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
NS2022-168(2023-03)

Delayed Hit 効果抑制のためのバーストスコア集計の処理負荷の抑制法

フェリファアリアント[†]　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　キャッシングは高人気コンテンツに対するネットワーク品質を向上させるが，キャッシュ制御法，特にキャッシュ置換アルゴリズムはキャッシュヒット率に大きな影響を与える．これまでに，キャッシュの性能を低下させる１つの要因として，Delayed Hit 効果が指摘されている．そこで著者らは，Delayed hit 効果によるキャッシュの性能低下に対処するために、バーストスコア集約 (BSA) キャッシュ置換アルゴリズムを提案した．BSA は特にコンテンツ人気度分布の歪度パラメータが 0.9 を超える Zipf 分布に従う場合に，ヒット率低下の抑制に効果的である．しかし BSA は従来のキャッシュ置換法と比較して，多くの処理資源を必要とする．そこで本稿では，BSA と比較して処理負荷を軽減する R-BSA (Reduced Burst Score Aggregation) キャッシュ置換法を提案する．計算機シミュレーションにより，R-BSA は BSA と比較して、ヒット率の低下を約 1%に抑えながら，約 5%，高速な処理時間を低減する．

キーワード　オンライン キャッシング システム、ディレイド ヒット キャッシング、LRU、バースト スコア集計

Reduced Burst Score Aggregation in Suppressing Delayed-Hit Caching Effects

Feri FAHRIANTO[†] and Noriaki KAMIYAMA[††]

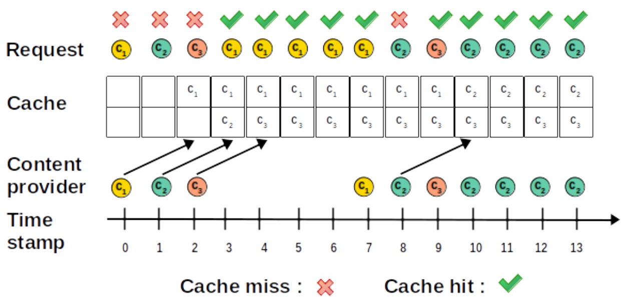
[†] Graduate School of Engineering, Fukuoka University
8–19–1, Nanakuma, Jounan, Fukuoka 814–0180
[††] College of Information Science and Engineering, Ritsumeikan University
1–1–1 Nojihigashi, Kusatsu, Shiga 525–0058
E-mail: [†]td196502@cis.fukuoka.ac.jp, [††]kamiaki@fc.ritsumei.ac.jp

Abstract　Caching has been proven to increase network connectivity against popular content. Caching strategy, notably the cache replacement algorithm, contributes substantially to hit ratio performance at the caching system. Several authors have introduced delayed-hit caching that causes a severe hit-ratio decline in the caching performance. We proposed a burst score aggregation (BSA) cache replacement algorithm to cope with the performance degradation because of delayed-hit caching. BSA has been proven effective against the hit ratio degradation, especially when request contents follow Zipf distribution at skewness parameter greater than 0.9. Unfortunately, BSA requires more intensive processing resources than conventional cache replacement algorithms. Therefore, we introduce reduced burst score aggregation (R-BSA) cache replacement algorithm that promotes processing load reduction compared to BSA. R-BSA shows a faster processing time of about 5% as opposed to BSA in a computer simulation. As a trade-off, the hit ratio of RBA also experiences a slight downturn of about 1% from BSA. To explain this behavior, we investigate and analyze the R-BSA cache replacement algorithm's performance and its hit ratio drops.

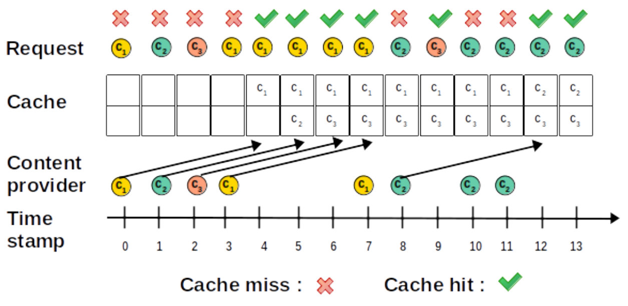
Key words　Online-caching system, Delayed-hit caching, LRU, Burst Score Aggregation

1. Introduction

Online-caching systems like Content Delivery Networks (CDN), Information-centric Networking (ICN), and Proxies support a caching system to enhance network performance so the user can experience a low-latency connection. The discussion of online content caching is being investigated actively in the computer networking research area. The typical objective of an online caching system is to reduce the overall cost of retrieving data. Instead of obtaining from the original content provider, the user can retrieve the content from the intermediary caching server, which shortens the distance between the requester and provider.



(a) Cache misses in case of content download time of about 2 times from request inter arrival time



(b) Cache misses in case of content download time of about 4 times from request inter arrival time

Fig. 1: Cache misses increase due to delayed-hit caching

Caching strategy is substantial to the development and design of the online-caching system. Typically, cache replacement algorithms enhance the cache hit ratio of requested content by storing the content items that are most likely to be accessed in the near future. However, the future data reference pattern is obviously difficult to predict. A generic strategy is to cache the most frequently cited contents from the historical data. This method is utilized by classic algorithms such as the Least Recently Used (LRU) and Least Frequently Used (LFU). Even though the classic replacement algorithm can improve the hit ratio of caching system, it is not considered the online caching environment. Therefore, the conventional cache eviction algorithms do not anticipate the impact of a delay hit in caching. The conventional cache replacement algorithm, LRU, experiences almost two times cache misses whenever content download time takes two times longer with the same cache size and request pattern, as shown in Figure 1. Moreover, the LRU cache replacement algorithm can be suffered from hit ratio degradation of about 30% because of delayed-hit caching, as reported in [1].

A strategy to suppress the degradation in hit ratio of the online caching system has been investigated by several scholars, as reported in [4], [5], and [6]. Most of the proposed methods require an intensive calculation that requires more processing resources. We have proposed 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 as reported in [2]. However, BSA still consumes lots of computation resources. In this paper, we introduce the Reduced Burst Score Aggregation (R-BSA) eviction algorithm that promotes a reduction of calculation resources with a slight decrease in hit ratio performance. A numerical evaluation with computer simulation is given as justification to show the finding. The Organization of this paper consists of five sections: introduction, related work, reduced burst score aggregation, evaluation, and conclusion.

2. Related work

The effect of delayed caching has been the subject of published research [1]. Implementing a caching system must, therefore, additionally consider the delayed caching issue. We suspect that the root cause of the issue is a disparity in throughput and latency between the requester and content providers. During a fetching interval, just a few requests can arrive if the requester throughput is low compared to the content provider. If the requester's throughput is greater than the provider's, more requests will arrive during the fetching time, resulting in greater cache misses. Many researchers have sought a cache replacement strategy to address the problem of delayed caching. The findings discuss a caching replacement strategy that reduces the impact of delayed caching to minimize a decline in the cache hit ratio.

Atre et al. in [5] determined each unique content aggregation delay and subsequent turn request or future request as crucial parameters for the cache replacement technique. Even though the technique is classified as both offline and online file caching, it is only suitable for systems with a large buffer to store incoming requests before they are fetched from the cache. Moreover, the calculation of aggregation delay is quite difficult. It requires a longer calculation time, which is unsuitable for a router with a fast data throughput and a small memory buffer. Another author, Chang et al. in [6], proposed delivering material with a predefined delay metric directly to the source server rather than storing it in the caching system. The metric is constrained to a particular

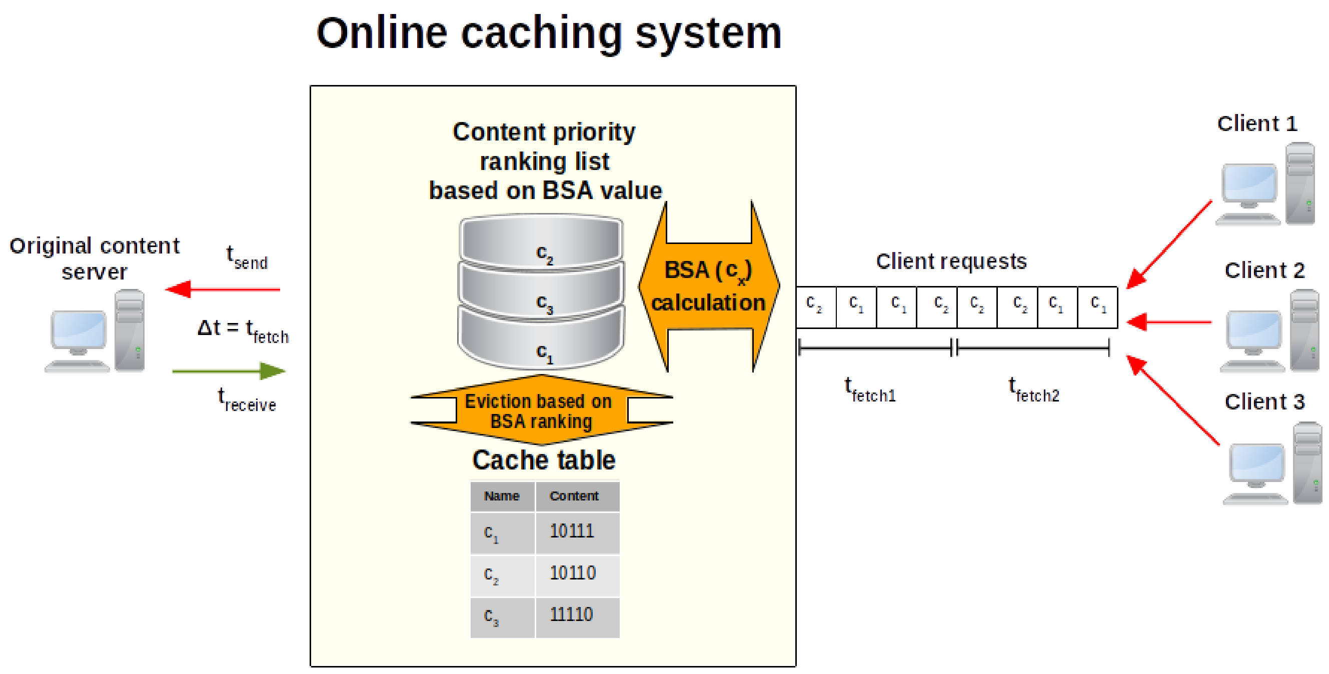


Fig. 2: Architecture of BSA cache replacement algorithm

threshold of delay and file size to resist the overall drop in hit ratio.

Moreover, we proposed a replacement algorithm that considers the burstiness of contents in [2] and [3]. The BSA level was 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. Reduced burst score aggregation

Reduced burst score aggregation (R-BSA) was a simplified form of the BSA replacement algorithm. The technique of the BSA algorithm relies on the burstiness of contents as a key parameter to prioritize the residency in the cache system. Requests from clients at the online caching system are captured at every content download time. All content burstiness levels are calculated and ranked during this interval in the BSA database table. The cache uses this ranking table to prioritize content residency in the cache table. The lower ranking of contents is deleted whenever the cache capacity is full. The procedure of the BSA cache replacement algorithm can be generally described in Figure 2.

BSA and R-BSA cache replacement algorithms are necessary to measure the burstiness of content. This can be described by burst score. The burst score is measured by comparing the occurrences of content during content download time from the original content provider with its total occurrences in the past. Hoonlor et al. in [11] introduced the burst score of a particular content x within content download interval, t_{fetch}, in an online-caching server, which can be formulated by

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

where $E_{t_{fetch}}$ is the total number of occurrences of event $x \in \{c_1, .., c_N\}$ within time frame of t_{fetch} and E is the total number of occurrences of $x \in \{c_1, .., c_N\}$ in total elapse time T that is relative to inter-arrival time of request, $t_{arrival}$. The set of $\{c_1, .., c_N\}$ express the number of N total unique contents. R-BSA intentionally reduces the number of processor computations for creating a ranking list database of content. Therefore, burst score calculations are reduced from the set of $\{c_1, .., c_N\}$ to the set of $\{c_1, .., c_S\}$ where S shows the unique set of contents during an interval of t_{fetch}. As a result, the number of calculation loads in each t_{fetch} can be reduced from N times to S times where S is much smaller than N.

Burst score is difficult to compare between one unique content to another since it focuses only on specific content occurances within t_{fetch} interval. Thus, the burst score of each content needs to be accumulated from the begining up to the present time, called as BSA value. Since the burst score of unique contents have been reduced from N to S in every t_{fetch}, we denote RBSA as the reduced aggregation of content burstiness level. The RBSA of the specific content can be defined by

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

where M is the total number of t_{fetch} sequence up to elapse time T. Thus, M is ratio between T and t_{fetch}. This RBSA is the burstiness level of content used by the R-BSA cache replacement algorithm.

4. Numerical evaluation

We evaluate the performance of R-BSA using a computer simulation. The simulation software was a multi-threaded application programmed by python 3.8. We compare the simulation result with BSA and LRU cache replacement algorithm as a based line. The setting parameters of the experiments are shown in Table 1. Furthermore, we collected the data from one million requests generated by the client. The performance of the cache hit ratio and simulation running time obtained were plotted against the different cases of request skewness (α) that obey the Zipf distribution. We analyzed the performance of R-BSA starting with α greater and equal to 0.9 because the suppression effect of delayed-hit caching is effective starting at this point, as reported in [2].

Tab. 1: Simulation parameter setting

Paramater	Value
Total unique content items (N)	10,000
Cache size	10
Request skewness of Zipf distribution (α)	0.9 - 1.5
Request rate	10,000 requests/second
Total requests	1,000,000 requests
t_{fetch}	10 ms
$t_{arrival}$	0.1 ms

4.1 Processing time

From the simulation result, the proposed method, R-BSA, can reduce the simulation running time by about 5% on average from the BSA cache replacement algorithm, as depicted in Figure 3. This simulation running time continues to decline whenever the skewness parameter of content, α, increases constantly. We analyze that the running time decrease is mainly caused by the reduction of BSA calculation in each interval t_{fetch}. To clarify this finding, we conduct a further simulation to capture the number of BSA calculations.

As mentioned earlier, we conducted a simulation to capture the computation processing load. The calculation processing load mostly comes from the number of unique content in t_{fetch} time frame. For every interval of content download time, t_{fetch}, the burst score aggregation level of unique contents must be calculated. Afterward, these results are used to construct a database of content ranking. Since BSA demands that all unique contents must be calculated even though those contents do not appear during the t_{fetch} time frame, this causes high connectivity between the processing load and the total unique contents. If the number of unique content rises in the online caching system, then the calculation processing load is also increasing.

On the other hand, R-BSA does not necessitate all unique

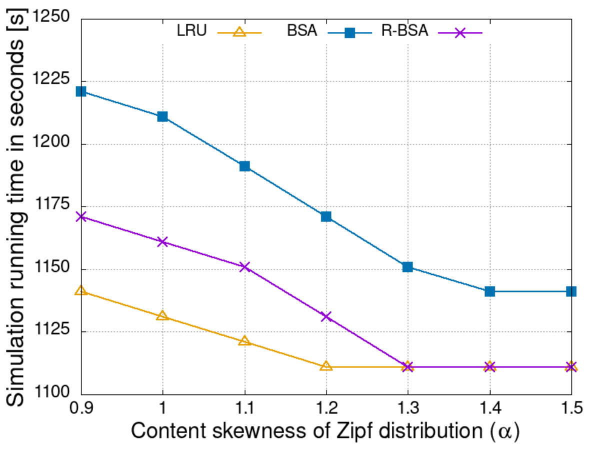


Fig. 3: Comparison of running simulation time for one million requests between BSA, R-BSA, and LRU cache replacement algorithm

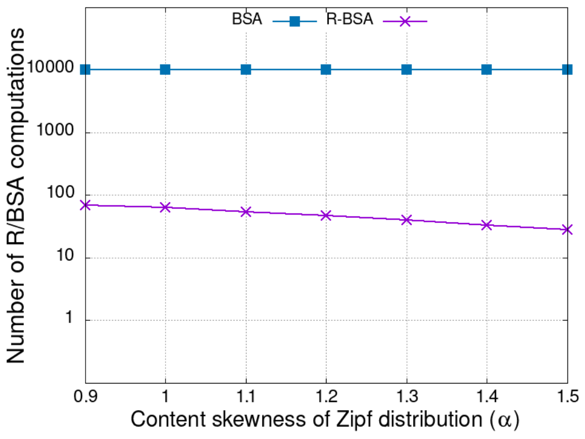


Fig. 4: Comparison of number BSA computation between BSA and R-BSA cache replacement algorithm in every content download timeframe against different case of skewwness parameter (α) of Zipf distribution

content calculations. R-BSA offers a simple form of BSA processing load reduction by calculating the BSA value of the unique content that only appears during a specific interval of t_{fetch}. As a result, when the total unique contents grow, the number of burst score aggregation calculations of contents does not grow. It means that R-BSA has decoupled the correlation between total unique contents and processing load, i.e., for 10,000 total unique contents, the BSA algorithm necessary to calculate of 10,000 BSA value calculation in each t_{fetch} time frame while R-BSA computes in about 60 averagely as seen in Figure 4. Therefore, this computation decrement creates a faster processing time.

4.2 Hit ratio

The hit ratio of R-BSA is slightly below the BSA hit ratio. However, the gap between BSA and R-BSA is about below 1%. Moreover, this gap constantly shrinks whenever the α increases, i.e., at α = 1.5, the hit ratio of BSA and R-BSA

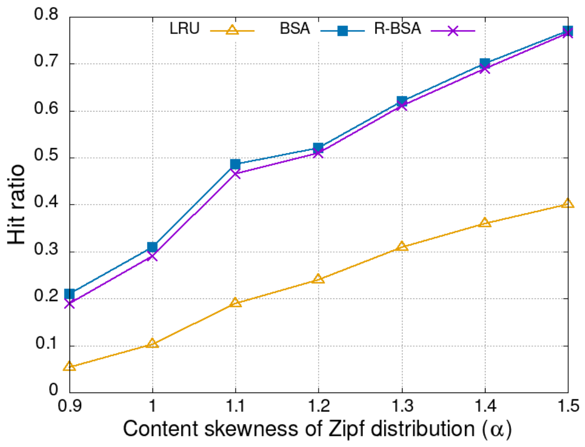


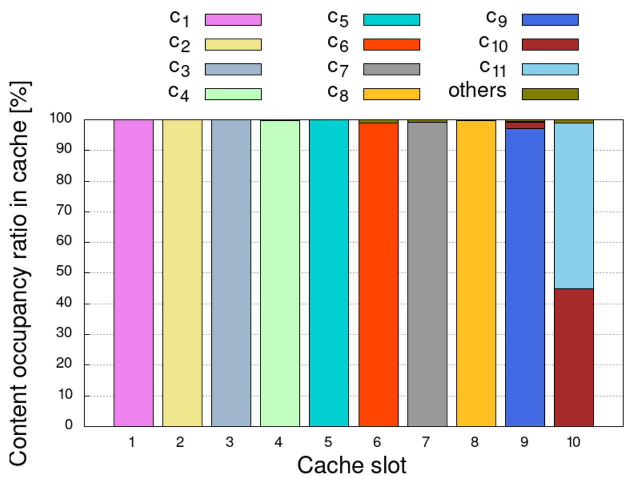
Fig. 5: Comparison of cache hit ratio for one million requests between BSA, R-BSA, and LRU cache replacement algorithm against different case of skewwness parameter (α) of Zipf distribution

is nearly zero, as shown in Figure 5. The gap of hit ratio reduction between BSA and R-BSA follows the increase of α. To understand more about this behavior, we implemented a script in our simulation application to read all the content inside the cache every t_{fetch} for further investigation.

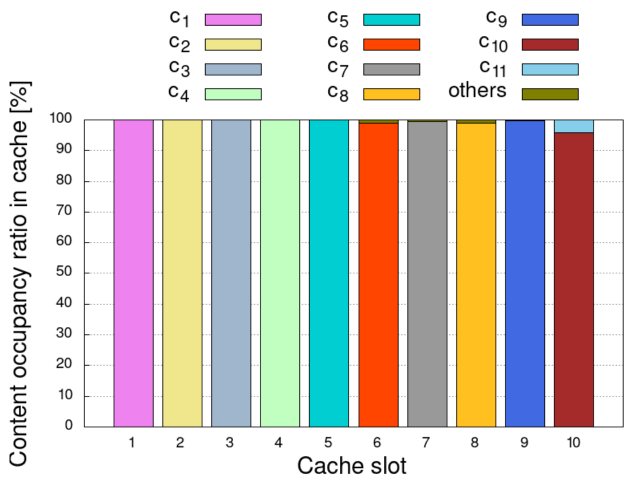
We observed the contents inside the cache for all contents of two different cache replacement algorithms, BSA and R-BSA. Furthermore, two different values of α were monitored in this investigation namely $\alpha = 0.9$ and $\alpha = 1.5$. We collected about 10,000 pieces of data from the experiment. This number was obtained from 1 million requests divided by requests obtained during t_{fetch}, which is about 100 requests. From 1st to 10,000th of t_{fetch}, we observed the cache containment for all contents. The occupancy time of contents residing in the cache was deeply analyzed.

About eleven unique contents, from c_1 to c_{11}, compete with each other to reside in the cache in two different cases, $\alpha = 0.9$ and $\alpha = 1.5$, as seen in Figure 6 and Figure 7. The cache size was enabled to store ten contents. We measured the competition among contents residing in the cache using the content occupancy metric. A higher percentage means that the content always stays inside the cache most of the time, while a lower percentage means that the content was outside the cache most of the time.

Figure 6 shows the top eleven of content ranking competition for both cache replacement algorithms in the cache when $\alpha = 0.9$. It shows that the BSA cache replacement algorithm is more sensitive to the request trend changing than R-BSA. The last slot of cache capacity is competed by several content requests. In the BSA, the last slot of cache is competed by c_{10} and c_{11} with content occupancy time of about 54% and 45%, respectively, as shown in Figure 6(a). This means that the BSA value for these two contents has dynamically



(a) BSA cache replacement algorithm

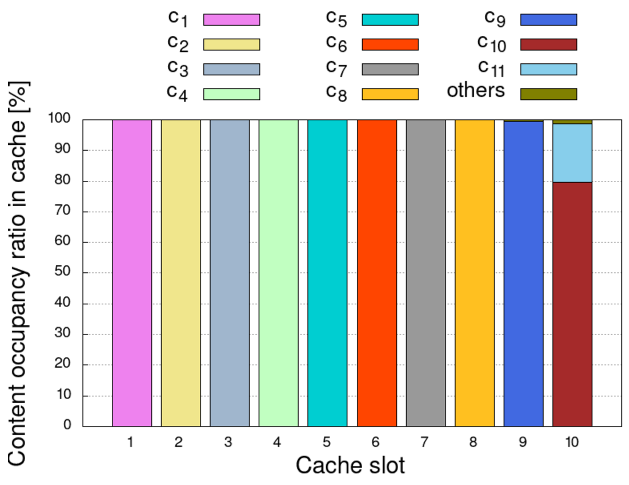


(b) R-BSA cache replacement algorithm

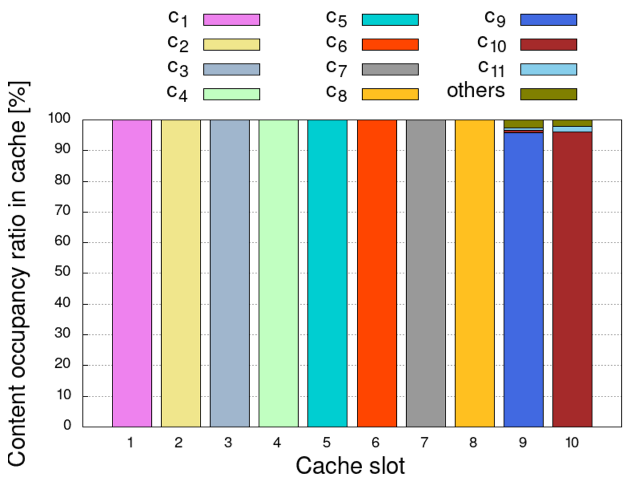
Fig. 6: Content occupancy ratio in cache in case of content request skewness (α) equal to 0.9

changed with a relatively same level, particularly in the BSA cache replacement algorithm. On the other hand, the R-BSA cache replacement algorithm has lower sensitivity to the request trend changing. it is proven by the last slot mostly occupied by c_{10} with occupancy time about 94% as shown in Figure 6(b). This situation can be interpreted that the disparity of RBSA value for c_{10} and c_{11} in the R-BSA cache replacement algorithm is higher than in the BSA algorithm, even though the request distribution of those two contents are relatively the same.

Furthermore, Figure 7 also shows the top eleven most popular content competitions between BSA and R-BSA cache replacement algorithm in case of $\alpha = 1.5$. In this case, BSA and R-BSA have relatively the same behavior regarding the last slot occupancy of the cache. The last slot of cache is competed by c_{10} and c_{11} with most of the occupation time reserved to c_{10} in about 80% for BSA as seen in Figure 7(a) and 90% for R-BSA as seen in Figure 7(b). However, the content c_{11} only occupies the last slot in about 19% and 4%



(a) BSA cache replacement algorithm



(b) R-BSA cache replacement algorith

Fig. 7: Content occupancy ratio in cache in case of content request skewness (α) equal to 1.5

for BSA and R-BSA, respectively. This can be interpreted that the higher α, i.e., in $\alpha = 1.5$, the disparity BSA level of contents for both cache replacement algorithm, BSA and R-BSA, are widened. As a result, the final position of content ranking in the database is relatively more stable and faster to achieve in both algorithms.

5. Conclusion and Future work

The R-BSA cache replacement algorithm effectively suppresses hit ratio decline in the online caching system by using a content ranking based on RBSA value. The proposed method, R-BSA, has proven to significantly reduce the number of processing loads to compute the RBSA value for every content download interval due to its detachment from the total unique contents. As a result, it can reduce the processing time by about 5% from the BSA cache replacement algorithm. Even though the R-BSA cache replacement algorithm has less sensitivity to the content's trend changing, it still can endure against the significant hit ratio decline compared to LRU and unsubstantial hit ratio drop in about less than 1% averagely compared to the BSA cache replacement algorithm.

The investigation to optimize the RBSA value in the change of request trend in every content download interval will be deeply examined in the future. Moreover, the limitation to the elapsed time, T, i.e., the reset time, will also be considered for the next work.

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 cooperated with Indonesia Endowment Fund for Education (LPDP).

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] Feri Fahrianto and Noriaki Kamiyama, "Suppressing Effect of Delayed Caching in ICN Router by Burst Score Aggregation", IEICE 2022 Society Conference, B-14-1, Online, Sep. 2022.

[3] Feri Fahrianto and Noriaki Kamiyama, "Cache replacing method using burst score aggregation to suppress delayed caching effects", 電子情報通信学会 ネットワークシステム（NS）研究会, NS2022-91, 札幌/オンライン, 2022 年 10 月

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

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

[11] A. Hoonlor et al., " An Evolution of Computer Science Research, " Communications of the ACM, 56(10), pp. 74-83, Oct. 2013.

[12] Manohar, Peter and Williams, Jalani, " Lower Bounds for Caching with Delayed Hits, " arXiv, May 2020.

[13] K. Elsayed and A. Rizk, "Time-to-Live Caching With Network Delays: Exact Analysis and Computable Approximations" in IEEE/ACM Transactions on Networking, vol. , no. 01, pp. 1-14, 5555, 2022.