

Towards Splitting Monolithic Workflows Into Serverless Functions and Estimating Their Run-Time in the Earth Observation Domain

Dennis Kaiser¹, Bohdan Dovhan¹, André Bauer and Samuel Kounev

University of Würzburg, Sanderring 2, 97070 Würzburg, Germany

¹Both authors contributed equally to this paper

Keywords

Serverless, serverless functions, runtime estimation, monolithic workflows, splitting monoliths

In the Earth observation domain, monolithic legacy workflows are still prevalent to this day. As a result, runtimes of such scientist workflows range from hours to weeks or months, depending on the area being investigated and/or the architecture of the algorithms. As minor optimizations and improved scalability can reduce runtimes considerably, we envision a novel full-stack execution platform Earth observation scientific workflows in the long term. This work introduces two cornerstones of our vision: (i) Splitting monolithic workflows into smaller, more scalable, and manageable functions. More precisely, we aim to result in workflows consisting of serverless functions since the user is not concerned with the operational part (NoOps). (ii) Estimating the runtime of the extracted functions based on different aspects to provide optimal scheduling.


Firstly, we have to investigate different aspects to port workflows to serverless functions. However, before we can extract serverless functions from these legacy monoliths, which mainly comprise serial program code, we have to solve the essential intermediate step of splitting the scientific workflow into parallelizable parts or applying other measures to improve the scalability and runtime. As part of the solution, parallelization is sought because it offers many possibilities for speedup and allows for easier distribution onto different threads and or processor cores during execution. In recent decades, the speedup of single CPU cores per new generation flattened. In contrast, the number of cores increased [1, 2, 3], promoting the trend to invest in parallel executable software development further. Another essential part of a possible solution is the optimization or reduction of serial code. As seen by Amdahl's law [4], code that needs to be processed in serial, even if it only represents 10% of the overall workflow code has significant implications for the speedup of the system. Therefore, we primarily have to shrink the percentage of serial code and assess optimizations for the remainder while keeping parallelization techniques in mind. After optimizing, parallelizing, and reducing the serial code, our vision is to construct graphs for the specific workflow, split off nodes by using a predefined ruleset that correlates with the previously used strategies, and in a final step map these new

SSP'21: Symposium on Software Performance, November 09–10, 2021, Leipzig, Germany

✉ dennis.kaiser@uni-wuerzburg.de (D. Kaiser); bohdan.dovhan@uni-wuerzburg.de (B. Dovhan); andre.bauer@uni-wuerzburg.de (A. Bauer); samuel.kounev@uni-wuerzburg.de (S. Kounev)



© 2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

resulting nodes to functions that can be executed according to the serverless paradigm.

Secondly, we have to examine approaches to estimate the runtime of serverless functions by leveraging decision-making procedures based on workload evaluation, custom metrics, and system feedback followed by time versus cost estimation analysis. The importance of the workload evaluation and serverless function runtime estimation in emerging serverless computation was presented in [5, 6]. The run time estimation requires constant and holistic input data scheduling and validation that play a crucial role in reliable performance. We will review and analyze estimation processes' metrics as well as research and compare decision-making procedures. The decision-making procedures have a crucial role in ensuring flow-less functions execution. Following the decision-making feedback combined with metrics and third-party parameters where applicable, the estimation procedure represents a self-sufficient cycle. The continuous feedback improvement and implementation of third-party AI prediction engines and/or static parameters in accordance with specific industry needs are potential areas for future research extension. The initial bootstrap procedure can have multiple implementations. However, it should follow the superfluous principle to avoid function calls loss or inability to accommodate the requested workload. We have to ensure that the runtime function estimation process should not cause any significant delays in the overall function execution time. Thus, the estimation process should be rather an independent, parallel process that might require dedicated resources. We will look at the cost prediction of serverless computation and its importance for business decision-making and cost/time analysis [7]. In addition, we aim to show the challenges that we are facing and discuss their potential remedies, future works as well as the general applicability of our approach.

References

- [1] M. M. Waldrop, More than Moore, *Nature* 530 (2016) 144–148.
- [2] P. Gepner, M. F. Kowalik, Multi-core processors: New way to achieve high system performance, in: *International Symposium on Parallel Computing in Electrical Engineering (PARELEC'06)*, IEEE, 2006, pp. 9–13.
- [3] J. S. Vetter, E. P. DeBenedictis, T. M. Conte, Architectures for the post-moore era, *IEEE Micro* 37 (2017) 6–8. doi:10.1109/MM.2017.3211127.
- [4] G. M. Amdahl, Validity of the single processor approach to achieving large scale computing capabilities, in: *Proceedings of the April 18-20, 1967, Spring Joint Computer Conference, AFIPS '67 (Spring)*, Association for Computing Machinery, New York, NY, USA, 1967, p. 483–485. URL: <https://doi.org/10.1145/1465482.1465560>. doi:10.1145/1465482.1465560.
- [5] A. Chirkin, A. Belloum, S. Kovalchuk, M. Makkes, M. Melnik, A. Visheratin, D. Nasonov, Execution time estimation for workflow scheduling, *Future Generation Computer Systems* 75 (2017). doi:10.1016/j.future.2017.01.011.
- [6] N. Akhtar, A. Raza, V. Isahagian, I. Matta, Cose: Configuring serverless functions using statistical learning, 2020, pp. 129–138. doi:10.1109/INFOCOM41043.2020.9155363.
- [7] S. Eismann, J. Grohmann, E. Eyk, N. Herbst, S. Kounev, Cost prediction of serverless workflows, 2020. doi:10.1145/3358960.3379133.