Priority-based Dynamic Replacement Cache

Arushi Grover and Soumyaroop Dutta

As the Internet is growing, huge amount of content is generated and stored in the datacenters everyday. Further, the network is responsible for delivery of this content to the applications at the end points. Two factors which come into play for the performance of this delivery are bandwidth and latency. While bandwidth has been growing exponentially over the years, latency has been a bottleneck, primarily due to the properties of the propagating medium and other uncontrollable factors. Applications, these days, use caching and pre-fetching at the end points (or CDN servers closer to the end points) to avoid going to the content provider, and thus, improve response time for the users.

Caches have limited space; we need to manage its contents by using techniques that use the knowledge of potential cache access patterns. There have been different cache eviction techniques, but most of them are based on LRU and MRU policies. Most of the applications that we use today do not benefit from a simple LRU based cache, and sometimes even create an extra overhead. For example, a popular website's index.html page should be always cached, even if it temporarily the least recently used object.

There have been other techniques introduced recently, like the Adaptive Replacement Cache algorithm, which provides better performance by keeping track of both frequently used and recently used pages, including a recent eviction history for both. This separates the accesses using spatial locality from those using temporal locality, which in turn helps in avoiding cache eviction by a large sequential access when combined with an iterative access. Yet, this does not avoid cache misses in various other use cases. For example, a CDN's objects is accessed by users in different parts of the globe at different times of the day. In this case, the CDN datacenters can use their cache differently, due to the different usage patterns.

We aim to implement a caching technique that dynamically adapts the cache replacement policy, based on various factors; including historical access patterns, geographical location of the caching server, time-of-day usage of the application or other priorities; and help reduce the miss ratio, thereby increasing cache efficiency.