Compiling Actors to Stateful Streaming Programs

Marcus Schutte [1], George Christodoulou [1], and Asterios Katsifodimos [1]

[1] Delft University of Technology, The Netherlands
{m.schutte,c.christodoulou,a.katsifodimos}@tudelft.nl

Transitioning to the cloud has numerous benefits including infinite scalability and reduced costs with a pay-as-you-go pricing model. Containerization and building microservices is a widespread approach toward creating scalable cloud applications. Reasoning about correctness across the boundary of microservices is extremely hard [3]. As an alternative, serverless computing - such as Function-as-a-Service (FaaS) - is gaining traction. Nonetheless, FaaS is missing execution guarantees such as durable execution progress, exactly once executing and synchronization [1]. In both cases, the burden of correctness falls on the programmer's shoulders resulting in business logic being cluttered with error-handling code.

Datastream systems are a suitable execution engine for serverless applications since they provide efficient message passing, state management, and exactly-once processing guarantees. However, programming streaming systems is hard. One needs extensive knowledge of functional programming and dataflow libraries or streaming query languages. Moreover, programming a streaming system is tailored toward big data analytics and not specifically toward serverless applications.

Our goal is to raise the abstraction of distributed systems programming and handle the complexity of scalability and correctness. We propose a compiler pipeline that lets programmers write stateful entities. The compiler turns entities into stateful dataflow programs. Stateful entities are long-lived objects containing a state, similar to the actor programming model [2]. Each entity is responsible for its state and can only interact with another entity by message passing. Our compiler executes a dataflow analysis of the programs and produces stateful dataflow graphs. The compiler extracts all calls to other stateful entities and automatically turns them into asynchronous remote function calls. Because of the dataflow analyses remote function calls are joined at the right place and continue with the appropriate context. This allows a method to make multiple remote function calls asynchronously. By using a streaming system as an execution engine. the program runs in a distributed environment with strong consistency guarantees and programmers do not need to concern themselves with correctness.

References

[1] Burckhardt et al. Durable functions: Semantics for stateful serverless. *Prod. ACM on Programming Languages*, 5(OOPSLA):1–27, 2021.

[2] Bykov et al. Orleans: cloud computing for everyone. In *Prod. ACM Symposium on Cloud Computing*, pages 1–14, 2011.

[3] Ghemawat et al. Towards modern development of cloud applications. In *Proceedings of the 19th Workshop on Hot Topics in Operating Systems*, pages 110–117, 2023.