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
NS2023-89(2023-10)

［依頼講演］遅延ヒットキャッシュ効果を抑制するための縮小バーストスコア集計における経過時間長の分析

フェリファリアント†　上山　憲昭††

† UIN Jakarta　〒 154–12 l. Ir. H. Djuanda No. 95
†† 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: †feri.fahrianto@uinjkt.ac.id, ††kamiaki@fc.ritsumei.ac.jp

あらまし　キャッシュは特に人気のあるコンテンツのネ可用性を向上するが，キャッシュ置換アルゴリズムはキャッシュのヒット率に大きな影響を与える．キャッシュミス時には，キャッシュはオリジンサーバからコンテンツを取得する必要があるが，その際に生じる遅延はキャッシュの性能を大幅に低下させる．本問題に対処するため，著者らは BSA (Burst Score Aggregation) 法と，その処理負荷を低減した E-BSA (Reduced BSA) を提案した．本稿では R-BSA のデータベースがリセットされる間隔が性能に与える影響を分析する．そしてそして BSA データベースが短い間隔でリセットされる場合，R-BSA はより低いコンテンツ歪度パラメータに対してより高い感度を示す傾向があり，逆に BSA データベースが長い間隔でリセットされる場合，より高いコンテンツ歪度パラメータに対する感度が高まることを明らかにする．そのため BSA データベースのリセット間隔の選択は，コンテンツの要求頻度分布に応じて適切に設定する必要がある．

キーワード　オンラインキャッシュシステム、遅延ヒットキャッシュ、LRU、BSA、R-BSA

[Invited Lecture] Analysis of Elapsed Time Length in Reduced Burst Score Aggregation for Suppressing Delayed-hit Caching Effects

Feri FAHRIANTO† and Noriaki KAMIYAMA††

† Faculty of Science and Technology, State Islamic University Syarif Hidayatullah Jakarta
Jl. Ir. H. Djuanda No. 95, Tangerang Selatan, Banten 154–12
†† College of Information Science and Engineering, Ritsumeikan University
1–1–1 Nojihigashi, Kusatsu, Shiga 525–0058
E-mail: †feri.fahrianto@uinjkt.ac.id, ††kamiaki@fc.ritsumei.ac.jp

Abstract　Caching is empirically proven to enhance network connectivity, particularly for popular content. The cache replacement algorithm is crucial in hit ratio performance within caching systems. Delayed-hit caching, introduced by some authors, significantly degrades caching performance. To address this, we propose the BSA (Burst Score Aggregation) cache replacement algorithm, which effectively mitigates hit ratio decline due to delayed-hit caching, as empirically validated. Then, we enhance our algorithm to R-BSA (Reduced BSA), which manages to reduce the processing load. This paper focuses on investigating the intervals at which the BSA (Burst Score Aggregation) database level is reset. The research findings a pivotal role played by the reset interval in influencing the behavior of the R-BSA algorithm concerning variations in content skewness parameters. Specifically, when the BSA database level is reset at shorter intervals, the R-BSA algorithm tends to display greater sensitivity to lower content skewness parameters. Conversely, opting for a longer reset interval for the BSA database level results in an increased sensitivity to higher content skewness parameters. Consequently, the choice of reset interval for the BSA database level substantially impacts the algorithm's adaptability to shifts in the popularity of requested content trends.

Key words　Online-caching system, Delayed-hit caching, LRU, BSA, R-BSA

1. Introduction

Delayed-hit caching is a phenomenon observed in online caching systems like Content Delivery Networks (CDN), Information-centric Networking (ICN), and Internet Proxies. It has been established that this phenomenon significantly reduces the hit ratio performance when conventional cache replacement algorithms such Least Recently Used (LRU) and Least Frequently Used (LFU) that cache frequently accessed content based on historical data, are employed. Due to an extreme delay from original content server, the same content that is requested previously in a cache server experienced a cache missed. To address this issue, specific anti-delayed-hit caching replacement algorithms have been developed. One such algorithm that we have introduced is named the Reduced Burst Score Aggregation (R-BSA) algorithm that is reported in references [1] and [2], which effectively mitigates the impact of delayed-hit caching.

While classic replacement algorithms can enhance the caching system's hit ratio, they are not designed for the delayed-hit caching. Consequently, conventional cache eviction algorithms fail to account for the impact of delayed hits in caching. For instance, the LRU cache replacement algorithm can experience nearly double the cache misses when content download times are twice as long with the same cache size and request pattern, as depicted in Figure 1. Furthermore, the LRU algorithm can suffer a hit ratio decline of approximately 30% due to delayed-hit caching, as we reported in [2].

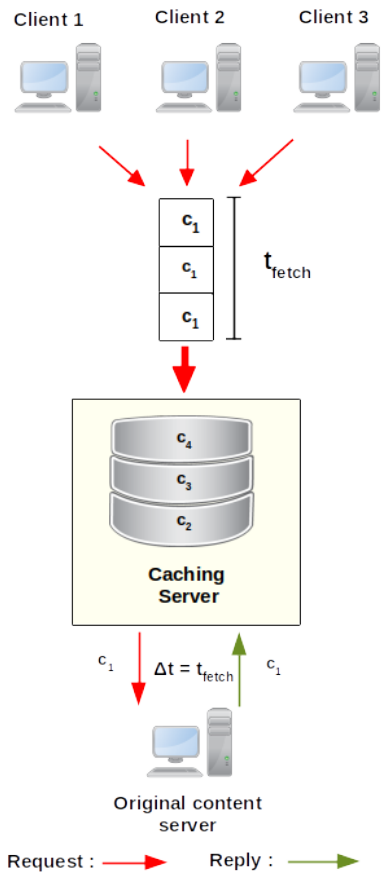

Fig. 1: Delay-hit in online caching system

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 [3]. Furthermore, we have improved it becoming the Reduced Burst Score Aggregation (R-BSA) eviction algorithm that promotes a reduction of calculation resources with a slight decrease in hit ratio as reported in [2].

Realizing that in the RBSA still has some flaws especially concerning elapses time and aggregation score database, we investigate the influence of different length of elapse time against the hit ratio performance in RBSA. A simulation is conducted to analyze the hit ratio. A numerical evaluation is used as justification in this paper. The structure of this paper consists of five sections: introduction, related work, reduced burst score aggregation, evaluation, and conclusion.

2. Related work

Published research [2] has explored the consequences of delayed caching, which needs to be taken into account when implementing a caching system. It appears that the primary issue stems from a disparity in throughput and latency between the requester and content providers. When the requester's throughput is significantly lower than that of the content provider, only a few requests may arrive during a fetching interval. Conversely, if the requester's throughput surpasses that of the provider, more requests are likely to arrive during the fetching period, leading to a higher rate of cache misses. To address this delayed caching problem, numerous researchers have been working on cache replacement strategies. Their findings discuss a cache replacement approach aimed at mitigating the impact of delayed caching to minimize any reduction in the cache hit ratio.

In the study conducted by Atre et al. [6], the authors identified the distinctive content aggregation delay associated with each individual request, along with subsequent request sequences, as fundamental parameters within the framework of cache replacement techniques. While this methodology encompasses both offline and online file caching paradigms, it is notably tailored to systems endowed with substantial buffering capabilities, enabling the temporary storage of incoming requests prior to their retrieval from the cache repository. Furthermore, it is imperative to acknowledge that the computational determination of the aggregation delay is characterized by a non-trivial complexity, necessitating extended processing times. This renders the approach less amenable to

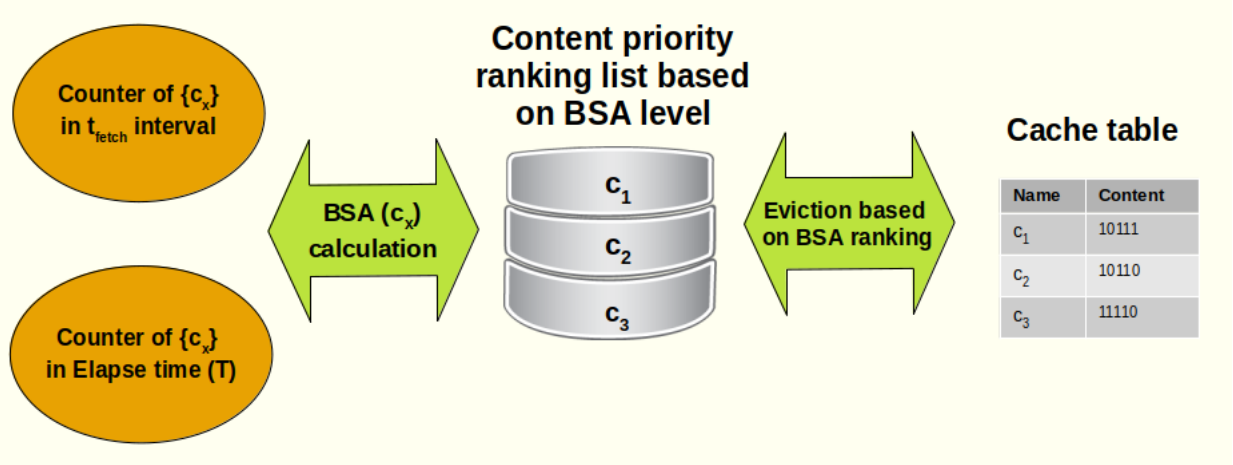

Fig. 2: Architecture of BSA cache replacement algorithm

deployment in routers characterized by high data throughput and limited memory buffers.

In a separate scholarly endeavor, Chang et al. [7] proposed an alternative strategy entailing the direct delivery of content to the source server, contingent upon a predefined delay metric, as opposed to traditional caching within the storage system. This metric is expressly bounded by predefined thresholds pertaining to delay and file size, thereby serving as a mechanism to mitigate the potential reduction in the cache hit ratio across the system.

Moreover, we proposed a replacement algorithm that considers the burstiness of contents in [3] and [4]. 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) is a simplified version of the cache replacement algorithm that prioritizes content within the cache based on the burstiness of content requests, considering it a key parameter. Within the online caching system, client requests are captured at regular intervals during content download sessions. These intervals facilitate the calculation and ranking of all content burstiness levels in the burstiness database table. The cache employs this ranking table to determine the order in which content resides in the cache. When the cache reaches full capacity, content with the lowest score is removed. The components of the R-BSA cache replacement algorithm are broadly depicted in Figure 2.

The implementation of R-BSA cache replacement algorithms is of paramount importance in evaluating the burstiness of content, a concept aptly quantified through burst scores. Burst scores are derived by comparing the occurrences of specific content items during content download time from the original content provider with their cumulative occurrences in the past. Hoonlor et al., as documented in [12], introduced a comprehensive methodology for calculating the burst score of a particular content item, denoted as x, within the content download interval, t_{fetch}, within an online-caching server. This score can be precisely formulated as follows:

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

Here, $E_{t_{fetch}}$ signifies the total number of occurrences of event $x \in c_1, .., c_N$ within the time frame of t_{fetch}, while E represents the overall count of occurrences of $x \in c_1, .., c_N$ over the entire elapsed time T, which is relative to the inter-arrival time of requests, $t_{arrival}$. The set $c_1, .., c_N$ denotes the collection of N unique contents. To optimize computational efficiency, R-BSA strategically streamlines the burst score calculations by reducing the scope from the complete set of $c_1, .., c_N$ to a more focused set, $c_1, .., c_S$. Here, S corresponds to the unique set of contents observed during an interval of t_{fetch}. Consequently, the computational burden in each t_{fetch} interval is significantly diminished, transitioning from N calculations to a mere S computations, where S typically represents a considerably smaller subset compared to N.

Burst score is difficult to compare between one unique content to another since it demands the historical occurance tracking of total unique contents 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 R-BSA as the reduced aggregation of content burstiness level. The R-BSA value of the specific content can be defined by

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

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 R-BSA value is the burstiness level of content used by the R-BSA cache replacement algorithm.

It is logic that the quantity denoted as M will inevitably experience growth over time. Consequently, M has the potential to reach exceedingly large values, rendering it impractical for real-world implementation. The value of M is constrained by the limitations imposed by hardware and software specification. This paper examine into this reality, establishing a foundation for exploring the impact of setting M to a finite value and its impact into the algorithm performance.

4. Numerical evaluation

To comprehensively assess the performance of the Reduced Burst Score Aggregation (R-BSA) algorithm, we conducted an in-depth evaluation through computer simulations. Our simulation software was developed as a multi-processed application using Python 3.8, ensuring both efficiency and accuracy in our assessments. Three process was constructed in our simulation application, namely request generator, caching server, and content provider as illustrated in Figure 3. Furthermore, we compared our R-BSA algorithm against the established LRU cache replacement algorithm.

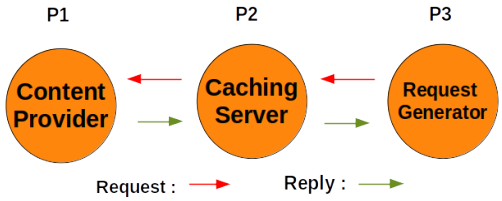

Fig. 3: Computer simulation setting

4.1 Simulation set-up

The experiments were thoroughly configured, and you can find the specific parameter settings detailed in Table 1 for hardware and software, and Table 2 for parameter set-up. In addition, we collected data from an extensive dataset of one million requests, systematically generated by the client. This voluminous dataset served as the foundation for our thorough investigation.

Tab. 1: Hardware and software specification

Hardware/Software	Specification
Processor	Core-i5
RAM	8 GB
Hard Drive	512 GB
Host OS	Ubuntu 20.04
Programming Software	Python 3.8

To justify the effectiveness of R-BSA across a spectrum of scenarios, we charted the cache hit ratio and the simulation running time against varying degrees of request skewness (α), conforming to the Zipf distribution. Moreover, we compare the hit ratio with LRU cache replacement algorithm. It is noteworthy that our analysis focused on α values greater than or equal to 0.7. This choice was rooted in prior research findings, as documented in [3], which indicate that the suppression effect of delayed-hit caching becomes notably influential starting at this specific threshold. Consequently, by examining R-BSA under these conditions, we aimed to provide a comprehensive understanding of its performance in realistic scenarios.

Tab. 2: Simulation parameter setting

Paramater	Value
Total unique content items (N)	1000
Cache size	10
Request skewness of Zipf distribution (α)	0.7 - 1.5
Request rate	10,000 requests/second
Total requests	1,000,000 requests
t_{fetch}	10 ms
$t_{arrival}$	0.1 ms
M	10, 100, 1000, 10000

4.2 Performance of hit ratio

We observed the performance of the hit ratio in three different cases. First, we set the value of M equal to 10. This means that the database and the counter of elapse time are reset every multiple of 10 from content download time, t_{fetch}. Then, we set the value of M equal to 100, which demands the BSA database to be set to zero for every multiple of 100 from content download time. Finally, the value of M is configured into 1000, so the BSA database value is reset in multiple of 1000 from content download time. We obtained the data from one million requests sent by the client generator request. β, the total hit ratio and $\Delta\beta$, the gap between the hit ratio obtained by the LRU and R-BSA cache replacement algorithm, are plotted against the content request skewness, α, of Zipf distribution.

The performance of the R-BSA hit ratio when M equals 10 is quite promising in α lower than 0.9. The hit ratio is about doubled from the hit ratio of LRU in intervals of α equal to 0.7 and 0.9. It indicates that R-BSA with M equal to 10 has outperformed LRU in lower content skewness. In the higher α, R-BSA also significantly surpass the LRU with the average of $\Delta\beta$ at about 25% higher. Generally, The gap of hit ratio reduction, $\Delta\beta$, R-BSA and LRU follows the increase of α as seen in Figure 4. From this simulation result, it is shown that the small value of M has a higher response to small content skewness, α, because of the fast reset in the BSA database.

Figure 5 shows the result of the simulation R-BSA cache replacement algorithm when M is equal to 100. When the

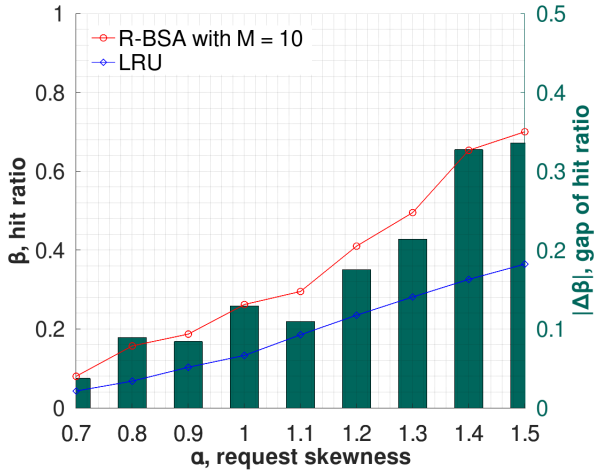

Fig. 4: Comparison of cache hit ratio for one million requests between R-BSA with M=10 and LRU cache replacement algorithm against different case of skewwness parameter (α) of Zipf distribution

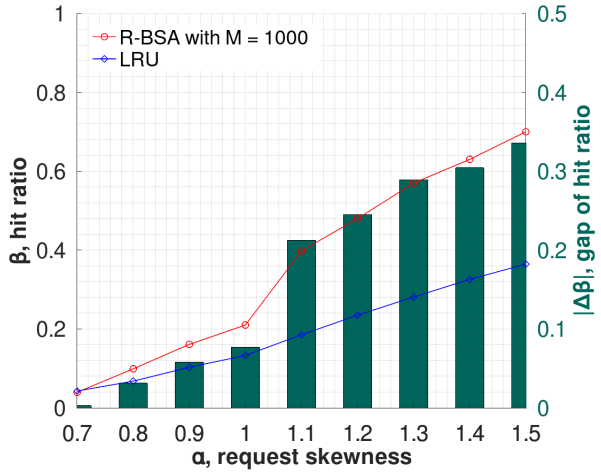

Fig. 6: Comparison of cache hit ratio for one million requests between R-BSA with M=1000 and LRU cache replacement algorithm against different case of skewwness parameter (α) of Zipf distribution

BSA database was cleared every 100 times of content download time, t_{fetch}, there was a significant drop of hit ratio, about 15%, in the lower content skewness, α, i.e., from α equal to 0.7 up to α equal to 0.9 compared to previous simulation scenario when the value of M is equal to 10. However, The higher α, i.e., from α equal to 1.0 up to α equal to 1.3, experienced a slight increase of about 20%. The increase of M improved the hit ratio performance in the middle range of α, but it shrinkage the performance of the hit ratio in the lower range of α.

We increased the value of M from 100 to 1000. Therefore, the BSA database level was cleared every 1000 of content download time, t_{fetch}. Figure 6 depicts the simulation result of hit ratio performance. The result shows that a deep correction occurred in the lower α. The gap between the hit ratio of LRU and R-BSA decreased in the range of α between 0.7 to 1.0, about 5% from the scenario when M equal to 100 and 30% averagely from the scenario when M equal to 10. On the other hand, the higher α, i.e., from α equal to 1.1 up to 1.5, the gap of hit ratio performance was widened between R-BSA and LRU with about 10% averagely from the scenario where M equal to 100.

The last computer simulation scenario where M equal to 10000 was conducted. Figure 7 showed the result of the simulation R-BSA cache replacement algorithm when the BSA database was cleared every 10000 times of content download time, t_{fetch}. Since we ran the computer simulation for one million requests, the BSA-level database was never actually

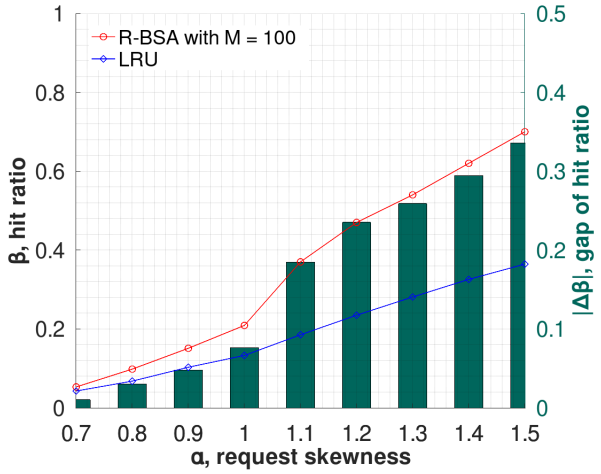

Fig. 5: Comparison of cache hit ratio for one million requests between R-BSA with M=100 and LRU cache replacement algorithm against different case of skewwness parameter (α) of Zipf distribution

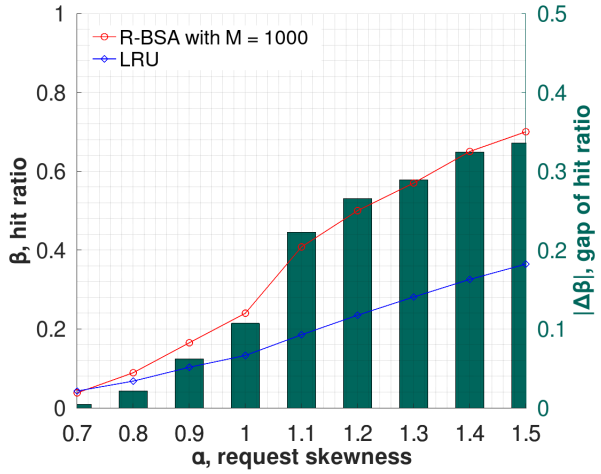

Fig. 7: Comparison of cache hit ratio for one million requests between R-BSA with M=10000 and LRU cache replacement algorithm against different case of skewwness parameter (α) of Zipf distribution

reset when the simulation was running for this scenario. The simulation showed that the gap of hit ratio between R-BSA and LRU, $\Delta\ \beta$, continued to widen, especially in the interval of α from 1.1 to 1.5. In general, there was a hit ratio improvement in the higher value of α compared to the previous simulation scenario when the value of M equals 1000. However, The higher α, i.e., from α equal to 1.1 up to α equal to 1.3, experienced a slight increase of about 20%.

Overall, the simulation result shows that the lower M has higher sensitivity against requests with lower content skewness, α. On the other hand, the higher M has a higher response to a middle and high range of α. In the lower α, the content requests are more varied within a content download time, t_{fetch}. Thus, the difference in BSA levels among specific content is negligible. When the database of the BSA level is reset every short interval, the content ranking position changes dynamically. However, in the higher α that the content requests more homogenous and bursty within content download time, t_{fetch}. Therefore, even though the BSA database is reset every short or long interval, it does not influence the ranking position of the BSA level.

5. Conclusion and Future work

5.1 Conclusion

The R-BSA cache replacement algorithm is an effective strategy employed within online caching systems to mitigate the decline in hit ratios. It achieves this by employing a content ranking approach based on the BSA value. This innovative technique, known as the R-BSA algorithm, has demonstrated its ability to notably reduce the computational workload associated with determining the BSA value for each content download interval. Moreover, the elapsed time in the R-BSA cache replacement algorithm must follow the implementation in the real world. Therefore, the database of the BSA level must be clear at every specific interval due to hardware or software constraints. The research findings indicate the interval duration of resetting the BSA database level has an impact on the R-BSA algorithm's sensitivity to variations in content skewness parameters. In cases where the BSA database level is reset at shorter intervals, the R-BSA algorithm tends to exhibit higher sensitivity to lower content skewness parameters. Conversely, when the reset interval for the BSA database level is longer, it results in greater sensitivity to higher content skewness parameters. Consequently, the choice of reset interval for the BSA database level significantly influences the algorithm's responsiveness to changes in the popularity of requested content trends.

5.2 Future work

The investigation to optimize the interval of BSA level database reset in the change of request trend will be deeply examined. Moreover, the sudden or extreme change of content requests against the different intervals of BSA database reset will be studied in the future.

Acknowledgement

This work was supported by JSPS KAKENHI Grant Number 21H03437 as well as State Islamic University Syarif Hidayatullah Jakarta.

References

[1] F. Fahrianto and N. Kamiyama, "Reduced Burst Score Aggregation in Suppressing Delayed-Hit Caching Effects", IEICE Tech. Rep., vol. 122, no. 406, NS2022-168, pp. 7-12, March 2023.

[2] 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.

[3] 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.

[4] 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

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

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

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

[14] 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.

[15] P. Manohar and J. Williams, "Lower Bounds for Caching with Delayed Hits," in CoRR, vol. abs/2006.00376, 2020, doi: 10.48550/arXiv.2006.00376 .