Demo Abstract: ACTiManager: An end-to-end interference-aware cloud resource manager

Stratos Psomadakis, Stefanos Gerangelos, Dimitrios Siakavaras, Ioannis Papadakis, Marina Vemmou, Aspa Skalidi, Vasileios Karakostas, Konstantinos Nikas, Nectarios Koziris, Georgios Goumas
National Technical University of Athens, ICCS
actimanager@cslab.ece.ntua.gr

ABSTRACT

Cloud service providers (CSPs) rely mostly on simplistic and conservative policies regarding resource management, to minimize interference of shared resources between multiple VMs and to provide acceptable performance. However, such approaches may lead to sub-optimal allocation and resource underutilization. In this demonstration we present ACTiManager, an end-to-end interference-aware manager for cloud resources. Our preliminary results compared to vanilla OpenStack are promising in terms of CSPs' profit while also retaining average user's satisfaction in the set of top priorities.

CCS CONCEPTS

• **Computer systems organization** → **Cloud computing**; • **Software and its engineering** → **Scheduling**.

KEYWORDS

cloud computing, resource management, interference-aware

ACM Reference Format:
Stratos Psomadakis, Stefanos Gerangelos, Dimitrios Siakavaras, Ioannis Papadakis, Marina Vemmou, Aspa Skalidi, Vasileios Karakostas, Konstantinos Nikas, Nectarios Koziris, Georgios Goumas. 2019. Demo Abstract: ACTiManager: An end-to-end interference-aware cloud resource manager. In *20th International Middleware Conference Demos and Posters (Middleware Demos and Posters '19), December 9–13, 2019, Davis, CA, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3366627.3368114

1 INTRODUCTION

Motivation. The cloud paradigm has created a rapid technology shift from small, in-house private infrastructures to large-scale, public or private datacenters. To accommodate the unprecedented demand for cloud computing and maximize clients' satisfaction, cloud service providers (CSPs) have been relying on continuous infrastructure expansions and on simplistic policies to manage resources. While this was a reasonable strategy for early adoption, it fails to sustain the growing demands for cloud computing. Recent studies have shown that resource utilization ranges from 10% to 50% [2, 6] in most cloud datacenters.

Research Problem. The root cause for resource waste is the conservative allocation policies of CSPs. The co-existence of multiple workloads within the same physical server can create performance degradation and unpredictability due to contention in shared resources (e.g., CPUs, caches, memory links, etc.) [1, 4, 5, 7, 10, 13, 15]. A straightforward and frequently preferred solution attempt is to completely isolate latency-critical virtual machines (VMs) either by granting them an entire physical server or large parts of it (e.g., a NUMA node [12]). Clearly, such attempts leave precious resources underutilized in the fear of interference.

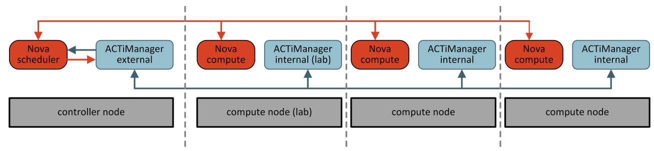

Figure 1: ACTiManager system architecture.

Related Work. The main approaches towards dealing with interference are *avoidance*, i.e., allocate resources avoiding interference, and *mitigation*, i.e., apply isolating actions when interference is detected. Several research resource managers have incorporated interference avoidance and/or mitigation in their operations [1, 3, 6, 8, 9, 11, 14]. However, those systems are based on a number of constraints that limit their applicability to operational cloud environments. In particular, some systems rely on offline profiling information that need to be collected before VM execution [1, 8, 11, 12, 14], and/or require the availability of a Quality-of-Service (QoS) reporting mechanism from the applications [6, 14]. In addition, some systems lack support for multi-phase applications [11], or need to apply online probes to support accurate predictions [9, 14]. Finally, prior systems either seek to guarantee the performance of latency-critical VMs or to increase the total system utilization without considering how application performance translates to profit from the CSP perspective.

Our goal is a practical interference-aware resource manager that: (i) performs both interference avoidance and interference mitigation, (ii) requires no offline application profile, (iii) operates in an application transparent way, (iv) avoids artificial interference through probes, and (v) optimizes for datacenter profit.

Contribution. In this demonstration we present *ACTiManager*, an end-to-end interference-aware cloud resource manager. With ACTiManager, VMs start their lifetime in a cloud environment within a protected, *isolated island*, which typically is a NUMA socket. This unleashes the potential to collect online information in an interference-free environment. We characterize the behavior of the VM using machine-learning techniques and create the *resource fingerprint* of the VM regarding its potential to create or suffer from interference. Similarly, we build a model to describe its *healthy*

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
Middleware Demos and Posters '19, December 9–13, 2019, Davis, CA, USA
© 2019 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-7042-4/19/12.
https://doi.org/10.1145/3366627.3368114

state which later on will be utilized to detect interference. From that point on, the VM can be placed in a symbiotic environment and proceed with its execution under ACTiManager's supervision.

2 RESEARCH & TECHNICAL APPROACH

Overview. As shown in Figure 1, ACTiManager follows a two-tier approach and comprises two components: *ACTiManager.external* and *ACTiManager.internal*. *ACTiManager.external* interfaces with the cloud operating system and decides the placement of VMs on the available servers, maintaining a global view of the datacenter and targeting coarse-grain workload consolidation and interference avoidance. *ACTiManager.internal* operates within each server and enforces fine-grain workload consolidation, targeting both interference avoidance and mitigation. *ACTiManager.internal* comes in two flavors: (i) the *laboratory* version, which analyzes newly arrived VMs in an environment free of interference, monitors them, characterizes them in terms of their potential to create or suffer from interference, and creates a model that describes the VM's healthy state; (ii) the *standard* version, which pins the VMs in specific cores, monitors them, and takes actions (e.g., re-pinning or informing *ACTiManager.external*) in case of anomaly detection.

Target Optimization Function. ACTiManager differentiates from previous resource managers on the target function it applies to guide its decision making. Prior work [6, 14] distinguishes between *latency-critical* and *best effort* VMs, and guarantees the QoS of the former while attempting to co-locate them with the latter to maximize utilization. ACTiManager still prioritizes VMs based on their Service Level Objectives (SLOs), but decisions are made instead by combining pricing models with estimation of VMs' slowdown due to co-location. Thus, ACTiManager is flexible enough to support numerous optimization policies, including maximizing CSPs' profit.

Interference Avoidance and Mitigation. In the *laboratory* version of *ACTiManager.internal*, the first goal is to create the resource fingerprint of each VM by coarsely classifying it as noisy/quiet and sensitive/insensitive (two labels per VM). The second goal is to create a healthy state model for each VM so that any anomaly in the VM's performance can be detected during its execution. To achieve these goals, we build two classifiers, one for each label, and an anomaly detection model, based on machine-learning techniques. Then, for each VM we collect measurements from the system's performance counters while the VM runs in the isolated environment of the *laboratory* version. We feed these measurements to the two classifiers to produce the two labels which will serve towards fulfilling the goal of interference avoidance. Subsequently, we use the same measurements to train the VM's healthy state model whose goal is to assist ACTiManager in identifying interference incidents and perform the appropriate actions to mitigate interference.

VM lifecycle. Once a user creates a VM, ACTiManager places it on a laboratory node for *ACTiManager.internal (lab)* to analyze it. Note that with this placement, the VM starts executing at full speed. Upon completion of the characterization, *ACTiManager.external* is notified and selects to place (migrate) the VM on a physical server, together with other VMs in a symbiotic environment. *ACTiManager.internal* pins the newly arrived VM to the appropriate cores of the selected server and periodically monitors its behavior. If a performance anomaly is detected, *ACTiManager.internal* attempts to place the VM in different cores within the server. If unsuccessful, *ACTiManager.external* is informed that a server overload has occurred and migrates a VM to a different node. When faced with the decision of where to pin a VM, *ACTiManager.external/internal* attempts to find the most suitable set of servers/cores by considering all the VMs' resource fingerprints. In particular, a score is calculated for each server/core, which indicates a combined estimated slowdown; looping over the requested and available resources, the server/core with the minimum score will be selected.

Evaluation. We are currently in the process of performing extensive evaluation of ACTiManager and comparison with other approaches. Our preliminary results are promising, achieving increased CSPs' profit while respecting workload prioritization.

3 CONCLUSIONS

Improving the resource efficiency of datacenters is of paramount importance due to economies of scale. Towards that goal, we presented ACTiManager that enables efficient resource management.

ACKNOWLEDGMENTS

This research has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement no. 732366 (ACTiCLOUD).

REFERENCES

[1] Christina Delimitrou and Christos Kozyrakis. 2013. Paragon: QoS-aware Scheduling for Heterogeneous Datacenters. In *ASPLOS'13*.
[2] Christina Delimitrou and Christos Kozyrakis. 2014. Quasar: Resource-efficient and QoS-aware Cluster Management. In *ASPLOS'14*.
[3] Tyler Dwyer, Alexandra Fedorova, Sergey Blagodurov, Mark Roth, Fabien Gaud, and Jian Pei. 2012. A practical method for estimating performance degradation on multicore processors, and its application to hpc workloads. In *SC'12*.
[4] Alexandros Haritatos, Georgios Goumas, Nikos Anasopoulos, Konstantinos Nikas, Kornilios Kourtis, and Nectarios Koziris. 2014. LCA: A memory link and cache-aware co-scheduling approach for CMPs. In *PACT'14*.
[5] Ram Srivatsa Kannan, Animesh Jain, Michael Laurenzano, Lingjia Tang, and Jason Mars. 2018. Proctor: Detecting and Investigating Interference in Shared Datacenters. In *ISPASS'18*.
[6] David Lo, Liqun Cheng, Rama Govindaraju, Parthasarathy Ranganathan, and Christos Kozyrakis. 2015. Heracles: Improving resource efficiency at scale. In *ACM SIGARCH Computer Architecture News*, Vol. 43. ACM, 450–462.
[7] Andreas Merkel, Jan Stoess, and Frank Bellosa. 2010. Resource-conscious scheduling for energy efficiency on multicore processors. In *Proceedings of the 5th European conference on Computer systems*. ACM, 153–166.
[8] Ripal Nathuji, Aman Kansal, and Alireza Ghaffarkhah. 2010. Q-clouds: Managing Performance Interference Effects for QoS-aware Clouds. In *EuroSys '10*.
[9] Dejan Novaković, Nedeljko Vasić, Stanko Novaković, Dejan Kostić, and Ricardo Bianchini. 2013. DeepDive: Transparently Identifying and Managing Performance Interference in Virtualized Environments. In *USENIX ATC'13*.
[10] Ioannis Papadakis, Konstantinos Nikas, Vasileios Karakostas, Georgios Goumas, and Nectarios Koziris. 2017. Improving QoS and Utilisation in modern multi-core servers with Dynamic Cache Partitioning. In *COSH 2017*.
[11] Francisco Romero and Christina Delimitrou. 2018. Mage: Online and Interference-aware Scheduling for Multi-scale Heterogeneous Systems. In *PACT'18*.
[12] Alan Roytman, Aman Kansal, Sriram Govindan, Jie Liu, and Suman Nath. 2013. PACMan: Performance aware virtual machine consolidation. In *Proceedings of the 10th International Conference on Autonomic Computing ({ICAC} 13)*. 83–94.
[13] Lingjia Tang, Jason Mars, and Mary Lou Soffa. 2011. Contentiousness vs. sensitivity: improving contention aware runtime systems on multicore architectures. In *EXADAPT '11*.
[14] Hailong Yang, Alex Breslow, Jason Mars, and Lingjia Tang. 2013. Bubble-flux: Precise Online QoS Management for Increased Utilization in Warehouse Scale Computers. In *ISCA '13*.
[15] Sergey Zhuravlev, Sergey Blagodurov, and Alexandra Fedorova. 2010. Addressing Shared Resource Contention in Multicore Processors via Scheduling. In *ASPLOS'10*.