Resource Profile Advisor for Containers in Cognitive Platform

Mehmet F. Aktaş[⋆], Chen Wang[†], Alaa Youssef[†], Malgorzata (Gosia) Steinder[†]

[⋆]Rutgers University, [†]IBM Thomas J. Watson Research Center
mehmet.aktas@rutgers.edu,chen.wang1@ibm.com,{asyousse,steinder}@us.ibm.com

ABSTRACT

Containers have transformed the cluster management into an application oriented endeavor, thus being widely used as the deployment units (i.e., micro-services) of large scale cloud services. As opposed to VMs, containers allow for resource provisioning with fine granularity and their resource usage directly reflects the micro-service behaviors. Container management systems like Kubernetes and Mesos provision resources to containers according to the capacity requested by the developers. Resource usages estimated by the developers are grossly inaccurate. They tend to be risk-averse and over provision resources, as under-provisioning would cause poor runtime performance or failures.

Without actually running the workloads, resource provisioning is challenging. However, benchmarking production workloads at scale requires huge manual efforts. In this work, we leverage IBM Monitoring service to profile the resource usage of production IBM Watson services in rolling windows by focusing on both evaluating how developers request resources and characterizing the actual resource usage.

Our resource profiling study reveals two important characteristics of the cognitive workloads. 1. **Stationarity**. According to Augmented Dickey-Fuller test with 95% confidence, more than 95% of the container instances have stationary CPU usage while more than 85% have stationary memory usage, indicating that resource usage statistics do not change over time. We find for the majority of containers that the stationarity can be detected at the early stage of container execution and can hold throughout their lifespans. In addition, containers with non-stationary CPU or memory usage are also observed to implement predictable usage trends and patterns (e.g., trend stationarity or seasonality). 2. **Predictability by container image**. By clustering the containers based on their images, container resource usages within the same cluster are observed to exhibit strong statistical similarity. This suggests that the history of resource usage for one instance can be used to predict usage for future instances that run the same container image.

Based on profiling results of running containers in rolling windows, we propose a resource usage advisory system to refine the requested resource values of the running and arriving containers as illustrated in Fig. 1. Our system continuously retrieves the resource usage metrics of running containers from IBM monitoring service and predicts the resource usage profiles in a *container resource usage prediction agent*. Upon the arrival of a new pod[1], the *resource profile advisor*, proposed as a module in the web-hooked admission controller in Kubernetes, checks whether the resource profile of each container in the pod has been predicted with confidence. If a container's profile has been predicted and cached in the *container resource profile database*, the default requested values of containers are refined by the predicted ones; otherwise, containers are forwarded to the scheduler without any change. Similarly, a *resource profile auto-scaler* is proposed to update the requested resource values of containers for running pods[2] as soon as the database is updated.

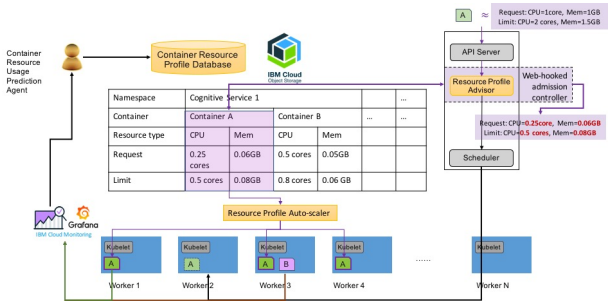

Figure 1: System overview

Our study shows that developers request at least 1 core-per-second (cps) CPU and 1 GB memory for ≥ 70% of the containers, while ≥ 80% of the containers actually use less than 1 cps and 1 GB. Additionally, ∼ 20% of the containers are significantly under provisioned. We use resource usage data in one day to generate container resource profiles and evaluate our approach based on the actual usage on the following day. Without our system, average CPU (memory) usage for >90% of containers lies outside of 50% - 100% (70% - 100%) of the requested values. Our evaluation shows that our system can advise request values appropriately so that average and 95th percentile CPU (memory) usage for >90% of the containers are within 50% - 100% (70% - 100%) of the requested values. Furthermore, average CPU (memory) utilization across all pods is raised from 10% (26%) to 54% (88%).

CCS CONCEPTS

• **Computer systems organization** → *Maintainability and maintenance*;

KEYWORDS

Container resource usage, stationarity, resource over-provisioning

[1]A Kubernetes pod is a group of containers deployed together on one host.
[2]Changes in Kubernetes are proposed to support the runtime vertical scaling in https://github.com/kubernetes/autoscaler/tree/master/vertical-pod-autoscaler.

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).
SoCC '18, October 11–13, 2018, Carlsbad, PA, USA
© 2018 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-6011-1/18/10…$15.00
https://doi.org/10.1145/3267809.3275448