Multimedia Proxy Caching Mechanism for Quality Adaptive Streaming Applications in the Internet

Tran Cong Thien Qui

1 Summary

Caching is one common technique for enhancing the scalability of media streaming systems. However, current caching techniques are not aware of quality adaptation. To deal with this problem, Rejaie et al. (1999) proposed an adaptive multimedia caching mechanism for layered encoded multimedia streams. This mechanism includes two techniques. The first is a pre-fetch scheme that enables the proxy to improve the quality of cached streams in a demand-driven fashion. The proxy always caches a stream during the first playback. During subsequent playbacks, at any playout time t_p, the proxy will pre-fetch any missing segments in the interval of $[t_p+T, t_p+T+\delta]$, called the *pre-fetching window*. This pre-fetching window will slide as fast as the playout point. Multiple requests of different clients are batched and sent to the server. The server will then send missing segments in decreasing priority. A new pre-fetching request will preempt the previous one. The second technique they proposed is the replacement algorithm. To choose a victim layer, they calculate the *whit* (weighted hit) for each layer during an interval, called *popularity window*. Whit is computed by T_{play}/T_{Total}. The popularity of a stream over a popularity window is the sum of its whits during this interval. The last segment of the least popular layer will be replaced first. However, while a stream is being accessed, all its layers are locked and cannot be removed. To evaluate the two above techniques, the authors conducted some simulations in *ns*. Their simulations showed that the interaction between the replacement and pre-fetching algorithm converges the state of the cache to an efficient state such that the quality of a cached stream is proportional to its popularity and the variations in quality of a cached stream is inversely proportional to its popularity.

2 Comments

In this paper, the authors proposed many effective techniques. First, the use of pre-fetching window can achieve the trade-off between the accurate of the pre-fetching prediction and the chance of receiving the requested segment in time. Second, they also suggested pre-fetching any missing segment of the new layer when the quality adaptation mechanism is close to adding a new layer. This technique can decrease the latency in streaming. Third, their preempting mechanism results in pre-fetching high priority segments while still limiting the pre-fetching rate to the congestion controlled rate limit. Next, their technique of batching all requests can reduce network congestion. Moreover, they also proposed keeping the first few segments of the base layer for each stream to hide the startup latency of future requests. Finally, their method to compute the popularity based on whit can exactly reflect the interest of clients to each layer.

However, there are still some drawbacks with this paper. Firstly, the authors did not investigate methods to pre-fetching missing segments during idle hours. Secondly, all pre-fetching requests are sent to one server, which can lead to network congestion. Instead, these requests can be alternatively sent to other caches that also contain required segments. Finally, their simulation topology is quite simple, including only one proxy. They should have run their simulation on more complicated topology to make their results more persuasive.