Multi-Objective and Adaptive Optimization of Controller and Datastore Placement Considering East-West Latency and Controller Load in Distributed SDN

Name: Kang Xingyuan
Laboratory's name: Software Design and Analysis Laboratory
Supervisor's name: Prof. Hajimu Iida

Abstract

Distributed SDN architecture addresses the scalability and reliability limitations of single-controller deployment by coordinating multiple controllers within a logically centralized control plane. Within this architecture, Controller Placement Problem (CPP) determines how many controllers to deploy, where to place them, and how switches are assigned among them; these decisions directly govern network latency, reliability, and load balancing. However, network latency in distributed SDN should not be limited to switch-to-controller communication alone; inter-controller communication must also be considered. As network scale grows, multiple controllers must coordinate through an East-West communication channel, which refers to communication across controllers for exchanging network state information, often supported by a distributed datastore, to maintain a consistent global state, introducing synchronization overhead that depends directly on how controllers and datastores are positioned. When placement decisions neglect the inter-controller communication, controllers experience elevated consensus delays, degrading flow setup performance and increasing risk of consistency bottlenecks. These challenges become more pronounced as traffic conditions fluctuate dynamically.

To address these challenges, this dissertation investigates the CPP considering East-West overhead and controller load balancing, through two complementary stages, progressing from static multi-objective optimization to adaptive, learning-based decision-making. The first stage jointly optimizes network latency and controller load balancing. Network latency is modeled through the Flow Setup Time (FST) framework, which decomposes end-to-end latency into switch-to-controller and inter-controller latencies, while controller load balancing is formulated as the Variance of Load Balancing (VOLB). Thirteen optimization methods are benchmarked across real-world topologies, revealing trade-offs and topology-dependent placement patterns. The second stage extends to dynamic conditions through a Deep Reinforcement Learning framework, formulating the joint placement as a Markov Decision Process trained on real traffic demand data, contributing one of the few systematic applications of reinforcement learning to CPP considering synchronization overhead.