IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2022.0122113

Analysis of Similarity Caching on General Cache Networks

RYO NAKAMURA[1], (Member, IEEE) and NORIAKI KAMIYAMA[2], (Member, IEEE)
[1]Faculty of Engineering, Fukuoka University, 8–19–1 Nanakuma, Jonan, Fukuoka, Fukuoka, Japan (e-mail: r-nakamura@fukuoka-u.ac.jp)
[2]College of Information Science and Engineering, Ritsumeikan University, 2–150 Iwakura, Ibaraki, Osaka, Japan (e-mail: kamiaki@fc.ritsumei.ac.jp)

Corresponding author: Ryo Nakamura (e-mail: r-nakamura@fukuoka-u.ac.jp).

This work was partly supported by JSPS KAKENHI Grant Number 23K21664 and 23K21665.

ABSTRACT Nowadays, content caching is essential to serve contents to users quickly and efficiently. In recent years, a technique to improve content caching, *similarity caching*, which focuses on the similarity among contents, has emerged. The feature of similarity caching is that a cache provides any content similar to a user's request even though the cache does not have the request itself, i.e., cache miss. In the literature, the effectiveness of similarity caching is investigated mainly through system-level metrics such as the cache hit ratio. However, its effectiveness from the point of view of users remains hardly understood despite its importance. Therefore, we try to quantify the user-level performance of similarity caching while focusing on a general cache network comprised of multiple cache nodes. To this aim, we introduce a user-oriented metric called *similar-content delivery delay* and analytically derive that; we also analyze the relation between the delivery delay observed by a user and the utility of the content similarity through several numerical evaluation. Our key findings are summarized as follows; (i) the effectiveness of similarity caching is dominated by the similarity which is acceptable to users; (ii) the effectiveness is also dependent on the location of users, i.e., distance to a contents server.

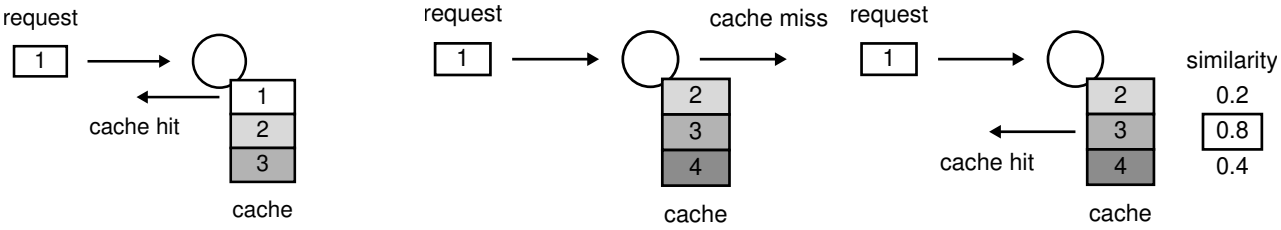
INDEX TERMS Content distribution networks, information-centric networking, mathematical analysis, quality of service.

I. INTRODUCTION

WITH the continuous growth of content space, a way to flexibly resolve user's content requests, *similarity searching* [1], is expected to be one of promising approaches. Similarity searching is a technique to, from the large-scale content space, find contents that have similarity which is defined according to a given criteria. Its effectiveness stems from the fact that if a user can retrieve similar content instead of what the user actually wants, the user can gain a certain level of satisfaction.

At the same time, to efficiently deliver content to users, we have devoted significant effort to design cache networks which utilize *content caching* [2]–[5]. Prominent architectures of the cache network are CDN (Content Delivery Network) [6], MEC (Mobile Edge Computing), and CCN (Content-Centric Networking) [7]/NDN (Named Data Networking) [8]. As we know, CDN distributes content from cache servers scattered around the world. MEC utilizes caches located at edge of a network, e.g., access points [9]; thus, users can acquire content from nearby caches. In contrast, CCN/NDN is a cache network comprised of routers equipped with a cache, which enables us to retrieve content from everywhere [10]. Although CCN/NDN is a future network architecture, effort has been made (see, for instance, [11]–[15]). The benefit gained by these cache networks are, for instance, to reduce traffic transferred through a network and to shorten the content delivery delay.

Alongside such developments, a caching technique which applies the concept of similarity searching to content caching, *similarity caching* [16], [17], has been attracting attention in recent years [18]–[23]. The fundamental difference of similarity caching from conventional content caching called *exact caching* is that a cache can provide content similar to a requested one. To explain this difference more specifically, we briefly describe the mechanisms of exact caching and similarity caching (see Fig. 1). In exact caching, when a request from a user arrives at a cache node, i.e., a node with a cache, this cache node performs as follows; (i) it checks whether to store the requested content or not; (ii) if so, i.e., cache hit, it immediately returns the content; otherwise, i.e., cache miss, it forwards the request to other nodes, e.g., a node which stores the original content. In contrast, similarity caching differs



(a) cache hit with exact caching (b) cache miss (c) cache hit with similarity caching

FIGURE 1. Overview of conventional exact caching and similarity caching. Figures 1(a) and 1(b) illustrate cache hit and cache miss with exact caching, respectively; namely, when a request to content 1 arrives at a cache node, the node immediately returns the content 1 if caching it. Otherwise, the node forwards the content request to other nodes. In contrast to exact caching, in the case of similarity caching as shown in Fig. 1(c), the node can return similar content, the content 3 with maximum similarity, instead of the content 1.

from exact caching in terms of the operation at cache miss; namely, when the cache node does not store a content exactly-matching the user's request, it looks up similar content within its own cache and responds with that instead of discovering the exact content. A representative application of similarity caching is, for instance, video streaming [24]. In this context, videos with different bit rates can be thought of as similar contents. Hence, in a scene where a user requests video content, similarity caching allows a cache node to server a video with a bit rate different from what the user actually wants, as similar content. As a result, the user can more quickly start watching it.

Similarity caching offers flexibility for content caching, but it involves a *trade-off* between the communication quality observed by users and the content similarity. For a user who values the communication quality, it is preferable that similar content is quickly delivered from a nearby cache even if this content differs from what the user wants. On the other hand, for a user who prefers the exactness of content, it is attractive that the requested content itself is returned from a node such as an origin server rather than a nearby cache, even if it sacrifices the communication quality.

Therefore, the purpose of this paper is theoretical understanding of similarity caching, in particular, to clarify the relationship between communication quality and content similarity while focusing on a general cache network. From our viewpoint, the situation in which such a quantitative evaluation of similarity caching has hardly performed comes up in the complexity of cache networks. In fact, the communication quality and content similarity as observed by the user are affected by various factors, for instance, which caches in the network serve content and what similar content they serve. This motivates us to reveal this complicated interaction on the general network with similarity caching.

To this end, we introduce a user-oriented metric for similarity caching, *similar-content delivery delay* and analytically derive that on a general cache network. The definition of similar-content delivery delay is the time taken from when a user sends a content request to a network until when it initially retrieves any content satisfying a given threshold. Here, the threshold is corresponding to the minimum of content similarity that the user can accept. Through several numerical examples, we also analyze the property of similarity caching in a multi-stage cache network from the perspective of the similar-content delivery delay.

Among several applications of similarity caching such as recommender system and contextual advertising [18], our paper focuses on multi-media retrieval, especially, query-based retrieval [16]. Namely, a user retrieves content by issuing a query. In this scenario, similarity caching becomes effective when a nearby cache stores similar content, even if it does not store the exact requested content, and when the user is satisfied with similar content provided in a short delay. This effectiveness arises from cases where, if a user requests a particular song, they might be satisfied with an arranged version of the song, as similar content, instead of the original.

The contributions of the present paper is summarized as follows.

- We introduce the similar-content delivery delay as a user-oriented metric for similarity caching and algebraically derive it on a general cache network.
- We provide an analytical framework for similarity caching. This enables us to optimize control parameters of similarity caching. For instance, our analysis can derive the similarity acceptable to a user if the user's allowable delay is given.
- We reveal the effectiveness of similarity caching from the perspective of similarity-content delivery delay. As a consequence, a summary our insights is that (i) the effectiveness of similarity caching is dominated by several factors, in particular, the similarity which is acceptable to users, and (ii) the location of users, i.e., distance to contents server, is also important factor.

The remainder of this paper is organized as follows. Section II introduces previous works related to this paper. Section III describes our analytic model used in this paper and explains the derivation of similar-content delivery delay. Section IV analyzes the property of similarity caching in a multi-stage cache network through numerical evaluation. Finally, Section V presents the summary of this paper and addresses future works.

TABLE 1. Comparison between related work and present paper.

	contribution	target system	performance metric
[16]	proposal of algorithm for similarity caching	single cache	cache hit ratio / similarity
[17]	proposal of algorithm for similarity caching	single cache	cache hit ratio / utility
[18]	mathematical formulation of similarity caching proposal of algorithms for similarity caching	single cache	cost formulated by cache hit ratio and similarity
[19]	optimization of content placement	single cache	cost formulated by cache hit ratio and similarity
[20]	optimization of content placement	single cache	caching gain
[21]	joint optimization of content placement and delivery path	cache network	weighted sum of delivery delay and similarity
[22]	optimization of content placement	cache network	cost formulated by cache hit ratio and similarity
present paper	theoretical understanding of similarity caching	cache network	s-content delivery delay

II. RELATED WORK

In this section, we introduce previous studies related to similarity caching from two perspectives: a single cache and general cache networks. Then, we describe the position of the present paper against these studies. A summary of comparison between previous studies and this paper is shown in Tab. 1.

A. SIMILARITY CACHING FOR A SINGLE CACHE

The pioneering works in the field of similarity caching are, to the best of our knowledge, Refs. [16], [17]. Both of these studies proposed the concept of similarity caching and designed cache replacement algorithms for it. Specifically, Ref. [16] applied similarity caching to a query-based content retrieval system. Also, Ref. [16] proposed a cache replacement algorithm based on the k-nearest neighbor and demonstrated its effectiveness through experiments using a large-scale dataset of photographs. Similarly, Ref. [17] introduced similarity caching to a content-match system on web applications which presents advertisements suitable to a user's visiting web page. This study also proposed cache replacement algorithms for similarity caching which are based on two typical algorithms, Least-Recently Used (LRU) and Least-Frequently Used (LFU). Although systems and cache replacement algorithms addressed in these studies are different, the commonality among them is to apply the concept of similarity into contents caching.

Recently, similarity caching has been attracted attention by Ref. [18] again. This study focused on similarity caching systems and tried to optimize contents placement on a cache. To this aim, this study formulated an optimization problem so as to minimize cost, which is formulated as an objective function, incurred on cache hit/miss. This optimization problem is proved to be NP-hard, so this study proposed heuristic algorithms called q LRU-ΔC and DUEL as practical solutions to this problem, and demonstrated these effectiveness through mathematical analysis and experiments.

Ref. [18] gives us understanding mathematical property of similarity caching and activates design of cache replacement algorithms for similarity caching as shown in Refs. [19], [20]. The overview of proposed algorithms in Refs. [19], [20] is as follows. The algorithm of Ref. [19] is to gradually update a cache state such that it caches contents that can serve as many different types of requests as possible; in contrast, that of Ref. [20] is to update a cache state in an online manner so as to minimize the cost formulated in Ref. [18]. These algorithms can succeed in achieving better performance than the conventional one proposed in Refs. [16], [17].

B. SIMILARITY CACHING FOR GENERAL CACHE NETWORK

A few studies [21], [22] have already tried to expand similarity caching into general cache networks comprised of multiple nodes with a cache. The focus of these studies is to find an optimal content placement on a cache network [21], [22]. Reference [21] proposed an algorithm for finding content placement so as to minimize a user-level metric formulated as the weighted sum of content delivery delay and content dissimilarity. But, a limitation of this algorithm is scalability in terms of the number of contents; in other words, this algorithm cannot be applicable to a large-scale content space [22]. To resolve this issue of scalability, Ref. [22] assumed not only discrete but also continuous content space. Furthermore, this study proposed two types of solutions according to network topology: an optimal solver for tree networks and a heuristic solver for arbitrary networks.

C. POSITION OF PRESENT PAPER

The differences between aforementioned studies and this study can be summarized as follows.

- This paper focuses on not a single cache, but rather a network of multiple caches.
  The motivation behind our focus stems from a situation where property of similarity caching in general cache networks remain hardly understood. As described in Section II, the focus of most previous studies is the design of caching strategy for a single cache. We believe that, in the future, similarity caching will be deployed on cache networks and that similarity caching has a favorable property to cache networks such as CCN [7]/NDN [8]. From this viewpoint, we think that it is important to understand property of similarity caching in cache networks. It is worth of noting that similarity caching cannot be directly applied to CCN/NDN which resolves a request based on exact matching of content identifier, but it can be applied to network architectures taking account of the similarity of contents, e.g., [25]–[27].
- On analyzing property of similarity caching, this paper

takes account of user's behavior: evaluation of the content similarity and retransmission of content requests.

The fundamental difference from previous studies is to newly introduce a user-oriented performance metric called similar-content delivery delay. In most of previous studies, the effectiveness of similarity caching is measured mainly through the cache hit ratio. In fact, the cache hit ratio is one of typical metrics for evaluating cache networks. Although the cache hit ratio is one of useful metrics, from the viewpoint of designing communication networks, metrics directly-related to QoS (Quality of Service) and QoE (Quality of Experience) observed by users, e.g., latency and throughput, are also important.

In a sense, the most relevant to this paper is Ref. [21] which used a user-oriented performance metric considering both of the content delivery delay and the similarity rather than the cache hit ratio. In this study, however, we assume a user's behavior to evaluate the similarity between what the user wants and what the user actually retrieves and to verify that this similarity suffices the user's demand. Moreover, we express that if the user is not satisfied, the user repeats content requests.

As a consequence of this consideration, we define the similar-content delivery delay as a performance metric including such a user's behavior, and quantitatively analyze, in terms of the similar-content delivery delay, property of similarity caching.

III. ANALYSIS

In this section, we introduce an analytic model that describes similarity caching on a general cache network, and then explain our definition of the similar-content delivery delay and its derivation. Table 2 summarizes mathematical symbols and these definitions used throughout this paper.

A. ANALYTIC MODEL

We focus on a cache network composed of nodes with a finite cache, i.e., *cache nodes*, and nodes holding original contents, i.e., *origin nodes*. Also, we represent the cache network as an undirected graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ represent a set of nodes and a set of links, respectively. We should note that a set of nodes $\mathcal{V}$ is a union of the set of cache nodes $\mathcal{V}_c \subset \mathcal{V}$ and the set of origin nodes $\mathcal{V}_o \subset \mathcal{V}$ and that cache nodes $\mathcal{V}_c$ and origin nodes $\mathcal{V}_o$ do not overlap; in other words, $\mathcal{V}_c \cup \mathcal{V}_o = \mathcal{V}$ and $\mathcal{V}_c \cap \mathcal{V}_o = \phi$ hold. We denote propagation delay of a link between nodes u and v as $\tau_{u,v}$. We suppose that bandwidth of links is sufficiently large; hence, we have two assumptions: (i) request/content loss within the network never occur and (ii) queuing delay at nodes can be negligible.

In this paper, we consider content delivery based on the request-and-response communication model employed in CCN [7]/NDN [8]. More specifically, the content delivery is realized in a following manner; a user issues a content request; the request is repeatedly transferred among nodes within a network until it arrives at a node holding the requested content; then, the node returns it as a response to the requester.

TABLE 2. Definition of symbols.

Cache network	
$G = (\mathcal{V}, \mathcal{E})$	undirected graph representing cache network
$\mathcal{V}_c$	set of cache nodes
$\mathcal{V}_o$	set of origin nodes
v_c	origin node storing content c
$\tau_{u,v}$	propagation delay of link between nodes u and v
$\mathbf{P}_{v,c}$	path from node v to origin node storing content c
Content	
C	number of contents
$\mathcal{C}$	set of contents $\{1, \ldots, C\}$
$s(c, c')$	similarity between two arbitrary contents c and c'
$\mathcal{C}_c^{s(<)}$	set of contents whose similarity to content c is smaller than s
Cache node	
B	cache size of cache nodes
α	probability whether cache node serves similar content or not
Client	
$D_{v,c}$	similar-content delivery delay observed by client v requesting content c
$D_{v,c}^{s(\geq)}$	s-content delivery delay observed by client v requesting content c
λ_c	request ratio of content c
δ	retransmission interval

We suppose that a content request is transferred along a predetermined path and that the content as a response is transferred by tracing back a path that its request is forwarded. This assumption is traditional in the area of cache networks such as CCN/NDN. We denote a path from a node v to a node $v_c\,(\in \mathcal{V}_o)$ storing an original content c as $\mathbf{P}_{v,c} = (v, \ldots, v_c)$. Hence, $\mathbf{P}_{v,c}[i]$ means the i-th node on the path $\mathbf{P}_{v,c}$. From the definition, $\mathbf{P}_{v,c}[1] = v$ and $\mathbf{P}_{v,c}[|\mathbf{P}_{v,c}|] = v_c$.

We denote a set of contents as $\mathcal{C} = \{1, \ldots, C\}$, where C is the number of contents. Following Refs. [18], [28], [29], we assume that the size of contents is equal. Also, we define the similarity among contents as $s\,(0 \leq s \leq 1)$ and denote the similarity between arbitrary two contents as $s(c, c')$. From definition, $s(c, c) = 1$. Using the similarity, we denote a subset of contents whose similarity to content c is smaller than s as $\mathcal{C}_c^{s(<)}$. Formally, the set $\mathcal{C}_c^{s(<)}$ is defined as $\{c' \,|\, s(c, c') < s, \forall c' \in \mathcal{C}\}$. In the same way, we have symbols $\mathcal{C}_c^{s(\leq)}$, $\mathcal{C}_c^{s(>)}$, and $\mathcal{C}_c^{s(\geq)}$.

A cache node has a finite cache of size B [content] and it can cache relaying contents. If the cache is completely occupied with other contents, the cache node evicts any content from its own cache according to a cache replacement algorithm and then newly inserts the relaying content.

When a content request arrives at a cache node, the node decides whether to return cached content with a probability $\alpha\,(0 \leq \alpha \leq 1)$. More specifically, with the probability α, it returns the most similar content among contents in its own cache; formally speaking, the cache node receiving a request to content c provides content c' satisfying $\arg\max_{c' \in \mathcal{B}} s(c, c')$, where $\mathcal{B} \subseteq \mathcal{C}$ is a set of cached contents. In contrast, with the probability $1-\alpha$, the cache node forwards the content request to one of adjacent nodes instead of providing similar content.

A client, i.e., requester, is connected to a cache node and

issues a content request. The request ratio to content c is denoted as λ_c.

More specifically, the client's request model assumed in this paper is as follows; (i) the client requests content c; (ii) the client retrieves any content and evaluates its similarity; (iii) if the client is not satisfied with the content, after a fixed interval, it re-issues a request to the content c; (iv) steps (i) $\sim$ (iii) is repeated until the client obtains content sufficing the client. Hereinafter, we denote the aforementioned fixed interval, i.e., the time interval between when the client receives content and when it re-sends a content request, as δ.

Below, we mention the rationale behind employing this request model because it is not general to take account of the client's retransmission behavior in the field of modeling the cache network. The rationale derives from a fact that, in similarity caching, it is not guaranteed that a client always retrieves exact content that the client requests, although this is natural in exact caching[1]. This is because the similar content provided by cache nodes may not be sufficient for the client. Moreover, we think that, if the client fails in obtaining contents satisfying the client, it retries to retrieve contents. As a consequence of this consideration, we introduce the retransmission behavior to the client's request model.

B. DEFINITION OF SIMILAR-CONTENT DELIVERY DELAY

In the following, we explain the definition of the *similar-content delivery delay* focused in this paper. Here, we focus on the similar-content delivery delay $D_{v,c}$ observed by a client v requesting content c. Hereinafter, we use the term "client v" to refer to a client connected to node v.

To define the similar-content delivery delay, we introduce *s-content delivery delay* $D_{v,c}^{s(\geq)}$, that is, the time from when a client v initially issues a request to content c until it obtains the content whose similarity is larger than or equal to a given similarity s. Because, as mentioned in Section III-A, a client repeats content requests until it obtains the content satisfying the similarity s, the s-content delivery delay consists of twofold: the content delivery delay — the time between when the client sends a request and when it receives content as a response to that request — and the fixed time interval between sending content requests (see Fig. 2). We will give the derivation of the s-content delivery delay later.

We formulate the similar-content delivery delay $D_{v,c}$ as the weighted average of the s-content delivery delay $D_{v,c}^{s(\geq)}$ and a parameter $\beta_v^{(s)}$ representing the client's acceptable degree of similarity:

$$D_{v,c} = \sum_s \beta_v^{(s)} D_{v,c}^{s(\geq)}. \tag{1}$$

We assume that $\beta_v^{(s)}$ is given by an arbitrary probability mass function. From the normalized condition, $\sum_s \beta_v^{(s)} = 1$. It is worth of noting that the similar-content delivery delay with a simple binary function $\beta_v^{(1)} = 1$ corresponds to that in the case of conventional exact caching.

[1]Of course, this establishes only when such a content exists in a network.

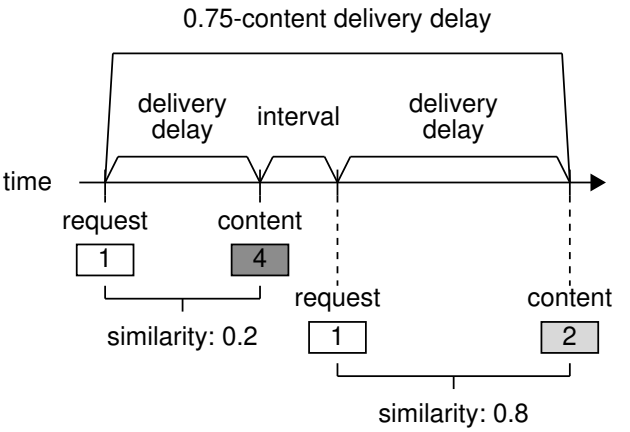


FIGURE 2. Example of s-content delivery delay with $s = 0.75$. In this example, a request to content 1 is sent and, as a response, similar content 4 is provided, but the content 4 does not satisfy the similarity. Hence, after the interval, the request to the content 1 is re-issued. As a result, similar content 2 whose similarity is larger than or equal to 0.75 is returned. In this case, the s-content delivery delay with $s = 0.75$ is given by the sum of two delivery delays and the one fixed interval.

C. DERIVATION OF S-CONTENT DELIVERY DELAY

As a preliminary to deriving the s-content delivery delay, we begin with obtaining a probability $p_c^{s(\geq)}$ that, when a request to content $c \in \mathcal{C}$ arrives at a cache node, it returns a content whose similarity is larger than or equal to s. Letting a probability that the cache node returns content with similarity s be $p_c^{(s)}$, the probability $p_c^{s(\geq)}$ is the sum of probabilities $p_c^{(s)}$ over a set of similarities $\mathcal{S}_c^{s(\geq)} = \{s(c,c') \,|\, s(c,c') \geq s, \forall c' \in \mathcal{C}\}$:

$$p_c^{s(\geq)} = \sum_{s \in \mathcal{S}_c^{s(\geq)}} p_c^{(s)}. \tag{2}$$

We proceed to derive the probability $p_c^{(s)}$ in the above equation. Recall that a cache node in our model returns the most similar content from its own cache. Based on this assumption, a phenomenon that the content with similarity s is chosen occurs only when the following two conditions establish simultaneously: if content with similarity s exists in the cache space and if the remaining cache space except for that content is fulfilled with contents only with similarity less than s. This can be expressed by a difference between a combination of the cache space comprised of contents whose similarity to content c is less than or equal to s, i.e., $\mathcal{C}_c^{s(\leq)}$ and that of contents only with similarity less than s, i.e., $\mathcal{C}_c^{s(<)}$. Hence, we have

$$p_c^{(s)} = \frac{\binom{|\mathcal{C}_c^{s(\leq)}|}{B} - \binom{|\mathcal{C}_c^{s(<)}|}{B}}{\binom{C}{B}}. \tag{3}$$

We should mention that the above equation holds only when the content popularity, i.e., request ratio for contents, is uncorrelated with the similarity among contents.

In the following, we focus on a client v requesting content c and derive its s-content delivery delay. Unless explicitly stated, we simplify some notations by omitting the notation v for a node and the notation c for content; for instance, we simplify the notation $D_{v,c}^{s(\geq)}$ to $D^{s(\geq)}$.

We can formulate the s-content delivery delay $D^{s(\geq)}$:

$$D^{s(\geq)} = \sum_{n=1}^{\infty} \Delta_n^{s(\geq)} R_n^{s(\geq)}. \tag{4}$$

In the above equation, $\Delta_n^{s(\geq)}$ is the time from when the client initially issues a request to when it retrieves content whose similarity is larger than or equal to s, i.e., $c' \in \mathcal{C}^{s(\geq)}$, at n-th request. Also, $R_n^{s(\geq)}$ is a probability that the client successfully retrieves that content at the n-th request.

To derive $\Delta_n^{s(\geq)}$, we consider two types of content delivery delays according to the similarity. We denote the delivery delay of content whose similarity is larger than or equal to s as $d^{s(\geq)}$ and that of content whose similarity is smaller than s as $d^{s(<)}$. Using these delivery delays and the retransmission interval δ, $\Delta_n^{s(\geq)}$ is given by

$$\Delta_n^{s(\geq)} = (n-1)\left(d^{s(<)} + \delta\right) + d^{s(\geq)}. \tag{5}$$

The first term in the above equation means the time from the client issues an initial request just before it issues the n-th request. Below, among two types of content delivery delays, i.e., $d^{s(\geq)}$ and $d^{s(<)}$, we explain only the derivation of the content delivery delay $d^{s(\geq)}$. This is because $d^{s(<)}$ can be obtained by replacing $\mathcal{S}_c^{s(\geq)}$ in Eq. (2) with the set $\mathcal{S}_c^{s(<)} = \{s(c,c') \,|\, s(c,c') < s, \forall c' \in \mathcal{C}\}$ in the process of the derivation of $d^{s(\geq)}$.

The content delivery delay is an expected value of the round-trip propagation delay between a client and a node serving the corresponding content. More specifically, by letting $\eta_i^{s(\geq)}$ be a probability that the i-th node serves content whose similarity is larger than or equal to s, we have

$$d^{s(\geq)} = \sum_{i=2}^{|\mathbf{P}|} \left(2\tau_{\mathbf{P}[i-1],\mathbf{P}[i]} \times \eta_i^{s(\geq)} \right). \tag{6}$$

In the above equation, $|\mathbf{P}|$ indicates the number of nodes composing the path $\mathbf{P}$ from the node v to the origin node v_c for content c.

Let us proceed with deriving the probability $\eta_i^{s(\geq)}$. As a preliminary, we derive a probability that a node v', i.e., a cache node or an origin node, provides content whose similarity is larger than or equal to s, and denote it as $q_{v'}^{s(\geq)}$. The probability $q_{v'}^{s(\geq)}$ depends on the node type: cache node or origin node. In the case of the cache node, this phenomenon occurs only when the cache node decides on responding with probability α and when it caches any content $c' \in \mathcal{C}^{s(\geq)}$; in contrast, the origin node always, i.e., with the probability of one, returns the original content. Hence, we have

$$q_{v'}^{s(\geq)} = \begin{cases} \alpha\, p^{s(\geq)} & \text{if } v' \in \mathcal{V}_c \\ 1 & \text{otherwise} \end{cases}. \tag{7}$$

Recall that $p^{s(\geq)}$ is the simplified notation of $p_c^{s(\geq)}$ (Eq. (2)). The probability $\eta_i^{s(\geq)}$ is the product of the probability that the i-th node on path $\mathbf{P}$ provides similar content $c' \in \mathcal{C}^{s(\geq)}$, i.e., probability $q_{\mathbf{P}[i]}^{s(\geq)}$, and probabilities that $j\,(1 \leq j \leq i-1)$-th nodes on the path do not return the content simultaneously, i.e., $1 - q_{\mathbf{P}[j]}^{s(\geq)}$. That is to say,

$$\eta_i^{s(\geq)} = \prod_{j=1}^{i-1} \left(1 - q_{\mathbf{P}[j]}^{s(\geq)}\right) q_{\mathbf{P}[i]}^{s(\geq)}. \tag{8}$$

Finally, we derive the probability $R_n^{s(\geq)}$ in Eq.(4) that, at the n-th request, the client obtains content whose similarity is larger than or equal to s, i.e., $c' \in \mathcal{C}^{s(\geq)}$. Letting $\rho^{s(\geq)}$ be the probability that such a content is provided from one of any nodes on a path, $R_n^{s(\geq)}$ is given by

$$R_n^{s(\geq)} = (1 - \rho^{s(\geq)})^{n-1}\, \rho^{s(\geq)}, \tag{9}$$

where

$$\rho^{s(\geq)} = \sum_{i=1}^{|\mathbf{P}|} \left((1-\alpha)^{i-1}\, q_{\mathbf{P}[i]}^{s(\geq)} \right). \tag{10}$$

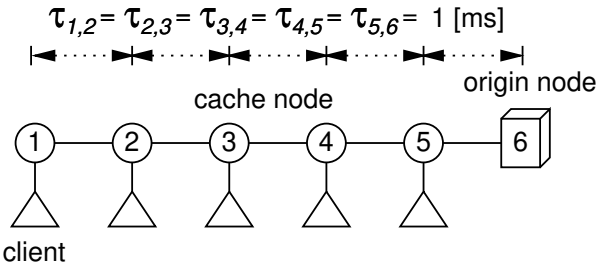


FIGURE 3. Network topology used in our numerical evaluation.

IV. NUMERICAL EVALUATION

In this section, through numerical evaluation, we investigate the property of similarity caching in terms of the similar-content delivery delay. Section IV-A explains the setting of parameters and Section IV-B presents the evaluation results and detailed discussion.

A. PARAMETER SETTING

The network topology used in this evaluation is a tandem topology as shown in Fig. 3; five cache nodes $v(1), \ldots, v(5)$ and one origin node $v(6)$ storing all the content are in series; a single client is connected to each cache node. We set the path $\mathbf{P}_{v(i),c}$ of client $v(i)$ to $(v(i), v(i+1), \ldots, v(6))$ regardless of content c. Also, we set the propagation delay $\tau_{u,v}$ between nodes to 1 [ms]. We note that the reason why we used the simple tandem topology, although it is typical to use more complex network topologies, e.g., a hierarchical topology [30], is to make our results interpretable.

To give the similarity among contents, we used two types of datasets.

- Synthetic dataset
  This dataset is constructed by following the methodology in Ref. [21]; namely, the dataset is artificially generated by calculating *distance*[2]. Formally speaking,

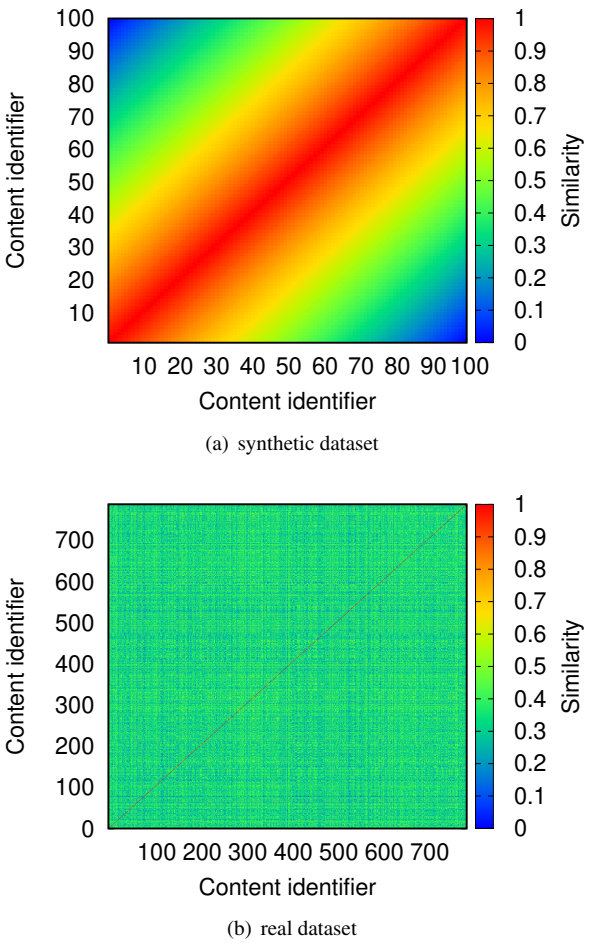
[2]The distance is also called *dissimilarity* or *cost*.

the distance $d(a,b)$ between any two content $a, b \in \mathcal{C}$ is given by $d(a,b) = |a-b|^{\gamma}$, where γ is a parameter that adjusts the distance between content. We used $\gamma = 1$ in our numerical evaluations. By normalizing the distance $d(a,b)$, we set the similarity $s(a,b)$ as $(d_{\max} - d(a,b))/(d_{\max} - d_{\min})$, where $d_{\min}$ and $d_{\max}$ represent the minimum and maximum distances, respectively.
In this case, we set the number of content C to 100 [content].

- Real dataset
  As a dataset, we used images published in kaggle[3], called Natural Images. This dataset contains images comprised of eight classes: airplane, car, cat, dog, flower, fruit, motorbike, and person. Among eight classes, the class "motorbike" with 788 images was used in our evaluation because the difference among eight classes is marginal.
  From this dataset, we obtained the similarity between images as follows.
  1) Detect the feature of an image via ORB algorithm.
  2) Calculate the Euclidean distance of features between images.
  3) Convert the distance to the similarity in the same way as in the synthetic dataset.

The similarity given by these two datasets is plotted in Fig. 4. In this figure, the term "content identifier" means content $c \in \{1, \ldots, C\}$.

Unless explicitly stated, we used the following parameters: cache size at cache node $B = 10$ [content]; content request ratio $\lambda_c = 1/C$; retransmission interval $\delta = 10$ [ms].



(a) synthetic dataset

(b) real dataset

FIGURE 4. Similarity among contents.

B. RESULTS

First of all, we confirm a general tendency of similarity caching from Fig. 5. The upper figure and the lower figure show the average over the s-content delivery delay of all the clients in the case of the synthetic dataset and the real dataset, respectively. In these figures, results with different α, i.e., the probability that an intermediate cache node decides to serve similar content, are plotted.

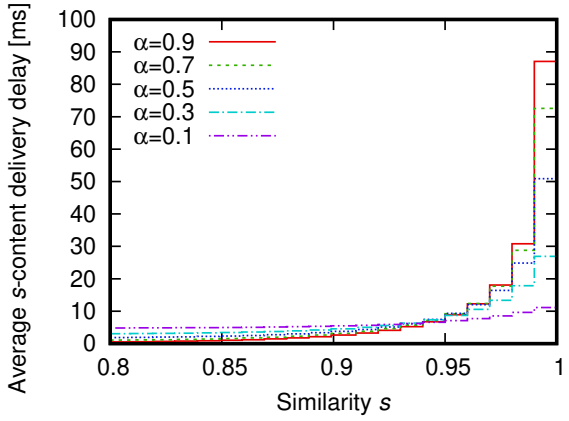
From Fig. 5(a), we discuss the relationship between the similarity s and s-content delivery delay. This figure reveals an intuitive tendency that the s-content delivery delay is increased with respect to the similarity s, regardless of α. In other words, the delivery delay for low-similar content is small whereas that for high-similar content is large. The reason of the increase in s-content delivery delay is caused by the client's retransmission behavior; namely, the client repeats content requests until the client retrieves any content satisfying a given similarity s.

Also, we ensure how the s-content delivery is affected by α. From Fig. 5(a), we can see that when α is low, i.e., $\alpha = 0.1$, the s-content delivery delay does not depend on the similarity s; in other words, the s-content delivery delay is not rapidly increased with respect to the similarity s. In contrast, when α is large, i.e., $\alpha = 0.9$, the s-content delivery delay exhibits the opposite tendency. This phenomenon is due to the nature of α. Recall that α is a parameter to adjust a probability that a cache node serves similar content; this means that as α increases, the cache node returns similar content to a requesting client more actively. Hence, with the increase in α, a client can quickly obtain low-similar content from intermediate cache nodes, whereas the client needs the longer time including the retransmission interval to obtain high-similar content.

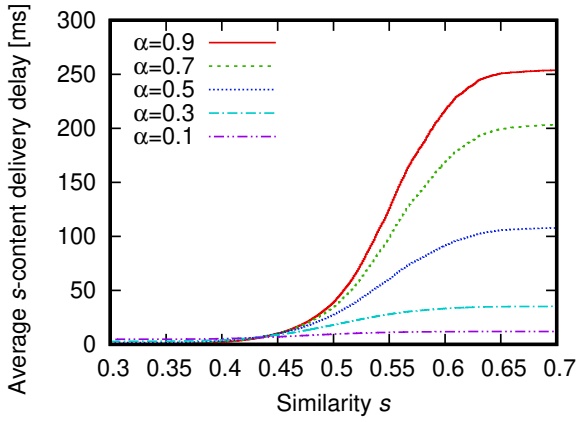
We validate that the above discussions can be applied to a real situation by comparing the result with synthetic dataset (Fig. 5(a)) and that with real dataset (Fig. 5(b)). From this comparison, we argue that our observations at the case with the synthetic dataset, i.e., the relationship between the similarity s and s-content delivery delay, and the effect of α, still hold at the case with the real dataset. As an additional remark, we note the reason why the s-content delivery delay with the real dataset is larger than that with the synthetic dataset. This is mainly due to the difference in the scale of content space; specifically, the number of content in synthetic dataset is 100 while that in real dataset is 788.

Next, we look at the s-content delivery delays of each

[3]https://www.kaggle.com/datasets/prasunroy/natural-images

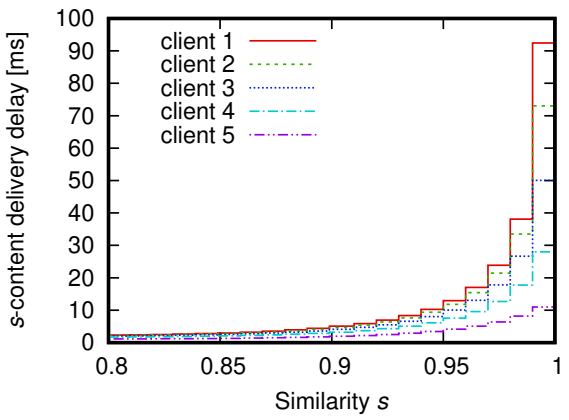


(a) synthetic dataset

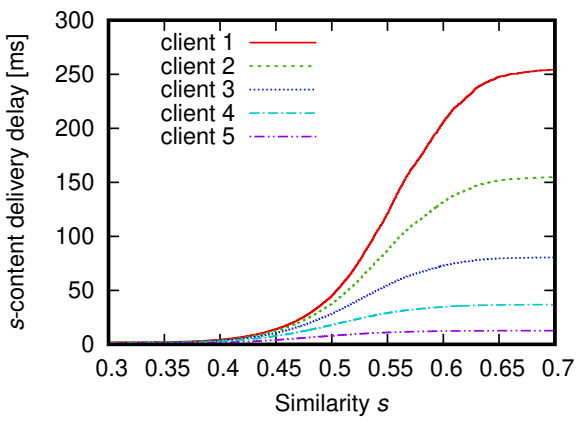


(b) real dataset

FIGURE 5. **Average *s*-content delivery delay. This figure indicates an intuitive tendency that the *s*-content delivery delay increases with the increase in similarity *s*.**



(a) synthetic dataset



(b) real dataset

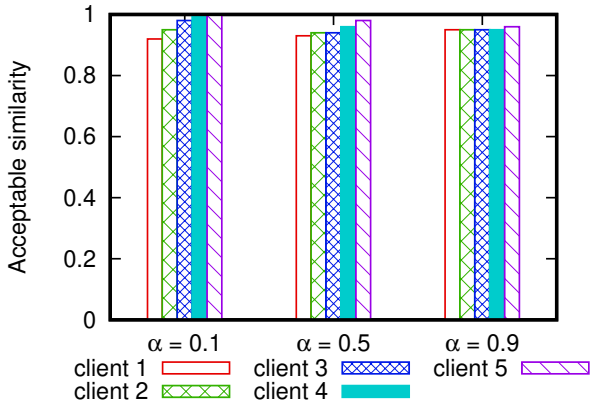
FIGURE 6. ***s*-content delivery delay of each client with $\alpha = 0.5$. Note that "client 1" is the client farthest from the origin node, whereas "client 5" is the nearest one. This figure depicts that the *s*-content delivery delay is closely related with the client's position.**

client as depicted in Fig. 6. Differently from the preceding figure (Fig. 5), Fig. 6 plots results focusing on each client when $\alpha = 0.5$. In this figure, the term "client 1" refers to the one farthest from the origin node, whereas "client 5" refers to the one nearest to the origin node (see Fig. 3).

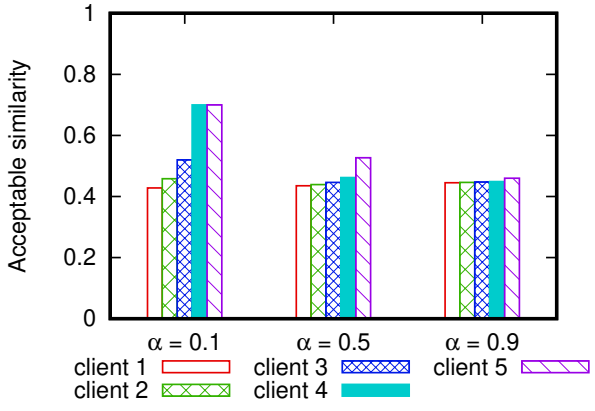
This figure shows a fact that the increase in *s*-content delivery delay is closely-related with the client's position, i.e., distance between a requesting client and an origin node. More specifically, the *s*-content delivery delay for client 1 is increased rapidly with respect to the similarity *s*; in contrast, that for client 5 does not exhibit such a tendency. That is to say, as a client gets closer to an origin node, the increase in *s*-content delivery delay is mitigated. This is because, when a client wants high-similar content such as content with $s = 1$, the origin node is more expected to return the content than intermediate caches. In this aspect, clients close to the origin node have an advantage that their content requests are more likely to reach the origin node.

To demonstrate the usability of our mathematical analysis, we reveal the similarity that a client can accept in a sense that the content delivery delay observed by the client is smaller than a given baseline. Hereinafter, we call this similarity as *acceptable similarity*. Formally, the acceptable similarity is defined as $\sup(\{s \,|\, D_v^{s(\geq)} \leq D_v^*, \forall s\})$, where $D_v^{s(\geq)} = \sum_c \lambda_c D_{v,c}^{s(\geq)}$ and D_v^* is the baseline. In this numerical evaluation, D_v^* is set to the round-trip propagation delay between a client and the origin node. Taking an example with clients 1 and 5, $D_{v(1)}^*$ and $D_{v(5)}^*$ are 10 $(= 5 \times 2)$ [ms] and 2 $(= 1 \times 2)$ [ms], respectively.

Figure 7 depicts the acceptable similarity of each client with different α. This figure reveals that the acceptable similarity is dominated by two factors: the probability α and the distance from a client to the origin node. When we look at the case with $\alpha = 0.1$, the acceptable similarity is unfair according to clients. Namely, the acceptable similarity for clients located far from the origin node, e.g., clients 1 and 2, is low whereas that for other clients is high. But, this unfairness is alleviated as α increases. This is because, with the increase in α, the cache node actively returns similar content; hence, clients located far from the origin node can retrieve similar content in a short delay, while other clients



(a) synthetic dataset



(b) real dataset

FIGURE 7. Acceptable similarity of each client. This figure suggests that the similarity acceptable to clients is dominated by the client's location and the parameter α.

tend not to retrieve exact content from the origin node. The aforementioned tendency is more noticeable in the case of real dataset (see Fig. 7(b)). Through this demonstration, we reveal the usability of our analysis. Namely, if the content delivery delay allowable to a requesting client, the acceptable similarity can be derived. This potentially contributes to, for instance, optimization of parameter α.

Finally, we highlight key insights obtained through our numerical evaluation.

- Benefits gained by similarity caching depend on the similarity which is acceptable to users. Namely, similarity caching is effective for a user who can accept low-similar content because such a content is delivered in the short delay. For a user who requires high-similar content, in particular, exact content, however, the effectiveness of caching is mitigated.
- The above benefits differ according to the user's location, which implies the importance that the cache node make a decision on whether serving similar content or not.

V. CONCLUSION

In this paper, we have studied the property of similarity caching on a general cache network, in particular, the trade-off between the communication quality and the similarity among contents. Specifically, we have formulated and derived the similar-content delivery delay as a user-oriented metric for similarity caching. Also, through numerical evaluation using the synthetic dataset and the real dataset, we have analyzed the interaction between several factors: similar-content delivery delay, content similarity, degree that intermediate caches provide similar content, and client's location. Our key findings are summarized as follows; (i) the effectiveness of similarity caching is dependent on the similarity allowable for users, and (ii) this tendency is also affected by the position of users, i.e., distance to an origin server.

As future works, we are planning to perform extensive evaluation while focusing on the network topology because the network topology used in this paper is quite simple. It is also important to design a specific network architecture as an application of similarity caching. Additionally, we extend the client's request model to incorporate some actions, for instance, retrieving other content with higher similarity, instead of waiting for repeating requests.

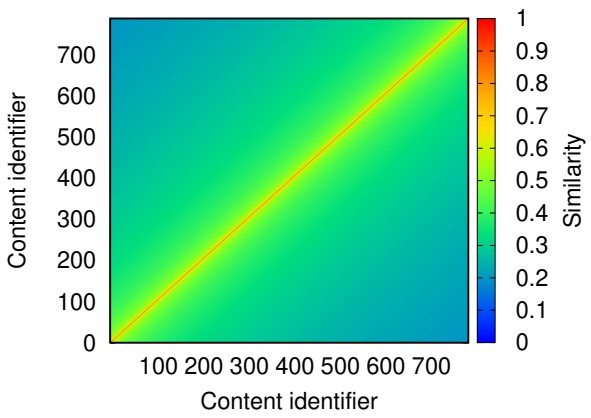
APPENDIX A
REPRODUCTION OF CONTENTS SIMILARITY OF REAL DATASET

In this Appendix, we introduce a method to synthetically reproduce the content similarity of real dataset. Our motivation is to fill a gap between the similarity with the synthetic dataset and that with the real dataset. As shown in Fig. 4, the tendency for the synthetic dataset diverges from that for the real dataset. To resolve this issue, we modify the synthetic method to generate the similarity. Specifically, our procedure is as follows.

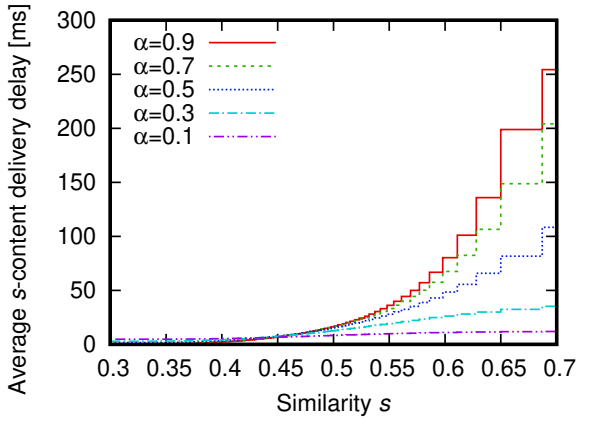
1) For a given two contents $a, b \in \mathcal{C}$, calculate the distance $|a-b|^\gamma$ with a parameter γ and normalize the distance to the similarity.
2) Shift this similarity by a given value θ. Formally speaking, $s(a,b) = \min(s(a,b)+\theta, 1)$.

The difference from the procedure in Section IV is introducing the "shift" operation.

Results using this method with $\gamma = 0.1$ and $\theta = 0.2$ are shown in Fig. 8; Figs. 8(a) and 8(b) plot the similarity among contents and the average s-content delivery delay, respectively. We note that the number of contents C is 788, which is equal to that of the real dataset. Comparing results with the real dataset, we can see that results with this synthetic dataset are close to those with the real dataset. In particular, looking at the s-content delivery delay (Fig. 8(b)), the tendency is almost the same as the real dataset, except for $\alpha = 0.7$ and $\alpha = 0.9$, due to manually adjusting the content similarity. As a consequence, we argue that it is able to sufficiently reproduce the content similarity observed at real world even though it needs manual tuning.



(a) similarity between content



(b) average s-content delivery delay

FIGURE 8. Results with modified synthetic dataset. This figure indicates that similarity of real dataset is reproducible with sufficient quality.

REFERENCES

[1] E. Chávez, G. Navarro, R. Baeza-Yates, and J. L. Marroquín, "Searching in metric spaces," *ACM Comput. Surv.*, vol. 33, pp. 273–321, Sept. 2001.
[2] J. Tang, J. Li, X. Chen, K. Xue, L. Zhang, and J. Lu, "Cooperative caching in satellite-terrestrial integrated networks: A region features aware approach," *IEEE Trans. Veh. Technol.*, vol. 73, pp. 10602–10616, July 2024.
[3] J. Zhang, Y. Yang, H. Sang, Z. Gao, and T. Song, "Content-aware proportional caching for efficient data delivery over satellite network," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, pp. 4890–4895, Dec. 2023.
[4] F. Faticanti and G. Neglia, "Optimistic online caching for batched requests," in *Proc. IEEE Int. Conf. Commun. (ICC)*, pp. 6243–6248, June 2023.
[5] S. Pires, A. Ribeiro, and L. N. Sampaio, "On learning suitable caching policies for in-network caching," *IEEE Trans. Mach. Learn. Commun. Netw.*, vol. 2, pp. 1076–1092, July 2024.
[6] A. Vakali and G. Pallis, "Content delivery networks: Status and trends," *IEEE Internet Comput.*, vol. 7, pp. 68–74, Dec. 2003.
[7] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proc. 5th Int. Conf. emerging Netw. EXP. Technol. (CoNEXT)*, pp. 1–12, ACM, Dec. 2009.
[8] L. Zhang *et al.*, "Named data networking," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, pp. 66–73, July 2014.
[9] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communications perspective," *IEEE Commun. Surv. Tutorials*, vol. 19, pp. 2322–2358, Aug. 2017.
[10] G. Xylomenos *et al.*, "A survey of information-centric networking research," *IEEE Commun. Surv. Tutorials*, vol. 16, no. 2, pp. 1024–1049, 2014.
[11] D. Kondo, T. Ansquer, Y. Tanigawa, and H. Tode, "Resource breadcrumbs: Discovering edge computing resources over named data networking," *IEEE Trans. Netw. Serv. Manage.*, vol. 21, pp. 3305–3316, June 2024.
[12] J. Hou, T. Tao, H. Lu, and A. Nayak, "An optimized GNN-based caching scheme for SDN-based information-centric networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, pp. 401–406, Dec. 2023.
[13] Y. Hayamizu, A. Ooka, K. Matsuzono, and H. Asaeda, "Controller-assisted adaptive video streaming experimented in cloud-native ICN platform," in *Proc. 5th Int. Workshop Intell. Cloud Comput. Netw. (ICCN)*, pp. 1–6, May 2023.
[14] L. Liu, Y. Li, Y. Xu, Q. Zhang, and Z. Yang, "Deep learning-enabled file popularity-aware caching replacement for satellite-integrated content-centric networks," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 58, pp. 4551–4565, Oct. 2022.
[15] M. Rodríguez-Pérez, S. Herrería-Alonso, A. Suárez-Gonzalez, J. C. López-Ardao, and R. Rodríguez-Rubio, "Cache placement in an NDN-based LEO satellite network constellation," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 59, pp. 3579–3587, Aug. 2023.
[16] F. Falchi, C. Lucchese, S. Orlando, R. Perego, and F. Rabitti, "A metric cache for similarity search," in *Proc. ACM Workshop Large-Scale Distrib. Syst. Inf. Retrieval (LSDS-IR)*, pp. 43–50, Oct. 2008.
[17] S. Pandey, A. Broder, F. Chierichetti, V. Josifovski, R. Kumar, and S. Vassilvitskii, "Nearest-neighbor caching for content-match applications," in *Proc. 18th Int. Conf. World Wide Web (WWW)*, pp. 441–450, Apr. 2009.
[18] G. Neglia, M. Garetto, and E. Leonardi, "Similarity caching: Theory and algorithms," *IEEE/ACM Trans. Networking*, vol. 30, pp. 475–486, Dec. 2021.
[19] A. Sabnis, T. S. Salem, G. Neglia, M. Garetto, E. Leonardi, and R. K. Sitaraman, "GRADES: gradient descent for similarity caching," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, pp. 1–10, May 2021.
[20] T. S. Salem, G. Neglia, and D. Carra, "AÇAI: Ascent similarity caching with approximate indexes," in *Proc. 33rd Int. Teletraffic Congr. (ITC)*, pp. 1–9, Aug. 2021.
[21] J. Zhou, O. Simeone, X. Zhang, and W. Wang, "Adaptive offline and online similarity-based caching," *IEEE Networking Lett.*, vol. 2, pp. 175–179, Dec. 2020.
[22] M. Garetto, E. Leonardi, and G. Neglia, "Content placement in networks of similarity caches," *Comput. Networks*, vol. 201, pp. 1–13, Dec. 2021.
[23] W. Xu, Z. Zhang, H. Song, S. Liu, Y. Feng, and B. Liu, "ISAC: In-switch approximate cache for IoT object detection and recognition," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, pp. 1–10, May 2023.
[24] A. Araldo, F. Martignon, and D. Rossi, "Representation selection problem: Optimizing video delivery through caching," in *Proc. 15th IFIP Netw. Conf. Workshops*, pp. 1–9, May 2016.
[25] M. Papalini, *TagNet: A Scalable Tag-Based Information-Centric Network*. PhD thesis, Faculty of Informatics of the Università della Svizzera Italiana, Oct. 2015.
[26] O. Ascigil, S. Rẽné, G. Xylomenos, I. Psaras, and G. Pavlou, "A keyword-based ICN-IoT platform," in *Proc. 4th ACM Conf. Inf.-Centric Netw. (ICN)*, pp. 22–28, Sept. 2017.
[27] R. Nakamura and N. Kamiyama, "Proposal of keyword-based information-centric delay-tolerant network," in *Proc. IEEE Commun. Soc. Int. Commun. Qual. Reliab. Workshop (CQR)*, pp. 1–7, May 2021.
[28] R. Nakamura and N. Kamiyama, "Content availability at network failure in information-centric networking," *IEEE Trans. Netw. Serv. Manage.*, vol. 18, pp. 3889–3899, Sept. 2021.
[29] V. Martina, M. Garetto, and E. Leonardi, "A unified approach to the performance analysis of caching systems," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, pp. 2040–2048, Apr. 2014.
[30] A. Ioannou and S. Weber, "A survey of caching policies and forwarding mechanisms in information-centric networking," *IEEE Commun. Surv. Tutorials*, vol. 18, pp. 2847–2886, May 2016.

RYO NAKAMURA received the M. E. and Ph. D. degrees from Kwansei Gakuin University, Japan, in 2017 and 2020, respectively. He is currently a lecturer at Faculty of Engineering, Fukuoka University, Japan. His research work is in the area of performance analysis and evaluation of Information-Centric Networking. He is a member of IEEE, Institute of Electronics, Information and Communication Engineers of Japan (IEICE), and Information Processing Society of Japan (IPSJ).

NORIAKI KAMIYAMA received his M.E. and Ph.D. degrees in communications engineering from Osaka University in 1994 and 1996, respectively. From 1996 to 1997, he was with the University of Southern California as a visiting researcher. He joined NTT Multimedia Network Laboratories in 1997, and he has been at NTT Network Technology Laboratories by 2016. He was also with the Osaka University as an invited associate professor from 2013 to 2014 and an invited professor in 2015. From 2017, he has been a professor of Fukuoka University. From 2021, he is a professor of Ritsumeikan University. He has been engaged in research concerning content distribution systems, network design, network economics, traffic measurement and analysis, and traffic engineering. He received the best paper award at the IFIP/IEEE IM 2013. He is a member of IEEE, ACM, and IEICE.