from this work is that: avoiding migration of VMs with steady capacity reduces the number of migrations with minimal penalty for the number of physical servers. The results obtained from these workloads are promising as data centers can easily benefit from our migration control mechanism since little modification is required to incorporate it into a standard resource management system.

Current commercial systems, such as VMware Distributed Power Management (DPM) [21] and Citrix XenServer Workload Balancing Power Management [22], provide automatic dynamic consolidation, however using a different approach than the one presented in this paper. These systems detect variations in VM resources utilization and dynamically migrate virtual machines to other servers to guarantee that the VM resources demands are satisfied while using only the necessary amount of servers, and when there is a reduction in resource utilization the VMs are consolidated in fewer servers again. These systems focus on reducing the number of physical servers required, therefore, when they find a server with a single VM that can be accommodated in another server with other VMs, they will migrate the VM and turn off the server. On the contrary, the algorithm proposed in this paper will only enable the migration if the VM requires a change in its capacity, otherwise it will remain in the same server.

Despite the difference, the algorithm can be easily implemented in commercial systems, such as VMware and Citrix, through the available open management interfaces provided by these systems. VMware provides APIs using several languages and interfaces that can be used to manage virtual machines in the infrastructure. The alerts and notifications provided by VMware vCenter Server can be used to identify situations where a change in the VMs mapping should be favorable and trigger automated workflows. The proposed approach could be implemented as a consolidation workflow which could be execute periodically in the virtualization infrastructure. A similar approach could be implemented in Citrix XenServer using its command line interface.

## References

[1] R. J. Creasy, The origin of the vm/370 time-sharing system, IBM Journal of Research and Development 25 (5) (1981) 483–490.

[2] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauery, I. Pratt, A. Warfield, Xen and the art of virtualization, in: Proceedings of 19th ACM Symposium on Operating Systems Principles (SOSP'03), 2003.

[3] I. VMware, Vmware - virtualization software http://www.vmware.com/ (2009).

[4] SWsoft, Welcome to openvz - server virtualization open source project, http://openvz.org/ (2009).

[5] Microsoft, Microsoft Hyper-V Server http://www.microsoft.com/hyper-v-server (2009).

[6] I. Corporation, Virtualization technologies from intel, http://www.intel.com/technology/virtualization/ (2009).

[7] I. Advanced Micro Devices, Amd's virtualization solutions, http://enterprise.amd.com/us-en/AMD-Business/Business-solutions/Consolidation/Virtualization.aspx (2009).

[8] G. Khanna, K. Beaty, G. Kar, A. Kochut, Application performance management in virtualized server environments, in: Proceedings of the 10th IEEE/IFIP Network Operations and Management Symposium (NOMS'06), 2006.

[9] M. Bichler, T. Setzer, B. Speitkamp, Capacity planning for virtualized servers, in: Proceedings of the 16th Annual Workshop on Information Technologies and Systems (WITS'06), 2006.

[10] B. Speitkamp, M. Bichler, A Mathematical Programming Approach for Server Consolidation Problems in Virtualized Data Centers, IEEE Transactions on Services Computing.

[11] S. Mehta, A. Neogi, ReCon: A Tool to Recommend Dynamic Server Consolidation in Multi-Cluster Data Centers, in: Proceedings of the IEEE Network Operations and Management Symposium (NOMS'08), Salvador, Bahia, 2008.

[12] T. Wood, P. J. Shenoy, A. Venkataramani, M. S. Yousif, Sandpiper: Black-box and gray-box resource management for virtual machines, Computer Networks 53 (17) (2009) 2923–2938.

[13] A. Verma, P. Ahuja, A. Neogi, pMapper: Power and migration cost aware application placement in virtualized systems, in: Proceedings of the ACM/IFIP/USENIX 9th International Middleware Conference, 2008.

[14] N. Bobroff, A. Kochut, K. A. Beaty, Dynamic placement of virtual machines for managing sla violations, in: Proceedings of the 10th IFIP/IEEE International Symposium on Integrated Network Management (IM'07), 2007.

[15] J. Rolia, A. Andrzejak, M. Arlitt, Automating enterprise application placement in resource utilities, in: 14th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM'03), 2003.

[16] L. T. Kou, G. Markowsky, Multidimensional bin packing algorithms, IBM Journal of Research and Development 21 (5).

[17] C. Clark, K. Fraser, H. Steven, Live migration of virtual machines, in: NSDI'05, 2005, pp. 1–11.

[18] W. Voorsluys, J. Broberg, S. Venugopal, R. Buyya, Cost of virtual machine live migration in clouds: A performance evaluation, in: Proceedings of the First International Conference on Cloud Computing (CloudCom'09), 2009.

[19] T. Koch, Rapid mathematical programming, Ph.D. thesis, Technische Universität Berlin, zIB-Report 04-58 (2004).
URL `http://www.zib.de/Publications/abstracts/ZR-04-58/`

[20] T. Achterberg, SCIP - a framework to integrate constraint and mixed integer programming, Tech. Rep. 04-19, Zuse Institute Berlin, `http://www.zib.de/Publications/abstracts/ZR-04-19/` (2004).

[21] VMware, Vmware distributed power management: Concepts and usage, http://www.vmware.com/files/pdf/Distributed-Power-Management-vSphere.pdf (2010).

[22] Citrix, Readme for citrix xenserver workload balancing 2.1, http://support.citrix.com/article/CTX127328 (2010).