Profiling-Based Adaptive GC Policy for Serverless

Milica Karličić

Faculty of Mathematics, University of Belgrade
Oracle Labs, Belgrade
e-mail: milica.karlicic@matf.bg.ac.rs

Ivan Ristović

Faculty of Mathematics, University of Belgrade
Oracle Labs, Belgrade
e-mail: ivan.ristovic@matf.bg.ac.rs

Milena Vujošević Janičić

Faculty of Mathematics, University of Belgrade
Oracle Labs, Belgrade
e-mail: milena.vujosevic.janicic@matf.bg.ac.rs

Abstract. Serverless computing is an innovative computing model for deploying cloud-based applications. It is driven by the recent transition of enterprise application structures towards containers and microservices [2]. In serverless, requests are typically independent, similar, single-threaded, and involve short-lived objects. Existing garbage collectors (GCs) [1] often lack comprehensive application context, resulting in excess collections that occur during request processing and polluting the heap. This negatively impacts the entire system's efficiency, i.e., the ratio of operational throughput and memory footprint.

In this work, we monitored and analyzed memory usage and request patterns in serverless applications to optimize efficiency. We implemented a GC policy that determines the optimal time for garbage collection based on the current workload. It also trades off operational throughput for vast memory savings. Our prototype is implemented within GraalVM [3, 4] Serial GC and evaluated using standard serverless benchmarks. Results show a significant improvement in efficiency, due to memory footprint savings. Furthermore, our prototype significantly reduces system latency by avoiding collections during requests.

Keywords: Serverless, Garbage Collector, Latency, Efficiency, *GraalVM*.

References

[1] David Detlefs, Christine Flood, Steve Heller, and Tony Printezis. Garbage-first garbage collection. In *Proceedings of the 4th international symposium on Memory management*, pages 37–48, 2004.

[2] Paul Castro, Vatche Ishakian, Vinod Muthusamy, and Aleksander Slominski. The rise of serverless computing. *Communications of the ACM*, 62(12):44–54, 2019.

[3] Oracle Company. Memory Managment, 2024. on-line at: `https://www.graalvm.org/latest/reference-manual/native-image/optimizations-and-performance/MemoryManagement/`.

[4] Christian Wimmer, et al. Initialize once, start fast: application initialization at build time. Proceedings of the ACM on Programming Languages 3. *OOPSLA* (2019): 1-29.