Optimizing the Performance of Fog Computing Environments Using AI and Co-Simulation

Shreshth Tuli
Imperial College London
London, UK
s.tuli20@imperial.ac.uk

Giuliano Casale
Imperial College London
London, UK
g.casale@imperial.ac.uk

ABSTRACT

This tutorial presents a performance engineering approach for optimizing the Quality of Service (QoS) of Edge/Fog/Cloud Computing environments using AI and Coupled-Simulation being developed as part of the Co-Simulation based Container Orchestration (COSCO) framework. It introduces fundamental AI and co-simulation concepts, their importance in QoS optimization and performance engineering challenges in the context of Fog computing. It also discusses how AI models, specifically, deep neural networks (DNNs), can be used in tandem with simulated estimates to take optimal resource management decisions. Additionally, we discuss a few use cases of training DNNs as surrogates to estimate key QoS metrics and utilize such models to build policies for dynamic scheduling in a distributed fog environment. The tutorial demonstrates these concepts using the COSCO framework. Metric monitoring and simulation primitives in COSCO demonstrates the efficacy of an AI and simulation based scheduler on a fog/cloud platform. Finally, we provide AI baselines for resource management problems that arise in the area of fog management.

CCS CONCEPTS

• **Computer systems organization** → *Embedded and cyber-physical systems*; • **Computing methodologies** → *Artificial intelligence.*

KEYWORDS

Performance Engineering, Fog Computing, Artificial Intelligence, Co-Simulation.

ACM Reference Format:
Shreshth Tuli and Giuliano Casale. 2022. Optimizing the Performance of Fog Computing Environments Using AI and Co-Simulation. In *Companion of the 2022 ACM/SPEC International Conference on Performance Engineering (ICPE '22), April 9–13, 2022, Bejing, China.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3491204.3527490

1 INTRODUCTION

In recent years, the landscape of computing technologies has seen a gradual yet remarkable shift from hand-encoded algorithms to Artificial Intelligence (AI) driven autonomous systems for Quality of Service (QoS) aware reliable resource management [3]. Further, research communities are now relying on coupled simulation engines (also referred to as co-simulators in literature) that act as digital-twins of the computing infrastructure, *i.e*, approximate models that facilitate testing several resource management decisions before executing them on physical deployments. The last few years have seen that the research and development has been focused on leveraging platforms such as Internet of Things (IoT), edge, fog and cloud for enhanced service delivery. The state-of-the-art prior work focuses on augmenting existing systems and using AI and co-simulators for a wide range of domains including efficient resource provisioning, application deployment, task placement and service management. This calls for the development of specialized modeling, testing and resource management techniques that can effectively harness AI and co-simulation technologies to reach optimum QoS in fog environments.

In this tutorial, we demonstrate tools and frameworks that aim to facilitate research for the development and deployment of AI and co-simulation based applications on fog environments with structured communication and platform-independent execution of applications. A recent such tool is the Co-Simulation based Container Orchestration (COSCO) framework [20].[1] Such tools connect all edge, fog and cloud nodes in a physical setup using lightweight communication APIs, such as HTTP REST, and allow AI-based resource management modules to not only utilize the system and workload resource utilization characteristics, but also simulated characteristics at a future state of the system. The interleaved execution of AI models and coupled simulation has been utilized by prior work for long-term performance optimization [6, 18], quick adaptation in volatile system settings [1] and fault-aware scheduling for resilient service delivery [8, 16, 19]. The objective of this tutorial is to demonstrate the various challenges posed by the integration of AI in fog environments and how they are addressed by the various contributions in recent work [17, 18, 20].

This tutorial initially introduces the fog architecture, the continuum of edge and cloud resources it presents, and the challenges posed in the context of performance engineering. The focus then shifts to the modeling of such distributed computing domains in the COSCO framework, with specific emphasis on the various inputs and outputs presented to an optimization engine, which in this context is an AI model. We discuss the various data sources, such as resource utilization characteristics, which can be used to train an AI agent using a co-simulator to act as a surrogate of the QoS metrics of future states of the fog systems. Leveraging the trained model, the tutorial demonstrates how we can optimize

ICPE '22 Companion, April 9–13, 2022, Bejing, China.

ACM ISBN 978-1-4503-9159-7/22/04…$15.00
https://doi.org/10.1145/3491204.3527490

[1] The code for the COSCO framework is publicly available under BSD-3 license `https://github.com/imperial-qore/COSCO`.

service delivery by making QoS aware resource management decisions such as scheduling of incoming tasks. To do this, we deploy popular fog benchmarking applications [10] and demonstrate the performance improvements compared to heuristic based methods using the monitoring tools in COSCO. A part of the tutorial, including demonstrations of QoS aware scheduling, are available as YouTube videos: `https://www.youtube.com/playlist?list=PLN_nzHzuaOBQijEwy2Fy8c09-dWYVe4XO`.

2 OVERVIEW OF FOG RESOURCE MANAGEMENT

Computing paradigms have changed from mainframes and client-server models to cloud and edge computing models. The fundamental driver for this shift has been the advent of offering compute as a service [10]. Cloud computing service providers offer utility for scalable and reliable computing leveraging technologies such as virtualization. Lightweight virtualization techniques such as containers, microservices and serverless computing further enhance the service quality of cloud and edge platforms. We are now in the era of IoT and big-data applications have given rise to the emergence of hybrid compute platforms, such as fog computing, that harness the resources of both edge and cloud layers. This has given rise the increasing complexity of contemporary compute environments that managing their resources cannot entirely rely on heuristics for optimum performance [20]. Thus, the community has now moved towards AI-centric approaches for optimal resource management [3].

Fog computing is an emerging paradigm in distributed systems, which encompasses all intermediate devices between the IoT layer and the cloud layer. It can drastically reduce latency of compute, network and storage services by placing them closer to end users, which is commonly called the *edge* of the network. This leads to a multitude of benefits such as reduced latencies, lower costs, more reliability and scalability. Considering the presence of billions of edge nodes in close proximity to the users, with only thousands of cloud nodes accessible to all, it becomes very expensive to run everything at the edge. To make deployments feasible, edge devices are often close to the user, but have limited processing capacities. On the other hand, cloud devices are far away, but can handle much heavier workloads. This gives rise to this research field on how to carefully balance task placement to be within the latency demands of users but also minimize deployment and execution costs [17].

However, the problem of resource management and specifically task placement is challenging. One of the challenges we face in such settings is to deliver low latencies for time-critical applications. For instance, many application domains, such as healthcare, manufacturing and smart-cities demand ultra low response times specifically for tasks that are sensitive to Service Level Agreements (SLAs). Other applications, in an effort to drive toward a more sustainable model, involve energy scavenging methods and renewable resources [20]. Using such energy sources leads to the requirement of supreme energy efficiency in task execution.

The challenge, however, of reaching low response times and energy consumption is exacerbated by modern-day applications, wherein the workloads are highly dynamic and host machines having non-stationary resource capabilities. Prior work that uses heuristics, reinforcement learning or other such methods, solves these problems to a limited extent by sometimes not being quick enough to adapt to volatile settings, being slow to converge or just that they cant keep up with the extreme user demands. Data driven methods, such as AI and deep learning aim to solve all these challenges to provide optimal QoS in fog environments.

3 TECHNOLOGICAL DRIVERS

3.1 Neural Networks as Surrogate Models

The fundamental core of most modern AI based resource management solutions is based on the idea of deep learning, which uses deep neural networks. A neural network, akin to a cognitive brain, is composed of neurons. Each neuron takes two or more inputs, performs some elementary operations on those, such as a simple dot product and produces a single output. When we stack multiple such neurons for the same set of inputs, we can generate multiple outputs and this is called a *layer* of neurons. When we stack such layers together, we call these as *deep* neural networks. The main advantages of a deep neural network is that, given enough data, it can be tuned to approximate any given function to arbitrary level of precision. When models approximate any given function, we call them *surrogates*. The functions they approximate could be of diverse kinds, including estimating the QoS parameters of a fog system, such as energy consumption, average response time, fraction of violations of the SLAs or say the operational costs of a hybrid edge-cloud setup.

Surrogate modelling has been popular prior work and different types of surrogate models have been used in the past. These include Support Vector Machines (SVMs), Gaussian Processes and polynomial models. However, the main advantage that neural networks provide is that they can be trained to accurately match a given dataset even with systems that dynamically change with time, or when there are several data sources leading to a high-dimensional data. These feature are canonical to heterogeneous fog environments; hence, in recent years, neural networks have gained popularity over the other types of models to act as surrogates of parameters of fog systems.

3.2 Coupled-Simulation

As described above, neural networks have properties that make them a good fit for surrogate modelling. However, simply using gradient-based optimization is not sufficient as data driven neural models can sometimes saturate. This is when feeding more data to the neural model does not improve the performance. To address this, we need *out-of-distribution* data, *viz*, data from settings that the neural network has not seen before [2]. A promising method to generate out-of-distribution data is to utilize a simulator. Such a simulator, akin to a digital-twin, allows us to run simulations in the background to facilitate decision making and injecting additional information regarding the system behaviour within the surrogate optimization methods. This is commonly referred to as *co-simulation* in literature [20]. A co-simulator could be used to virtually execute a sequence of decisions and observe how they affect the environment and the objective QoS scores. Through this, an optimum decision of a sequence of decisions may be obtained to explore out-of-distribution configurations. This enables neural

models to generalize to settings previously unseen during training and alleviate the problem of data saturation.

4 CASE STUDIES

The above two technological drivers have been integrated by the COSCO framework. This framework has been extended and applied to diverse resource management problems in fog computing. We now describe some case-studies arising from these technologies.

4.1 Bag-Of-Task Scheduling

Bag-of-task is a workload model wherein the individual tasks can be processed independently and there are no precedence orders between any two tasks in the system. Many neural network based surrogate modelling methods have been proposed in the past that leverage gradient-free optimization strategies such as evolutionary search and swarm optimization [11]. In discrete-time control settings [20], the idea of GOBI* integrates a neural network as a surrogate model with a co-simulator that provides a single-step look-ahead of QoS estimates. This concept stems from the Gradient based Optimization using Backpropagation to Input (GOBI) approach. In this approach, we train a neural network model, say $f(x, \theta)$, where θ are the parameters of the network and x is the *state* of the system. We characterize the state of a fog system using the collection of resource utilization metrics (of tasks and hosts) and the scheduling decision. The neural network f predicts the QoS metrics $\hat{y}$ at the end of the next time interval. This can be trained using multiple runs using a dataset of (x, y) pairs generated using a random scheduler to capture diverse system states and scheduling decisions x and the QoS score y. Then, θ parameters can be obtained such that the deviation between $\hat{y} = f(x, \theta)$ and y is minimized. Now that f acts as a differentiable surrogate of the QoS scores, we can use gradient optimization strategies such as stochastic-gradient-descent to update the scheduling decision as

$$x_n \leftarrow x_{n-1} - \nabla_x f(x, \theta), \tag{1}$$

where n denotes the iteration count. Running the above till convergence gives us a local-optimum schedule x. Advances such as root-mean-square propagation, momentum and warm restarts can be used to avoid getting stuck in local optima [7, 9, 12]. GOBI* extends this idea by also giving a QoS estimate from a co-simulator for the decision x. This helps us inject the system behaviour characteristics, to which such neural models are oblivious towards. Additionally, the co-simulated estimates facilitate solving the exposure bias and data saturation problems [13].

However, an issue with the GOBI and GOBI* models is that they utilize *deterministic* surrogate models. However, in pragmatic settings, the systems and workloads are seldom deterministic [17]. Further, most workloads follow high stochasticity in their behaviours. To model this, the GOBI and GOBI* models have been extended to utilize neural networks that generate stochastic outputs. One such example is the GOSH approach. GOSH uses Natural Parameter Networks (NPNs) that use weights as distributions themselves, and enable the use of a fairly general distribution of weights from one of the exponential class of distributions. Higher-order optimization strategies in tandem with stochastic surrogate models have shown improvements with respect to their deterministic counterparts [17].

4.2 Workflow Scheduling

Workflow applications, unlike bag-of-task models, impose precedence constraints over the tasks in the form of directed-acyclic-graphs (DAGs). similar to bag-of-task scheduling, the workflow scheduling problem aims to efficiently map the tasks onto the available resources to optimize the QoS metrics. Many works have been proposed that solve the workflow scheduling problem using AI models as surrogates [4, 21]. However, the state-of-the-art workflow scheduler, MCDS [18], aims to address the limitation of the GOBI/GOBI* approaches of only using single-step QoS predictions. For multi-stage applications, such as scientific workflows, such a myopic optimization models do not effectively capture the effects on QoS in case of chain of tasks [18]. To address this, MCDS executes additional simulation steps for a more informed decision. Such policies are also referred to as *long-horizon* policies in literature [5]. MCDS adapts Monte-Carlo tree search approach to train a surrogate for multi-step look-ahead estimates. Specifically, MCDS builds a tree of system states and possible actions. At each timestep, for the current system state, it selects a possible action using a balance of exploration and exploitation objectives using the Upper-Confidence Bound strategy [15]. It then expands the selected target state, runs several multi-step co-simulations and feeds the QoS scores back to the neural model. In the case of long-running workflow applications, we conjecture that such a long-term optimization model is helpful. This is corroborated by experiments that demonstrate that MCDS outperforms myopic optimization strategies that leverage single-step surrogates with evolutionary search strategies [18].

4.3 Fault-Tolerance Using Migrations

In fog computing environments, resource management also entails ensuring the system is able to avoid system or network level faults. In particular, modern application demands of low latency task execution and resource constraints of the edge devices exacerbate the problem of effective resource management. The increasing volumes of the data requiring immediate processing and the resource constraints at the edge are pushing the compute resources to their limits, giving rise to a high chance of resource contention and node downtimes. This leads to resource unavailability and violation of the SLAs that can possibly lead to significant financial losses. Further, the problem of developing a robust fault-tolerance framework requires us to proactively predict faults before they occur and diagnose the root-cause issues to be able to run appropriate remediation steps. For such systems to be within the strict specifications of modern industrial demands, they need to be able to resolve diverse kinds of system or network related faults. This may entail establishing the type of fault at the time of its prediction for a more informed recovery decision. Furthermore, to avoid overlooking any fault that may cause significant adverse effects later and to avoid the overhead of false-positive predictions, such prediction models need to be extremely accurate. To combat this, several fault-tolerance approaches have been proposed in the past [8]. Recent methods, such as PreGAN [19], utilize both neural networks as well as co-simulations to ensure fault-tolerance. Its neural network predicts preemptive migration decisions to migrate tasks from hosts that are predicted to have contentions in a future intervals and to avoid contentions. Such a neural model is trained using co-simulated

QoS estimates. This enables the preemptive migration decisions to be informed on the environment characteristics and workload behaviours. Such informed decisions enable PreGAN to outperform prior methods in terms of fault-prediction accuracy and QoS [19]. COSCO allows preemptive migrations to facilitate research in the direction of fault-tolerance.

4.4 Service Resilience in Edge Federations

Due to the inherent stochastic nature of workloads and fog environments, no fault-prediction method is perfect. Faults may still arise when prediction are not entirely accurate and there are false negatives. In such cases, it is possible that nodes break-down due to contentions. As fog systems typically follow a broker-worker architecture, worker failures are easier to address by having redundant workers or re-initializing the failed tasks on other workers. However, if a broker node fails, it renders useless all worker nodes handled by that broker as now the incoming workloads to that brokers cannot be assigned to any of those workers. This makes the broker resilience problem important to minimize service downtime and improve QoS. Recently, several works have been proposed to address this [8]. To test this, COSCO provides primitives to change the topology of a fog federation, wherein, nodes can be shifted from the broker to the worker layer and vice versa. A recently proposed method validated in COSCO, namely CAROL [16], leverages graph-neural-networks in tandem with co-simulations to predict changes in the fog topology in case of broker breakdowns, in order to optimize QoS in an online fashion.

5 DEMONSTRATIONS

The final part of the tutorial provided a hands-on experience using demos to demonstrate how the COSCO framework and its primitives can be utilized for research in fog computing.

COSCO Framework. COSCO[2] is a Python based framework that allows orchestration of docker containers in both simulated and real distributed computing environments. COSCO uses the assumption of the system model that the execution runs for a bounded time with the timeline divided into discrete intervals of fixed length. All the workload creation, and decision making and execution of task allocations and migrations are run at the start of an interval. Thus, all modules in COSCO are discrete time controllers. The framework provides an exhaustive wiki[3] a pre-configured Gitpod container[4] to quickly deploy and test schedulers, provisioners and fault-tolerance models. COSCO supports popular workload applications (BitBrain workload traces [14] and DeFog benchmarks [10]) with support for multiple Raspberry-Pi based edge and/or Microsoft Azure based public cloud VMs to run various tests.

Recordings of the tutorial demonstrations can be accessed at `https://youtu.be/osjpaNmkm_w`.

6 CONCLUSIONS

COSCO provides an interface to blend simulation driven feedback with AI based resource management solutions. This enables us to leverage neural networks as surrogate models to perform online optimization of QoS in heterogeneous and distributed fog environments. Further, co-simulated estimates aim us in ameliorating the limitations of pure AI based solutions by injecting system and workload behavioral information within black-box models such as neural networks. The COSCO framework offers a platform for performance engineering and extensions to other domains of dynamic resource management of computational systems.

[2]COSCO Framework: `https://github.com/imperial-qore/COSCO`.
[3]COSCO Wiki: `https://github.com/imperial-qore/COSCO/wiki`.
[4]COSCO Gitpod Container: `https://gitpod.io/#https://github.com/imperial-qore/COSCO/`.

REFERENCES

[1] Muder Almiani, Alia AbuGhazleh, Amer Al-Rahayfeh, Saleh Atiewi, and Abdul Razaque. 2020. Deep recurrent neural network for IoT intrusion detection system. *Simulation Modelling Practice and Theory* 101 (2020), 102031.

[2] David Berend, Xiaofei Xie, Lei Ma, Lingjun Zhou, Yang Liu, Chi Xu, and Jianjun Zhao. 2020. Cats are not fish: Deep learning testing calls for out-of-distribution awareness. In *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering*. 1041–1052.

[3] Luca Greco, Gennaro Percannella, Pierluigi Ritrovato, Francesco Tortorella, and Mario Vento. 2020. Trends in IoT based solutions for health care: Moving AI to the edge. *Pattern recognition letters* 135 (2020), 346–353.

[4] Goshgar Ismayilov and Haluk Rahmi Topcuoglu. 2020. Neural network based multi-objective evolutionary algorithm for dynamic workflow scheduling in cloud computing. *Future Generation computer systems* 102 (2020), 307–322.

[5] Adam Janiak, Tomasz Krysiak, and Radosław Trela. 2011. Scheduling problems with learning and ageing effects: a survey. *Decision Making in Manufacturing and Services* 5, 1-2 (2011), 19–36.

[6] Murat Kankal and Ergun Uzlu. 2017. Neural network approach with teaching–learning-based optimization for modeling and forecasting long-term electric energy demand in Turkey. *Neural Computing and Applications* 28, 1 (2017), 737–747.

[7] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[8] Abdeldjalil Ledmi, Hakim Bendjenna, and Sofiane Mounine Hemam. 2018. Fault tolerance in distributed systems: a survey. In *2018 3rd International Conference on Pattern Analysis and Intelligent Systems (PAIS)*. IEEE, 1–5.

[9] Ilya Loshchilov and Frank Hutter. 2016. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983* (2016).

[10] Jonathan McChesney, Nan Wang, Ashish Tanwer, Eyal de Lara, and Blesson Varghese. 2019. Defog: fog computing benchmarks. In *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*. 47–58.

[11] Zahra Makki Nayeri, Toktam Ghafarian, and Bahman Javadi. 2021. Application placement in Fog computing with AI approach: Taxonomy and a state of the art survey. *Journal of Network and Computer Applications* 185 (2021), 103078.

[12] Hengyue Pan and Hui Jiang. 2015. Annealed gradient descent for deep learning. In *The Thirty-First Conference on Uncertainty in Artificial Intelligence*. 652–661.

[13] Anna Rakitianskaia and Andries Engelbrecht. 2015. Measuring saturation in neural networks. In *2015 IEEE Symposium Series on Computational Intelligence*. IEEE, 1423–1430.

[14] Siqi Shen, Vincent van Beek, and Alexandru Iosup. 2015. Statistical characterization of business-critical workloads hosted in cloud datacenters. In *15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. IEEE, 465–474.

[15] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

[16] Shreshth Tuli, Giuliano Casale, and Nicholas R Jennings. 2022. CAROL: Confidence-Aware Resilience Model for Edge Federations. In *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE.

[17] Shreshth Tuli, Giuliano Casale, and Nicholas R. Jennings. 2022. GOSH: Task Scheduling using Deep Surrogate Models in Fog Computing Environments. *IEEE Transactions on Parallel and Distributed Systems* (2022).

[18] Shreshth Tuli, Giuliano Casale, and Nicholas R. Jennings. 2022. MCDS: AI Augmented Workflow Scheduling in Mobile Edge Cloud Computing Systems. *IEEE Transactions on Parallel and Distributed Systems* (2022).

[19] Shreshth Tuli, Giuliano Casale, and Nicholas R Jennings. 2022. PreGAN: Preemptive Migration Prediction Network for Proactive Fault-Tolerant Edge Computing. In *IEEE Conference on Computer Communications (INFOCOM)*. IEEE.

[20] Shreshth Tuli, Shivananda R. Poojara, Satish N. Srirama, Giuliano Casale, and Nicholas R. Jennings. 2022. COSCO: Container Orchestration Using Co-Simulation and Gradient Based Optimization for Fog Computing Environments. *IEEE Transactions on Parallel and Distributed Systems* 33, 1 (2022), 101–116. `https://doi.org/10.1109/TPDS.2021.3087349`

[21] Pengwei Wang, Yinghui Lei, Promise Ricardo Agbedanu, and Zhaohui Zhang. 2020. Makespan-driven workflow scheduling in clouds using immune-based PSO algorithm. *IEEE Access* 8 (2020), 29281–29290.