The Tail at Scale

Submitted by Ravi Prakash Giri (rgiri8)

The Tale at Scale published by Google shows great advices about how to handle latency variability for large distributed system. Predictable latency is quite hard to present to the users in distributed datacenters. We all are familiar with the idea of `fault-tolerance` but less well-known is `tail-tolerance`. This paper identifies latencies reasons such as shared nodes and shared global resources (such as networks and locks), daemons executing on nodes, maintenance activities (such as garbage collection), queuing delays.

The key takeaways from this paper is the two approaches to handle latency variability for large distributed system: reducing variation and living with variation. From the aspect of reduce variation differentiating service classes and higher-level queuing, reducing head-of-line blocking and managing background activities/synchronized disruption have been discussed. The paper introduces methods to handle the inevitable tail-latency events; additional strategies are proposed including hedged requests, tied-requests. If variation can't be avoided, live with it. Google didn't try to solve all of the things together, but develop tail-tolerant techniques that mask or work around temporary latency pathologies. While the overhead of ome of the most powerful tail-tolerant techniques can be modest, they require additional resources.

It is well known that straggler effects get amplified the more a system is scaled-out. This concept can be used in caching. To improve caching, the master tries to assign the same row shard to a given worker in each iteration of its main loop. If a row shard ends up being stolen by a different worker because of load balancing, the new worker can avoid accessing the DFS by requesting the row shard's file from the old worker's cache. When this happens, hedged-requests are used to avoid a worker that is slow to respond.