Towards Improved Cloud Function Scheduling in Function-as-a-Service Platforms

PhD research proposal

Edwin F. Boza
Escuela Superior Politécnica del Litoral, ESPOL
Guayaquil, Ecuador
eboza@fiec.espol.edu.ec

ABSTRACT

This early doctoral proposal describes several interesting questions on how to design better cloud function schedulers for Function-as-a-Service platforms. I have started tackling these open questions, specifically how to improve function latency via increasing code locality. Preliminary evaluation shows that our proposed algorithm can reduce latency by up to 60% at the expense of a higher level of node imbalance.

CCS CONCEPTS

• **Computer systems organization → Cloud computing**;

KEYWORDS

cloud, FaaS, scheduling, serverless

ACM Reference Format:
Edwin F. Boza. 2018. Towards Improved Cloud Function Scheduling in Function-as-a-Service Platforms: PhD research proposal. In *Proceedings of 12th EUROSYS Doctoral Workshop (EuroDW'18).* ACM, New York, NY, USA, 2 pages.

1 MOTIVATION

Recent trends in software development have moved away from complex monolithic applications, decomposing them and modularizing into smaller stateless services, currently known as microservices, each with a bounded context and able to communicate with other modules to compose the complete application.

Besides the advantages for developers to have isolated, easier to debug or upgrade modules, microservices leverage cloud platforms to provide a high level of scalability [6]. However, microservices, which usually run containerized, require to be active and listening for incoming connections, leading to possible unused/overprovided resources like the VMs running the containers.

Serverless computing is an emerging cloud architecture model where developers can focus on programming their applications instead of worrying about infrastructure concerns. In this model, abstraction reaches the server level, and developers can stop thinking about VM deployment, scalability, fault tolerance and other infrastructure issues [10].

Function-as-a-Service (FaaS) is a subclass of serverless platforms, where a stateless piece of code is executed in response to a triggered event [9, 10]. Major cloud services providers currently provide FaaS services, like AWS Lambda [1], IBM OpenWhisk [2], Microsoft Azure [3], and Google Cloud Functions [4]; and the open source project OpenLambda was created to work in research problems related to this architecture [7].

FaaS architectures are becoming increasingly more popular, as they provide the following advantages for developers and software architects:

- The abstraction level allows the developer to easily run their code in a cloud environment, without the need to learn about infrastructure deployment.
- Service elasticity will be managed by the cloud provider, freeing FaaS users from dealing with provisioning planning or unexpected workload peaks.
- FaaS users are billed on per-access basis. Functions are executed only when needed, and no idle time is accounted in the cost, which could lead to reduced hosting costs by between 66% and 95% [2, 4].

Despite the aforementioned benefits, FaaS architectures present disadvantages that must be taken into account when considering its adoption, and represent open research challenges. Concretely, there are several research challenges in serverless computing: Optimizing overheads, scheduling policies, cold starts, session management, code and data locality, load balancing, execution time constraints, among others [2, 3, 5, 7, 9].

2 PROPOSED WORK

For my doctoral research, I will focus on the performance related challenges. Specifically, I will study **how to intelligently schedule cloud functions, to improve performance metrics in FaaS architectures**.

For example, due to the event-based execution nature of FaaS architectures for a function invocation, an environment initialization may be required in the worker node that will carry out the job. Potential high latencies, derived from the initialization phase, make this architecture no suitable for some use cases [2].

The research questions that I will attempt to address in my research are:

(1) Can we reduce function launch time via intelligent scheduling decisions that seek to minimize control plane and data plane communications?

[1]https://aws.amazon.com/lambda/
[2]https://developer.ibm.com/openwhisk/
[3]https://azure.microsoft.com/en-us/services/functions/
[4]https://cloud.google.com/functions

(2) How can we reconcile possibly conflicting goals of the function scheduling process? (e.g., balance worker load, maximize data locality, maximize code locality, meet QoS guarantees)
(3) Could the provider offer differentiated services so that we only need to optimize the scheduling of those functions that need quick launch time?
(4) Could delaying function launch time, when the functions are part of complex workflows, help improve performance?

I believe these are important questions, and answering them will advance the state-of-the art in FaaS function scheduling.

3 PRELIMINARY RESULTS

In this section, I describe our preliminary work tackling question 1 above. Specifically, my current focus has been on how to improve function latency via increasing code locality (by running functions at workers that have already cached the packages required by those functions).

Cloud function latency relies heavily on initialization time, where the worker node receives the request and prepares the required environment to run the incoming cloud function. Initialization time increases when the user function requires large packages or libraries to be loaded before its execution. To overcome this issue, a proposed solution uses a cache at the worker nodes, to keep packages preloaded prior to function execution, which leads to speed-ups of up to 2000*x* for workloads that require only one package [8].

We believe that caching benefits can be better exploited by a function scheduler that takes into account cache capacity. In a current project, we found that existing schedulers for FaaS platforms are simplistic and only focused on load balancing between the worker nodes, and we have looked into the use of package-awareness scheduling to take advantage of any package possibly cached in response to a previous access [1]. Our proposed algorithm tries to co-locate functions with common package dependencies into the same worker nodes, while avoiding to imbalance the jobs between all available nodes.

Preliminary evaluation shows that our proposed algorithm can reduce latency by up to 60% at the expense of a higher level of node imbalance. These results strengthen our decision to explore ways to improve the performance of FaaS platforms through the construction of an efficient scheduler.

While the preliminary results are encouraging, I am working on a more comprehensive evaluation that will include a simulation-based exploration of the configuration space as well as real tests using OpenLambda in a public cloud.

REFERENCES

[1] Cristina L. Abad, Edwin F. Boza, and Erwin van Eyk. 2018. Package-Aware Scheduling of FaaS Functions. In *ICPE âĂŹ18: ACM/SPEC International Conference on Performance Engineering Companion*. ACM.
[2] Gojko Adzic and Robert Chatley. 2017. Serverless computing: economic and architectural impact. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*. ACM, 884–889.
[3] Ioana Baldini, Paul Castro, Kerry Chang, Perry Cheng, Stephen Fink, Vatche Ishakian, Nick Mitchell, Vinod Muthusamy, Rodric Rabbah, Aleksander Slominski, et al. 2017. Serverless computing: Current trends and open problems. In *Research Advances in Cloud Computing*. Springer, 1–20.
[4] E. F. Boza, C. L. Abad, M. Villavicencio, S. Quimba, and J. A. Plaza. 2017. Reserved, on demand or serverless: Model-based simulations for cloud budget planning. In *2017 IEEE Second Ecuador Technical Chapters Meeting (ETCM)*. 1–6. https://doi.org/10.1109/ETCM.2017.8247460
[5] Rajkumar Buyya, Satish Narayana Srirama, Giuliano Casale, Rodrigo Calheiros, Yogesh Simmhan, Blesson Varghese, Erol Gelenbe, Bahman Javadi, Luis Miguel Vaquero, Marco AS Netto, et al. 2017. A Manifesto for Future Generation Cloud Computing: Research Directions for the Next Decade. *arXiv preprint arXiv:1711.09123* (2017).
[6] Nicola Dragoni, Ivan Lanese, Stephan Thordal Larsen, Manuel Mazzara, Ruslan Mustafin, and Larisa Safina. 2017. Microservices: How to make your application scale. In *International Andrei Ershov Memorial Conference on Perspectives of System Informatics*. Springer, 95–104.
[7] Scott Hendrickson, Stephen Sturdevant, Tyler Harter, Venkateshwaran Venkataramani, Andrea C Arpaci-Dusseau, and Remzi H Arpaci-Dusseau. 2016. Serverless computation with openlambda. *Elastic* 60 (2016), 80.
[8] Edward Oakes, Leon Yang, Kevin Houck, Tyler Harter, Andrea Arpaci-Dusseau, and Remzi Arpaci-Dusseau. 2017. Pipsqueak: Lean Lambdas with Large Libraries.
[9] Erwin van Eyk, Alexandru Iosup, Simon Seif, and Markus Thömmes. 2017. The SPEC cloud group's research vision on FaaS and serverless architectures. In *Proceedings of the 2nd International Workshop on Serverless Computing*. ACM, 1–4.
[10] Blesson Varghese and Rajkumar Buyya. 2018. Next generation cloud computing: New trends and research directions. *Future Generation Computer Systems* 79 (2018), 849–861.