Application Aware Placement and Scheduling for Multi-tenant Clouds

Lalith Suresh[1] and Marco Canini[2]

[1]Technische Universität Berlin, Germany
[2]Université Catholique de Louvain, Belgium
{lsuresh@inet.tu-berlin.de, marco.canini@uclouvain.be}

Abstract

In an Infrastructure-as-a-Service (IaaS) environment, it is paramount to perform intelligent allocation of shared resources. Placement *is the problem of choosing which virtual machine (VM) should run on which physical machine (PM), whereas* Scheduling *is the problem of sharing resources between multiple co-located VMs. An efficient placement and scheduling is one, that in addition to satisfying all constraints, increases the overall utilization of physical resources such as CPU, storage, or network. Determining an efficient placement and scheduling is a very challenging problem, especially in face of conflicting goals and partially available information about workloads.*
In order to reason about placement, we first tackle the problem of performance interference that may affect co-located VMs—when there is more demand by multiple VMs for a resource than is available at a given instant of time. We thus characterize the performance of Hadoop in a shared and virtualized setting.

1. Introduction

The focus of our study is multi-tenant, multi-purpose Infrastructure-as-a-Service (IaaS) cloud data centers, wherein servers are virtualized, with multiple tenants deploying applications atop shared infrastructure. Utilizing compute, storage and network resources efficiently is a crucial objective in these environments as virtual machines (VMs) are collocated on physical servers and contend for a set of shared resources on the physical server and network.
However, it is well-known that virtualized computing environments can suffer from *performance interference* (see e.g., [4]). More precisely, due to imperfect performance isolation by the underlying hypervisors and lack of fine-grained bandwidth assurances across tenants within the network, the performance of a VM may suffer due to collocated VMs. Such performance interference can lead to unpredictable delays for the cloud tenant, which would ultimately manifest in failed service-level agreements (SLAs) at the application level and in loss of business revenue.

To mitigate performance interference, it is thus important to balance the different workloads of various tenants across the data center. While VMs from different tenants may be collocated on the same physical host, different tenants in the data center may have varying demands for resources depending on the applications that they are running. For instance, a tenant running a Hadoop [3] deployment on the cloud may have a greater demand for disk I/O bandwidth than a tenant managing a multi-tier website. Depending on the resource demands of different applications being run by different tenants, VMs may experience varying degrees of performance interference due to contention for shared resources. Thus, the performance degradation experienced by different VMs is sensitive to the specific placement of VMs in the data center [7].

Since this performance degradation depends on the applications being run within the VMs, understanding how applications behave and where their bottlenecks are may allow an operator to perform smarter placement of VMs within the data center. This project takes a fresh look at the fundamental problem of application placement in the private cloud environment. Application placement is the task of deciding which VM runs on which physical server. There have been proposals to address application placement from both theoretical and systems standpoints. For instance, theoretical approaches model the environment as a multi-dimensional bin packing problem [8] or provide schedules based on statistical models [1]. However, these models are too coarse to accurately account for performance interference, which require fine-grained experimentation to observe. On the other hand, [5, 6, 2, 9] propose techniques to identify performance interference by observing low-level system metrics on individual VMs and then making placement decisions to minimize it or allocating more resources to compensate for interference. However, these approaches do not exploit the specific characteristics and goals of the applications.

In fact, in today's cloud ecosystem, most applications are distributed across more than one VM. With service-oriented architectures becoming the norm, systems are decomposed into multiple, loosely coupled, communicating clusters of components. An individual logical component in an application could by itself be a self-healing system comprised of multiple nodes, which have built-in mechanisms to compensate for stragglers or slower nodes (e.g, MapReduce's scheduler). Taking these aspects into consideration, we argue that performance interference should be reasoned about at the granularity of *logical components of the application* using application-specific metrics, and not at the granularity of the individual VMs using system-level metrics. This project aims to demonstrate this fact experimentally.

In order to design *application- and interference-aware approaches* to plan application placement in multi-tenant clouds, we first try to characterize the performance of different systems under multi-tenant environments. This information can then be used to effectively place applications in a cloud, as show in the methodology illustrated in Figure 1. In the findings presented below, we present performance results of running Hadoop in shared environments.

2. Usage of Future SOC Lab

HPI FSOC Lab provided us access to a state-of-the art, 1000-core computing cluster. The cluster consists of 25 nodes, each equipped with 40 cores at 2.40 GHz, 1 TB RAM, 3.6 TB SSD storage and 10 Gbps Ethernet. To use this cluster for our experimental study, we needed it to be configured as a virtualized cloud environment, and we obtained simultaneous, dedicated access to all nodes.

We ran experiments of the Hadoop cluster under multiple configurations and co-location scenarios.

- Baseline: every VM of the Hadoop cluster is running on a separate physical machine.
- Co-located Datanodes: two Hadoop Datanode VMs from the same cluster are co-located each physical machines.
- Co-located Clusters: a VM each from two different Hadoop clusters are co-located on the same physical machine.

Through our runs, we vary the number of reducers used by Hadoop, the Hadoop scheduler being used (Capacity and Fair schedulers), and we enable/disable speculative execution (wherein tasks are speculatively cloned to account for stragglers). We use the following notations to describe the combinations of these parameters. *cpt* and *cpf* represent the use of the Capacity Scheduler along with speculative execution enabled and disabled, respectively. *fst* and *fsf* represent the use of the Fair Scheduler along with speculative execution enabled and disabled, respectively. The workload we run is the TeraSort benchmark that is packaged with the Hadoop distribution, with four jobs submitted in parallel.

3. Findings

In this section, we describe some of our findings.

Figure 2 represents measurements of the difference between the maximum and minimum job completion times when submitting four TeraSort jobs to the cluster at the same time, under varying scenarios. We use this as an indication of fairness in the system. We find that the baseline and co-located Datanodes scenarios provide identical fairness when using different schedulers and regardless of whether speculative execution is enabled or disabled. However, in the co-located clusters scenario, we observe that due to performance interference from each cluster on the other, the system is unable to guarantee fairness in job completion times even when using the Hadoop Fair Scheduler. Figure 3 describes the same set of results but when using 10 reducers instead of 1. Since Hadoop can now distribute the reduce tasks over multiple nodes, the skew in job completion times is minimized, but still remains significant in the co-located clusters scenario.

One challenge we faced in obtaining more results and measurements was that the specifications of the physical machines being used deviated significantly from those used in typical Infrastructure-as-a-Service deployments, where servers have a different CPU cores to I/O bandwidth ratio.

4. Summary and Future Work

Our studies indicate that performance interference not only affects the job completion times of Hadoop directly, but also leads to interference in the Hadoop schedulers, affecting the guarantees they are expected to provide (such as fairness). We plan to use these measurements in order to prepare measurement-driven interference models which can in turn be used as input to placement algorithms which can be used to map VMs to physical machines in cloud environments. Our study will also extend to other systems such as key-value stores and the interactions between these systems.

5. Acknowledgements

We would like to thank the system administrators of the Future SOC-Lab infrastructure for their support, in

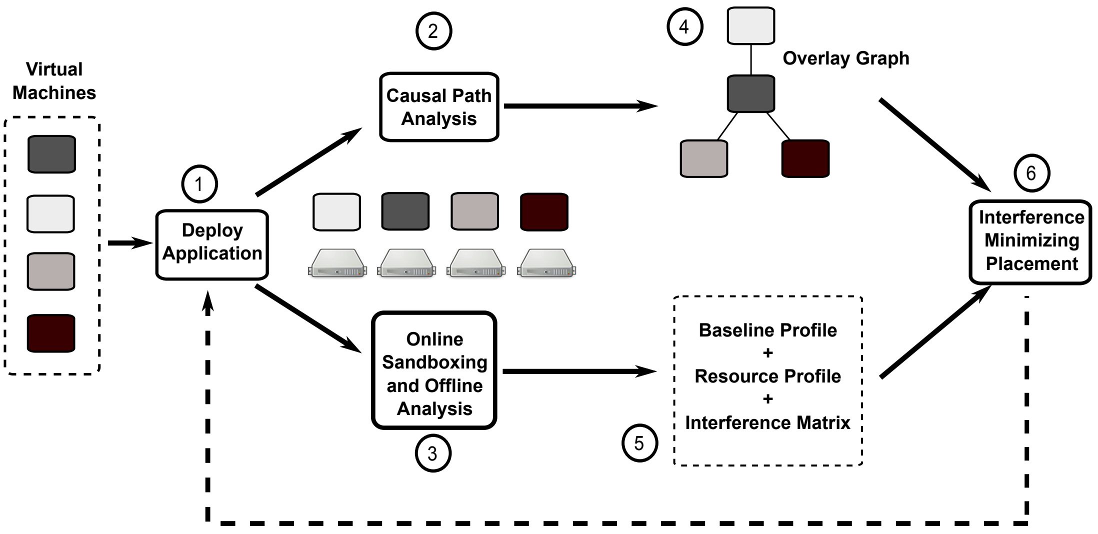
Virtual Machines
1
Deploy Application
2
Causal Path Analysis
3
Online Sandboxing and Offline Analysis
4
Overlay Graph
5
Baseline Profile + Resource Profile + Interference Matrix
6
Interference Minimizing Placement

Figure 1. At a high level, our workflow combines online and offline measurements of applications with specific information about the application's internals in order to perform interference minimizing placement.

particular with setting up OpenStack on the machines. We also thank Zubair Khwaja Sediqi for help with the setup of the measurement code and workloads.

References

[1] R. C. Chiang and H. H. Huang. Tracon: interference-aware scheduling for data-intensive applications in virtualized environments. In *Proc. ACM SC*, 2011.

[2] D. Novaković et al. Deepdive: Transparently identifying and managing performance interference in virtualized environments. In *Proc. USENIX ATC*, 2013.

[3] A. Hadoop. `http://hadoop.apache.org/`, last accessed Dec 20th, 2013.

[4] Y. Koh, R. Knauerhase, P. Brett, M. Bowman, Z. Wen, and C. Pu. An analysis of performance interference effects in virtual environments. In *Performance Analysis of Systems & Software, 2007. ISPASS 2007. IEEE International Symposium on*, pages 200–209. IEEE, 2007.

[5] Y. Koh, R. Knauerhase, P. Brett, M. Bowman, Z. Wen, and C. Pu. An analysis of performance interference effects in virtual environments. In *Proc. IEEE ISPASS*, 2007.

[6] R. Nathuji, A. Kansal, and A. Ghaffarkhah. Q-clouds: managing performance interference effects for qos-aware clouds. In *Proc. EuroSys*, 2010.

[7] A. Roytman, A. Kansal, S. Govindan, J. Liu, and S. Nath. Algorithm design for performance aware vm consolidation.

[8] B. Urgaonkar, A. Rosenberg, and P. Shenoy. Application placement on a cluster of servers. *Int. J. Found. Comput. Sci.*, 2007.

[9] Y. Xu, Z. Musgrave, B. Noble, and M. Bailey. Bobtail: avoiding long tails in the cloud. In *Proc. NSDI*, 2013.

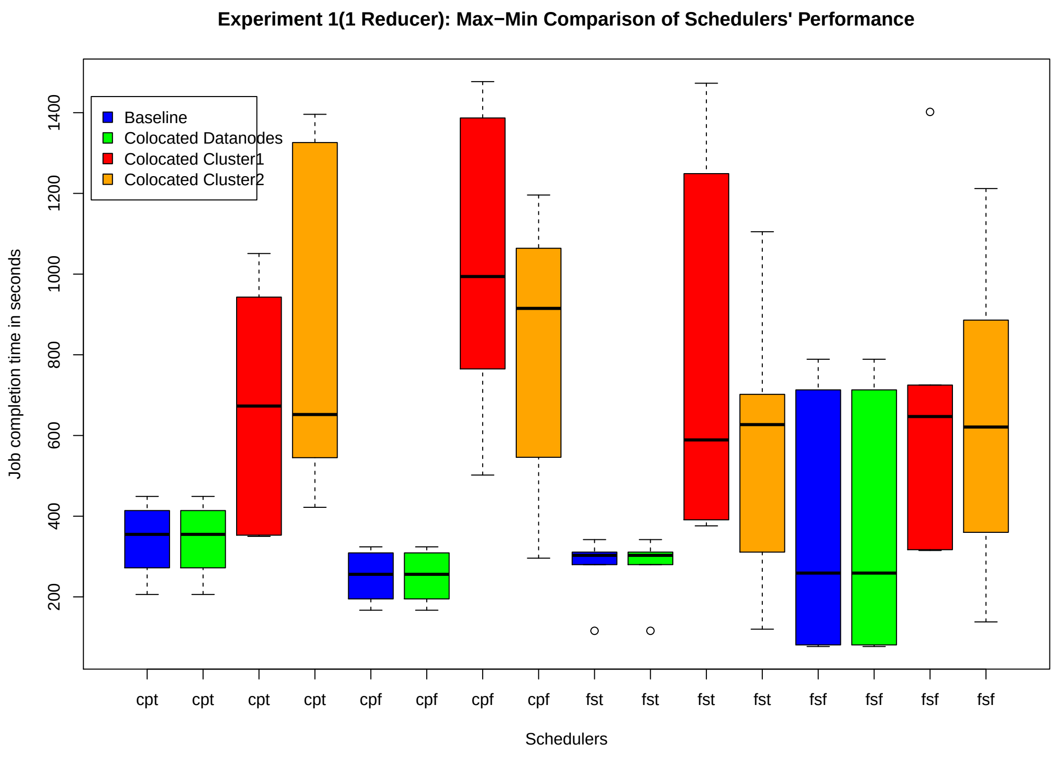

Figure 2. When using the capacity and fair scheduler for Hadoop, the systems are unable to guarantee fairness of job completion times when running as co-located clusters due to performance interference.

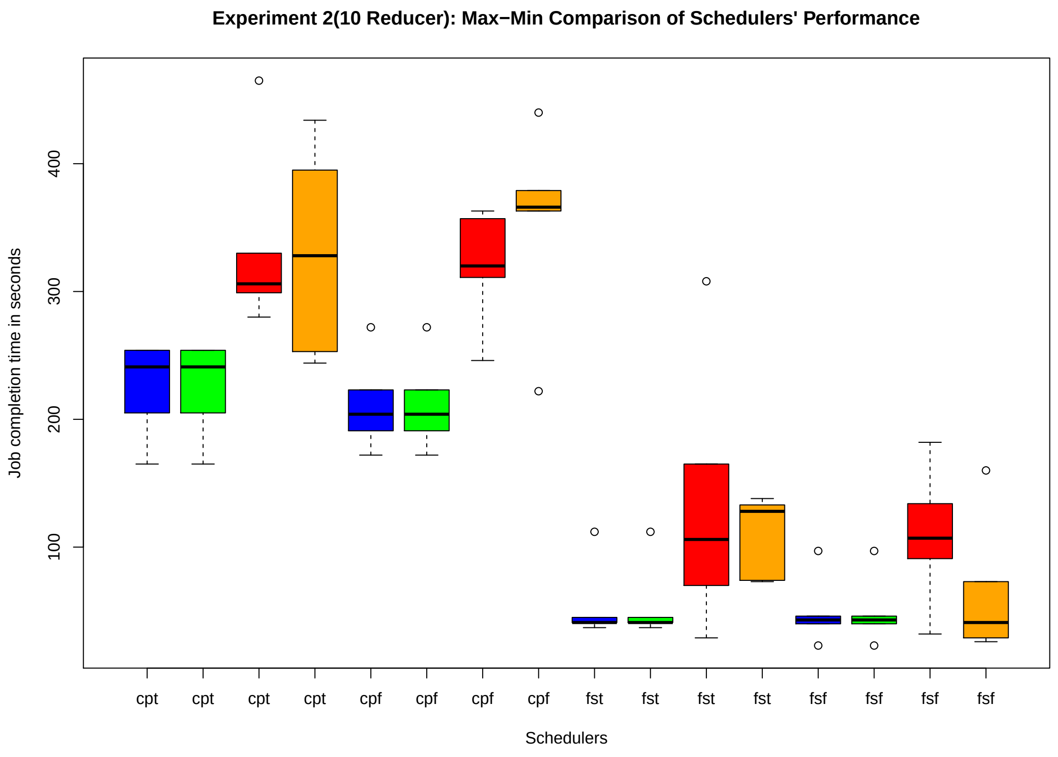

Figure 3. As with the single reducer case, the systems are unable to guarantee fairness of job completion times when running as co-located clusters. However, using more reducers reduces the skew in job completion times.