

ClaTex: LaTeX as a Service on the Cloud

John Doe¹ and Jane Monroe²

¹Dept. Cloud, Cloud University ²Cloud National Labs

Submission Type: Research

Abstract

The time has come to offer LaTeX on the cloud.

1 Introduction

Virtualization is a basic technology for the cloud computing. The growing awareness of the advantage provided by virtualization technology is brought about by economic factors of scarce resources, government regulation and more competition.

Virtualization provides the ability of consolidation for virtual machines (VM). Multiple VMs share the same physical machine without any interference of each other. However, virtualization providers suppose VMs on the same physical machine are unlikely to use up the whole memory at the same time. So VMs are often configured more memory in total than the capacity of the physical machine. But on the worse case, VMs compete with each other leading to shortness of memory resources. Other technology like migration also provided by virtualization must start up soon. Otherwise, some VMs may suffer memory starvation and lose the performance of the application on top of that.

With the development of hardware, a typical host machine in a datacenter now packs tens to hundreds of VMs in order to maximize resource utilization. VMware ESXi hypervisor has increased the number of VMs supported per host from 32 to 1024 in recent years[7]. Memory as shared resources needs to be allocated on demand in order to full resource utilization but still the minimum QoS has to be guaranteed since users buy for it. VM providers should be able to cope with the situation when memory increases suddenly.

In virtualization environment, guest OS is transparent to the underlying hypervisor. There are only some basic monitoring statistics exposed to hypervisor such as VSZ and RSS in linux system. VSZ and RSS are coarse-grained statistics which did not reflect the memory usage in a certain period. In the past, working set theory[1] has been widely used to capture applications' memory needs. Working sets is defined as the set of all pages accessed by a process

over a given epoch. However, working set doesn't relate directly to the performance of an application. For example, if an application scans 100 pages sequentially. In a short period, the working set is 100 pages. But allocating 1 page or 100 pages gives the same memory hit ratio because the miss ratio is always 100%. MRC is the alternative to working set theory. It plots the miss rate against given physical memory. MRC may differ every second reflecting the memory requirement changes of the application which gives us a hint to dynamic allocate more or reclaim spare memory in order to utilize the host machine's resources better.

MRC theory is a general method to curve the miss rate and each part of the memory hierarchy in computer storage including cache, memory and etc. Recently many research focus on the cache MRC construction. Centaur[3] implements host-side SSD cache partition for VMs that provides both lower cache miss rate and better performance isolation and performance control for VM workloads. Like Centaur, Multi-Cache[5] presents multiple cache management on a set of storage devices and intelligently manages how caches are shared by competing VMs. Moirai[6] presents a tenant- and workload-aware system that allows data center providers to control their distributed caching infrastructure.

2 Motivation

Since the needs of oversold of memory and still high performance assurance of VMs. Memory prediction is still a valuable research area especially on the area of cloud computing. To relate the memory size and the performance, a traditional approach is to use page miss ratio curve (MRC). MRC shows the miss rate under given memory size. A way to track MRC is to use LRU stack algorithm[4]. But the time complexity and space usage is high which makes it hard to track MRC online.

Recently there are some breakthroughs to track MRC in efficient ways. Counter Stacks[10] use probabilistic counters to estimate approximate MRC at a fraction of the cost of traditional techniques. SHARDS[8] simply uses sampling to reduce the size of input trace. Both

approach reduces the time complexity and space cost in constructing MRC and makes it possible to online tracking. AET[2] describes a new kinetic model for MRC construction of LRU caches based on average eviction time(AET). AET runs in linear time asymptotically and uses sampling to minimize the space overhead.

These novel techniques track MRC in low cost. However, none of them makes it in practice on real time system especially on constructing online MRC for virtual machines in virtualization environment. MEB[9] uses optimized balancing tree to construct MRC in virtualization and dynamically balance memory allocation across all virtual machines on top of a single physical machine. However there are still 30% overhead. Intermittent Memory tracking which turns off the tracking system during steady periods cuts down the overhead to acceptable range. So online nonstop efficient MRC tracking in virtualization is still a challenge.

3 Design

4 Conclusion

References

- [1] P. J. Denning. The working set model for program behavior. *Communications of the Acm*, 26(1):43–48, 1968.
- [2] X. Hu, X. Wang, L. Zhou, Y. Luo, C. Ding, and Z. Wang. Kinetic modeling of data eviction in cache. In *2016 USENIX Annual Technical Conference (USENIX ATC 16)*, pages 351–364, Denver, CO, 2016. USENIX Association.
- [3] R. Koller, A. J. Mashtizadeh, and R. Rangaswami. Centaur: Host-side ssd caching for storage performance control. In *IEEE International Conference on Autonomic Computing*, pages 51–60, 2015.
- [4] R. L. Mattson, J. Gecsei, D. R. Slutz, and I. L. Traiger. Evaluation techniques for storage hierarchies. *Ibm Systems Journal*, 9(2):78–117, 1970.
- [5] S. Rajasekaran, S. Duan, W. Zhang, and T. Wood. Multi-cache: Dynamic, efficient partitioning for multi-tier caches in consolidated vm environments. In *IEEE International Conference on Cloud Engineering*, pages 182–191, 2016.
- [6] I. Stefanovici, E. Thereska, G. O’Shea, B. Schroeder, H. Ballani, T. Karagiannis, A. Rowstron, and T. Talpey. Software-defined caching: managing caches in multi-tenant data centers. In *ACM Symposium on Cloud Computing*, pages 174–181, 2015.
- [7] vSphere 6.0. Configuration maximums. <https://www.vmware.com/pdf/vsphere6/r60/vsphere-60-configuration-maximums.pdf>, 2017.
- [8] C. A. Waldspurger, N. Park, A. Garthwaite, and I. Ahmad. Efficient mrc construction with shards. In *Usenix Conference on File and Storage Technologies*, pages 95–110, 2015.
- [9] Z. Wang, X. Wang, F. Hou, Y. Luo, and Z. Wang. Dynamic memory balancing for virtualization. *Acm Transactions on Architecture & Code Optimization*, 13(1):2, 2016.
- [10] J. Wires, S. Ingram, Z. Drudi, N. J. A. Harvey, and A. Warfield. Characterizing storage workloads with counter stacks. In *Usenix Conference on Operating Systems Design and Implementation*, pages 335–349, 2014.