Can VM Live Migration Improve Job Throughput? Evidence From a Real World Cluster Trace

Daon Park[0000−0003−2312−3049], Hyunsoo Kim[0000−0002−4660−1771], Youngsu Cho[0000−0002−5519−3562], Changyeon Jo[0000−0002−9707−1256], and Bernhard Egger ✉[0000−0002−6645−6161]

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

Abstract. Cloud resource providers are putting more and more emphasis on efficiently management the resources of their data centers to achieve high utilization while minimizing energy consumption. Despite these efforts, an analysis of recent data center traces reveals that the utilization of CPU and memory resources has not improved significantly over the past decade. Resource overcommitment is a promising approach to improve resource utilization, because most workloads show a significant gap between their guaranteed and actually consumed resources. A wrong prediction of the actual usage, however, can lead to a severe performance degradation on overloaded nodes. Combining resource overcommitment with live migration of tasks can alleviate the situation, but its prohibitively high cost has so far prevented a wide adoption. Recent and rapid advancements in networking technology, however, are changing the status quo. With throughputs surpassing 100 Gb/s in 2021, even large tasks can be migrated within a few seconds. In light of these improvements, we believe it is time to rethink the application of resource overcommitment and live migration to improve data center resource utilization. Based on real-world cluster traces published by Google in 2019, we show that combining resource overcommitment with task live migration can reduce the mean task turnaround time by 16%, demonstrating that further research in this direction is both warranted and promising.

Keywords: Live migration · Data center · Virtualization · Load balancing

1 Introduction

Due to the huge benefits of flexibility and efficiency, many users and organizations have come to cloud computing to either use or service various products [14,18]. During this trend, virtual machines (VMs) have been the fundamental unit of cloud computing with the advantages of security, performance isolation, and ease of management [5]. Nowadays, the growing popularity of resource-intensive applications such as artificial intelligence (AI), machine learning (ML), and big data analytics demand larger VM instances to accommodate their workloads. There is a clear trend of increasing instance sizes to run resource-intensive applications; the median memory size of a VM in AWS EC2 catalog is 64GB, and VMs with a memory size above 32GB in Microsoft Azure see an increase from about 5% [3] to 10% [4] of all instances.

Despite the growing demand for computing resources, machines in the data centers still suffer from low resource utilization. The recently published cluster trace from Google [16] reveals that the average utilization of CPU and memory stagnated at the 60% level, indicating significant room for improvement. Other providers such Microsoft Azure [7] and Alibaba [9] report similarly low average resource utilization.

Under utilization is typically caused by tasks that reserve more resources than they actually use. Since tasks are scheduled onto a machine based on the requested resources, it is not easy to avoid under utilization if the total sum of requested resources are to fit within the node's capacity. To tackle this issue, resource overcommitment [8] became a promising solution to improve the resource utilization. However, the huge gap between the reservation and the actual usage makes seamless overcommitment challenging. An inaccurate prediction of resource usage causes a waste of resources when overestimated, or a performance degradation when underestimated. Since no prediction can be perfect every time, load balancing the key to achieve high utilization of nodes [12].

Live migration enables load balancing and is thus a promising solution to alleviate the problem of low resource utilization. A task is live migrated from one machine to another by copying its entire volatile state, i.e., its memory, from one machine to another and thus is a resource-intensive operation in itself. With typical virtual machine memory sizes reaching 64 GiB, however, migrating a task can take several minutes and is thus not typically used for load balancing.

Recent advancements in networking technology change this situation. With the arrival of Terabit Ethernet [15], even large tasks can be migrated within a few seconds, opening up new possibilities for resource overcommitment and load balancing. An interesting question in this context is by how much resource utilization can be improved in such environments. This paper explores this question by simulating the execution of a 2019 Google cluster trace in a data center cluster with an 100Gb/s interconnection network.

The remainder of this paper is organized as follows: Section 2 discusses VM live migration and the 2019 Google cluster trace. Section 3 explains our simulator in detail, and Section 4 analyzes the experiments conducted with the simulator. Section 5 discusses related works, and Section 6 concludes this paper.

2 Background and Motivation

2.1 Cluster Scheduler and Scaling

The job of the cluster scheduler is to place incoming jobs on the cluster's nodes to improve throughput, turnaround time, and overall resource utilization. A job often comprises multiple tasks that each can be placed on different nodes. A node executes multiple tasks in parallel to maximize resource utilization and job throughput. Finding the proper balance when co-locating tasks is an important task of the cluster scheduler.

A major challenge in designing a cluster scheduler is accurately predicting the resource demands of each task. The mismatch between reservation and actual usage of resources makes it difficult to maximize the utilization of a cluster. For example, despite a task requesting 4 CPUs and 8 GiB of memory, the actual resources usage can be much lower than the requested amount of resources.

Task migration is an important tool to correct load mispredictions and allows rebalancing the load of each node. However, due to the high cost of migration, cluster schedulers try to minimize the number of migrated tasks. Instead, after assigning tasks, horizontal or vertical scaling or a combination of both [1, 13] are used to achieve higher utilization. While these techniques can exploit slack caused by low average utilization, such scaling techniques are challenging to be applied during load spikes [2].

2.2 VM Live Migration

VM live migration is a useful technique that enables relocating a running VM to another node without a significant downtime. It is especially useful when a cluster has to change the placement of VMs for load balancing. With VM live migration, a task running on a heavily loaded node is migrated to a lightly loaded node to utilize resources better and improve overall task completion time.

A challenge with VM live migration for load balancing is reducing the total time of the migration itself. Migrating a VM requires sending a copy of the entire VM state to the destination node; for current VMs, this can take several minutes even with 10 Gb/s networks. A fluctuating resource usage of a task also makes applying VM live migration difficult. If migration takes too long to complete, the VM can be migrated at the wrong time, resulting in a waste of resources.

Due to its prohibitively high cost, in general, VM live migration is not used for load balancing purposes [7]. The rising popularity of fast networks, however, is bringing this cost down by several orders of magnitude. It is thus important to re-evaluate live migration as a load balancing tool. Currently, network performance doubles every few years; network throughputs of 100 Gb/s are now available for inter node data transfer [15]. Such fast network dramatically reduce the overall VM migration time and thereby make live migration for load balancing feasible [11]. Considering such technique employed in data centers, operators may need to rethink the use of VM live migration for load balancing.

2.3 Google Cluster Trace 2019

Google has published cluster traces of their warehouses in 2011 [17] and 2019 [16]. In this paper, we use the cluster trace published in 2019 for our analysis. The main content of the cluster trace is per-task resource usage over time. The trace includes 5-minute averaged normalized CPU and memory usage with percentile values. By summing up the resource usage of tasks running on the same node, we can compute the total resource usages of each node over time. To analyze potential improvements in resource utilization, we simulate an overcommitted data center with and without live migration.

3 Cluster Simulation

To test the hypothesis of this work, we have implemented a cluster simulator. The simulator simulates a given number of heterogeneous physical machines for a number of tasks that are described in terms of their resource usage over time. In the following, the design and implementation of our simulator are explained in detail.

3.1 Simulation Parameters

The simulator is controlled with three main parameters. The *overcommit factor* determines by how much the resources of a physical node can be overcommitted. With an overcommit factor of 2, for example, the resources of a node can be reserved up to 200% of its actual capacity. The overcommit factor is a key parameter that significantly affects node utilization during the simulation. The *migration* parameter that controls whether tasks are allowed to be live migrated to other nodes for load balancing purposes. The *epochs* parameter, finally, sets the duration of the simulation in epochs. The number of simulated expochs should be sufficiently high to allow most tasks to terminate.

3.2 Tasks and Machines

Every task is associated with a resource reservation request and a resource utilization history. The resource reservation request defines how the amount of resources that the task reserves on a node. The resource utilization history of a task reflects the actual resource utilization of the task over time. Physical nodes are defined with a certain amount of resources. The sum of all co-located tasks' resource reservations on a node must not surpass the amount of resources multiplied by the overcommit factor. If the sum of the co-located tasks' actual resource utilization at a given point in time exceeds the physically available resources, the tasks experience a performance degradation (see below). All resources, resource requests, and resource utilizations are normalized to the largest amount of resources available in a single node.

3.3 Task Arrivals

We assume that all tasks are submitted at the beginning of the simulation (epoch 0). This is to have a large pool of schedulable jobs ready to get a better understanding the effect of live migration and resource overcommitment on the utilization of a data center. Simulations with task arrivals over time are left for future work.

3.4 Task Performance Degradation

As stated above, the sum of co-located tasks' resource reservations never exceeds the node's resource multiplied by the overcommit factor. It is, however, possible that the accumulated resource usage of all co-located tasks for a given epoch surpasses the physically available resources of the node. This situation is called *overload*. Since all resource metrics are normalized, *overload factor* is identical to the total of all co-located task's resource utilization. Whenever overload occurs, all co-located tasks on the overloaded node experience a performance degradation. The In this work, we slow down all co-located tasks equally by dividing the resource utilization by the overload factor. For example, if four co-located tasks request 50% of the physically available CPU resources on a node (an overload factor of 2), each task receives only $50/2 = 25\%$ CPU resources, effectively, halving their performance.

3.5 VM Live Migration Duration

For slower networks, the duration of live migration is an important metric since at least one full copy of a task's volatile state needs to be copied from the source to destination node. In dependence of the live migration algorithm, a significantly larger amount of data may get transferred [10]; this is especially true for algorithms based on pre-copy that that iteratively send the modified data from the source to the destination node [6].

With network bandwidths approaching (or even exceeding) 100 Gb/s, live migration becomes "instant" for most practical purposes. A VM with 64 GiB of RAM, for example, can be migrated in less than 10 seconds even when assuming a duplication factor of 1.5 for pre-copy (i.e., copying 96 GiB of data). Since the sampling interval of Google's trace data with one sample per 5 minutes is significantly larger, we ignore the duration of live migration in this work.

3.6 Simulation Procedure

The simulator runs from epoch 0 to *epochs* as defined in the simulation parameters (Section 3.1). We only consider CPU and memory resources in this work; other resources such as network bandwidth are left for future work. At the start of each epoch, overcommitment and VM live migration are simulated as follows:

1. **Migration candidate selection.** In this first step, all nodes are checked for overload by summing up the current resource usage of each co-located task. The co-located tasks of a node are visited in random order. As soon as overload occurs for either the CPU or memory resource, the current and all following tasks are marked as migration candidates.
2. **VM live migration.** In this step, the simulator tries to migrate all migration candidates to nodes with sufficient resources. The simulator tries to migrate each candidate task a node with idle resources; the order in which the nodes are visited is random. If no machine can host the task without getting overloaded, the task is not migrated and stays on its initial node.
3. **Task scheduling.** After all migration candidates have been processed, the scheduler tries to place tasks from the job wait queue onto the cluster. Tasks in the wait queue are not reordered, i.e., task scheduling stops at the first task that cannot be mapped onto the cluster anymore.
4. **Proportional scheduling.** In this last step, the performance of overloaded nodes is adjusted to simulate performance degradation. All co-located tasks on an overloaded node progress with the reciprocal of the overload factor for this epoch.

3.7 Implementation

The simulator is implemented in Python.

Table 1: Task completion and turnaround time in dependence of the over-commit factor.

	Completion Time					Turnaround Time				
	x1.0	x2.0	x2.5	x3.0	x3.5	x1.0	x2.0	x2.5	x3.0	x3.5
Median	7	8	9	12	14	38	10	11	13	15
Mean	39.4	40.6	43.1	46.2	50.4	2191.3	70.1	60.5	63.5	67
95%ile	77	82	89	98	109	9923	269	189	195	204
99%ile	645	647	698	658	742	11579	1284	980	965	981
Std. Dev.	250.5	250.9	252.4	251.8	255	3446.6	289.8	268.8	268.3	270.3

Table 2: Task completion and turnaround time depending on task live migration.

	Completion Time		Turnaround Time	
	W/o Migration	W/ Migration	W/o Migration	W/ Migration
Median	9	7	11	9
Mean	43.1	40.7	60.5	50.7
95%ile	89	87	189	145
99%ile	698	653	980	823
Std. Dev.	252.4	253.1	268.8	260

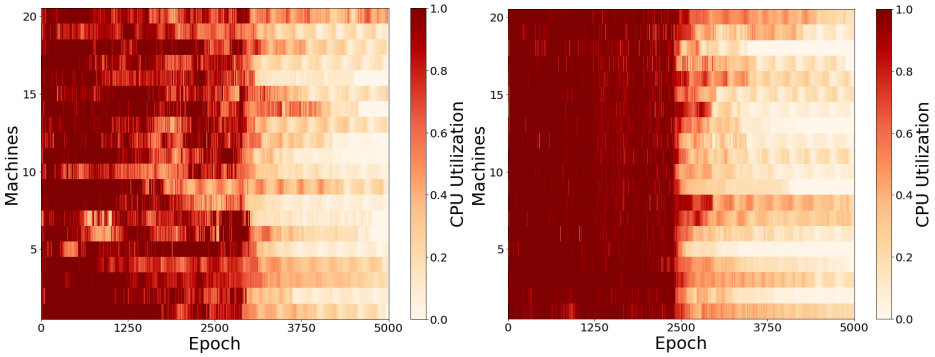
4 Evaluation

For the evaluation, we simulate 20 physical nodes and 400'000 tasks randomly selected from the 2019 Google Cluster Trace [16]. We first analyze how the overcommitment factor and task live migration affect the simulation in Section 4.1. Next, we investigate resource utilization over time with and without live migration in Section 4.2.

4.1 Task Completion and Turnaround Time

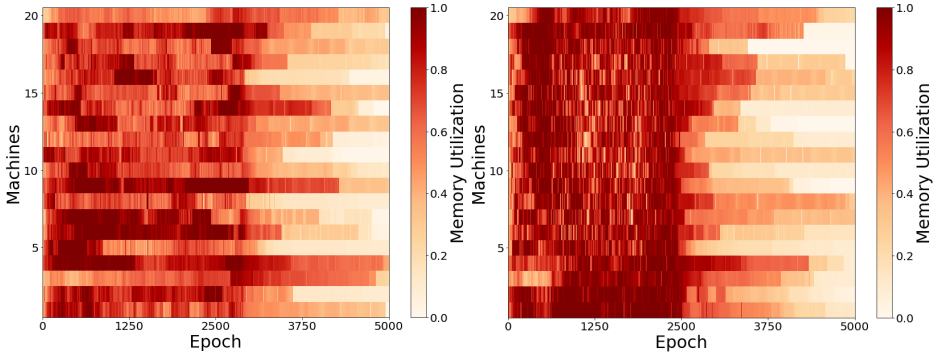
Table 1 shows the effects of different overcommit factors without task live migrations. The completion time denotes the time from then the task is scheduled to its completion. The turnaround time, on the other hand, reports the time from the task's submission time to its completion, i.e., includes the wait time between submission and the start of the execution. The results show that no overcommitment is best for tasks (as there is no performance degradation) but seriously harms the throughput of a data center. As the overcommit factor is raised, the data center's throughput increases significantly with a moderate increase in task completion time. We also observe that too much overcommitment starts to hurt data center performance but is still well above no overcommitment at an overcommit factor of 3.5.

Next, we analyze the effect of task migration on completion time and turnaround time. Table 2 shows the results for an overcommitment factor of 2.5 with and without



(a) Without task live migration (b) With task live migration

Fig. 1: Node CPU utilization heatmap with and without task live migration.



(a) Without task live migration (b) With task live migration

Fig. 2: Node memory utilization heatmap with and without task live migration.

task migration. We observe live migration that both completion time and turnaround time improve with live migration. The mean completion time is reduced by 5% while the mean turnaround time decreases by 16% compared to no task migration. While these numbers seem moderate, the benefits in terms of cost and energy savings are significant at warehouse-scale.

4.2 Resource Utilization

Cluster resource utilization and task throughput are visualized by Figures 1 and 2. The figures show heatmaps of CPU and memory utilization for simulations with an over-commit factor of 2.5 with and without task migration. Since all tasks are assumed to be submitted at the beginning of the simulation, the cluster utilization is high during the first 2500 epochs even without migration. We observe, however, that even in such an extreme scenario, it is not easy to continuously and fully utilize all resources of a node

without migration. With live migration, on the other hand, we clearly see that tasks are migrated to less-loaded nodes and thus, the overall utilization of the cluster goes up. Most tasks in the Google Cluster Trace are short-lived with a few very long running tasks. We observe that with live migration, most short-lived tasks are scheduled and complete by epoch 2500 during which the cluster utilization is close to 100%. After epoch 2500, few long-lived tasks keep running. Without live migration, on the other hand, cluster utilization is much lower on average during the first 2500 epochs. This effect is observable for both CPU and memory resource.

4.3 Discussion

The simulation in this work-in-progress paper are admittedly limited. Both the simulator itself, and the simulated scenario can be improved. Nevertheless, we believe that these initial results reveal the potential of live migration in fast networking environments. In future work, we plan to extend our simulator to include more realistic task scheduling and migration policies and also account for the overhead of live migration. This will require finer-grained resource usage statistics since the 5-minute interval of the Google Cluster Trace does not allow exact modeling of performance degradation under overload or live migration scenarios.

5 Conclusion

In this paper, we have shown that new and faster network technologies allow for a renaissance of live migration for load balancing and, thus, better resource utilization in data centers. Using real-world task traces provided by Google, our simulations show that live migration has the potential to significantly improve overall resource utilization, task completion time, and task turnaround time. For future work, we plan to extend our simulator into a full-fledged cluster simulator to serve as a test bed for exploring new algorithms and policies for live migration, job placement, and overcommitment.

Acknowledgments

We thank the anonymous reviewers for their helpful feedback and suggestions. This work was supported by the Korean government (MSIT) through the National Research Foundation by grants 0536-20210093 and 21A20151113068 (BK21 Plus for Pioneers in Innovative Computing - Dept. of Computer Science and Engineering, SNU). ICT at Seoul National University provided research facilities for this study.

References

1. New AWS auto scaling unified scaling for your cloud applications (May 2009), https://aws.amazon.com/blogs/aws/aws-auto-scaling-unified-scaling-for-your-cloud-applications/
2. Ari, I., Hong, B., Miller, E.L., Brandt, S.A., Long, D.D.: Managing flash crowds on the internet. In: 11th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer Telecommunications Systems, 2003. MASCOTS 2003. pp. 246–249. IEEE (2003)

3. Azure: Azure public dataset - trace analysis.ipynb (Oct 2017), https://github.com/Azure/AzurePublicDataset/blob/master/analysis/Azure%20Public%20Dataset%20-%20Trace%20Analysis.ipynb
4. Azure: Azure 2019 public dataset v2 - trace analysis.ipynb (Jul 2019), https://github.com/Azure/AzurePublicDataset/blob/master/analysis/Azure%202019%20Public%20Dataset%20V2%20-%20Trace%20Analysis.ipynb
5. Beloglazov, A., Buyya, R.: Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints. IEEE transactions on parallel and distributed systems **24**(7), 1366–1379 (2012)
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 (2005)
7. Cortez, E., Bonde, A., Muzio, A., Russinovich, M., Fontoura, M., Bianchini, R.: Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms. In: Proceedings of the 26th Symposium on Operating Systems Principles. pp. 153–167 (2017)
8. Ghosh, R., Naik, V.K.: Biting off safely more than you can chew: Predictive analytics for resource over-commit in iaas cloud. In: 2012 IEEE Fifth International Conference on Cloud Computing. pp. 25–32. IEEE (2012)
9. Guo, J., Chang, Z., Wang, S., Ding, H., Feng, Y., Mao, L., Bao, Y.: Who limits the resource efficiency of my datacenter: An analysis of alibaba datacenter traces. In: 2019 IEEE/ACM 27th International Symposium on Quality of Service (IWQoS). pp. 1–10. IEEE (2019)
10. Jo, C., Cho, Y., Egger, B.: A machine learning approach to live migration modeling. In: ACM Symposium on Cloud Computing. SoCC'17 (September 2017)
11. Jo, C., Kim, H., Egger, B.: Instant virtual machine live migration. In: International Conference on the Economics of Grids, Clouds, Systems, and Services. pp. 155–170. Springer (2020)
12. Kansal, N.J., Chana, I.: Cloud load balancing techniques: A step towards green computing. IJCSI International Journal of Computer Science Issues **9**(1), 238–246 (2012)
13. Rzadca, K., Findeisen, P., Swiderski, J., Zych, P., Broniek, P., Kusmierek, J., Nowak, P., Strack, B., Witusowski, P., Hand, S., et al.: Autopilot: workload autoscaling at google. In: Proceedings of the Fifteenth European Conference on Computer Systems. pp. 1–16 (2020)
14. Subramanian, N., Jeyaraj, A.: Recent security challenges in cloud computing. Computers & Electrical Engineering **71**, 28–42 (2018)
15. Technology, L.: Terabit ethernet: The new hot trend in data centers (Mar 2020), https://www.lanner-america.com/blog/terabit-ethernet-new-hot-trend-data-centers/
16. Tirmazi, M., Barker, A., Deng, N., Haque, M.E., Qin, Z.G., Hand, S., Harchol-Balter, M., Wilkes, J.: Borg: the next generation. In: Proceedings of the Fifteenth European Conference on Computer Systems. pp. 1–14 (2020)
17. Verma, A., Pedrosa, L., Korupolu, M., Oppenheimer, D., Tune, E., Wilkes, J.: Large-scale cluster management at google with borg. In: Proceedings of the Tenth European Conference on Computer Systems. pp. 1–17 (2015)
18. Zhang, X., Wu, T., Chen, M., Wei, T., Zhou, J., Hu, S., Buyya, R.: Energy-aware virtual machine allocation for cloud with resource reservation. Journal of Systems and Software **147**, 147–161 (2019)