一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
NS2022-91(2022-10)

［依頼講演］遅延キャッシング効果の抑制を目的とした集約バーストスコアによるキャッシュ置換方式

フェリファリアント†　　上山　憲昭††

† 福岡大学大学院 工学研究科 電子情報工学専攻　〒 814–0180 福岡市城南区七隈 8–19–1
†† 立命館大学 情報理工学部　〒 525–8577 滋賀県草津市野路東 1–1–1
E-mail: †td196502@cis.fukuoka.ac.jp, ††kamiaki@fc.ritsumei.ac.jp

あらまし　コンテンツ配信ネットワーク (CDN)、プロキシ、またはゲートウェイなどのオンラインキャッシュ システムは，要求データがキャッシュに存在しないとき (キャッシュミス) は，コンテンツのオリジナルを保有するサーバ (オリジンサーバ) から要求されたデータを取得し，キャッシュした後，要求ユーザに送信する．そのためオリジンサーバからデータを取得する前に，同じデータに対し複数の要求が発声した場合，これらもキャッシュミスとなるが，従来のキャッシュヒット率の近似式やシミュレーション評価では，後続する要求はキャッシュヒットしたと見なされる結果，ヒット率が高く見積もられる結果となる．本現象を本稿では Delayed caching(遅延キャッシング) 効果と呼ぶ．本稿では，キャッシュサーバやゲートウェイでの遅延キャッシングの影響を評価し，オンラインキャッシングシステムの遅延キャッシングキャッシュの影響を抑制するキャッシュ方式を提案する．
キーワード　オンラインキャッシングシステム，遅延キャッシング，LRU，ヒット率低下

[Invited Lecture] Cache replacing method using burst score aggregation to suppress delayed caching effects

Feri FAHRIANTO† and Noriaki KAMIYAMA††

† Graduate School of Engineering, Fukuoka University
8–19–1, Nanakuma, Jounan, Fukuoka 814–0180
†† College of Information Science and Engineering, Ritsumeikan University
1–1–1 Nojihigashi, Kusatsu, Shiga 525–0058
E-mail: †td196502@cis.fukuoka.ac.jp, ††kamiaki@fc.ritsumei.ac.jp

Abstract　An online-caching system such as Content Delivery Network (CDN), Proxy, or Gateway, uses a caching method to enhance network connectivity. It can store temporary data packets from an original content producer by utilizing the caching system. In order to receive the requested data content, the client transmits a request packet to the intermediate caching server or gateway. The different throughput between the request packet from the client and the data packet from the original server, in addition to the cache capacity size, seem to have an impact on the hit-ratio degradation defined as delayed caching. This phenomenon occurs when the average response time of the original server's data packet is slower than the average request time of the client's request packet. Therefore, the paper investigates the consequences of delayed caching in the caching server or gateway and proposes a suppressing method in the cache replacement algorithm in the online-caching system. We clarify our finding with simulation with different values of request skewness (α) representing the request content popularity that follows Zipf distribution, against the arrival request time and original server response time.
Key words　Online-caching system, Delayed caching, LRU, Hit-ratio decline

1. Introduction

An online-caching system such as Content Delivery Network (CDN) or proxy server promotes a caching system to improve network performance so the user can experience a low latency connection. Moreover, the issue of online content caching is a critical problem studied in computer-networking systems. The conventional objective of caching is to mini-

mize the cache misses or the total cost of data retrievals. An exquisite online caching algorithm should provide a lower average content access latency, resulting in a better user experience.

Cache replacement algorithms play a central role in the design of any caching component. Cache replacement algorithms usually maximize the cache hit ratio by attempting to cache the data items most likely to be referenced in the future. Since the future data reference pattern is typically difficult to predict, a common approach is obtained from the past by caching the data items which were referenced most frequently. This approach is implemented by, e.g., the LRU cache replacement algorithm. However, maximizing the cache hit ratio alone does not guarantee the best client response time in the online environment. In addition to maximizing the cache hit ratio, a cache replacement algorithm for contents should also minimize the cost of cache misses, i.e., the delays caused by fetching documents not found in the cache. Clearly, the content that took a long time to retrieve and, more often, should be preferentially retained in the cache.

This article proposes the Burst Score Aggregation (BSA) that represents the high occurrence degree during a delay in obtaining the content so that it can suppress the effect of delayed caching. The metric is used to modify the cache replacement algorithm in the caching system, which can prevent hit-ratio degradation because of cache misses due to delayed caching. We use a computer simulation to clarify our method with request distribution that follows the Zipf distribution with skewness parameter (α). The Organization of this paper consists of six sections: introduction, related work, delayed caching in the online-caching system, delayed caching suppressing method, numerical evaluation, and conclusion.

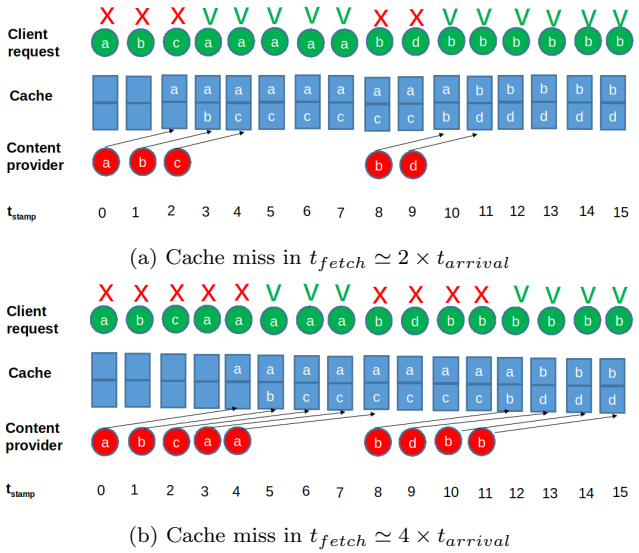
2. Related work

Investigation of the effect of delayed caching has been published in the literature [1]. Therefore, implementing a caching system must also consider the delayed caching problem. We presume that the fundamental problem arises from an imbalance in throughput and latency between the requester and the content provider. If the requester throughput is poor relative to the content provider, only a few requests may be able to arrive during a fetching interval. If the requester's throughput is greater than the provider's, then more requests come during the fetching period causing more cache misses. Several researchers have sought a caching replacement approach to address the issue of delayed caching. The authors propose the caching replacement technique that enables a reduction of the effect of delayed caching so that it can prevent the cache hit ratio degradation.

Atre et al. in [3] calculated each unique content aggregation delay and its subsequent turn request or future request as key parameters to the cache replacement algorithm. Even though the technique is considered both offline and online file caching, it is suited only for a system with a significant buffer to store the incoming request before it is requested in the cache. Moreover, calculating aggregation delay is a relatively complex computation that requires a longer calculation time which is not suited for a router that operates with high data throughput and less memory buffer. Another author, Chang et al. in [4], considered passing a particular content directly to the original server that has a specific delay metric instead of storing it in the caching system. The metric is bound to a specific threshold of delay and file size to resist the overall hit-ratio decline. As an alternative for suppressing the delay caching problem, we proposed a replacement algorithm that considers the burstiness of content. The BSA level is used as a metric for content eviction in the cache. Additionally, a database is provided so the caching server can periodically update and track the content's BSA.

3. Delayed caching in online-caching system

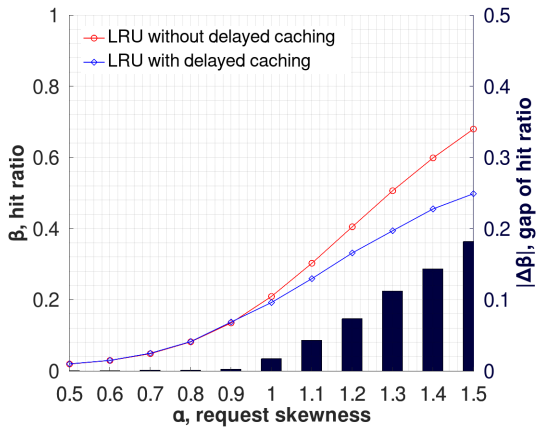
This section describes the effect of delayed caching in an online-caching system. Figure 1 depicts a typical cache with a delay caching that occurred in the caching system. Whenever a client requests content but the content absences in the cache, the proxy or gateway forwards this request packet to the original content provider to fetch the data content. The retrieval process takes some time, known as the producer response time (t_{fetch}). And the average arrival time between the client's requested packets, known as $t_{arrival}$. If a new request for the same data content arrives at the cache before t_{fetch} is completed, this new request suffers from a cache miss in the cache. Then, if the condition repeatedly occurs in the online-caching server, it causes a significant hit ratio



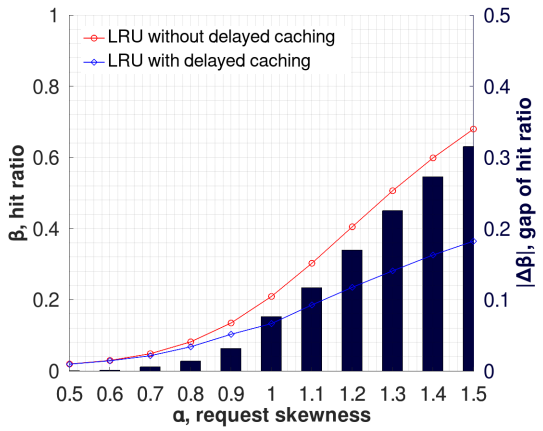
(a) Cache miss in $t_{fetch} \simeq 2 \times t_{arrival}$

(b) Cache miss in $t_{fetch} \simeq 4 \times t_{arrival}$

Fig. 1: Cache miss because of delayed caching in different length of t_{fetch} relative to $t_{arrival}$



(a) In case of $t_{fetch} \simeq 100 \times t_{arrival}$



(b) In case of $t_{fetch} \simeq 1000 \times t_{arrival}$

Fig. 2: The hit-ratio (β) decline in LRU in different value of t_{fetch} relative to $t_{arrival}$.

decline.

The hit ratio is defined as the number of requests satisfied by the cache divided by the number of all requests that arrived at the cache. The higher the hit ratio is or the lower the miss ratio is, the more requests are satisfied by the cache. The hit-ratio degradation that occurs due to caching delay is depicted in Figure 1. Figure 1(a) shows the case when t_{fetch} is two times of $t_{arrival}$, whereas Figure 1(b) shows the case when t_{fetch} is four times of $t_{arrival}$. In the first case, just eleven requests are satisfied by the cache, whereas seven requests are satisfied by the cache in the second case. The longer t_{fetch} relative to $t_{arrival}$ can cause more cache miss. As a result, the hit ratio is declining in the online-caching server.

Figure 2 shows the hit degradation that exist in online-caching server that we have presented in [1]. The hit degradation is subjected to the incoming request that obeys Zipf distribution representing the content popularity and the comparison between t_{fetch} and $t_{arrival}$. It shows that the smaller α creates smaller gap compare to higher α in every different case of t_{fetch} relative to $t_{arrival}$. Furthermore, $\Delta\beta$, the gap between idealized LRU and delayed caching LRU, monotonically increased as t_{fetch} increased. The average of $\Delta\beta$ were about 15% and 30% when $t_{fetch} \simeq 10 \times t_{arrival}$, and $t_{fetch} \simeq 100 \times t_{arrival}$ respectively. We have found that the more popular content was higher likely to experience delayed caching. This is because more requests for popular content were more likely to be requested during the t_{fetch} interval, so they experienced cache miss in the cache.

4. Delayed caching suppressing method

A specific strategy to suppress the effect of delayed caching is developed. The technique relies on the burstiness of contents as a key parameter to prioritize the residency in the cache system. The procedure of the proposed method can be generally described in Figure 3. First, we need to capture L number of requests during fetching time t_{fetch}. Next, a burst score is calculated and aggregated from the L number content's request. A database consisting of records from all the BSA of unique contents is built. Finally, the cache implements a cache replacement algorithm that evicts the content with the lowest BSA whenever the capacity is full after receiving content from the producer. Therefore, the parameter burst score and BSA are the essential keys in this suppressing method.

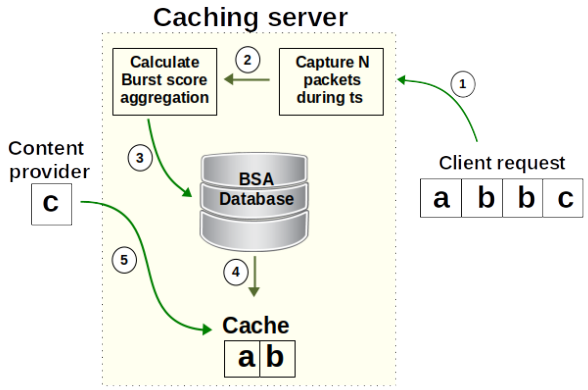


Fig. 3: Architecture of BSA based cache replacement algorithm

4.1 Burst score

We have shown that the popular content has a higher probable experiencing delayed caching in the previous section. Under certain conditions, a particular content occurrence is dominant at a specific sampling time compared to other sampling times. This situation is defined as a burst period. The burst score of the particular content can be measured by comparing the occurrences of that content between sampling time and elapsed time. Hoonlor et al. in [5] introduced the burst score of a particular content x for the specific sampling period, which is similar to t_{fetch} in an online-caching server, which can be defined by

$$Burst(x, t_{fetch}) = (\frac{E_{t_{fetch}}}{E} - \frac{1}{T}), \quad (1)$$

where $E_{t_{fetch}}$ is the total number of occurrences of event $x \in \{c_1, .., c_N\}$ in sampling time interval of t_{fetch} and E is the total number of occurrences of $x \in \{c_1, .., c_N\}$ in total elapse time T. The set of $\{c_1, .., c_N\}$ express the number of N total unique contents.

As for example, lets consider five sampling periods namely t_{fetch1}, t_{fetch2}, t_{fetch3}, t_{fetch4}, and t_{fetch5} respectively, with three different contents c_1, c_2, and c_3 as seen in Figure 4. The average number of content in each sampling time is five contents. This value can be obtained by dividing t_{fetch} with $t_{arrival}$. Using formula 1, it is obtained that the content's burst score is at maximum when the content appears for the first time during the sampling period. Afterwards, it is about in the range between zero and the maximum value, or always in positive value. On the other hand, the burst score becomes negative if the content does not appear at any rate in the sampling period. Furthermore, the burst score produces identical scores for contents that have a different number of occurrences within the sampling period and are present for the first time. As a result, the burst score can not describe the magnitude of occurrence of a specific content compared to others. It is a metric to identify a particular content's occurrence in the sampling period relative to the total of previous occurrences.

We could interpret the burstiness of a certain content when we track the burst score from the beginning sampling period up to the latest. Therefore, we introduce burst score aggregation (BSA) to measure the burst level of content by accumulating from previous burst scores up to elapse time. The BSA of the specific content can be defined by

$$BSA(x, t_{fetch}) = \sum_{i=1}^{M} Burst(x, t_{fetch}), \qquad (2)$$

where M is the total number of t_{fetch} sequence up to elapse time T. Thus, M is ratio between T and t_{fetch}.

BSA necessities the accumulation of the previous burst score up to elapse time to interpret the trend of specific content being frequently requested or not. If the frequency of specific content gradually increases over time and has never been absent in all sampling period sequences, then the BSA level will reach the maximum value. Using previous example as shown in Figure 4, BSA of c_1 has the highest value at t_{fetch5} because c_1 consistently presents in all sampling period during the elapse time. Moreover, the number of occurrences is also gradually increasing. On the contrary, even though c_2 has the same BSA level at the beginning of sampling time, it remains stagnant because its frequency is in a declining trend.

On the other hand, c_3 becomes the second highest at t_{fetch5} even though it has a negative burst score in the beginning, but it is compensated with the next positive frequency trends. As a result, the BSA is significantly improved at t_{fetch5}. Therefore, the high BSA means that the content has a high probable being dominantly requested in the future, so this parameter can be used to determine which content

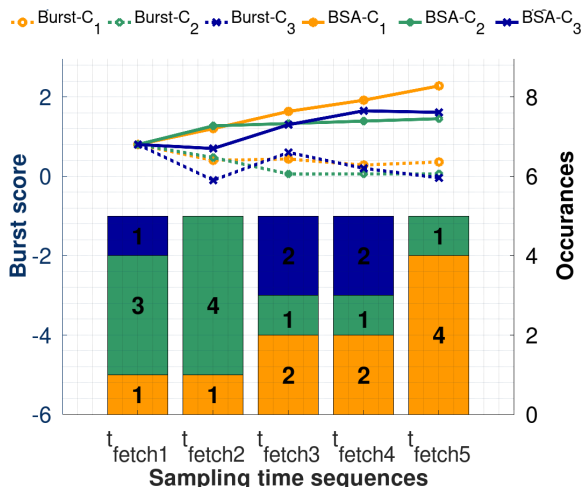
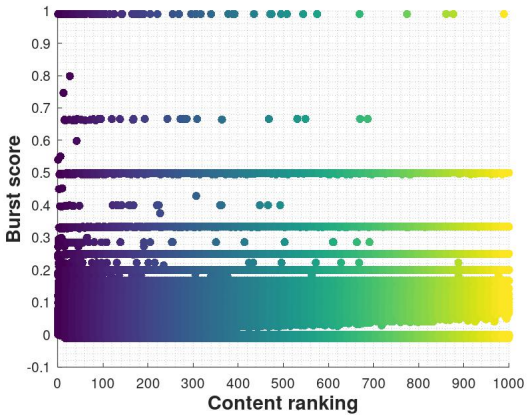
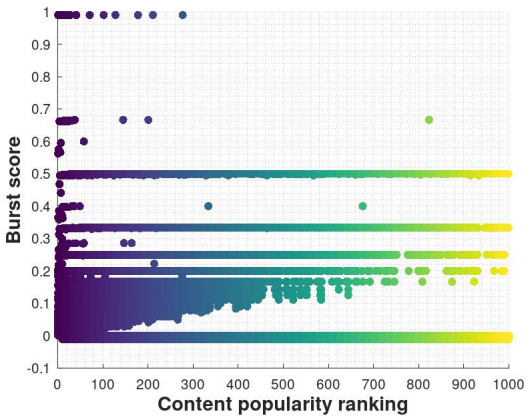


Fig. 4: Burst score and Burst score aggregation (BSA) with five sampling period, t_{fetch}, and three unique contents.



(a) When α=1.0



(b) When α=1.5

Fig. 5: Burst score distribution against skewness parameter (α) of Zipf distribution for 1000 unique contents (N)

should be prioritized in the cache.

To understand the characteristic of BSA against request distribution, Figure 5 shows the burst score distribution of content that follows the Zipf distribution. Two skewness parameters of Zipf distribution for α equal to 1.0 and 1.5 are taken as a comparison. The total elapsed time is about 10 seconds with a sampling interval, t_{fetch}, of about 10 ms. From the figure 5, we know that popular contents have a more positive burst score than unpopular content. As the value of α increases, the number of positive burst scores is concentrated on the most popular content. According to Zipf distribution characteristics, the most popular content is more likely to appear more frequently. Thus, it is always found in every sequence of the sampling period.

4. 2 Cache replacement algorithm

As we have mentioned earlier, the cache replacement algorithm is essential in optimizing the cache hit ratio in the caching system. We know that a cache miss in delayed caching is mainly caused by the massive arrival of the same particular content in the unfinished fetching process. Therefore, modifying the cache replacement algorithm can effectively prevent the hit-ratio decline. If a caching system prioritizes content based on its burst score aggregation, i.e., content with the lowest burst score aggregations will be evicted from the cache system. The most requested content in every sampling period will always remain in the cache. To implement this idea, the CDN server or proxy must first construct a database that records the BSA of each unique content. Second, the cache replacement policy of caching system evicts content with the lowest content burst score aggregation, as seen in Figure 5.

Algorithm 1: Burst score database.

1: $ts \leftarrow$ Average content provider response time;
2: $T \leftarrow$ Elapse time;
3: Burst database $\leftarrow$ Burst(x, t_{fetch})=0 $\forall x \in \{c_1, .., c_N\}$;
4: **while** $T \mod t_{fetch} = 0$ **do**
5: Count interest Burst(x, t_{fetch}) $\forall x \in \{c_1, .., c_N\}$;
6: Aggregate burst score = Burst(x, t_{fetch})[previous]+Burst(x, ts)[current] $\forall x \in \{c_1, .., c_N\}$ and store in Burst database;
7: **end while**

Algorithm 1 shows the pseudo-code in calculating the burst scores for all available contents. Initially, the online-caching server set the value of t_{fetch} with the average producer response time. Afterward, the router calculates all burst scores in every t_{fetch} interval. The calculation result is aggregated by accumulating the previous burst score with the current value for all content items. Then, it is stored in the database.

Algorithm 2 describes the pseudo-code of content replacement in the caching server. When it receives content from a original content provider, it stores it in cache. However, if the capacity of cache is oversized, the online-caching server executes a content replacement algorithm that removes a particular content with the lowest BSA.

Algorithm 2: Content replacement

1: **while** Cache receives content **do**
2: **if** Cache size is full **then**
3: Evict content in cache with lowet BSA;
4: Store content in cache;
5: **else**
6: Store content in cache;
7: **end if**
8: **end while**

5. Numerical evaluation

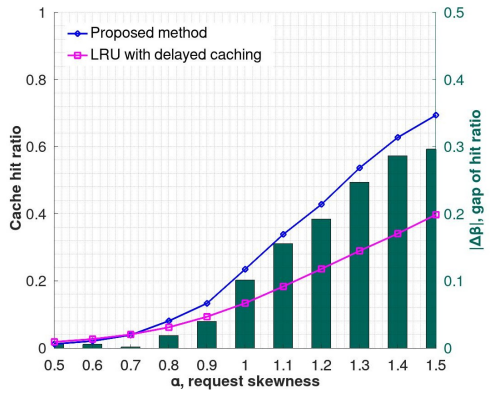
We evaluate and clarify our hypothesis using computer simulation consisting of 3 parallel processes programmed by Python 3.8. Furthermore, We compare the hit ratio performance between the proposed method and LRU in the online-caching server. The sampling delay, t_{fetch}, was set to different scenarios. Table 1 shows the metric parameters used in the experiment.

Tab. 1: Setting values of main parameters

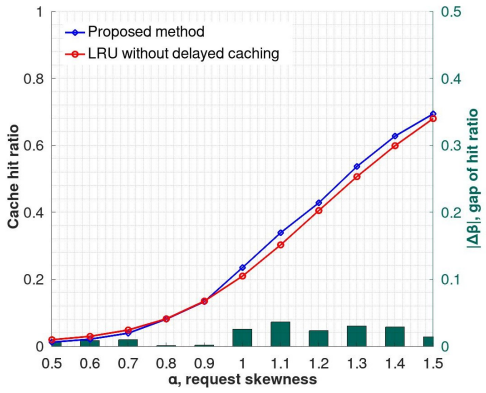
Paramater	Value
Number of unique content items (N)	1000
Cache size	10
Request skewness (α)	0.5 - 1.5
Interval(ts)	10 ms and 100 ms
Interest rate	10000 interests/s

We collected the data from 100000 requests sent by the requester or client. The cache hit ratios obtained by the simulator were plotted against the request skewness, α, for each of the three different cases, namely LRU without delayed caching, LRU with delayed caching, and the proposed method as seen in Figure 6. In general, the result shows that the proposed method enable to prevent the cache hit ratio decline averagely 30% from LRU with delayed caching in case of t_{fetch} equal to 10 ms as seen in Figure 6(a), and 40% in case of t_{fetch} equal to 100 ms as seen in Figure 6(c).

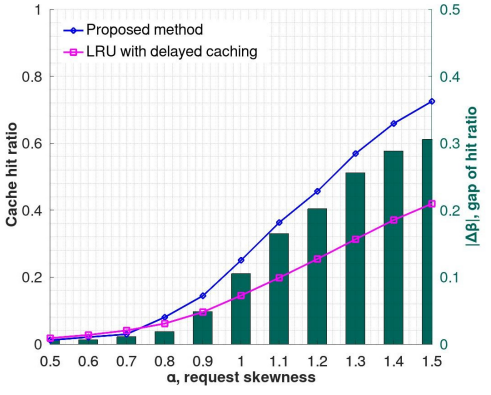
The hit ratio of the proposed method outperformed the LRU without delayed caching, about 1% in case of t_{fetch} equal to 10 ms as shown in Figure 6(c)and 3% in case of t_{fetch} equal to 100 ms respectively as shown in Figure 6(d). This is because the proposed method utilizes the burst scores that are highly correlated with content popularity to evict the content in the cache, so the order of arrival request sequence does not affect the eviction in the cache. In addition, a high α indicates that a few numbers of popular content dominate requests. The overall cache-hit ratio increases significantly when cache capacity is merely sufficient to store those popular contents. On the other hand, the LRU algorithm caches the content in accordance with the order of



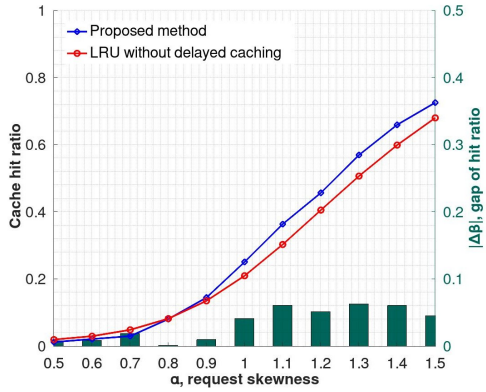
(a) When t_{fetch}=10 ms



(b) When t_{fetch}=10 ms



(c) When t_{fetch}=100 ms



(d) When t_{fetch}=100 ms

Fig. 6: Comparison of hit ratio between LRU without delayed caching, proposed method, and LRU

arrival request sequences, i.e., if the unpopular content is sent between popular content, then LRU will cache it for sure since it is the most recent. As a result, it creates more cache misses.

6. Conclusion and Future work

The numerical evaluation shows that the BSA is effective in preventing the hit-ratio decline because of delayed caching. The BSA suppressing method is optimum to suppress the effect of the delayed caching when the content request distribution follows the Zipf distribution with skewness parameter α greater than 0.9 and with a small number of unique contents.

The investigation to optimize the Burst Score Aggregation database will be deeply examined in the future. The technique to reduce calculation complexity against the increasing number of unique contents is also considered for future works.

Acknowledgement

This work was supported by JSPS KAKENHI Grant Number 18K11283 and 21H03437, as well as the Ministry of Religious Affairs of the Republic of Indonesia (MORA 5000 Doktor).

References

[1] F. Fahrianto and N. Kamiyama, "Impact of Delayed Caching on Hit-ratio of ICN Router", IEICE 2022 General Conference, BS-3-5, Online, Mar. 2022.

[2] P. Scheuermann, J. Shim, and R. Vingralek, "A case for delay-conscious caching of Web documents", The sixth international conference on World Wide Web, Elsevier Science Publishers Ltd., 1997.

[3] N. Atre, J. Sherry, W. Wang, and D. S. Berger, "Caching with Delayed Hits", In Proceedings of the Annual conference of SIGCOMM '20, ACM, New York, NY, USA, 2020.

[4] C. Zhang, H. Tan, G. Li, Z. Han, S. H. . -C. Jiang and X. -Y. Li, "Online File Caching in Latency-Sensitive Systems with Delayed Hits and Bypassing," IEEE INFOCOM 2022 - IEEE Conference on Computer Communications, 2022.

[5] A. Sabnis and R. K. Sitaraman, "TRAGEN: a synthetic trace generator for realistic cache simulations," In Proceedings of the 21st ACM Internet Measurement Conference (IMC '21), ACM, New York, NY, USA, 2021.

[6] Sundarrajan et. al, "Footprint descriptors: Theory and practice of cache provisioning in a global cdn," In Proceedings of the 13th International Conference on emerging Networking Experiments and Technologies, pp. 55-67. 2017.

[7] B. Wissingh, C. Wood, A. Afanasyev, L. Zhang, D. Oran, and C. Tschudin, "Information-Centric Networking (ICN): ContentCentric Networking (CCNx) and Named Data Networking (NDN) Terminology", RFC 8793, June 2020.

[8] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, R. L. Braynard, "Networking Named Content," CoNEXT 2009, Rome, December 2009.