Demo Paper: FECBench – A Framework for Measuring and Analyzing Performance Interference Effects for Latency-Sensitive Applications

Yogesh D. Barve, Shashank Shekhar, Ajay D. Chhokra, Shweta Khare,
Anirban Bhattacharjee and Aniruddha Gokhale
Dept of EECS, Vanderbilt University, Nashville, TN 37212, USA
Email:{yogesh.d.barve,shashank.shekhar,ajay.d.chhokra, shweta.p.khare,anirban.bhattacharjee,a.gokhale}@vanderbilt.edu

***Abstract*—Cloud-enabled latency-sensitive applications are increasingly exploiting fog/edge computing resources to meet their latency requirements while still benefiting from the elastic properties of the cloud. Fog/edge computing enable the services to perform the processing at the edge rather than sending the data to the remote distant cloud, which can be susceptible to high communication latencies. However, as more latency-sensitive services get deployed on these fog/edge resources, performance interference effects caused due to sharing of the limited fog/edge resources such as cache, memory, disk can lead to these services missing their deadlines and hence violation of their service level objectives. Presently, there is a general lack of tools that can enable developers and system administrators to study, understand and predict such performance interference issues when their services are deployed in the presence of a variety of other services that share these resources. To address these gaps, this paper presents a framework called Fog/Edge/Cloud Benchmark (FECBench). FECBench allows users to build performance models of latency-sensitive fog/edge-based applications. To predict the sensitivity of the target application to co-located workloads and pressure that it imposes on co-located workloads, FECBench maintains an extensible knowledge base that captures utilizations for different resources under a variety of application co-location combinations that includes the target application. To enable this, FECBench provides a benchmarking and remote monitoring infrastructure to conduct these benchmarking experiments in an automated fashion, and collect the results from remote sites. FECBench also supports a visual domain-specific language that eases the construction and execution of performance interference experiments for the users. This paper describes the FECBench framework and the demonstration scenario.**

***Keywords*—Performance analysis, DSML, Interference, Monitoring, Cloud, Fog, Edge Computing, Resource Management.**

I. Introduction

Fog/Edge computing is gaining tremendous traction for latency-sensitive applications due to its elastic but locally available compute processing capability. Applications, such as medical patient monitoring, industrial internet of things, transit vehicle monitoring applications[1], preventive equipment repair monitoring services [2] and hardware-in-loop distributed simulations are increasingly benefitting from fog/edge computing resources [3].

With rapid growth in such technologies and newer types of application workloads running on these platforms, fog/edge providers must provide effective resource management to meet the service level objectives (SLOs) of these applications. Like the cloud, the fog/edge supports multi-tenancy, however, in more resource-constrained environments because of which applications are susceptible to more pronounced effects of performance interference. Performance interference is caused due to sharing of resources such cache, network, disk, etc which are difficult to partition/isolate among applications [4]. In this context, effective schedulers need to be designed that can mitigate the impact of performance interference thereby providing desired level of QoS to such latency-sensitive applications [5], [6].

Developing effective resource management solutions, e.g., schedulers, requires an accurate understanding of application performance under different co-location scenarios which can give rise to different performance interference patterns. To that end, the use of performance models learned from such a benchmarking effort can allow resource management solutions to rapidly make intelligent resource allocation decisions and enforce effective application placement on the runtime platforms. One approach to creating such performance interference models was proposed by the DeepDive project [7], which uses the system resource metrics to infer application performance. However, to create such performance models is a challenging task for a variety of reasons. First, the application performance needs to be analyzed under varying levels of system resource utilization. However, system resource utilization is a multi-dimensional space due to the presence of multiple types of resources, and hence creating varying levels of resource utilizations spanning this large design space is a difficult task. Compounding this problem is the fact that it is hard for users to define the right kinds of software workloads that can cover this multi-dimensional resource utilization design space. Secondly, running such performance interference tests is a very difficult task due to an overall lack of software tooling infrastructure and its runtime complexity.

To address these concerns, we present FECBench (Fog/Edge/Cloud Benchmark), which is a framework to build performance interference models for latency-sensitive applications that can be co-located with a variety of different applications on the fog/edge resources.

II. Tool Demonstration

Figure 1 shows the high level architecture of FECBench. The experiment modeling provides intuitive abstractions to the user to configure and automate the benchmarking experiments and collect the desired resource utilization metrics. The generative aspects of the framework synthesize the artifacts needed to automate the entire performance modeling and analysis process.

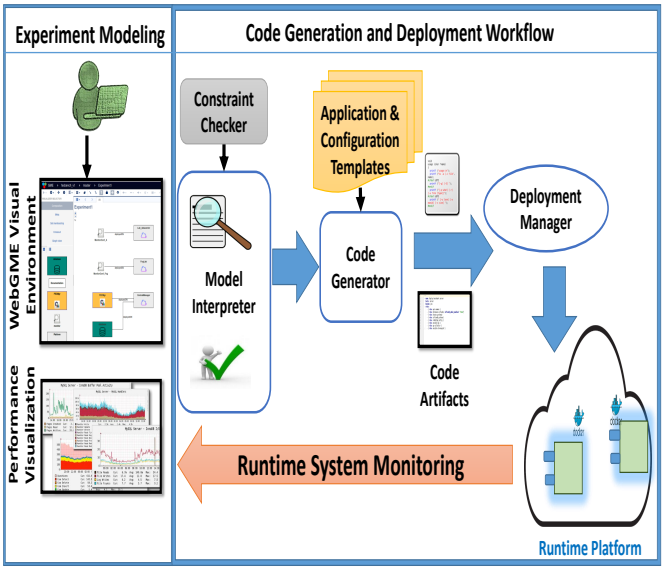

Fig. 1. High Level Overview of FECBench Workflow

The demonstration we will show at the conference will include the following steps.

1) **Modeling of the Experiment**: In this step we will showcase the visual domain specific modeling language (DSML) that can be utilized to specify different metrics of interest to be monitored on the runtime platform. These system metrics are then utilized to monitor in real time the resource consumption metrics.
2) **Executing Benchmarks of Applications**: Using FECBench, we will demonstrate how one can run performance benchmarks for the latency-sensitive application when executing in the context of colocated workloads. This step will also showcase how the collection and aggregation of metrics takes place, and how we can visualize them using a graphical dashboard.
3) **Displaying the Effects of Performance Interference**: Using a latency-sensitive application, such as machine learning inference, we will demonstrate the effect of performance interference on the application's execution time. The goal of this step is to show how the interference effects manifest as performance degradation in the application during its execution on the underlying platform. As an example, consider a machine learning prediction service running on an Intel Xeon platform and co-located along with some long running batch applications. As seen in Figure 2, the response time of the machine learning inference step can be severely degraded due to performance interference effects caused due to the co-located workloads.

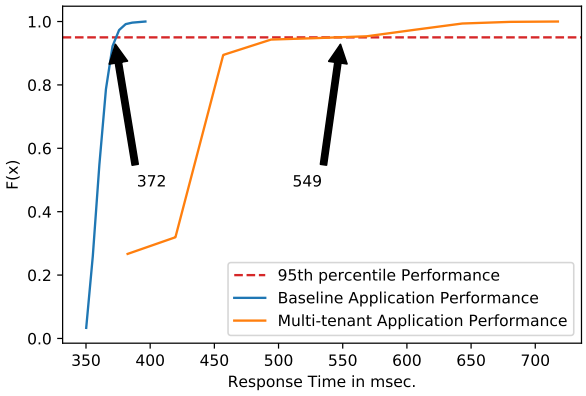

Fig. 2. CDF Representation of Prediction Inference Response Times for Inception RESNETv2 Keras Model

4) **Strategies for minimizing performance interference** : In this, we will show how using an intelligent placement and job scheduler that we can minimize performance interference effects in a multi-tenant environment.

Acknowledgments

This work is supported in part by NSF US Ignite CNS 1531079 and AFOSR DDDAS FA9550-18-1-0126. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of NSF or AFOSR.

References

[1] Y. Simmhan, P. Ravindra, S. Chaturvedi, M. Hegde, and R. Ballamajalu, "Towards a data-driven iot software architecture for smart city utilities," *Software: Practice and Experience*, 2018.
[2] A. Bayoumi and R. McCaslin, "Internet of things–a predictive maintenance tool for general machinery, petrochemicals and water treatment," in *Advanced Technologies for Sustainable Systems*. Springer, 2017, pp. 137–146.
[3] S. Jeschke, C. Brecher, T. Meisen, D. Özdemir, and T. Eschert, "Industrial internet of things and cyber manufacturing systems," in *Industrial Internet of Things*. Springer, 2017, pp. 3–19.
[4] J. Mars, L. Tang, R. Hundt, K. Skadron, and M. L. Soffa, "Bubble-up: Increasing utilization in modern warehouse scale computers via sensible co-locations," in *Proceedings of the 44th annual IEEE/ACM International Symposium on Microarchitecture*. ACM, 2011, pp. 248–259.
[5] S. Shekhar, A. D. Chhokra, A. Bhattacharjee, G. Aupy, and A. Gokhale, "Indices: exploiting edge resources for performance-aware cloud-hosted services," in *Fog and Edge Computing (ICFEC), 2017 IEEE 1st International Conference on*. IEEE, 2017, pp. 75–80.
[6] C. Delimitrou and C. Kozyrakis, "Paragon: Qos-aware scheduling for heterogeneous datacenters," in *ACM SIGPLAN Notices*, vol. 48, no. 4. ACM, 2013, pp. 77–88.
[7] D. Novakovic, N. Vasic, S. Novakovic, D. Kostic, and R. Bianchini, "Deepdive: Transparently identifying and managing performance interference in virtualized environments," in *Proceedings of the 2013 USENIX Annual Technical Conference*, no. EPFL-CONF-185984, 2013.