Mitigating Congestion with Explicit Cache Placement Notification for Adaptive Video Streaming over ICN

Rei NAKAGAWA∗a, Satoshi OHZAHATA†, Ryo YAMAMOTO†, and Toshihiko KATO†, Members

SUMMARY  Recently, information centric network (ICN) has attracted attention because cached content delivery from router’s cache storage improves quality of service (QoS) by reducing redundant traffic. Then, adaptive video streaming is applied to ICN to improve client’s quality of experience (QoE). However, in the previous approaches for the cache control, the router implicitly caches the content requested by a user for the other users who may request the same content subsequently. As a result, these approaches are not able to use the cache effectively to improve client’s QoE because the cached contents are not always requested by the other users. In addition, since the previous cache control does not consider network congestion state, the adaptive bitrate (ABR) algorithm works incorrectly and causes congestion, and then QoE degrades due to unnecessary congestion. In this paper, we propose an explicit cache placement notification for congestion-aware adaptive video streaming over ICN (CASwECPN) to mitigate congestion. CASwECPN encourages explicit feedback according to the congestion detection in the router on the communication path. While congestion is detected, the router caches the content to its cache storage and explicitly notifies the client that the requested content is cached (explicit cache placement and notification) to mitigate congestion quickly. Then the client retrieves the explicitly cached content in the router detecting congestion according to the general procedures of ICN. The simulation experiments show that CASwECPN improves both QoS and client’s QoE in adaptive video streaming that adjusts the bitrate every video segment download. As a result, CASwECPN effectively uses router’s cache storage as compared to the conventional cache control policies.

key words: information centric networking, adaptive video streaming, cache management, congestion control, quality of experience

1. Introduction

With significant increase of video traffic over the Internet [1], information centric network (ICN) has attracted attention as a new content distribution infrastructure [2]. In ICN, each network node associates a content-based and decentralized network of a content request by Interest packets and a content distribution by Data packets. An ICN router temporarily caches the transferring Data packets to its cache storage and enables content delivery from cache storage to clients. By delivering content quickly from router’s cache storage close to the client, the ICN router can reduce redundant traffic to the upstream. Thus, since ICN has a key advantage of improving QoS with effective use of router’s cache storage, video streaming over ICN is applied for improvement of QoS and client’s QoE [2], [3].

In conventional single bitrate video streaming, however congestion leads to degradation of QoS and QoE because congestion reduces throughput and the reduced throughput causes video buffer starvation in the video player resulting in playback interruption (stall time) [4]. To avoid congestion, adaptive video streaming, which is the most popular video streaming method to adjust the bitrate in multiple bitrate levels, is proposed [5]. In adaptive video streaming, an adaptive bitrate (ABR) algorithm in a client can avoid congestion by reducing the bitrate of the video segment according to congestion estimation. However, congestion causes a bitrate reduction and frequent variation of the bitrate in adaptive video streaming [6], [7], and still reduces QoE [8].

In ICN, since content delivery from router’s cache storage enables fast delivery from routers closer to the client and reduces redundant traffic to upstream, many cache control policies are applied to adaptive video streaming to improve QoS and QoE [9]–[13]. These previous cache policies stand for an implicit cache placement that the router caches a content requested by a user for the other users who may request the same content subsequently. However, since adaptive video streaming dynamically selects the bitrate level even for the same content with congestion of the network, the other users obtain the cached contents with the other bitrate levels. As a result, these cache policies are not always effective for improvement of QoS and QoE in adaptive video streaming over ICN. In addition, since the previous cache policies are designed to increase throughput without considering the network congestion state, the ABR algorithm works incorrectly (e.g., overestimating the bitrate) even if network is congested and causes congestion as a result, and then QoS and QoE degrade due to congestion [9].

As an alternative approach for transport control in ICN research areas, an explicit congestion control, which applies the idea of explicit congestion notification (ECN), is proposed for better communication performance than that of conventional implicit congestion control [14]–[16]. These approaches propose feedback-enabled congestion control in which routers explicitly notify the local congestion state to the clients on a communication path. By detecting congestion according to the explicit feedback, the client can avoid congestion by decreasing its transmission rate of Interest packet according to the explicit congestion state at the router. However, the idea of ECN has not been applied to the implicit cache policy of ICN router for adaptive video streaming.

In this paper, we propose congestion-aware adaptive video streaming over ICN with explicit cache placement no-
tification (CASwECPN). CASwECPN applies the idea of ECN to explicit cache placement notification (ECPN) for mitigating QoS and QoE degradation due to congestion. The design of CASwECPN employs an ECN framework based on the explicit congestion detection measuring packet queue length of each router’s network interface [15], [16]. According to the explicit congestion detection, the router avoids congestion by caching the arrived contents to its cache storage and explicitly notifies the client of that the requested content is cached (explicit cache placement and notification). According to the feedback, the client retrieves the explicitly cached content in the router after congestion is mitigated. As a result, the router caches the contents that the users surely request after mitigating congestion for effective use of cache storage. In the simulation experiments, we implement CASwECPN with each of two representative cache placement methods, leave copy everywhere (LCE) and Probcache [10], and evaluate its effectiveness for improvement of QoS, QoE and cache efficiency in adaptive video streaming.

This paper is organized as follows. Section 2 describes backgrounds and related work. Section 3 proposes CASwECPN. Then we show the evaluation of CASwECPN from the point of QoS metrics, the complex metrics for client’s QoE, and cache efficiency in Sect.4. Section 5 provides discussion about the behavior of CASwECPN according to the results in Sect.4. Finally, we conclude CASwECPN in Sect.6.

2. Backgrounds and Related Works

2.1 Adaptive Video Streaming over ICN

Recently, adaptive video streaming has become the most popular video streaming method because the client improves QoE by adjusting the bitrate according to the estimated congestion state as compared to conventional single bitrate video streaming [3]. Therefore, ICN is also applied to adaptive video streaming for further client’s QoE improvement [2]. Figure 1 shows an overview of adaptive video streaming over ICN [17]. In Fig.1, the server stores the video contents composed of the media presentation description (MPD) that includes video contents information (e.g., a number of video segments, stored multiple bitrate levels, audio information) and N video segments of short play time encoded in M bitrate levels1. The server and the router with cache storage transfer the video contents to the client over ICN. In addition, the router caches transferring Data packets to its cache storage according to the preinstalled cache control policy. The client starts video streaming with the video player engine in the application layer. The video player engine equips a fixed-length video buffer which stores pre-downloaded video segments and starts the playback of the stored video segment after the video buffer length becomes larger than a predefined threshold. In addition, the ABR algorithm is installed to the video player engine for adaptive video streaming. The ABR algorithm adjusts the bitrate for each video segment before the video player engine downloads the video segment according to the following procedures through the transport control in the ICN transport layer.

Figure 2 shows the sequential procedures in the Interest/Data packet communication for the video contents download in adaptive video streaming. The video player engine first downloads MPD for streaming initialization ((0) in Fig.2) and the transport control manages the communication to receive MPD ((0-0) to (0-k)) in Fig.2. In the communication, the transport control first sends an Interest packet for requesting a Data packet of a first data chunk of MPD ((0-0) in Fig.2) using a name address composed of the content information of MPD and the first chunk name (e.g., video/MPD/#1). When the router receives the Interest packet, the router first looks up the corresponding Data packet in its cache storage ((0-1) in Fig.2). If the Data packet is cached, the router returns the cached Data packet to

1The bitrate represents the average throughput required to download the video segment encoded in the bitrate.
the client ((0-5) in Fig. 2). If not, the router transfers the Interest packet to the server for requesting the Data packet ((0-2) in Fig. 2); when the router receives the Data packet, the router caches the Data packet to its cache storage according to the preinstalled cache control policy ((0-4) in Fig. 2) and transfers the Data packet to the client ((0-5) in Fig. 2). After acquiring the first data chunk of MPD, the transport control acquires the remaining data chunks to download MPD ((0-6) to (0-k) in Fig. 2) as the same procedure for the first data chunk. As soon as downloading MPD, the video player engine with ABR algorithm starts the bitrate adaptation of each video segment ((1-0) to (N-0) in Fig. 2) and download of the video segment sequentially ((1-1) to (N-1) in Fig. 2). For instance, in the first video segment download ((1-0) and (1-1) in Fig. 2), the ABR algorithm first selects the bitrate (Bitrate0) of the first video segment according to the estimated congestion state in the previous video content download (i.e., throughput in MPD download) ((1-0) in Fig. 2), and then the video player engine requests the communication procedure to download the first video segment ((1-1) in Fig. 2) as the same procedure for MPD download.

In this way, the video player engine with ABR algorithm sequentially downloads MPD and the video segments through the ICN transport layer as long as the video buffer is not occupied, and the server/router’s cache storage transfers Data packets of MPD and video segments to the client. In addition, the transport control is installed into the ICN transport layer for management of the communication procedure in downloading video segments. In the following subsections, we describe the details of cache control policy for router’s cache storage (in section 2.2), the ABR algorithm (in section 2.3) in the application layer, and the transport control in the ICN transport layer (in section 2.4).

2.2 Cache Control Policy with Implicit Cache Placement for Clients

Most of the cache control policies are composed of a cache placement policy and a cache replacement policy on each router’s cache storage along the download path[12]. The cache placement policy determines whether a router should cache the arrived Data packets in its cache storage ((0-4) in Fig. 2). The replacement policy determines relocation of the Data packets in cache storage on cache eviction. In addition, the eviction of Data packets when cache storage is occupied at cache placement.

Algorithm 1 shows the basic procedures of the cache placement and replacement policy in cache storage. If a Data packet arrives (line 1 in Algorithm 1), the cache placement policy first checks if cache storage is occupied. If occupied (line 2 in Algorithm 1), cache placement of the Data packet (line 5 in Algorithm 1) is scheduled after the cache eviction according to the cache replacement policy (lines 3 in Algorithm 1). If an Interest packet arrives (line 6 in Algorithm 1), the cache replacement policy first looks up cache storage for the corresponding Data packet and makes a copy of the Data packet (lines 7-8 in Algorithm 1). If the Data packet is not null (cache hit), the cached Data packet is replaced in cache storage according to the cache replacement policy (lines 9-10 in Algorithm 1).

There are two representative algorithms in implicit cache placement policy for clients.

- **Leave copy everywhere (LCE):** LCE is the most basic and simplest cache placement policy to cache all the arrived Data packets to cache storage. LCE enables all the routers on the communication path to cache all the transferring Data packets.
- **Probcache[10]:** Probcache is a probabilistic approach to determine whether a router should cache a Data packet passing through it. To reduce redundant placement, Probcache enables all the routers on the communication path to calculate probability of caching based on its distance from the client and the cache capacity on the communication path.

The above implicit cache placement policies caches a content for the clients who may request the same content subsequently. Since the clients do not always request the cached content, the implicit cache placement policies causes frequent cache misses.

The most representative cache replacement policy is Least Recently Used (LRU), which is designed to leave the most recently requested Data packet in cache storage [18]. LRU moves the hit Data packet to the start of the cache queue on cache hit. In addition, LRU purges the end of the cache queue on cache eviction.

2.3 ABR Algorithm for Congestion State Estimation

The ABR algorithm is designed to select the bitrate before the video segment download according to the congestion state estimation. There are two representative ABR algorithms, the rate-based bitrate adaptation algorithm (RBA) [19] and the buffer-based bitrate adaptation algorithm (BBA) [20]. RBA refers to the average throughput of the last video segment request for congestion state estimation. Dash.js [21], which is the open source project of dynamic adaptive streaming over HTTP (DASH) [4], implements RBA. On the other hand, BBA refers to the remaining video buffer length for congestion state estimation.

Algorithm 1 Cache placement and replacement in router’s cache storage.

```
1: if a Data packet arrives at a router then
2: if cache storage is occupied then
3: eviction of Data packet(s) according to cache replacement policy;
4: end if
5: cache the Data packet according to cache placement policy;
6: else if an Interest packet arrives at the router then
7: look up cache storage for the Interest packet;
8: cache = the matched data packet when looking up;
9: if cache is not Null then
10: replace cache according to cache replacement policy;
11: end if
12: end if
```
Algorithm 2 Rate-based bitrate adaptation of nth video segment.
1: for \( n \in [1, N] \) do
2: \( \{rate_1, ..., rate_n\} \leftarrow \) available bitrate levels in MPD;
3: \( \text{intervalTime} \leftarrow \) constant for time interval to start the next segment download;
4: \( \text{segments}_n \leftarrow \) nth video segment;
5: \( B_v \leftarrow \) video buffer length (sec) at time \( n \);
6: \( B_{\text{max}} \leftarrow \) max length of video buffer;
7: \( T_{\text{set}} \leftarrow \) average throughput in \( \text{segments}_{n-1} \) download;
8: \( T_{\text{max}} = \text{EWMA}(T_{\text{set}}) \);
9: \( \text{Bitrate}_n = \text{max}(rate_a | rate_a \leq T_{\text{set}}) \)
10: \( I_n = \{\text{intervalTime}; (B_v \geq B_{\text{max}}) \}
11: \) end for

Algorithm 2 shows the procedures of RBA when requesting the nth video segment. For the bitrate adaptation of the nth video segment, RBA actually refers to the smoothed average throughput at the request of the last video segment \( (T_{\text{set}}) \) using an exponentially weighted moving average (EWMA) (lines 7-8 in Algorithm 2), and selects the highest possible bitrate \( (\text{Bitrate}_n) \) of nth video segment according to \( T_{\text{set}} \) from the multiple bitrate levels described in MPD (line 9 in Algorithm 2). After the bitrate adaptation, the time interval length \( (I_n) \) to start downloading the nth video segment is calculated according to the remaining video buffer length \( (B_v) \) (line 10 in Algorithm 2). Since RBA implicitly estimates congestion using throughput, RBA causes congestion by selecting excessively high bitrate for available bandwidth [22, 23].

Algorithm 3 shows the procedures of BBA when requesting the nth video segment. For the bitrate adaptation of the nth video segment, BBA refers to remaining video buffer length (sec) at the request of the nth video segment \( (B_v) \), and selects the bitrate \( (\text{Bitrate}_n) \) of nth video segment according to the mapping function from video buffer length to available bitrate levels \( (f(B_v)) \) and two buffer threshold, the reservoir \( (r) \) and the cushion \( (cu) \) (lines 20-30 in Algorithm 3). First, to avoid the cause of stall time, the reservoir is set to fill at least one chunk in the buffer. Therefore, when the buffer is filling up the reservoir (i.e., \( 0 \leq B_v \leq r \)), BBA selects minimum bitrate (lines 20-21 in Algorithm 3). Once the reservoir is reached, BBA then increases the bitrate according to \( f(B_v) \) that increases the bitrate passively as the buffer increases (lines 24-29 in Algorithm 3). Then, when the buffer is filled up enough (i.e., \( r + cu \leq B_v \leq B_{\text{max}} \)), BBA selects maximum bitrate (lines 22-23 in Algorithm 3). After the bitrate adaptation, the time interval length \( (I_n) \) to start downloading the nth video segment is calculated according to the remaining video buffer length \( (B_v) \) (line 31 in Algorithm 3). In this way, the BBA is designed to reduce stall time, thus preventing QoE degradation [20].

2.4 Implicit/Explicit Congestion Control in the ICN Transport Layer
The congestion control in the ICN transport layer stands for the Interest packet transmission control for congestion avoidance. In other words, the congestion control reduces the transmission amount of Interest packets according to the congestion detection. In ICN research areas, the methods of congestion detection are classified into two approaches, an implicit approach and an explicit one. The implicit approach [24-26] is the implicit congestion detection determining the cause of congestion based on packet loss at a client. Specifically, the implicit approach detects congestion when the client does not receive Data packet even after a predefined time length passed since sending the Interest packet as described in Sect. 2.1. The explicit approach is the ECN-like explicit congestion detection in which the router explicitly notifies the local congestion state to the client. Since the explicit approach enables it to notify the congestion state of the network explicitly with short communication delay between the client and the router, communication performance is better than that of the implicit approach [14-16].

To improve client’s QoE with effective congestion avoidance, the idea of explicit congestion control is applied to the implicit ABR algorithm (e.g., RBA), which implicitly estimates congestion using the end-to-end throughput at the client, by some references [22, 23]. However, the idea of ECN has not been applied to the implicit cache control.
3. Congestion-Aware Adaptive Video Streaming over ICN with Explicit Cache Placement Notification

3.1 Overview

To mitigate congestion with effective use of router’s cache storage, we propose congestion-aware adaptive video streaming over ICN with explicit cache placement notification (CASwECPN). CASwECPN follows the prescribed operation of video streaming over ICN, and further introduces congestion avoidance enabling the explicit cache placement notification (ECPN) from routers to clients. Figure 3 shows an overview of CASwECPN. In Fig. 3, the server stores the video contents composed of MPD and the video segments of short play time, and the server and the router with cache storage transfer the video contents to the client over ICN. In the client, the video player engine with ABR algorithm starts adaptive video streaming. Then, the video player engine starts playback of the pre-downloaded video segment in the video buffer according to the prescribed procedure in Sect. 2. In addition, for download processes, the ECP Interest transmission control based on ECP Signal is newly installed to the ICN transport layer for cooperating with the following ECP control in the router.

The ECP control in the router is processed according to the following communication procedures for the video contents download. Figure 4 shows a packet sequence for downloading the video contents with the ECP control. As soon as the video player engine starts downloading MPD ((0) in Fig. 4), every time the router transfers a Data packet to the client, the ECP control checks the local congestion state of the network interfaces for transferring ((0-0-4) in Fig. 4). While congestion is detected on the network interfaces, the router forces the preinstalled cache policy to cache the Data packet to its cache storage ((0-0-4) in Fig. 4) and transfers the Data packet with no payload (as an ECP Signal) ((0-0-5) in Fig. 4) to the client (explicit cache placement and notification). As soon as the client receives the ECP Signal, the ECP Interest transmission control immediately transmits the Interest packet (as an ECP Interest) ((0-0-6) in Fig. 4) to retrieve the explicitly cached Data packet (as an ECP Data) in the router. When the router receives the ECP Interest, the router looks up the ECP Data in cache storage ((0-0-8) in Fig. 4) and checks local congestion when transferring the ECP Data ((0-0-11) in Fig. 4). Then, if congestion is detected, the router transfers the ECP Signal to the client again ((0-0-12) in Fig. 4). In this way, the ECP Signal/ECP Interest communication ((0-0-6) to (0-0-12) in Fig. 4) is repeated as long as the client receives the ECP Data ((0-0-k) in Fig. 4). Thus, to mitigate congestion, the ECP control enables routers to reduce traffic for the congested network interface by caching the requested contents to its cache storage and notifies the explicit feedback of cache placement for the clients who surely retrieve the cached contents.

In the following subsections, we describe the details of the ECP control in the routers (in Sect. 3.2) and the ECP Interest transmission control based on ECP Signal in the client (in Sect. 3.3), which are the functions unique to CASwECPN in Fig. 3.

3.2 ECP Control in the Router on a Communication Path

In CASwECPN, the ECP control works on each network interface. The ECP control checks the local congestion state of the network interfaces for transferring Data packet to the client and forces the preinstalled cache policy to cache Data packet to its cache storage as long as congestion is detected. The ECP control also sends ECP Signal to the client to inform the precached Data packet and checks the local congestion state of the network interfaces for transferring ECP Data packet to the client. The ECP control repeats this process as long as the client receives ECP Data packet.
interface in each router on a communication path. The network interfaces are equipped with a priority queue for preferentially sending ECP packets (ECP Signal/ECP Interest/ECP Data). In the router, the ECP control is composed of the explicit cache placement notification (ECPN) procedure according to the congestion detection and the ECP Data transmission.

The ECPN procedure according to the congestion detection is scheduled every time the router transfers a Data packet through an equipped network interface. Algorithm 4 describes the ECP procedure for transferring Data packet according to the congestion detection. If a transferring Data packet is the ECP Signal, the router does not cache the Data packet (lines 4–5 in Algorithm 4) because the data packet has already been cached in the upstream router and the ECP Signal has no content. If not, the router checks congestion by comparing packet queue length \( \frac{Q_L}{Q_{L_{\text{max}}}} \) and congestion threshold \( C_h \) (lines 7–10 in Algorithm 4). If congestion is detected, the router forces the preinstalled cache policy to cache the Data packet (line 11 in Algorithm 4), removes the content of the Data packet, and creates the Data packet with the ECP flag (ECP Signal) (lines 12–13 in Algorithm 4) to notify ECP for the client. If non-congestion, the router caches the Data packet according to the preinstalled cache policy (lines 14–15 in Algorithm 4). Then, the router finally forwards the Data packet to the prescribed transfer procedures through the network interface (line 18 in Algorithm 4).

The ECP Data transmission is processed when the router receives the ECP Interest from the client. Algorithm 5 describes ECP Data transmission. When an Interest packet arrives, the router first looks up cache storage for the Interest packet (line 3 in Algorithm 5). If cache hit (line 4 in Algorithm 5), the router first checks the congestion state of the network interface to transfer the cached Data packet (lines 5–9 in Algorithm 5). If congestion, the router forwards the cached Data packet to the ECPN procedure (line 10 in Algorithm 5). If not congestion, the router checks whether the Interest packet is the ECP Interest or not (lines 11–12 in Algorithm 5). If the ECP Interest, the router creates the Data packet with the ECP flag (ECP Data) (line 13 in Algorithm 5) and finally forwards the Data packet/ECP Data to downstream (line 15 in Algorithm 5). If cache miss (line 17 in Algorithm 5), the Interest packet is immediately forwarded to upstream to receive the corresponding Data packet (line 18 in Algorithm 5).

### Algorithm 4 ECPN procedure according to the congestion detection in the router

```plaintext
1: interface \( \rightarrow \) network interface for transferring Data packet;
2: cache policy \( \rightarrow \) preinstalled cache control policy;
3: if a Data packet arrives at interface then
4:   if the Data packet is ECP Signal then
5:     do not cache the Data packet in cache storage;
6:   else
7:     \( C_h \rightarrow \) congestion threshold in \([0, 1]\);
8:     \( Q_L \rightarrow \) packet queue length of interface;
9:     \( Q_{L_{\text{max}}} \rightarrow \) max packet queue length of interface;
10:    if \( \frac{Q_L}{Q_{L_{\text{max}}}} \geq C_h \) then
11:      force cache policy to cache the Data packet to cache storage;
12:      remove content payload from the Data packet;
13:      add ECP flag to the Data packet (ECP Signal);
14:    else
15:      cache the Data packet to cache storage according to cache policy;
16: end if
17: end if
18: forward the Data packet to the prescribed transfer process;
19: end if
```

### Algorithm 5 ECP Data transmission in a router

```plaintext
1: cache policy \( \rightarrow \) preinstalled cache control policy;
2: if an Interest packet arrives at the router then
3:   cachehit \( \rightarrow \) the matched Data packet when looking up cache storage for the Interest packet;
4:   if cachehit is not Null then
5:     interface \( \rightarrow \) network interface for transferring cachehit;
6:     \( C_h \rightarrow \) congestion threshold in \([0, 1]\);
7:     \( Q_L \rightarrow \) packet queue length of interface;
8:     \( Q_{L_{\text{max}}} \rightarrow \) max packet queue length of interface;
9:     if \( \frac{Q_L}{Q_{L_{\text{max}}}} \geq C_h \) then
10:    forward cachehit to ECPN procedure;
11:  else
12:    if the Interest packet is ECP Interest then
13:      add ECP flag to Cachehit (ECP Data);
14:    end if
15:    forward cachehit to the prescribed transfer process;
16:  end if
17: end if
18: forward the Interest packet to the prescribed transfer process;
19: end if
```

As soon as the video player engine with ABR algorithm starts downloading the first video content (i.e., MPD), the ECP Interest transmission control is scheduled before the prescribed transport control in the ICN transport layer as below.

When an ECP Signal arrives at the ICN transport layer, a new ECP Interest is created to retrieve the content of ECP Data. When the other Data packet arrives at the client, the new Interest packet follows the prescribed procedures through the transport control. Then, the ICN transport layer finally transfers the newly created Interest packet/ECP Interest by using the priority queue for preferentially sending ECP packets. In this way, the ECP Interest transmission control enables the client to retrieve the explicitly cached content quickly from the router notifying the ECP Signal.

### 4. Evaluation

We evaluate behavior of congestion-aware adaptive video streaming over ICN with explicit cache placement notification (CASwECPN) on QoS, QoE and cache efficiency...
through amus-ndnSIM, the event-driven ICN simulator for video streaming [27].

Figure 5 shows the evaluation topology composed of 5 (white square) servers, 20 routers (black circle) and 25 clients (white circle) [3]. The link speed between the router and the client and between each router is 10 Mbps, and the link speed between the router and the server is 100 Mbps. Each server stores the video content, and each router works in basic functionalities for ICN routing and equips cache storage with a capacity of 15 Mbytes. The preinstalled cache placement policy in cache storage is composed of the implicit cache placement policy of LCE/Probcache and a cache replacement policy of least recently used (LRU). The ECP control ($C_{th} = 0.1$) is installed on all routers for CASwECPN. Each client employs a video player engine with a video buffer with 15 video segments (30 sec). The ICN transport layer in each client controls the video contents download through the ECP Interest transmission control based on ECP Signal and the transport control.

In the simulation we set up the adaptive video streaming scenario with 100 kinds of video content (a time length of each video content is 600 sec), composed of 300 video segments (a time length of each video segment is 2 sec). Each server stores 3.7 Gbytes of video contents (MPD and 300 video segments encoded in 20 bitrate levels (from 45 Kbps to 4.2 Mbps)) [28], and one of the ABR algorithm (RBA or BBA) is installed with the video player engine to each client. RBA determines the bitrate based on end-to-end throughput in the previous download, and BBA determines the bitrate based on the remaining video buffer length¹.

For comparison, we conduct 8 methods based on the aforementioned adaptive video streaming scenario, adaptive video streaming with RBA and LCE (RBA LCE), CASwECPN with RBA and LCE (CASwECPN RBA LCE), adaptive video streaming with RBA and Probcache (RBA Probcache), CASwECPN with RBA and Probcache (CASwECPN RBA Probcache), adaptive video streaming with BBA and LCE (BBA LCE), CASwECPN with BBA and LCE (CASwECPN BBA LCE), adaptive video streaming with BBA and Probcache (BBA Probcache), CASwECPN with BBA and Probcache (CASwECPN BBA Probcache).

4.1 Metrics for Evaluation

To evaluate QoS, we evaluate throughput (Mbps) for both the video content download and the video segment download respectively.

To evaluate QoE, we use QoE-lin scoring [29]–[31] in adaptive video streaming. A QoE-lin is a complex QoE metric based on quantitative log information on an adaptive streaming application in a client. QoE-lin is a linear combination of three QoE metrics, a bitrate, a bitrate magnitude, a stall time defined in (1).

1In the evaluation topology and content requests, we confirmed that network is congested.

$$QoE_{lin} = \sum_{n=1}^{N} q(R_n) - \lambda \sum_{n=1}^{N-1} |q(R_{n+1}) - q(R_n)|$$

$$- \mu \sum_{n=1}^{N} b_n - \mu_D D$$

In QoE-lin, The stall time experience is composed of the playback stop time and the wait time for streaming initialization. $[b_n | 0 \leq b_n < \infty]$ of the third term in $QoE_{lin}$ is a sum of playback stop (stall) time length due to video buffer exhaustion (sec) at a nth video segment playback. In addition, $[D | 0 \leq D < \infty]$ is a sum of stop time length (sec) due to the initialization of adaptive streaming. $b_n$ and $D$ represent “QoE(stall time).”

$\{\lambda, \mu, \mu_D\}$ are non-negative weighting factors for QoE(bitrate magnitude) and QoE(stall time) set against the bitrate metric. We use the predefined values in [27], $\{\lambda = 1, \mu = 4.3, \mu_D = 4.3\}$ for adaptive video streaming.

To evaluate cache efficiency, we use the cache hit rate on each router and the details of received ECP Data at all the clients.

4.2 Evaluation for Adaptive Video Streaming

Firstly, we evaluate the effect of CASwECPN on QoS metric, throughput in 8 methods. Table 1 and Fig. 6 show the comparison of an average and a cumulative distribution function (CDF) of throughput (Mbps).

In Table 1, all the CASwECPN methods do not affect the average throughput metrics as compared to all the regular methods. However, in Fig. 6 (a), (b), we can see that throughput per video segment in all the CASwECPN methods do not become excessively small/large and proportion of
around 2 Mbps is larger as compared to all the regular methods. This is because the router adjusts the throughput by caching the traffics without transferring them to the network interface detecting congestion, and then transfers the cached traffic after congestion is mitigated. In addition, since the ABR algorithm works correctly under the proposed control, in Fig. 6 (c), (d), all the CASwECPN methods are better in around throughput of per video content is larger than 3 Mbps as compared to all the regular methods.

Secondly, we evaluate the effect of CASwECPN on QoE metrics, QoE-lin, QoE(bitrate), QoE(bitrate magnitude), and QoE(stall time) in 8 methods. Table 2 and Fig. 7 show the comparison of an average and CDF of QoE metrics.

First, we evaluate the comprehensive QoE metric, QoE-lin. In Table 2, the CASwECPN methods improves 94 % in the RBA LCE method, 7 % in the BBA LCE method, 114 % in the RBA Probcache method, and 8 % in the BBA Probcache method about the average QoE-lin. Besides, in Fig. 7 (a), (b), all the CASwECPN methods avoids excessively high QoE-lin as compared to all the regular methods. In addition, in Fig. 7 (a), all the CASwECPN RBA methods are not excessively low QoE-lin, and thus QoE-lin becomes more fair for all the clients as compared to all the regular methods. Therefore, all the CASwECPN methods enable the client to improve the comprehensive QoE because the router mitigates QoE degradation due to congestion (details are described in the following).

Second, we evaluate QoE(bitrate). In Table 2, all the CASwECPN RBA methods do not affect the average QoE(bitrate). However, in Fig. 7 (c), all the CASwECPN RBA methods avoid the excessive high QoE(bitrate) larger than 600 as compared to all the regular RBA methods. This is because CASwECPN avoids the excessively high throughput by caching the traffics during congestion (This behavior is shown in Fig. 6 (a)). As a result, all the CASwECPN RBA methods avoid the excessively high bitrate. On the other hand, while the BBA also avoids the excessive high QoE(bitrate) (600 to 900) as well as RBA, the CASwECPN BBA methods are better in the high QoE(bitrate) (larger than 900) compared to all the regular BBA methods. This is because BBA selects the lower bitrate passively and continuously after the bitrate drops excessively due to congestion. As a result, in Fig. 7 (d), by

Table 1: Average throughput.

|                | RBA LCE | CASwECPN RBA LCE | RBA Probcache | CASwECPN RBA Probcache | BBA LCE | CASwECPN BBA LCE | BBA Probcache | CASwECPN BBA Probcache |
|----------------|---------|-------------------|---------------|------------------------|---------|-------------------|---------------|------------------------|
| Throughput per video segment | 3.147   | 3.034             | 3.119         | 3.019                  | 3.120   | 2.981             | 3.119         | 2.955                  |
| Throughput per video content  | 2.508   | 2.596             | 2.486         | 2.590                  | 2.406   | 2.526             | 2.387         | 2.513                  |
mitigating congestion, and then all the CASwECPN BBA methods improves 7% in all the regular methods about the average QoE(bitrate) in Table 2.

Third, we evaluate QoE(bitrate magnitude). In Table 2, the CASwECPN methods improves about 16% in all the regular RBA methods, and about 33% in all the regular BBA methods about the average QoE(bitrate magnitude). Besides, in Fig. 7 (e), (f), we can see that all the CASwECPN RBA methods are smaller than 300 in QoE(bitrate magnitude) as compared to all the regular RBA methods, and all the CASwECPN RBA methods are smaller than 50 in QoE(bitrate magnitude) as compared to all the regular BBA methods. Therefore, CASwECPN enables the ABR algorithm to adjusts the bitrate correctly by adjusting the throughput due to congestion avoidance (in Fig. 6 (a), (b)). As a result, all the CASwECPN methods improve the QoE-lin.

Fourth, we evaluate QoE(stall time). In Table 2, all the CASwECPN RBA methods improve 50% in the average QoE(stall time) as compared to all the regular RBA methods, and all the CASwECPN RBA methods improve 70% in the average QoE(stall time) as compared to all the regular BBA methods. Therefore, CASwECPN enables the ABR algorithm to adjusts the bitrate correctly by adjusting the throughput due to congestion avoidance (in Fig. 6 (a), (b)). As a result, all the CASwECPN methods improve the QoE-lin.

![Fig. 7 CDF for QoE metrics. (a) and (b) QoE(linear combination), (c) and (d) QoE(bitrate), (e) and (f) QoE(bitrate magnitude), (g) and (h) QoE(stall time).](image-url)
Table 2  Average QoE metrics.

| Method          | RBA LCE | CASwECPN RBA | RBA Probcache | CASwECPN RBA Probcache | BBA LCE | CASwECPN BBA | BBA Probcache | CASwECPN BBA Probcache |
|-----------------|---------|--------------|---------------|------------------------|---------|--------------|---------------|------------------------|
| QoE-lin         | 179.787 | 348.640      | 163.206       | 348.452                | 607.308 | 647.303      | 598.217        | 642.898                |
| QoE (bitrate)   | 729.860 | 715.900      | 719.643       | 709.753                | 653.852 | 684.016      | 644.988        | 679.091                |
| QoE(bitrate magnitude) | 257.170 | 219.343      | 253.962       | 214.049                | 38.345  | 28.922       | 38.292         | 28.319                |
| QoE (stall time) | 292.904 | 147.918      | 302.475       | 147.252                | 8.120   | 7.792        | 8.479          | 7.874                |

Consequently, RBA and BBA with the implicit cache policy works incorrectly and causes congestion as a result, and then QoE-lin degrades due to congestion. In contrast, CASwECPN enables RBA and BBA to adjust the bitrate correctly for the in-network congestion state (in Fig. 7 (c), (d)) because the router reduces the throughput quickly by caching data packets in its cache storage to mitigate congestion (in Fig. 6 (a), (b)). Thus, CASwECPN mitigates the bitrate magnitude (in Fig. 7 (e), (f)), enables RBA to reduce the stall time experience (in Fig. 7 (g)), and enables BBA to increase the bitrate (in Fig. 7 (d)). As a result, CASwECPN mitigates comprehensive QoE-lin degradation due to congestion (in Fig. 7 (a), (b)).

Finally, we evaluate the effect of CASwECPN on cache efficiency metrics, cache hit rate on each router and the details of received ECP Data at all the clients.

Table 3 and Fig. 8 show the comparison of an average and CDF of cache hit rate per 10 seconds on each router in 8 methods.

Table 3  Average cache hit rate on each router.

| Method          | Cache hit rate |
|-----------------|----------------|
| oneBit RBA      | 0.059          |
| CASwECPN RBA LCE | 0.073          |
| RBA Probcache   | 0.017          |
| CASwECPN RBA Probcache | 0.071          |
| oneBit BBA      | 0.059          |
| CASwECPN BBA LCE | 0.071          |
| BBA Probcache   | 0.016          |
| CASwECPN BBA Probcache | 0.068          |

In the congestion situation. Besides, the all the Probcache method excessively reduce cache hit rate because Probcache reduces the cache amount on the communication path as
compared to LCE\(^1\). However, all the CASwECPN Prebcache methods improve the average cache hit rate as compared to all the regular Prebcache methods (in Table 3) because CASwECPN forces Prebcache to cache the traffic during congestion and enables the client to request the cached content surely by ECPN after congestion is mitigated.

Table 4 shows the details of received ECP Data at all the clients composed of reception rate (RxRate) of ECP Data \(\frac{\text{Num of received ECP Data}}{\text{Sum of Data Interest}}\) and ratio of content sources, downstream from the router detecting congestion (the congested R), the congested R, upstream from the congested R, in all the received ECP Data.

In Table 4, RxRate is about 0.5 and the content source of ECP Data is almost 100 % of the congested router in all the CASwECPN methods. In other words, the client sends the ECP Interest twice (on average) until receiving the ECP Data from the congested router. Therefore, in CASwECPN, the router does not sufficiently mitigate the congestion at the first transfer of each ECP Data, and then explicitly caches each ECP Data and sends the ECP Signal again to mitigate congestion\(^1\). As a result, the download time during congestion is delayed and the download throughput is reduced (in Fig. 6), and then RBA and BBA avoids excessively high bitrate (in Fig. 7 (c), (d)). Thus, CASwECPN enables RBA and BBA to adjust the bitrate correctly according to congestion state.

Consequently, the conventional and implicit LCE and Prebcache, which caches a content requested by a client for the other clients that may request the content subsequently, is unable to keep caching the contents for the other users due to the very large video content size with comparing the cache size of the router. In contrast, ECPN enables the router that detects congestion to keep caching the content requested by a client until congestion is mitigated, and surely delivers the cached content to the client after congestion is mitigated. As a result, CASwECPN improves cache hit rate of the routers notifying ECPN.

5. Discussion

5.1 Effect of Implicit/Explicit Cache Policy on Bitrate Adaptation

To analyze the effect of the implicit/explicit cache policy on bitrate adaptation, we evaluate the time-series graph of video segment numbers that plots the bitrate, throughput, FromCacheRate of each video segment in the CASwECPN RBA LCE method and the CASwECPN BBA LCE method.

\[
\text{FromCacheRate} = \frac{\text{Data Num from cache}}{\text{Sum of Data Num}} \quad (2)
\]

First, Fig. 9 (a), (b) show the time-series graph of video segment numbers with the bitrate, throughput, and FromCacheRate of each video segment in the CASwECPN RBA LCE method and the CASwECPN RBA LCE method.

In Fig. 9 (a), we can see that RBA selects the excessively high bitrate for the throughput after the throughput increases in the previous segment download with the high FromCacheRate (at segment number 4, 6, 11, 19). Therefore, since the implicit cache policy increases the throughput without considering in-network congestion state, it causes the bitrate overestimation in RBA. Then, the bitrate drops excessively, and the stall time experience is caused due to congestion, QoE degrades as a result.

On the other hand, in Fig. 9 (b), we can see that

| Table 4 | Details of Received ECP Data. |
|---------|-----------------------------|
| Method  | RxRate | Ratio of content sources in received ECP Data |
|         |        | Downstream | Congested R | Upstream |
| CASwECPN RBA LCE | 0.591 | 1.47 \times 10^{-3} | 0.999 | 1.53 \times 10^{-6} |
| CASwECPN RBA Prebcache | 0.595 | 1.41 \times 10^{-3} | 0.999 | 5.42 \times 10^{-6} |
| CASwECPN BBA LCE | 0.592 | 0.85 \times 10^{-3} | 0.999 | 2.50 \times 10^{-7} |
| CASwECPN BBA Prebcache | 0.594 | 0.96 \times 10^{-3} | 0.999 | 2.69 \times 10^{-6} |

\(^1\)Reduced cache hit rate of the regular Prebcache method results in a little degradation of QoE-lin as compared to the regular LCE method in Fig. 7 (a), (b).

\(^1\)Since the ECP Signal is the Data packet with the content payload removed, the size of ECP Signal is small as compared to the size of normal Data packet. Therefore, the traversing of the ECP Signal/Interest reduces the network load while congestion is detected (in Fig. 6).
Table 5  Average QoE metrics on changing the congestion threshold (Cth).

|               | CASwECPN RBA LCE Cth0.1 | CASwECPN RBA LCE Cth0.5 | CASwECPN BBA LCE Cth0.1 | CASwECPN BBA LCE Cth0.5 |
|---------------|--------------------------|--------------------------|--------------------------|--------------------------|
| QoE-lin       | 348.640                  | 308.523                  | 647.303                  | 647.349                  |
| QoE (bitrate) | 715.900                  | 726.258                  | 684.016                  | 684.403                  |
| QoE(bitrate magnitude) | 219.343                  | 234.058                  | 28.922                   | 29.589                   |
| QoE (stall time) | 147.918                  | 183.677                  | 7.792                    | 7.874                    |

Fig. 10  Time-series graph of video segment numbers with the bitrate, throughput, video buffer length (normalized in [0,1]), FromCacheRate of each video segment in (a) the BBA LCE method and (b) the CASwECPN BBA LCE method.

CASwECPN reduces the throughput with the high FromCacheRate (at segment number 12, 13, 17, 18). Therefore, CASwECPN reduces the throughput and avoids the bitrate overestimation in RBA by downloading traffic during congestion with a delay for congestion avoidance. As a result, CASwECPN mitigates QoE degradation due to congestion as compared to the regular RBA methods (in Fig. 7 (e)).

Second, Fig. 10 (a), (b) show the time-series graph of video segment numbers with the bitrate, throughput, video buffer length (normalized in [0,1]), FromCacheRate of each video segment in the BBA LCE method and the CASwECPN BBA LCE method.

In Fig. 10 (a), BBA overestimates the bitrate for the throughput and then has to select the lower bitrate after the period with the high FromCacheRate (in a sequence of segment numbers 10 to 15). Therefore, since the implicit cache policy works without considering the network congestion state, it also causes the incorrect bitrate adaptation in BBA as well as RBA.

Besides, in Fig. 10 (b), BBA avoids the excessively low bitrate with the high FromCacheRate (in a sequence of segment numbers 7 to 20). Therefore, since the router in CASwECPN adjusts the throughput by caching the traffics without transferring them to the network while congestion is detected, BBA works correctly under the control of CASwECPN. As a result, CASwECPN mitigates QoE degradation due to congestion as compared to the regular BBA methods (in Fig. 7 (e)).

5.2 Effect on Changing Congestion Threshold

To analyze the effect on changing the congestion threshold, we evaluate the CASwECPN RBA LCE method and the CASwECPN BBA LCE method with congestion threshold changed to 0.5 (0.1 in default). In Table 5, QoE(bitrate magnitude) and QoE(stall time) degrade in the CASwECPN RBA LCE Cth0.5 method as compared to the CASwECPN RBA LCE Cth0.1 method. Therefore, since the congestion detection is delayed due to the increased congestion threshold, QoE metrics degrade in the CASwECPN RBA LCE Cth0.5. In contrast, the low congestion threshold in the control of CASwECPN enables the router to early detect congestion and enables the client to surely retrieve the explicitly cached contents after congestion is mitigated quickly.

On the other hand, since BBA is insensitive to the change of in-network congestion state with the congestion threshold increased, the CASwECPN BBA LCE method does not affect the QoE metrics when increasing the congestion threshold in Table 5. However, in the case of the video streaming scenarios with the smaller video buffer size (e.g., 6 sec video buffer in HTTP live streaming[^3]), BBA selects the bitrate more sensitively to the in-network congestion state due to the shorter video buffer.[^3]

[^3]: In the evaluation, we assume on-demand video streaming, which allows the client to set a longer video buffer length (i.e., 30 sec) than HTTP live streaming.
when BBA works for video streaming with shorten video buffer such as HTTP live streaming, CASwECPN should be applied to BBA to select bitrate correctly.

6. Conclusion

In this paper, we propose congestion-aware adaptive video streaming over ICN with explicit cache placement notification (CASwECPN) to mitigate congestion. Through the simulation experiments, CASwECPN mitigates congestion because the router reduces the throughput by temporarily caching the data packets during congestion and deliver the cached Data packets by ECPN immediately after congestion is mitigated. As a result, CASwECPN mitigates client’s QoE degradation due to congestion. In addition, since the explicit cache placement policy in CASwECPN enables a manner that the users always request the explicitly cached contents in the router detecting congestion, router’s cache storage is effectively used even for video content with large file sizes.

In the future, for further improvement of cache efficiency in CASwECPN, we will study a cooperative system in which routers offload traffic during congestion to cache storage closer to the client.

References

[1] Cisco, “Cisco Visual Networking Index: Forecast and Trends, 2017–2022,” 2017.
[2] C. Westphal, S. Lederer, D. Posch, C. Timmerer, A. Azgin, W. Liu, C. Mueller, A. Detti, D. Corujo, J. Wang, M. Montpetit, and N. Murray, “Adaptive video streaming over information-centric networking (ICN),” RFC 7933, 2016.
[3] B. Rainer, D. Posch, and H. Hellwagner, “Investigating the performance of pull-based dynamic adaptive streaming in NDN,” IEEE JSAC, vol.34, no.8, pp.2130–2140, Aug. 2016.
[4] P. Juluri, V. Tamarapalli, and D. Medhi, “Measurement of quality of experience of video-on-demand services: A survey,” IEEE Commun. Surv. Tutor., vol.18, no.1, pp.401–418, 2016.
[5] ISO-IEC 23009-1:2012 Information Technology. Dynamic Adaptive Streaming over HTTP (DASH).
[6] N. Barman and M.G. Martini, “QoE modeling for HTTP adaptive video streaming-A survey and open challenges,” IEEE Access, vol.7, pp.30831–30859, 2019.
[7] R.K.P. Mok, E.W.W. Chan, X. Luo, and R. K. C. Chang, “Inferring the QoE of HTTP video streaming from user-viewing activities,” Proc. SIGCOMM W-MUST, pp.31–36, Aug. 2011.
[8] S. Akhshabi, L. Anantakrishnan, C. Dovrolis, and A.C. Begen, “What happens when HTTP adaptive streaming players compete for bandwidth?,“ Proc. International Workshop on Network and Operating System Support for Digital Audio and Video, pp.9–14, June 2012.
[9] D.H. Lee, C. Dovrolis, and A.C. Begen, “Caching in http adaptive streaming: Friend or foe?,” Proc. NOSSDAV, p.31, March 2014.
[10] L. Psaras, W.K. Chai, and G. Pavlou, “Probabilistic in-network caching for information-centric networks,” Proc. ACM ICN WRKSHP, pp.55–60, Aug. 2012.
[11] D. Nguyen, K. Sugiyama, and A. Tagami, “Congestion price for cache management in information-centric networking,” Proc. INFOCOM WRKSHP, pp.1–6, 2015.
[12] W. Li, S.M.A. Oteafy, and H.S. Hassanein, “StreamCache: Popularity-based caching for adaptive streaming over Information-centric networks,” Proc. IEEE ICC, pp.1–6, 2016.
[13] M.A. Naeem, M.A.U. Rehman, R. Ullah, and B.-S. Kim, “A comparative performance analysis of popularity-based caching strategies in named data networking,” IEEE Access, vol.8, pp.50057–50077.
[14] J. Zhou, Q. Wu, Z. Li, M.A. Kaafar, and G. Xie, “A proactive transport mechanism with explicit congestion notification for ndn,” Proc. IEEE ICC, pp.5242–5247, 2015.
[15] K. Schneider, C. Yi, B. Zhang, and L. Zhang, “A practical congestion control scheme for named data networking,” Proc. ACM ICN, pp.21–30, Sept. 2016.
[16] J. Miyoshi, S. Kawauchi, M. Bandai, and M. Yamamoto, “Congestion control for multi-source content retrieval in content centric networks,” IEICE Trans. Commun., vol.E102-B, no.9, pp.1832–1841, Sept. 2019.
[17] M. Conti, R. Droms, M. Hassan, and C. Lal, “Fair-RTT-DAS: A robust and efficient dynamic adaptive streaming over ICN,” Computer Communications, vol.129, no.1, pp.209–225, Sept. 2018.
[18] M. Yamamoto, “A survey of caching networks in content oriented networks,” IEICE Trans. Commun., vol.E99-B, no.5, pp.961–973, May 2016.
[19] J. Kua, G. Armitage, and P. Branch, “A survey of rate adaptation techniques for dynamic adaptive streaming over HTTP,” IEEE Commun. Surv. Tutor., vol.19, no.3, pp.1842–1866, 2017.
[20] T. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, “A buffer-based approach to rate adaptation: evidence from a large video streaming service,” SIGCOMM Comput. Commun. Rev. vol.44, no.4, pp.187–198, Oct. 2014.
[21] DASH Industry Forum, https://github.com/Dash-Industry-Forum/dash.js.
[22] R. Nakagawa, S. Ohzahata, R. Yamamoto, and T. Kato, “A congestion-aware adaptive streaming over ICN combined with explicit congestion notification for QoE improvement,” IEICE Trans. Inf. & Syst., vol.E104-D, no.2, pp.264–274, Feb. 2021. [Preprint]. Available at: https://search.ieice.org/bin/index.php?category=D&lang=E&curr=1#latest (Accessed: 10 Nov. 2020).
[23] D. Nguyen, J. Jin, and A. Tagami, “Cache-friendly streaming bitrate adaptation by congestion feedback in ICN,” Proc. the 3rd ACM Conf. Information-Centric Networking, pp.71–76, Sept. 2016.
[24] G. Carofiglio, M. Gallo, and L. Muscariello, “ICP: Design and evaluation of an interest control protocol for content-centric networking,” Proc. IEEE INFOCOM Workshop, pp.304–309, 2012.
[25] L. Saino, C. Cocora, and G. Pavlou, “CCTCP: A scalable receiver-driven congestion control protocol for content centric networking,” Proc. IEEE ICC, pp.3775–3780, 2013.
[26] C. Xia, M. Xu, and Y. Wang, “A loss-based TCP design in ICN,” Proc. 22nd Wireless and Optical Communication Conference, pp.449–454, 2013.
[27] B. Rainer, C. Kreuzberger, D. Posch, and H. Hellwagner, Demo: amus-ndnSIM-adaptive multimedia streaming simulator for ndn.
[28] MPEG-DASH at IETF/AAU, http://dash.itec.aau.at
[29] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. Joseph, A. Ganjam, J. Zhan, and H. Zhang, “Understanding the impact of video quality on user engagement,” Proc. ACM SIGCOMM, pp.362–373, Aug. 2011.
[30] Z. Akhter, Y.S. Nam, R. Govindan, S. Rao, J. Chen, E. Katz-Bassett, B. Ribeiro, J. Zhan, and H. Zhang, “Oboe: Auto-tuning video ABR algorithms to congestion status,” Proc. ACM SIGCOMM, pp.44–58, 2018.
[31] H. Mao, R. Netravali, and M. Alizadeh, “Neural adaptive video congestion control for multi-source content retrieval in content centric networking,” IEICE Trans. Commun., vol.133, no.1, pp.133–141, 2020.
[32] T.C. Thang, H.T. Le, A.T. Pham, and Y.M. Ro, “An evaluation of bitrate adaptation methods for HTTP live streaming,” IEEE J. Sel. Areas Commun., vol.32, no.4, pp.693–705, April 2014.
Rei Nakagawa received the B.S. and M.S. degrees from the Tokyo University of Science, Chiba, Japan, in 2016, and 2018. He is currently a D. E. candidate at the Graduate School of Informatics and Engineering, the University of Electro-Communications, Tokyo, Japan. His research interests include an ICN technology for adaptive streaming application.

Satoshi Ohzahata received the B.S., M.E., and D.E. degrees from the University of Tsukuba, Ibaraki, Japan, in 1998, 2000, and 2003, respectively. From 2003 to 2007 and from 2007 to 2009, he was a Research Associate in the Department of Computer, Information and Communication Sciences and an Assistant Professor, respectively, at Tokyo University Agriculture and Technology. Since 2009, he has been an Associate Professor at the University of Electro-Communications, Tokyo, Japan. His interests are mobile ad hoc networks, the Internet architecture in mobile environments, and Internet traffic measurement. Dr. Ohzahata is a member of ACM and IPSJ.

Ryo Yamamoto received his B.E. and M.E. degree in electronic information systems from Shibaura Institute of Technology, Tokyo, Japan, in 2007 and 2009. He received D.S. in global telecommunication studies from Waseda University, Tokyo, Japan, in 2013. He was a research associate at Graduate School of Global Information and Telecommunication Studies, Waseda University, from 2010 to 2014, and has been engaged in researching in wireless communication networks. He is presently an associate professor at Graduate School of Informatics and Engineering, The University of Electro-Communications. His current research interests are ad hoc networks, sensor networks, IoT/M2M networks, and network protocols for the networks. He is a member of IEICE and IEEE.

Toshihiko Kato received the B.E., M.E., and Dr. Eng. degrees from the University of Tokyo, Tokyo, Japan, in 1978, 1980, and 1983, respectively, all in electrical engineering. In 1983, he joined KDDI and worked in the field of OSI and Internet until 2002. From 1987 to 1988, he was a Visiting Scientist at Carnegie Mellon University. He is currently a Professor of the Graduate School of Informatics and Engineering, The University of Electro-Communications, Tokyo. His current research interests include protocols for mobile Internet, high-speed Internet and ad hoc network.