Contention resilience in overcommitted serverless deployments

Al Amjad Tawfiq Isstaif, Richard Mortier
University of Cambridge

Serverless computing is becoming an increasingly popular deployment model which allows developers to focus on application functionality while delegating server resource management to a service provider. Serverless functions often experience extended periods of inactivity so a serverless platform that respects resource reservations may lead to wasted CPU resources. This problem can be exacerbated by the function concurrency feature, which requires users to reserve excess CPU resources to accommodate surges in incoming invocations. To address this, one potential approach is to overcommit cluster CPU resources, increasing the efficiency of serverless deployments at the cost of potentially increasing resource contention. In this talk, I present my work on a contention-resilient Linux kernel scheduler which addresses the scheduling overhead associated with overcommitted serverless deployments. We observe that the risk of contention increases in a high density deployment due to the large turnaround time under fairness policy implemented in modern schedulers. The proposed scheduling pushes the envelope of utilisation of CPU resources while ensuring predictable and stable system behaviour during overload. This extension complements work-conserving heuristics implemented in modern scheduler which enables opportunistic use of resource slack of colocated workloads. The key idea behind the extension is to the fairness requirement and prioritise the long tail of low concurrency functions. We observe that this makes a cluster worker more resilient to contention, allowing contended CPU run queues to drain more quickly, and other functions to execute with significantly less interruption. The proposed scheduler seamlessly integrates with Knative and is portable to any Linux cgroup-based serverless framework. I provide an extensive evaluation based on real-world workloads, and demonstrate a 20\% reduction in server cost.