# MARS: Measurement-based Allocation of VM Resources for Cloud Data Centers

Chiwook Jeong, Taejin Ha, Jaeseon Hwang, Hyuk Lim, and JongWon Kim
School of Information and Communications
Gwangju Institute of Science and Technology (GIST), Republic of Korea
Email: hlim@gist.ac.kr

## ABSTRACT

High performance data centers use virtualization technique which enables each physical server machine to host multiple virtual machines (VMs) to achieve highly efficient resource utilization. In this paper, we propose a measurement-based approach for efficient allocation of virtualized resources in hyper-convergence environments where virtualized computing, networking, and storage resources are unified and converged. Using real-time measurements of service performance metrics, our proposed approach identifies the VM with the worst performance resulting from over-utilized resource, and gradually adjusts the amount of resources allocated to it in order to improve its performance. The results of empirical evaluations conducted indicate that our proposed approach can realize efficient resource allocation among VMs with varying resource demands.

## Categories and Subject Descriptors

C.4 [**PERFORMANCE OF SYSTEMS**]: Measurement techniques; K.6.2 [**MANAGEMENT OF COMPUTING AND INFORMATION SYSTEMS**]: Installation Management—*Performance and usage measurement; Pricing and resource allocation*

## Keywords

Hyper-convergence; cloud computing; resource allocation; performance measurement

## 1. INTRODUCTION

Rapidly increasing demands for the construction of cost effective data centers have led to a wide deployment of virtualization technology [1] and a development of new hyper-convergence technology [2]. Hyper-convergence is an advanced resource virtualization technique that enables the cost effective integrated management of virtualized resources such as computing, networking, and storage. Consequently, small and medium-sized data centers have adopted hyper-convergence technology in order to provide cost effective services.

Research on virtual resource allocation is categorized into two major approaches: reactive-based approaches [3–6] and predictive-based approaches [7–9]. In reactive-based approaches, virtual resources allocation is performed according to the latest monitored utilization values or pre-defined rules. VMware [3] proposed a distributed resource scheduler (DRS) that dynamically balances resource allocation among VMs by exploiting their resource utilization information. Amazon [4] also provides an *autoscaling* service that correlates with user demand by allocating the amount of resources and the time period specified by each paying user. Nathani *et al.* [5] proposed a planning-based deadline-sensitive scheduling algorithm that is based on the deadline and resource utilization information of VMs. Han *et al.* [6] proposed a cost-aware scaling algorithm for elastic resource allocation that considers both short-term and long-term workload variations.

In predictive-based approaches, resource allocation is performed in accordance with anticipated near-future needs. Song *et al.* [7] proposed a multi-tiered resource scheduling scheme called the global resource flowing algorithm (GRFA), which re-allocates resources to VMs according to their own resource-flowing model based on the anticipated number of requests for service. Weng *et al.* [8] proposed a tuning strategy that estimates the amount of resources and the workload on VMs using the modified Roth-Erev learning algorithm. Jiang *et al.* [9] proposed a cloud resource autoscaling scheme that predicts the number of requests and estimates future resource demands, with a trade-off between cost and latency.

## 2. PROPOSED RESOURCE ALLOCATION STRATEGY

To achieve higher resource utilization efficiency than that achieved by previous methods, we propose a measurement-based resource allocation strategy (MARS). The proposed method first directly measures service performance and then adjusts the virtual resources allocated to the VMs to increase the performance of the VM with the worst performance resulting from over-utilization of the assigned resources. In contrast to conventional approaches, which primarily use resource utilization and workload statistics, MARS uses real-time measurement of service-specific performance metrics.

Let us first consider a single physical server machine running $n$ VMs. Suppose that each VM provides a service with different demands for virtual resources such as CPU, memory, and network bandwidth. Because of the different re-


*CoNEXT Student Workshop'13*, December 9, 2013, Santa Barbara, CA, USA.

http://dx.doi.org/10.1145/2537148.2537161.

**Algorithm 1** Proposed resource allocation strategy

1: // *res*: CPU, memory, or network bandwidth.
2: **for** VM $i$=1 to $n$ **do**
3: $r(i) \leftarrow$ response time for service requests on VM $i$
4: $T_{res}(i) \leftarrow$ total over-utilized resource duration when $U_{res}(i) > U_{res}^{thr}$ on VM $i$
5: **end for**
6: $j \leftarrow \arg\max_i r(i)$
7: $res \leftarrow \max_{res} T_{res}(j)$
8: **for** VM $i$=1 to $n$ **do**
9: **if** $(U_{res}(i) < U_{res}^{thr})$ **then**
10: $R_{res}(i) \leftarrow (1 - \alpha_{res}) * R_{res}(i)$
11: $R_{res}(j) \leftarrow R_{res}(j) + \alpha_{res} * R_{res}(i)$
12: **end if**
13: **end for**
14: Resource re-allocation for all VMs in accordance with $R_{res}$.

source demands by the VMs, resource utilization per VM will be highly imbalanced if the resources are evenly distributed among the VMs. By using service performance measurements, we identify the response time and the over-utilized time for different resources as performance metrics. Then, on the basis of the real-time measurements, we select the VM with the worst performance and its highly over-utilized resource, because resource over-utilization is a dominant factor that causes significant degradation in service performance. Finally, if there are under-utilized resources in other VMs, we re-allocate them to the VM identified above in order to improve its performance.

Algorithm 1 is the pseudo-code for MARS. A central controller periodically invokes the resource allocation algorithm. The algorithm first measures the service response time and the total time during which the resource is over-utilized above the utilization threshold, $U_{res}^{thr}$. On the basis of the real-time measurements obtained, it selects the worst performing VM (denoted by $j$) and identifies the resource (denoted by $res$) that is being over-utilized for the longest time. Then, if another VM has the identified resource that is under-utilized below $U_{res}^{thr}$, i.e., the resource with a low resource demand, a certain portion $\alpha_{res}$ of that resource is inserted into the resource pool. Finally, the resource in the pool is reallocated to all the VMs in accordance with $R_{res}$.

## 3. PERFORMANCE EVALUATION

To evaluate the performance of MARS, we constructed a small cloud testbed using KVM [10], OpenStack [11], and OpenvSwitch [12], and compared MARS with the naive (1/n) scheme, which evenly distributes virtualized resources to $n$ VMs at initialization. The scenario used in the the experiment was that each VM is running a web server, which handles 10,000 HTTP requests. We used the *weighttp* tool to generate HTTP workloads for performance benchmark, the *virt-top* tool to measure the network resource, and the *top* tool to measure the CPU and memory resources. We also measured the average response time as a performance metric. The system parameters $\alpha_{res}$ and $U_{res}^{thr}$ for all the resources are set to 0.1 and 0.85, respectively.

Figure 1 depicts the worst and average response times with respect to the number of VMs. Note that the worst response time is that of the VM that has the longest response

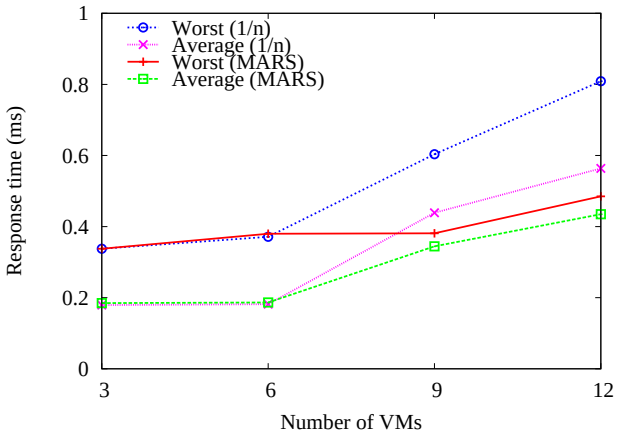


**Figure 1: Average response time with respect to the number of VMs.**

time among all the VMs. When the number of VMs is small (i.e., three and six), the worst and average performances of each scheme are almost the same because all the VMs have enough resources to serve their workload. As larger numbers of VMs are configured, the performance of MARS gets better than that of the (1/n) scheme because MARS identifies the VM with the worst performance and allocates more resources to it. It is seen that MARS achieves improvements of approximately 40 % and 21.5 % in the the worst and average performances, respectively, compared to the (1/n) scheme. Using MARS, when the number of VMs is large, the discrepancy between the worst and average performances is quite small, with a small average response time. This indicates that the resources are appropriately allocated among all the VMs with different resource demands.

## 4. CONCLUSIONS

In this paper, we proposed a measurement-based resource allocation strategy (MARS) that reallocates under-utilized resources to the VM with the worst performance in terms of the measured service performance and the length of time during which its resources are over-utilized. By means of experiments conducted on a small cloud testbed, we verified that MARS achieves efficient resource allocation by effectively utilizing computing and network resources on VMs. In future work, we plan to extend MARS to manage the storage resources in hyper-convergence environments, where the computing, networking, and storage resources are unified among VMs.

## 5. ACKNOWLEDGMENTS

This research was supported in part by National Information Society Agency (KOREN project 13-951-00-001), and by the Industrial Strategic Technology Development Program (10047577) funded by the Ministry of Science, ICT and Future Planning, Korea.

## 6. REFERENCES


[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. A view of cloud computing. *Communications of the ACM*, 53(4):50–58, 2010.

[2] Hyperconvergence for Virtualization. http://www.scalecomputing.com/files/documentation/hc3-whitepaper-hyperconvergence.pdf.

[3] VMware. http://www.vmware.com/.

[4] Amazon Autoscaling. http://aws.amazon.com/autoscaling/.

[5] A. Nathani, S. Chaudhary, and G. Somani. Policy based resource allocation in IaaS cloud. *Elsevier Future Generation Computer Systems*, 28(1):94–103, 2012.

[6] R. Han, M. M. Ghanem, L. Guo, Y. Guo, and M. Osmond. Enabling cost-aware and adaptive elasticity of multi-tier cloud applications. *Elsevier Future Generation Computer Systems*, 2012.

[7] Y. Song, H. Wang, Y. Li, B. Feng, and Y. Sun. Multi-tiered on-demand resource scheduling for VM-based data center. In *Proc. of IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (CCGrid)*, 2009.

[8] C. Weng, M. Li, Z. Wang, and X. Lu. Automatic performance tuning for the virtualized cluster system. In *Proc. of IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2009.

[9] J. Jiang, J. Lu, G. Zhang, and G. Long. Optimal Cloud Resource Auto-Scaling for Web Applications. In *Proc. of IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (CCGrid)*, 2013.

[10] Kernel-based Virtual Machine (KVM). http://www.linux-kvm.org/.

[11] OpenStack. http://www.openstack.org/.

[12] Open vSwitch. http://openvswitch.org/.