Abstract

Serverless computing emerges as a new standard to build cloud applications, where developers write compact stateless functions that respond to events in the cloud infrastructure. Function-as-a-Service(FaaS) realization of serverless promotes an application in the form of independent granular components called functions. Clients can solely focus on developing applications in a serverless ecosystem, passing the overburden of resource governance to the service providers. In addition to this, the popularity of serverless computing is greatly attributed to its auto-scalability, on-demand resource provisioning, event-driven architecture, and the pay-as-you-go billing model.

This study focuses on the heightened latency experienced in serverless computing environments, arising from the diverse challenges inherent in the architecture and implementation of serverless systems. One key limitation in serverless computing is that it disregards the significance of data. All existing serverless architectures are based on the data shipping architecture. The thesis presents an alternate architecture to state-of-the-art data shipping architecture of serverless computing. The proposed architecture enables the computation to be performed at the side of data by permitting the code to flow across regions. Another limitation linked to serverless computing is the "limited lifespan" of containers. Therefore, managing incoming function requests results in an increased number of container creations, leading to cold starts. An extensive solution to handle the added up latency due to the cold start problem is proposed in our work. The thesis proposes a scheduling approach to reduce the cold start occurrences by keeping the containers alive for a longer period of time using the Least Recently Used warm Container Selection (LCS) approach on Affinity-based scheduling for worker node selection. Additionally, an orchestration service is crucial for running applications in a serverless environment. The thesis suggests an orchestration service with the ability to access inter-region resources simply by deploying the application in a single region with restricted data transmission latency. The orchestration service not only gives access to inter-region resources but also optimizes the performance of application by restricting the communication latency. Subsequently, a framework for a task-allocation crowdsourcing application is introduced, leveraging the capabilities of serverless computing to assess advantages of serverless computing environment over the traditional local monolithic implementation.

Keywords: Cloud Computing, Serverless Computing, FaaS, Latency, Cold Start, Container.