Interference-aware Scheduling for Data-processing Frameworks in Container-based Clusters

Miguel G. Xavier and Cesar A. F. De Rose
Pontifical Catholic University of Rio Grande do Sul (PUCRS)
Faculty of Informatics, Porto Alegre, Brazil

***Abstract*—With the emergence of data-processing frameworks like Hadoop and Spark, a new concept of a cluster resource manager was necessary to deliver per-application container-wrapped on-demand resources with high scalability on a large scale. The container-based ”Big Data” Operating System concept arose (e.g. YARN and Mesos) and brought along with it the long-standing inter-instance performance interference issues from virtualization technologies. As a result, the performance interference effects between co-located data-processing applications become an uncertain issue in container-based clusters, leading performance to fluctuate unpredictably and the guarantees to be likely violated. To work around this, an interference-aware scheduling algorithm is necessary to mitigate interference-related performance degradation and wisely schedule tasks on the best-suited compute nodes—the nodes whose performance is maximized and the makespan is minimized.**

I. Introduction

As the popularity of large-scale data analysis increases, the emergence of data-processing frameworks and programming models beyond MapReduce also grows. For example, Storm, Tez and Spark that consolidate a new applications' ecosystem, referred to as data-processing applications. Nothing prevents the same or different data from being simultaneously accessed by multiple applications from different frameworks in a cluster. This requires a holistic, centralized entity to manage cluster's resource slots and schedule not just MapReduce-centric application tasks, but an entire ecosystem of data-processing application. YARN and Mesos are the next-generation ”Big Data” Operating Systems. They implement a multi-tenancy platform on which tasks from multiple frameworks are scheduled and deployed onto the same shared physical cluster of computers. Virtualization based on containers is the key enabling technology behind such systems.

II. Problem Statement

Despite offering high-performance levels and a lightweight alternative to the traditional hypervisor-based systems, such as Xen, container-based virtualization raised the inter-virtual instance/-(VI) performance interference problem of traditional virtualization technologies. Inter-VI performance interference happens when two co-located VIs (i.e. provisioned in the same physical server) share the same hardware (e.g. CPU, cache, memory, I/O buffer, etc) and due to resource contention the VIs adversely affect the performance between one another. The problem may be even worse in data operating systems. Having many co-located containers may cause unexpected performance variations due to framework-specific resource demands and the workload that fluctuates unpredictably. For instance, Spark is an interactive query backed framework that performs in-memory computing; Stream is a framework that produces CPU-/memory-bound applications to process unbounded streams of data in real time. These load variations may compromise the efficiency and effectiveness of the traditional task schedulers (FIFO/Fair/Capacity), and affect the overall job performance. Unexpected task slowdown caused by interference in a cluster of multiple frameworks may render traditional task schedulers no longer effective. As a result, the performance guarantee is susceptible to violation, impacting on the jobs makespan and users experience.

III. Proposal Statement

We propose a novel interference-aware scheduler algorithm for data-processing applications in container-based clusters. The scheduler determines which node affects task performance the least and then deploys there. The scheduler's decision rely on an interference prediction model, which estimates interference influences per task for each node in the cluster.

We conducted a set of preliminary experiments in a controlled container-based environment in order to assess interference-related performance degradation a disk-intensive workload tolerates from other co-located workloads. We intend to ascertain any perturbations in database operations while handling transaction-oriented floods carried from micro-benchmarks.

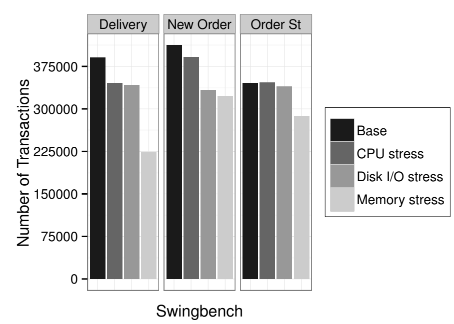

Fig. 1. Inter-VI performance interference on LXC sourced from co-located micro-benchmarks (CPU-/Disk-/Memory-intensive). Base workload means that application run alone on the machine.

We believe that interferences in LXC (Figure 1) are due to the shared OS that had to handle much more instruction during high load conditions, being unable to efficiently handle the database transactions. Theses results are the first steps and show the interference level (about 38% performance overhead in the worst case) a disk-intensive application can suffer when co-located on the same machine. We plan to profile applications by interference level and use the results as input for our prediction model.