On an Impact of Large Content on Packet-level Caching of Information Centric Networking

Yoji Yamamoto, Junji Takemasa, Yuki Koizumi, and Toru Hasegawa
Graduate School of Information Science and Technology, Osaka University
Email: {y-yamamoto, j-takemasa, ykoizumi, t-hasegawa}@ist.osaka-u.ac.jp

***Abstract*—On the one hand *packet-level caching* of Information-Centric Networking (ICN) is a key to accommodating Video on Demand (VoD) movies thanks to its fine granularity caching, but on the other hand it would cause degradation in cache hit probability because each of packets constituting a content object is dealt individually. This paper analytically reveals that caching large content objects by using packet-level caching degrades caching hit probability.**

I. Introduction

The growth of VoD streaming of movies and TV programs, such as Netflix, makes content size larger than that of user generated content, such as YouTube [1]. The advent of 4K/8K resolution Ultra High Definition (UHD) videos will further accelerate this trend of the content size explosion.

Packet-level caching, where content is cached on a packet-by-packet basis rather than content objects, is a key to accommodating the increasing VoD videos because it eliminates redundant content transfer at a finer granularity [2]. A content object is simply referred to as an object, hereafter. ICN names each packet, thereby naturally allowing routers to support packet-level caching, and hence it is a promising approach to accommodating the increasing VoD videos.

Despite the efficiency of packet-level caching, several problems of packet-level caching have been pointed out [3]. Among the problems, *looped replacement* is caused by caching large objects, and it degrades cache hit probability due to the fact that a newly arrived packet evicts packets of the same object from the cache. They have developed object-oriented packet caching, which is a hybrid of packet-level and object-level caching, to cope with the performance degradation. Nevertheless, it remains open to analyze how large objects results in looped replacement and how it degrades cache hit probability.

This paper revisits the issue of looped replacement. The key contributions of this paper are summarized as follows: First, we develop an analytical model to derive probability of occurrence of looped replacement. Second, we evaluate probability of cache hit and looped replacement under a situation where a huge number of large objects, such as UHD videos, are deployed.

II. Definition of Looped Replacement

1) System Model: We consider a situation where each ICN router is equipped with a cache adopting the least recently used (LRU) algorithm. A schematic of a cache in an ICN router is shown in Fig. 1. The size of the cache is C slots, each of which can store one packet. Without loss of generality, we assume that packets are sorted in descending order of their recency in the cache. That is, the most recently requested packet is stored at the leftmost side of the cache. The set of all objects is denoted by $\mathcal{O}$. Object i consists of m_i packets, which are denoted by an ordered set $\mathcal{P}_i = \{p_1^i, \ldots, p_{m_i}^i\}$, and they are requested in the order of their IDs. The set of all packets is denoted as $\mathcal{N} = \bigcup_{i \in \mathcal{O}} \mathcal{P}_i$.

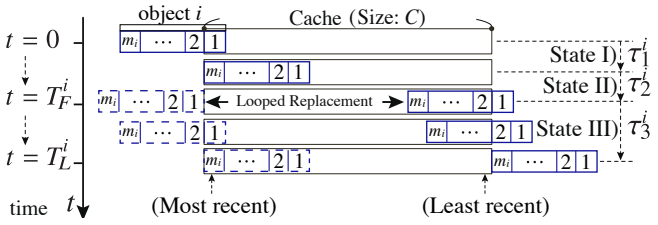

Fig. 1: Positions of packets of object i in a cache

2) Looped Replacement: Unlike object-level caching, where an entire object is inserted or evicted with a single operation, there are time gaps between insertion and eviction of the first and the last packet in the case of packet-level caching. These time gaps cause a situation where a request to the first packet arrives while some of the packets constituting the same object is being evicted from the cache. In this case, all the requests to the object do not hit even though some of the packets are in the cache. This phenomenon is named *looped replacement*.

Looped replacement is explained more precisely by using an example in Fig. 1. Note that the cache always starts evicting object i with p_1^i, as shown in the figure at time $t = T_F^i$. If a request to p_1^i arrives after the eviction of p_{k-1}^i $(1 < k \leq m_i)$, the request misses and p_k^i is replaced with p_1^i. This causes sequential cache misses and replacements, and hence all the packets $p \in \mathcal{P}_i$ miss even though some of the packets are in the cache. Looped replacement occurs when a request to p_1^i arrives in the case that packets in $\mathcal{P}_i$ are partially in the cache.

Looped replacement might degrade cache hit probability since it wastes multiple slots of the cache. Precisely, some of packets in $\mathcal{P}_i$ remain in the cache even though they never hit.

III. Analytical Model of Looped Replacement

1) Overview: We develop an analytical model to derive two kinds of probability of packet-level caching. One is the probability π_i that looped replacement of object i occurs and

978-1-5090-6501-1/17/$31.00 ©2017 IEEE

the other is the probability ψ_i that $p \in \mathcal{P}_i$ hit. The proposed model is similar to the model proposed by Che et al. [4]. Unlike object-level caching, there are intervals between insertion and eviction of the first and the last packets in the case of packet-level caching. Our model, hence, derives π_i and ψ_i by modeling the average of the intervals and the probability that a next request to object i arrives in the intervals.

Factors that push a certain packet to the rightmost side of the cache are different depending on where the packet is in the cache. As shown in Fig. 1, situations can be categorized into the following three states: *State I)* packets $p \in \mathcal{P}_i$ are arriving at the cache, *State II)* all packets $p \in \mathcal{P}_i$ are in the cache, and *State III)* packets $p \in \mathcal{P}_i$ are being evicted. π_i is derived as the probability that a request to p_1^i arrives during State III.

2) Assumptions: Requests to object i follow a Poisson process with rate λ_i. That is, requests to the first packet p_1^i also follows the Poisson process. Consecutive packets are requested with constant interval $1/\mu_i$. In the case of a streaming video, μ_i is determined according to its bit rate.

3) Looped Replacesment Probability: The terms τ_1^i, τ_2^i, and τ_3^i denote expected time when object i stays at each of the three states. Using the terms, the interval between the insertion and the eviction of the first packet p_1^i is derived as $T_F^i = \tau_1^i + \tau_2^i$. Similarly, the interval between the insertion of p_1^i and the eviction of the last packet $p_{m_i}^i$ is $T_L^i = \tau_1^i + \tau_2^i + \tau_3^i$.

Looped replacement of object i occurs if an interval t between two consecutive requests to i satisfies $T_F^i < t < T_L^i$, and thus the probability π_i is calculated by using the cumulative distribution function of an exponential distribution $\mathbb{P}$ as follows:

$$\pi_i = \mathbb{P}_i(t < T_L^i) - \mathbb{P}_i(t < T_F^i) = e^{-\lambda_i T_F^i} - e^{-\lambda_i T_L^i}. \quad (1)$$

Since intervals between two packets are constant $1/\mu_i$, τ_1^i is derived as $\tau_1^i = (m_i - 1)/\mu_i$. τ_2^i is the length of time that $C - m_i$ packets except for $\mathcal{P}_i$, i.e., $p \in \mathcal{N} \setminus \mathcal{P}_i$, arrive at the cache. According to the approximation in [4], τ_2^i is obtained by solving $\sum_{j \in O \setminus \{i\}} m_j \mathbb{P}_j(t < \tau_2^i) = C - m_i$. τ_3^i is the duration that $m_i - 1$ packets including object i, i.e., $p \in \mathcal{N}$, arrive at the cache, and thus it is derived by solving $\sum_{j \in O} m_j \mathbb{P}_j(t < \tau_3^i) = m_i - 1$.

4) Exception for Unpopular Objects: For unpopular objects, there is a case that the insertion and the eviction of object i occur simultaneously, i.e., p_1^i is evicted before $p_{m_i}^i$ arrives. This case occurs if C or more packets, more precisely, $C - m_i$ packets $p \in \mathcal{N} \setminus \mathcal{P}_i$ and m_i packets $p \in \mathcal{P}_i$, arrive at the cache during the interval between the arrivals of p_1^i and $p_{m_i}^i$. The condition is expressed as $m_i + \sum_{j \in O \setminus \{i\}} m_j \mathbb{P}_j(t < \tau_1^i) > C$. In this case, State II does not happen, and hence T_F^i is derived by solving $\mu_i T_F^i + \sum_{j \in O \setminus \{i\}} m_j \mathbb{P}_j(t < T_F^i) = C$. Similarly, T_L^i is derived as $T_L^i = \tau_1^i + \tau_{2'}^i$, where $\tau_{2'}^i$ is obtained by solving $\sum_{j \in O} m_j \mathbb{P}_j(t < \tau_{2'}^i) = C$, since arrivals of any packets in $\mathcal{N}$ push $\mathcal{P}_i$ to the rightmost side in this case.

IV. Numerical Results

1) Evaluation Conditions: We determine properties of deployed objects according to [1], assuming that VoD streaming of movies and TV programs is widely spread. 18,000 video objects, of which request rates λ_i follow a Zipf distribution with parameter $\alpha = 1$, are deployed. Each video is divided into 1 KBytes packets. The total request rate is set to 0.05 requests/s, assuming the cache is located at an access network having a few thousands of subscribers. The lengths of videos are 30, 60, and 120 minutes. The number of videos of 30, 60, and 120 minute lengths are the same, i.e., 6,000. The bit rates of videos are determined according to a truncated normal distribution with the mean of β, the standard deviation of 10, and the range between 0.5 and 25 Mbps according to the bit rates of Netflix videos [5].

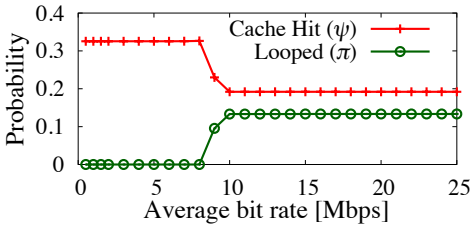

Fig. 2: Probability of cache hit and looped replacement

2) Looped Replacement Probability: The horizontal axis of Fig. 2 indicates the average bit rate. The vertical axis does the average of the cache hit and looped replacement probabilities (ψ_i and π_i) weighed with the arrival rate λ_i of each object i. In the case that the average bit rate β exceeds 10 Mbps, the cache hit probability steeply drops, whereas the looped replacement probability increases. The differences of both the probabilities with $\beta = 8.0$ and $\beta = 10.0$ are the same (approximately 0.13), and hence looped replacement causes the degradation in the cache hit probability. The observations are summarized as follows: First, looped replacement occurs if the average bit rate of video is high, i.e., the average object size is large. Second, looped replacement degrades cache hit probability.

V. Conclusion

This paper develops an analytical model to obtain the probability of looped replacement in the case of packet-level caching. Through numerical evaluations assuming a large-scale VoD is deployed, we have revealed that looped replacement is not negligible if high bit rate video objects, such as UHD videos, are cached, and it causes the degradation in cache hit.

Acknowledgements

This work was supported by Grant-in-Aid for Challenging Exploratory Research (16K12417).

References

[1] W. Bellante *et al.*, "On netflix catalog dynamics and caching performance," in *Proceedings of IEEE CAMAD*, Sep. 2013.

[2] A. Anand *et al.*, "Packet caches on routers: The implications of universal redundant traffic elimination," in *Proceedings of ACM SIGCOMM*, Aug. 2008.

[3] Y. Thomas *et al.*, "Object-oriented packet caching for ICN," in *Proceedings of ACM ICN*, Sep. 2015.

[4] H. Che *et al.*, "Hierarchical web caching systems: Modeling, design and experimental results," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 7, pp. 1305–1314, Nov. 2002.

[5] Netflix, "Internet connection speed recommendations," (Accessed Aug. 6, 2017). [Online]. Available: https://help.netflix.com/en/node/306/