Towards Performance and Cost Simulation in Function as a Service

Johannes Manner

Distributed Systems Group, University of Bamberg, Germany
johannes.manner@uni-bamberg.de

Abstract. Function as a Service (FaaS) promises a more cost-efficient deployment and operation of cloud functions compared to related cloud technologies, like Platform as a Service (PaaS) and Container as a Service (CaaS). Scaling, cold starts, function configurations, dependent services, network latency etc. influence the two conflicting goals cost and performance. Since so many factors have impact on these two dimensions, users need a tool to simulate the function in an early development stage to solve these conflicting goals. Therefore, a simulation framework is proposed in this paper.

Keywords: Serverless Computing, Function as a Service, FaaS, Benchmarking, Load Pattern, Cold Start, Pricing, Simulation Framework

1 Introduction

Function as a Service (FaaS) [3] is a new, event-driven computing model in the cloud, where single functions are executed in containers on a per request basis. It promises a faster, easier and more cost-efficient development, deployment and operation due to the abstraction of operational tasks by the FaaS provider. Scaling to zero as one of the game changers avoids running instances of cloud functions unnecessarily. Also the most granular pay-per-use model in the overall cloud stack leads to a serious cost reduction for suited use cases. Therefore, it is not surprising that early cost studies [1, 12] compared this new paradigm with established ones, like monolithic architectures or microservices deployed on virtual machines and container infrastructure.

The position paper is structured as follows. Sect. 2 puts the work in relation to already conducted studies and approaches, which are important for a simulation framework for FaaS. Based on these insights and the lack of a reproducible and structured approach, Sect. 3 describes the main objectives of the dissertation plan and concludes with a short discussion.

2 Related Work

One of the first cost comparisons between monolithic, microservice and cloud function architecture was done by Villamizar et al. [12]. They state that cloud functions saved more than 70% cost compared to the other implementations in their use case scenario. Another case study [1] reduced cost up to 95%. But, there are also cases, where cloud functions are more costly than traditional Virtual Machine (VM) based solutions. An example use case is a large distributed data processing application by Lee et al. [6], which is ten times more expensive.

The overall price is calculated by multiplying the execution time and the price for the function configuration. Performance, i.e. execution time, of a cloud function directly influences the price calculation. Due to this billing model, there exist a lot of publications, which focus on performance. McGrath and Brenner [11] implemented a performance-oriented FaaS platform on top of a cloud provider to gain control over the infrastructure and improve the throughput and scaling property. Lloyd et al. [8] identified a performance variation of 1500% w.r.t. the cold and warm infrastructure components of a FaaS platform.

They also assessed the throughput by concurrent access of cloud functions on different FaaS platforms, especially the commercial ones, like Amazon Web Services (AWS) Lambda, Google Cloud Functions, Azure Functions and IBM OpenWhisk. A similar multi-provider study proposed a CPU-intensive benchmark [9] to compare the different FaaS offerings and support customers to select the right platform for their needs.

Pricing and performance is "difficult to decipher", as Back and Andrikopozlos [2] stated. They also implemented a first minimal wrapper to harmonize the different function handler interfaces of the major FaaS platforms. Applications with bursty workloads are in focus of their work since these applications could especially profit from the characteristics of FaaS.

3 Simulation Framework

Since the conducted research reveals a mixed picture so far, there is a need to combine all these directions together in a structured way. By now, the related work investigates aspects in isolation and preconditions for benchmarks and other experiments are often not clear to readers. Besides the abstraction of operational tasks in FaaS, cost and performance are important considerations when deciding to build or migrate an application which includes cloud function building blocks. To shed light on the cost and performance perspective of FaaS, this paper proposes a simulation framework.

This framework is of conceptual nature to assess single cloud functions. The overall idea is to simulate a single cloud function in isolation under various circumstances in an early development phase. Chaining or orchestration of functions is out of scope. Influential factors, like the cold start, are investigated by conducting benchmarks [10] on the different FaaS platforms. Results of these experiments are aggregated to mean values and their deviations to cover the best, worst and average case. These values serve as an input for the simulation framework and enable local testing of a cloud function on a developers machine. It also shows the variation in price and performance w.r.t. the function characteristics, e.g. CPU-bound functions or IO-bound functions.

Therefore, the following research objectives are of particular interest. They are preparatory work to get a solid foundation for the overall goal of the dissertation.

Load Patterns - Benchmarking applications is a problematic field as Huppler [4] noted. Repeatability, verifiability and economical considerations are some of his requirements for a good benchmark. Since FaaS introduces a completely different scaling notion than related paradigms, such as PaaS or CaaS, the requirements for a suited FaaS benchmark are different. There is a lack of standardized load patterns for the cloud and especially for the event-triggered execution of cloud functions. The idea is to extract standardized load patterns via a literature study or real world use cases, group them and define template patterns. The catalog contains a few generic, parameterized load patterns, such as linear or bursty workloads, and could also serve as a reference for other benchmarks in the cloud area. Based on such a load pattern catalog, experiments are controllable and comparable.
Each function is executed in a lightweight container environment. If the load pattern leads to a lot of up and downscaling, the execution time is directly influenced due to the cold start overhead and communication setup to dependent services. To understand the impact of these application load patterns on cloud function price and performance compared to IaaS, PaaS or CaaS, their characteristics [2] need deeper investigation and the proposed load pattern research is the first step towards this understanding.

Cold Start - Cold starts are an inherent problem of every virtualization technology. Discussed in the previous aspect, the scaling property of FaaS results in a lot of cold starts. Based on our previous work [10] and results conducted similarly [5], there is a cold start overhead present ranging between 300ms up to seconds for a single function. This execution time overhead has a direct performance and cost impact.

Pricing - Pay-per-use billing model leads to a simple pricing for cloud function at a first glance. Only execution time, memory setting and number of invocations are necessary. But a single cloud function is rarely an application in the sense of serving business value to the customer. To get FaaS in production, a lot of additional services are mandatory, like databases, API gateways etc. Cost models like the CostHat model [7] could be adapted from the microservice to the nanoservice scope and include such mandatory services.

Portability - Portability is another important aspect. Only when portability is ensured, a simulation is useful to test, if the function shows a better performance on another platform. Therefore, the transformation effort and the estimated savings have to be considered w.r.t. cost or performance. Wrapper utilities [2] are a first step to enable portable functions and allow a comparison between custom functionality. Since cloud functions profit from rich provider ecosystems and are tightly integrated with other services, like databases, messaging systems etc., the main problem of portability are the custom interfaces of these services.

A simulation framework for cloud functions to solve conflicting goals between cost and performance is only realizable, if the items of the previous section are assessed in detail. The presented research objectives are a starting point and cover, to the best of the author's understanding, the most important objectives relevant to the proposed framework. This leads to the conclusion, that the author is aware, that there might be missing dimensions and aspects, which will be also considered when they arise. Similar to the problem of dev-prod parity in software engineering, a proof of concept is necessary. This validation step is at first a simulation, second conducting experiments and third, compare the simulated values to the metered ones in a real life experiment (sim-prod parity).

Typically, a function with fewer resources allocated is slower, but cheaper and vice versa. The catalog of load patterns, the configurations of cloud functions, the cold start values for different languages, providers and other dimensions and the price structure are the input besides the cloud function code for simulating the cost. Additional meta data are needed in form of a model, which depicts the interaction with other services in the provider's ecosystem. A small-sized benchmark is the processing step of our simulation framework. The simulation is reproducible since the parameters are constant and only the local execution on a client's machine could influence the outcome slightly. A report serves as the output, where the user can see the best configuration and provider for his use case dependent on his leading dimension.

References

1. Adzic, G., Chatley, R.: Serverless Computing: Economic and Architectural Impact. In: Proc. ESEC/FSE (2017)
2. Back, T., Andrikopoulos, V.: Using a Microbenchmark to Compare Function as a Service Solutions. In: Service-Oriented and Cloud Computing. Springer International Publishing (2018)
3. van Eyk, E., et al.: The SPEC Cloud Group's Research Vision on FaaS and Serverless Architectures. In: Proc. WoSC (2017)
4. Huppler, K.: The art of building a good benchmark. In: Performance Evaluation and Benchmarking (2009)
5. Jackson, D., Clynch, G.: An Investigation of the Impact of Language Runtime on the Performance and Cost of Serverless Functions. In: Proc. WoSC (2018)
6. Lee, H., Satyam, K., Fox, G.: Evaluation of Production Serverless Computing Environments. In: Proc. CLOUD (2018)
7. Leitner, P., Cito, J., Stöckli, E.: Modelling and Managing Deployment Costs of Microservice-Based Cloud Applications. In: Proc. UCC (2016)
8. Lloyd, W., et al.: Serverless Computing: An Investigation of Factors Influencing Microservice Performance. In: Proc. IC2E (2018)
9. Malawski, M., et al.: Benchmarking Heterogeneous Cloud Functions. In: Euro-Par 2017: Parallel Processing Workshops (2018)
10. Manner, J., et al.: Cold Start Influencing Factors in Function as a Service. In: Proc. WoSC (2018)
11. McGrath, G., Brenner, P.R.: Serverless Computing: Design, Implementation, and Performance. In: Proc. ICDCSW (2017)
12. Villamizar, M., et al.: Infrastructure Cost Comparison of Running Web Applications in the Cloud Using AWS Lambda and Monolithic and Microservice Architectures. In: Proc. CCGrid (2016)