Parallelization and Scheduling Techniques for Managing Tail Latency in Interactive Services

Md E. Haque
Rutgers University

9/9/2015 at 11:30 am
Core B (Room 305)

Abstract

Interactive services, such as Web search, recommendations, games, and finance, must respond quickly to satisfy customers. Achieving this goal requires optimizing tail (e.g., 99th+ percentile) latency. Interactive services often show wide variability in service demand; typically, most requests are short but a few long requests may run for much longer. Services can manage tail latency by giving extra resources to long requests and controlling resource usage of short requests. This is challenging because (1) service demand is unknown when requests arrive; (2) blindly giving extra resources to all requests quickly oversubscribes resources; and (3) giving extra resources to the numerous short requests will not improve tail latency.

In this thesis, we propose judicious usage of parallelization and scheduling to manage tail latency in interactive services. First, we introduce Few-to-Many (FM) incremental parallelization, which dynamically increases parallelism to utilize idle cores for reducing tail latency. FM uses request service demand profiles and hardware parallelism in an offline phase to compute a policy, represented as an interval table, which specifies when and how much software parallelism to add. At runtime, FM adds parallelism as specified by the interval table indexed by dynamic system load and request execution time progress. The longer a request executes, the more parallelism FM adds. Second, we propose exploiting emerging heterogeneous processors with fast cores and energy-efficient slow cores on the same chip. We introduce Adaptive Slow-to-Fast (A2SF) optimization algorithm, which gives powerful cores to long requests by migrating requests from slow to fast cores. We configure threshold-based migration policies using control theory to optimize a range of performance and energy objectives.

We evaluate FM and A2SF in Lucene, an open-source enterprise search engine, and Bing, a commercial Web search engine. Our results show that FM and A2SF can significantly improve tail latency. In addition, our results show
that A2SF can significantly reduce energy consumption for specific target tail latency. These results show that FM and A2SF are powerful strategies for managing tail latency in interactive services.