Towards Economical Live Migration in Data Centers

Youngsu Cho[0000−0002−5519−3562], Changyeon Jo[0000−0002−9707−1256], Hyunik Kim[0000−0001−7740−8344], and Bernhard Egger✉[0000−0002−6645−6161]

Seoul National University, Republic of Korea
{youngsu,changyeon,hyunik,bernhard}@csap.snu.ac.kr

Abstract. Live migration of virtual machines (VMs) enables maintenance, load balancing, and power management in data centers. The cost of live migration on several key metrics combined with strict service-level objectives (SLOs), however, typically limits its practical application to situations where the underlying physical host has to undergo maintenance. As a consequence, the potential benefits of live migration with respect to increased resource usage and lower power consumption remain largely untouched. In this paper, we argue that live migration-aware SLOs combined with smart live migration algorithm selection provides an economically viable model for live migration in data centers. Based on a model predicting key parameters of VM live migration, an optimization algorithm selects the live migration technique that is expected to meet client SLOs while at the same time to optimize target metrics given by the data center operator. A comparison with the state-of-the-art shows that the presented guided live migration technique selection achieves significantly fewer SLO violations while, at the same time, minimizing the effect of live migration on the infrastructure.

Keywords: Live migration · Live migration modeling · Data center

1 Introduction

Millions of virtual machines run in data centers of large Infrastructure as a Service (IaaS) providers. Virtualization enables service providers to achieve a higher utilization of their infrastructure whilst providing isolation between the multiple co-located tenants [4]. A key technology to achieve high availability and resource utilization in warehouse-scale computing (WSC) is live migration. Live migration transfers a running VM from one host to another and enables operators to balance the load between hosts, to perform maintenance on hard- and software, and to reduce the data center's energy consumption by consolidating VMs onto fewer machines. Various cloud resource management systems propose employing live migration for consolidation and load balancing [35,29,13,26], however, live migration in commercial data centers is still mostly used for maintenance only. Google, for example, migrates over a million VMs every month to perform maintenance on hard- and software in its data centers [28,11].

Despite the economical benefits of reduced energy consumption and higher resource utilization, a number of reasons wide-spread adoption of live migration a means to load balancing and power savings in data centers. First, live migration causes a small performance reduction of the migrated VM. This makes it difficult to fulfill end-user SLOs guaranteeing a certain throughput and high availability. Second, live migration increases the network traffic and CPU load in the data center. Violated SLOs and additional resource usage both incur a short-term financial cost to the operators while

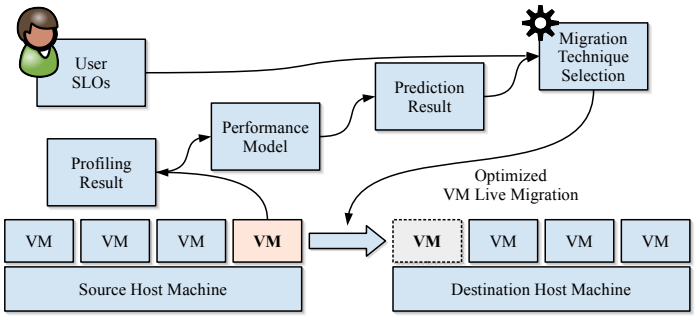

Fig. 1. High-level organization of the presented framework.

the mid- to long-term benefits are harder to quantify [34]. The resulting immobility of VMs is one of the root causes for resource underutilization in data centers as is evident from Alibaba and Google clusters utilization data that shows large opportunities for improvement [4,2,12].

This paper makes the case that new SLOs that take into account the characteristics of live migration combined with smart live migration open new opportunities for economical live migration in WSC. SLOs that permit a certain performance degradation enable live migration of such VMs without incurring a financial penalty. This allows data center operators to apply live migration more frequently and achieve higher resource utilization and a lower total cost of ownership (TCO). This, in turn, allows for new VM pricing models and increased competitiveness.

The main obstacle to live migration-aware SLOs the inability to accurately quantify the effect of live migration on the VM and the data center. The performance degradation of a VM during live migration depends on multiple factors that include the employed live migration technique, the size of the VM, and the workload running in the VM. Similarly, the effect on resources of the data center such as consumed network bandwidth and CPU utilization needs to be quantifiable.

To this end, this work presents a framework for economic live migration in data centers; Figure 1 shows its high-level organization. At the core of the framework is a model that predicts several key metrics related to live migration under consideration of the live migration technique, the VM, and the data center. The framework considers the SLOs of the VMs and migrates VMs whose migration is expected to fulfill the SLOs of all involved VMs. The model is able to predict five key metrics for the nine most common live migration techniques in a heterogeneous data center. The proposed live migration-aware SLOs define the maximum duration and performance degradation an end-user is willing to accept during live migration. The presented framework is evaluated in a small heterogeneous data center. The framework is able to migrate VMs with far fewer SLO violations than any single static live migration technique, demonstrating that, when applied judiciously, live migration has the potential to lead to more flexibility, higher resource utilization, and a lower TCO in commercial data centers.

The remainder of this paper is organized as follows. Section 2 discusses live migration and its metrics. Sections 3 and 4 presents the model and the end-user SLOs. Section 5 details the experimental setup and evaluates the presented framework. Section 6 discusses relevant related work, and Section 7 concludes this paper.

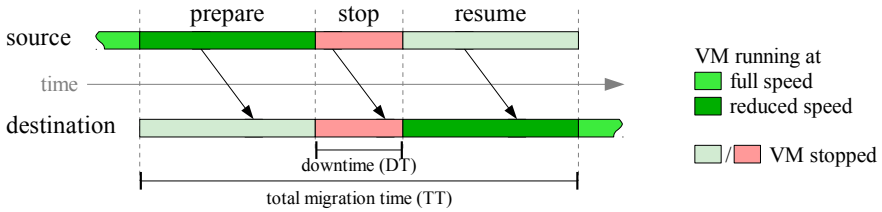

Fig. 2. Live migration phases.

2 Background

2.1 The Anatomy of Live Migration

Migrating a VM from one physical host to another requires transferring the entire volatile state of the VM. The volatile state of a VM includes its memory contents, the execution context, and device buffers. With typical VM memory sizes from four to tens of gigabytes the transfer of the main memory dominates the migration overhead [7].

Live migration can be divided into three distinct phases (Figure 2): *prepare*, *stop-and-copy*, and *resume*. In the *prepare* phase, part of the VM's memory is transferred to the destination host, while the VM keeps running on the source. It is possible that the VM experiences a performance degradation that is caused by the additional resource consumption of the migration technique. The *prepare* phase is succeeded by the *stop-and-copy* phase during which the VM is completely stopped and the execution control is transferred from the source to the destination host. In the *resume* phase, finally, the VM is restarted on the destination and the remaining volatile state is fetched from the source host. Some migration techniques do not copy the entire volatile state in the prepare phase. This can lead to a severe performance degradation during the resume phase.

2.2 Live Migration Techniques

Live migration techniques can be classified by the point in time when the memory of a VM is transferred. One extreme is to send all data before the VM instance is moved to the destination host; this technique is called *pre-copy* [6]. Since the VM keeps running while the memory is being transferred, dirtied memory pages (i.e., pages that are modified while a transfer is ongoing) need to be copied more than once. Pre-copy iteratively copies dirtied memory pages to the destination host until the amount of dirtied pages reaches a given threshold. The VM is then stopped on the source host and the remaining dirty pages and the VM context are transferred to the destination. Finally, the VM is restarted on the destination host. Since the entire volatile state of the VM has been transferred, pre-copy has a very short resume phase.

At the other end of the spectrum lies *post-copy* [14] that immediately transfers the minimal context and restarts the VM on the destination host. The VM's main memory is fetched in the background and on-demand from the source host. Post-copy transfers each memory page exactly once, however, the long resume phase can lead to severe performance degradation in the VM. As usual in engineering, the optimum is seldom found at the extremes; several hybrid live migration techniques have been proposed that split the transfer of the volatile state between the prepare and the resume phase.

Abbreviation	Description
PRE	pre-copy
POST	post-copy
DLTC	pre-copy with delta compression
DTC	pre-copy with data compression
DLTC.DTC	pre-copy with delta compression followed by data compression
THR	throttled pre-copy
THR.DLTC	throttled pre-copy with delta compression
THR.DTC	throttled pre-copy with data compression
THR.DLTC.DTC	throttled pre-copy with delta compression followed by data compression

Table 1. Evaluated migration techniques

In addition to the decision when to transfer the volatile state, there exist several orthogonal optimizations that improve the performance of live migration. *CPU throttling* limits the performance of the VM's CPUs to reduce the rate of dirtied memory. While this optimization can improve the performance of the pre-copy technique, it may also lead to significant performance degradation in the VM. Data compression is another optimization aiming at reducing the amount data to be transferred. Typically, two variants are supported: data compression and delta compression. *Data compression* uses a standard block compression algorithm to compress the data before sending it over the network. Data compression works especially well if the entropy of the data is low, however, the computational overhead incurred by the compression algorithm may cause a performance reduction of the migrated and the co-located VMs. *Delta compression* tries to solve this problem by using the computationally light delta compression algorithm. The significant memory requirements of this algorithm may prohibit its application in situations when live migration is initated because the source host is low on memory.

This paper explores the two basic algorithms, pre-copy and post-copy, and seven combinations of pre-copy with the optimizations CPU throttling, data compression, and data compression as shown in Table 1. The presented framework is implemented in the KVM/QEMU virtual machine manager (VMM) [5] that supports all of these algorithms out-of-the-box. A number of alternative optimization techniques for live migration have been proposed [15,17,19,20,23,31] but are not considered in this work because they are not supported by the virtualization layer employed in industrial data centers.

3 Live Migration Model

The model employed in this paper is able to predict the performance of several live migration metrics (Section 3.1) for each of the nine live migration techniques (Section 2.2) for a given VM. The model takes as input the relevant parameters of the workload running inside a VM, information about the involved source and destination hosts, and the available network bandwidth. These parameters are gathered prior to the migration during a brief profiling phase. The model extends the state-of-the-art in live migration modeling [18]. In contrast to [18], the presented model has been extended to support heterogeneous physical nodes and more live migration techniques. The extended model distinguishes between different CPU vendors and types, and the CPU clock frequency.

Metric	Description
Total migration time (TT)	Total elapsed time from start to completion of a migration.
Downtime (DT)	Time duration of the *stop-and-copy* phase, i.e., the time during which the VM is stopped.
Transferred data (TD)	Total amount of transferred from the source to the destination host.
CPU utilization (CPU)	Additional CPU load incurred on the source host during migration.
Memory utilization (MEM)	Amount of memory consumed by the live migration algorithm.
Performance degradation (PERF)	Performance degradation experienced by migrated VM measured in instructions per second (IPS).
Degradation time (DegT)	Duration of migrated VM's performance degradation.

Table 2. Measured and modeled live migration metrics.

Given the wide range of live migration techniques, physical hosts, and characteristics of VMs, the model is machine-trained. A large database of several 100'000 live migrations performed in our research cluster using all migration techniques is used as the training data set. Each record contains the characteristics of the VM obtained through black-box profiling, the state of the involved physical hosts, the live migration technique, as well as the actual metrics observed during the migration. In addition to real-world benchmarks and to better cover the large parameter space, the training dataset also contains data obtained from migrating VMs running synthetic workloads. Support vector regression (SVR) [30]) is employed to train the model parameters. A separate model is trained for each combination of the live migration technique, the predicted metric, and the type of the source and destination hosts.

3.1 Live Migration Metrics

Table 2 lists the seven VM live migration metrics measured and predicted in this paper. Metrics of interest to the end-user include the downtime *DT*, the performance degradation *PERF*, and the duration of the degradation *DegT*. Metrics of interest to the data center operator are the total migration time *TT*, the amount of transferred data *TD*, and the CPU and memory utilization incurred by the live migration (*CPU* and *MEM*). For all seven metrics lower is better.

3.2 Analytically Modeling Performance Degradation

End-user SLOs considered in this work are the *minimally guaranteed CPU performance* and the *duration of the degradation* while a VM is migrated. Users running a throughput-oriented workload may be willing to accept a 90% degradation in performance for a short duration while end-users for latency-critical workloads may prefer an SLO guaranteeing a minimal 10% performance degradation for a longer period of time. Other SLOs such as the frequency of migration are part of future work.

Live migration-aware SLOs are not only of interest to end-users but also to data center operators. VMs that accept a performance degradation during live migration can be offered at a lower price than VMs that are required by SLOs to provide strict tail-latencies. This flexibility allows the data center operators to classify VMs into *stationary VMs* that cannot be migrated and *mobile VMs* whose SLOs allow migration. These mobile VMs can then be targeted for migrations for the purpose of load balancing, increasing utilization and consolidation in a warehouse.

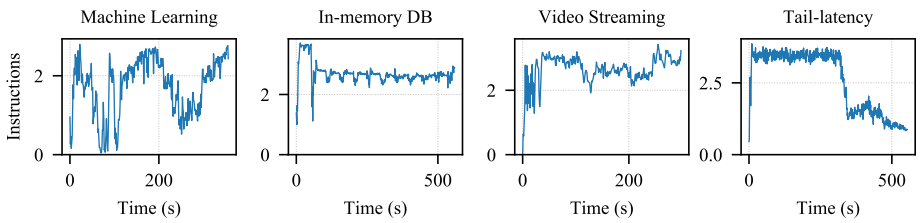

Fig. 3. Workload IPS Pattern

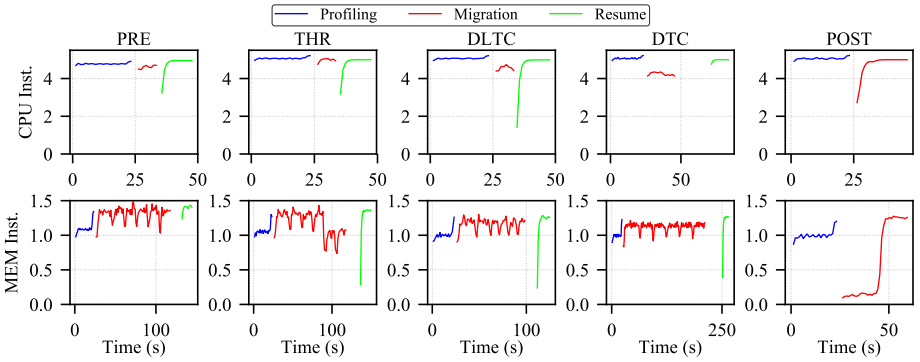

Fig. 4. VM performance in dependence of the migration technique.

An analysis of real-world workload patterns reveals that predicting the maximal performance degradation based on profiled performance is difficult [7]. Figure 3 shows that workloads exhibit large IPS variations as execution transitions between different phases. Since it is impossible to know whether the profiled IPS was obtained during an idle or a high-load period, predicting the IPS based on such data is not a viable approach. This paper thus predicts the relative performance degradation *PERF* and the duration of the degradation *DurT* based on analytical models that are fine-tuned by measuring performance degradation of synthetic benchmarks during live migration with all techniques and between all combinations of hosts.

Figure 4 plots the IPS of a CPU-intensive (upper row) and memory-intensive (lower row) workload for different migration techniques. The plots reveal that the performance during migration is strongly affected by the host CPU load, the VM CPU utilization, the VM memory size, the working set size, and the page dirty rate. The effect of CPU throttling on performance is visible for memory-intensive workloads migrated using the THRottling technique. Similarly, the severe performance degradation caused by the POST-copy technique is pronounced especially for memory-intensive workloads.

To predict the *guaranteed minimal performance* and the *duration of the degradation* for a workload, live migration technique, and involved hosts, the workload generator is executed with four intensity levels ranging from 0 (CPU bound) to 3 (memory bound) with different working set size and page dirty rates. The results of these experiments are consolidated into a Linear Regression model that allows to compute the guaranteed minimal performance of a VM. Figure 5 shows the estimated starting point of throttling for the different memory intensities indicated by the magenta vertical line.

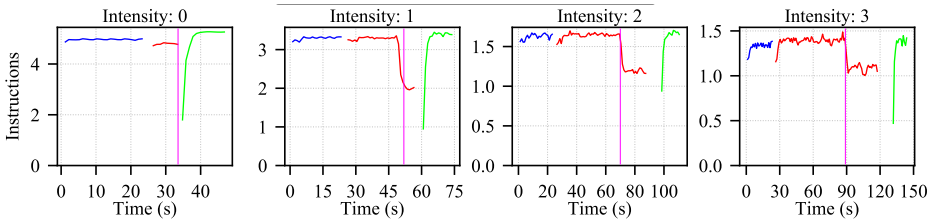

Fig. 5. Estimated start of performance degradation for `THR` (purple line).

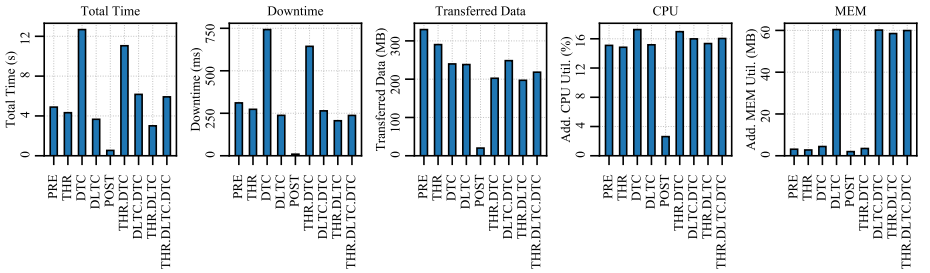

Fig. 6. 10-Fold Cross Validation of the Model (Geo Mean Absolute Error)

3.3 Predicting Key Live Migration Metrics using Machine-Learned Models

The remaining five parameters (total migration time, downtime, transferred data, CPU and memory utilization) are predicted by the machine-learned SVR model. The model is trained with around 90'000 live migration instances for each technique and physical host type. In addition to the 20 input features from prior work [18], the model also takes the processor type of the source and destination host as additional inputs to account for the heterogeneity of the physical hosts involved. Figure 6 displays the result of 10-fold cross validation of the presented model trained using a SVR Bagging regressor; details of model training are given in prior work [18]. The accuracy of the model is slightly below that of prior work - which is expected due to the higher number of features caused by heterogeneous hosts and the nine predived live migration techniques.

4 Service Level Objectives and Migration Technique Selection

Existing Service Level Agreements in commercial data centers focus foremost on service uptime [3,10] and allow for little performance variation [8]. As a consequence, virtual machines are *stationary*: once placed on a host, a VM is immobile and migrated only when the host has to undergo maintenance [9]. Stationary VMs stifle efficient load balancing and consolidation, both operations that are beneficial for data center operators to lower the TCO. One of the main reasons of the reluctance to pro-actively live migrate VMs is that the effects of live migration on the migrated VM and the data center are difficult to quantify. The model presented in the preceding section demonstrates that is it possible to predict several key metrics of live migration in an accurate and low-overhead manner. To benefit from this ability to accurately predict the effect of live migration, is is necessary to rethink existing migration policies and service level agreements (SLAs).

The ability to predict downtime, guaranteed performance, the duration of the performance degradation of live migration allows service providers to offer what we call SLOs for *mobile* VMs. Service level objectives can be defined from the perspective of the the end-user and the service provider. The former are part of the SLA between the end-user and the service provider and typically cause a financial loss when violated. The latter are objectives whose optimization leads to a lower TCO due to reduced resource usage; their violation, however, does not directly lead to contractual violations with financial reparations.

SLOs that enable mobile VMs are beneficial for both the service provider and the end-user. The service provider benefits from a pool of movable VMs that can be migrated to to achieve load balancing or node consolidation. The end-user benefits from cheaper VMs since mobile VMs are expected to be priced below stationary VMs.

5 Evaluation

5.1 Cluster Configuration and VM Workloads

VMs are deployed on our internal research cluster comprising eight heterogeneous physical nodes. Four nodes are equipped with a quad-core Intel Skylake i5-6600 processor and 16GB of physical memory. The other four nodes contain an octa-core AMD FX-8300 processor and 16GB of physical memory. The nodes are interconnected with three separate networks, one for application traffic, one for migration traffic, and one for remote storage traffic. The nodes run Ubuntu 16.04 with the QEMU-KVM 2.8.5 hypervisor that supports all evaluated live migration techniques from Table 1. Each VM is configured with 1 VCPU and 2 GBs of memory. The migration network has a bandwidth of 1Gbit/s.

The workloads executed in the VMs include Intel-Hadoop Hibench [16], Tailbench [21], YCSB memcached [36], and Mplayer [33]. The Hibench benchmark suite contains several micro-benchmarks and machine learning workloads that are widely used in cloud services. Tailbench consists of latency-critical workloads, and the YCSB Memcached benchmark is designed to evaluate the performance of in-memory database workloads. Mplayer, finally, represents a video-streaming workload.

5.2 Migration Polices

Table 3 lists the SLO policies evaluated in this paper. All policies except *Min TT* contain user and provider SLOs that must be met. The optimization goal lists the metric that is to be minimized if several live migration techniques are expected to meet all SLOs. The policies are constructed to cover several scenarios. *Min TT* migrates VMs as fast as possible by minimizing the total migration time *TT*; this policy is useful if the machine has to undergo an emergency shutdown. The other five policies contain more or less stringent SLOs for both the user and the provider with different optimization goals to cover various scenarios.

Policy Name	User SLO	Provider SLO	Opt.Goal	Description
Min TT	-	-	TT	Minimize total time.
High Performance	DT $\leq$ 1s PERF $\geq$ 95% DegT $\leq$ 5s	CPU 100% MEM 512MB	DT	Minimize downtime under strict limits for downtime and performance
Least Traffic	DT $\leq$ 2s PERF $\geq$ 60% DegT $\leq$ 20s	CPU 100% MEM 512MB	TD	Minimize transferred data with relaxed downtime and performance limits
Least Operation Cost	DT $\leq$ 3s PERF $\geq$ 75%	CPU 100% MEM 512MB	TT	Minimize migration time with performance guarantees
Reduced Downtime	DT $\leq$ 1s PERF $\geq$ 50% DegT $\leq$ 30s	TT 60s	DT	Minimize downtime while limiting migration time and performance degradation
Traffic Overload	PERF $\geq$ 80% DegT $\leq$ 60s	TD 3GB	TD	Limit transferred data while maintaining performance

Table 3. SLOs for mobile VMs.

5.3 Live Migration Sequence and Technique Selection

The experiments evaluate a sequence of 480 migrations that occurred in real-live on the cluster. In every case, the state of the cluster is recreated as faithfully as possible with the same number of co-located VMs running the same workloads, nevertheless, differences in the actual resource utilization are possible due to different execution phases within the workloads. When replaying the migration sequence, the management framework (Figure 1) first profiles all VMs on the source host as discussed in [18]. Next, the profiles are fed into the model to predict the metrics for all VMs and live migration techniques. If several techniques are expected to meet all SLOs, the technique that minimizes the optimization goal is chosen. If all techniques are predicted to violate one or more SLOs, the technique with the smallest average relative violations is chosen. Note that in a real-world scenario, the framework may choose not to migrate a VM in such a case; since this paper aims to demonstrate that the number of SLO violations can be minimized with the presented model, the framework always migrates a VM.

5.4 Analysis of SLO Violations

Figure 7 shows the average relative score of SLO violations (i.e., the average severity of the violations) and total number of SLO violations for the different live migration techniques. The *Min TT* scenario is not shown because it has no SLOs. The white bar shows the *single* static live migration technique that incurs the minimal relative SLO violation score, the presented model-*guided* technique, and results for an *oracle* model that can predict all metrics without any error. We observe that the guided approach clearly outperforms the choice of a single technique by a large margin and comes relatively close to the oracle model. *Guided* outperforms *static* in all cases except the absolute number of SLO violations for the *Least Traffic* policy. While the *post-copy* technique is the single best technique that minimizes the number of SLO violations, the violations are much more severe than those incurred by the *guided* approach.

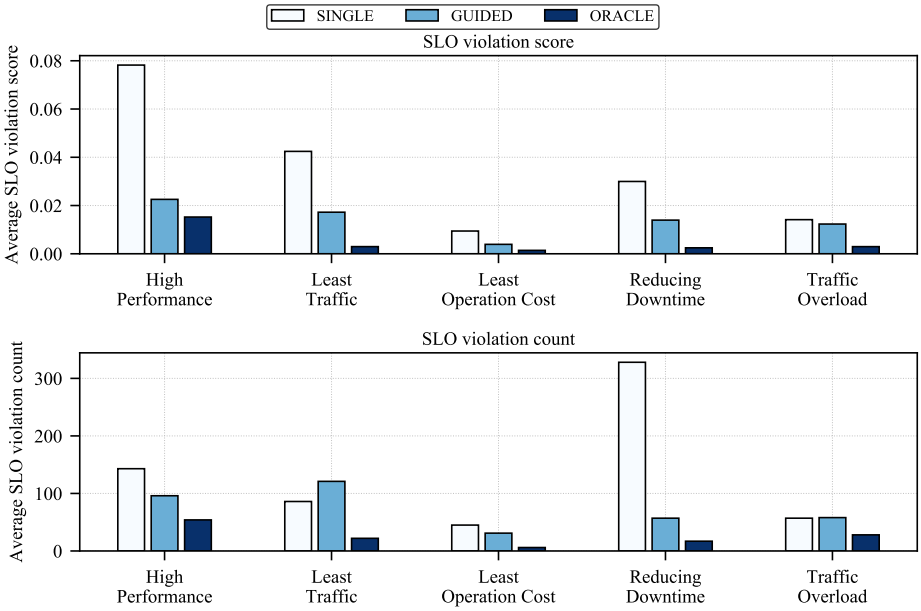

Fig. 7. Avg. relative SLO violations and total violation count for different technique selections.

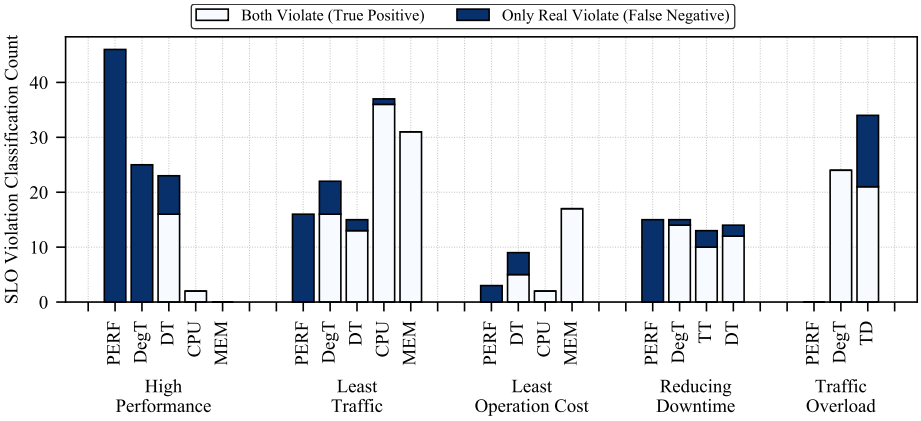

Fig. 8. Effect of modeling error on SLO violations.

While the model shows an average error of 10% over all predicted metrics, an interesting question is how this prediction error manifests in terms of absolute SLO violations. Figure 8 shows the results. The white bar denotes true positives, i.e., the selected technique was predicted to violate an SLO and actually did so, while the blue bar shows false negatives (violation occurred despite no violation predicted). Only false negatives are a concern since true positives were expected to occur. The results reveal that the prediction of performance under stringent constraints is difficult for the proposed model and leaves room for improvement in future work. Note, however, that to the best of our knowledge, there exists no model that can predict performance and the duration of the performance with higher accuracy than the model presented in this work.

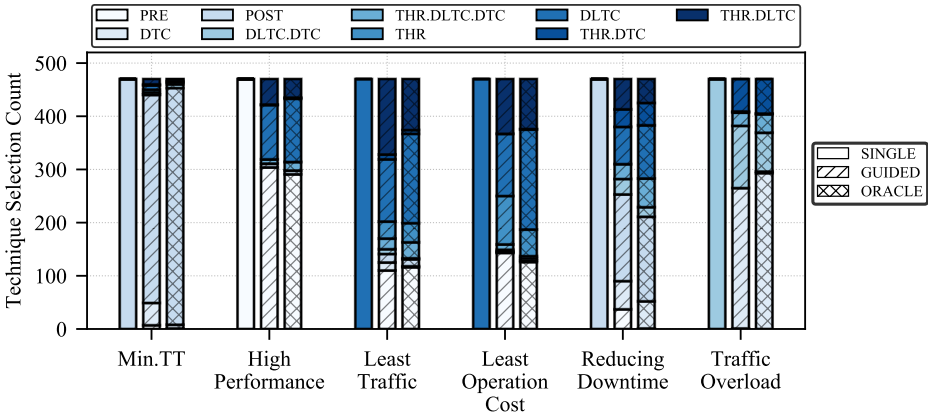

Fig. 9. The ratio of the selected techniques by the guided approach.

5.5 Importance of Model-Guided Live Migration Technique Selection

Figure 9 finally plots the selected live migration techniques for *single*, *guided*, and *oracle* technique selection for the different migration policies from Table 3. The figure reveals that selecting the appropriate technique depends less on the migration policy but rather on the workload running inside the VM and the state of the source host. In other words, this results clearly demonstrates the necessity of model-guided live migration technique selection to reduce not only the total number of SLO violations but also the severity of those violations.

6 Related Work

Since the introduction of various live migration techniques over the last decade, a number of researchers have tackled the problem of employing the different techniques properly. Koto [22], Nathan [25], and Svard [32] all propose guidelines for live migration technique selection that are, however, difficult to deploy in real systems due to the lack of a systematic way of choosing them. Similarly, there exist a number of works that model the performance of live migration techniques. Analytical models require complex calibration when deployed and are thus not of much practical use in heterogeneous data centers. Jo [18] has presented the first machine learning-based attempt to live migration modeling. That work only considers five techniques and four metrics but served as the basis for the model developed in this paper.

A number of migration frameworks have been presented that aim at resolving or avoiding hot-spots. Sandpiper [35] is an early dynamic migration system which is mitigating hotspots in the cluster on-demand. It periodically builds a per-node resource utilization profile and predicts hotspot occurrence in advance using a simple time-series prediction technique. Sandpiper uses a simple heuristics that selects the VM to migrate and only employs the pre-copy migration technique. CloudScale [27] presents a resource prediction technique to avoid SLO violations before they manifest. It increases resource capping when a peak resource demand is expected in the near future. CloudScale's design can tolerate a certain amount of resource demand misprediction by not fully committing all available resources. VMs are only migrated when no resources are

available on the node. While CloudScale takes the migration overhead into consideration to prevent possible SLO violations during live migration and employs a simple linear regression model to predict live migration cost. It has been shown, however, that a simple linear model is unable to predict the migration cost of a VM with high accuracy [1,18,24].

This work is based on prior work by Jo [18] that focuses on developing machine-learned models to predict six live migration metrics (total time, downtime, transferred data, CPU and memory utilization, and performance degradation). The modeling of live migration metrics is similar to prior work. This work uses the same machine-learned approach to model five of the six metrics, but supports nine instead of five migration techniques and heterogeneous data centers. Also, it was found that performance degradation and the new metric performance degradation duration are better predicted with analytical models. Other than prior work, this paper proposes and evaluates new live migration-aware SLOs that enable data centers to classify VMs into stationary and mobile to achieve flexible migration of VMs and thus better resource utilization.

7 Conclusion

Strict SLOs and the uncertainty of the effect of live migration prevent data centers from employing live migration to achieve better load balancing and higher resource utilization. This paper demonstrates that the combination of live migration-aware SLOs and an accurate model predicting several key metrics of live migration allows for an economically more efficient use of live migration in data centers while minimizing the number of SLA violations. An evaluation with six policies and 480 live migrations shows that the presented migration-aware SLOs and smart live migration framework is able to significantly reduce the total number of SLO violations and the relative violation of these SLO compared to any static single migration technique.

The live migration dataset, the model, and the source code of the framework are available at `https://csap.snu.ac.kr/software`.

Acknowledgements

This work was supported by the National Research Foundation of Korea (NRF) funded by the Korean government, in part, by grants NRF-2015K1A3A1A14021288, 2016R1-A2B4009193, by the BK21 Plus for Pioneers in Innovative Computing (Dept. of Computer Science and Engineering, SNU, grant 21A20151113068), and by the Promising-Pioneering Researcher Program of Seoul National University in 2015. ICT at Seoul National University provided research facilities for this study.

References

1. Akoush, S., Sohan, R., Rice, A., Moore, A.W., Hopper, A.: Predicting the performance of virtual machine migration. In: Proceedings of the 2010 IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems. pp. 37–46. MASCOTS '10, IEEE Computer Society, Washington, DC, USA (2010). https://doi.org/10.1109/MASCOTS.2010.13
2. `https://github.com/alibaba/clusterdata` (2018), online; acc. June 2020
3. `https://aws.amazon.com/legal/service-level-agreements/` (2020), online; accessed June 2020

4. Barroso, L.A., Hölzle, U., Ranganathan, P.: The datacenter as a computer: Designing warehouse-scale machines. Synthesis Lectures on Computer Architecture **13**(3) (2018)
5. Bellard, F.: Qemu, a fast and portable dynamic translator. In: USENIX Annual Technical Conference, FREENIX Track. vol. 41, p. 46 (2005)
6. Clark, C., Fraser, K., Hand, S., Hansen, J.G., Jul, E., Limpach, C., Pratt, I., Warfield, A.: Live migration of virtual machines. In: Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation - Volume 2. pp. 273–286. NSDI'05, USENIX Association, Berkeley, CA, USA (2005), http://dl.acm.org/citation.cfm?id=1251203.1251223
7. Egger, B., Gustafsson, E., Jo, C., Son, J.: Efficiently restoring virtual machines. International Journal of Parallel Programming **43**(3), 421–439 (2015). https://doi.org/10.1007/s10766-013-0295-0
8. https://cloud.google.com/compute/docs/benchmarks-linux (2020), online; accessed June 2020
9. https://cloud.google.com/compute/docs/instances/live-migration (2020), online; accessed June 2020
10. https://cloud.google.com/terms/sla/ (2020), online; accessed June 2020
11. https://goo.gl/Ui3HFd (2018), online; accessed June 2020
12. https://github.com/google/cluster-data (2019), online; acc. June 2020
13. Hermenier, F., Lorca, X., Menaud, J.M., Muller, G., Lawall, J.: Entropy: A consolidation manager for clusters. In: Proceedings of the 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments. pp. 41–50. VEE '09, ACM, New York, NY, USA (2009). https://doi.org/10.1145/1508293.1508300
14. Hines, M.R., Gopalan, K.: Post-copy based live virtual machine migration using adaptive pre-paging and dynamic self-ballooning. In: Proceedings of the 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments. pp. 51–60. VEE '09, ACM, New York, NY, USA (2009). https://doi.org/10.1145/1508293.1508301
15. Hirofuchi, T., Nakada, H., Itoh, S., Sekiguchi, S.: Reactive consolidation of virtual machines enabled by postcopy live migration. In: Proceedings of the 5th International Workshop on Virtualization Technologies in Distributed Computing. pp. 11–18. VTDC '11, ACM, New York, NY, USA (2011). https://doi.org/10.1145/1996121.1996125
16. https://github.com/intel-hadoop/HiBench (2018), online; acc. June 2020
17. Jin, H., Deng, L., Wu, S., Shi, X., Pan, X.: Live virtual machine migration with adaptive, memory compression. In: 2009 IEEE International Conference on Cluster Computing and Workshops. pp. 1–10 (Aug 2009). https://doi.org/10.1109/CLUSTR.2009.5289170
18. Jo, C., Cho, Y., Egger, B.: A machine learning approach to live migration modeling. In: ACM Symposium on Cloud Computing. SoCC'17 (September 2017)
19. Jo, C., Egger, B.: Optimizing live migration for virtual desktop clouds. In: IEEE 5th International Conference on Cloud Computing Technology and Science. CloudCom '13, vol. 1, pp. 104–111 (Dec 2013). https://doi.org/10.1109/CloudCom.2013.21
20. Jo, C., Gustafsson, E., Son, J., Egger, B.: Efficient live migration of virtual machines using shared storage. In: Proceedings of the 9th ACM SIGPLAN/SIGOPS international conference on Virtual execution environments. pp. 41–50. VEE '13, ACM, New York, NY, USA (2013). https://doi.org/10.1145/2451512.2451524
21. Kasture, H., Sanchez, D.: Tailbench: A benchmark suite and evaluation methodology for latency-critical applications. Proceedings of the 2016 IEEE International Symposium on Workload Characterization, IISWC 2016 pp. 3–12 (2016). https://doi.org/10.1109/IISWC.2016.7581261
22. Koto, A., Kono, K., Yamada, H.: A guideline for selecting live migration policies and implementations in clouds. In: Proceedings of the 2014 IEEE 6th International Conference on

Cloud Computing Technology and Science. pp. 226–233. CLOUDCOM '14, IEEE Computer Society, Washington, DC, USA (2014). https://doi.org/10.1109/CloudCom.2014.36
23. Liu, Z., Qu, W., Liu, W., Li, K.: Xen live migration with slowdown scheduling algorithm. In: Proceedings of the 2010 International Conference on Parallel and Distributed Computing, Applications and Technologies. pp. 215–221. PDCAT '10, IEEE Computer Society, Washington, DC, USA (2010). https://doi.org/10.1109/PDCAT.2010.88
24. Nathan, S., Bellur, U., Kulkarni, P.: Towards a comprehensive performance model of virtual machine live migration. In: Proceedings of the Sixth ACM Symposium on Cloud Computing. pp. 288–301. SoCC '15, ACM, New York, NY, USA (2015). https://doi.org/10.1145/2806777.2806838
25. Nathan, S., Bellur, U., Kulkarni, P.: On selecting the right optimizations for virtual machine migration. In: Proceedings of the12th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments. pp. 37–49. VEE '16, ACM, New York, NY, USA (2016). https://doi.org/10.1145/2892242.2892247
26. Novaković, D., Vasić, N., Novaković, S., Kostić, D., Bianchini, R.: Deepdive: Transparently identifying and managing performance interference in virtualized environments. In: Proceedings of the 2013 USENIX Conference on Annual Technical Conference. pp. 219–230. USENIX ATC'13, USENIX Association, Berkeley, CA, USA (2013), http://dl.acm.org/citation.cfm?id=2535461.2535489
27. Reiss, C., Tumanov, A., Ganger, G.R., Katz, R.H., Kozuch, M.A.: Heterogeneity and dynamicity of clouds at scale: Google trace analysis. In: Proceedings of the Third ACM Symposium on Cloud Computing. pp. 7:1–7:13. SoCC '12, ACM, New York, NY, USA (2012). https://doi.org/10.1145/2391229.2391236
28. Ruprecht, A., Jones, D., Shiraev, D., Harmon, G., Spivak, M., Krebs, M., Baker-Harvey, M., Sanderson, T.: Vm live migration at scale. In: Proceedings of the 14th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments. pp. 45–56. VEE '18, ACM, New York, NY, USA (2018). https://doi.org/10.1145/3186411.3186415
29. Shen, Z., Subbiah, S., Gu, X., Wilkes, J.: Cloudscale: Elastic resource scaling for multi-tenant cloud systems. In: Proceedings of the 2Nd ACM Symposium on Cloud Computing. pp. 5:1–5:14. SoCC '11, ACM, New York, NY, USA (2011). https://doi.org/10.1145/2038916.2038921
30. Smola, A.J., Schölkopf, B.: A tutorial on support vector regression. Statistics and Computing **14**(3), 199–222 (Aug 2004). https://doi.org/10.1023/B:STCO.0000035301.49549.88
31. Svärd, P., Hudzia, B., Tordsson, J., Elmroth, E.: Evaluation of delta compression techniques for efficient live migration of large virtual machines. In: Proceedings of the 7th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments. pp. 111–120. VEE '11, ACM, New York, NY, USA (2011). https://doi.org/10.1145/1952682.1952698
32. Svärd, P., Hudzia, B., Walsh, S., Tordsson, J., Elmroth, E.: Principles and performance characteristics of algorithms for live vm migration. SIGOPS Oper. Syst. Rev. **49**(1), 142–155 (Jan 2015). https://doi.org/10.1145/2723872.2723894
33. http://www.mplayerhq.hu/design7/news.html (2017), online; accessed June 2020
34. Voorsluys, W., Broberg, J., Venugopal, S., Buyya, R.: Cost of virtual machine live migration in clouds: A performance evaluation. In: Jaatun, M.G., Zhao, G., Rong, C. (eds.) Cloud Computing. pp. 254–265. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)
35. Wood, T., Shenoy, P., Venkataramani, A., Yousif, M.: Black-box and gray-box strategies for virtual machine migration. In: USENIX Symposium on Networked Systems Design and Implementation (NSDI) (Apr 2007), http://faculty.cs.gwu.edu/~timwood/papers/NSDI07-sandpiper.pdf
36. https://github.com/brianfrankcooper/YCSB/tree/master/memcached (2018), online; accessed June 2020