SUMMARY  Recently, adaptive streaming over information centric network (ICN) has attracted attention. In adaptive streaming over ICN, the bitrate adaptation of the client often overestimates a bitrate for available bandwidth due to congestion because the client implicitly estimates congestion status from the content download procedures of ICN. As a result, streaming overestimated bitrate results in QoE degradation of clients such as cause of a stall time and frequent variation of the bitrate. In this paper, we propose a congestion-aware adaptive streaming over ICN combined with the explicit congestion notification (CAAS with ECN) to avoid QoE degradation. CAAS with ECN encourages explicit feedback of congestion detected in the router on the communication path, and introduces the upper band of the selectable bitrate (bitrate-cap) based on explicit feedback from the router to the bitrate adaptation of the clients. We evaluate the effectiveness of CAAS with ECN for client’s QoE degradation due to congestion and behavior on the QoS metrics based on throughput. The simulation experiments show that the bitrate adjustment for all the clients improves QoE degradation and QoE fairness due to effective congestion avoidance.

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

1. Introduction

With the significant increase of video traffic over the Internet [1], Information Centric Network (ICN) has attracted attention as an approach to reduce the traffic of video content distribution [2]. ICN is a decentralized and content-based network that of procedures consist of the content request by Interest packets and content distribution by Data packets. ICN reduces redundant traffic for the duplicate content download by identifying a content-based name addresses assigned to Interest/Data packets. To further improve client’s QoE, adaptive streaming, which adjusts a bitrate of video content according to network congestion status, is applied for ICN [2], [3].

Adaptive streaming is widely used to deliver high QoE to clients in diverse network conditions, combined with HTTP (e.g., Dynamic Adaptive Streaming over HTTP (DASH)) [4]. In adaptive streaming, the server stores each video that is divided into segments of short play time and each video segment is encoded in multiple bitrate levels, and then the client estimates congestion status to select suitable bitrate for each video segment. The client enables to play the highest possible bitrate by a bitrate adaptation method and improves QoE due to the increase of bitrate in the play [5]. Since the bitrate adaptation is mainly responsible for QoE in adaptive streaming, many bitrate adaptation approaches have been proposed [6]–[8].

However, these studies are types of an implicit bitrate adaptation for congestion status that determines the bitrate based on the throughput measured at the client. The implicit bitrate adaptation often overestimates the bitrate for available bandwidth and causes congestion by streaming overestimated bitrate. As a result, congestion causes frequent variation of the bitrate (bitrate magnitude) and play stall due to video buffer exhaustion (stall time) in many scenarios [9]–[11], and finally degrades QoE metrics [12]. Thus, in adaptive streaming over ICN, there are problems with QoE degradation due to the implicit bitrate adaptation for congestion status [13]–[16]. Therefore, in ICN, adaptive streaming also requires control for congestion to improve QoE.

In ICN research fields, some explicit congestion control applying the idea of Explicit Congestion Notification (ECN) is proposed for better communication performance than that of implicit congestion control [17]–[19]. These approaches focused on the characteristic of ICN where routers work in a distributed manner and propose congestion control in which routers notify explicit feedback of congestion status to the clients. The client enables to avoid congestion by decreasing its transmission rate of the Interest packet. However, the idea of explicit congestion control is not well considered for adaptive streaming over ICN.

In this paper, we propose a congestion-aware adaptive streaming over ICN combined with explicit congestion notification (CAAS with ECN) for QoE, which applies the idea of ECN to an adaptive streaming application to avoid QoE degradation. The design of CAAS with ECN employs an ECN framework based on congestion detection measuring the packet queue length of each router’s interface [18], [19]. CAAS with ECN enables the router to notify an upper band of the selectable bitrate (bitrate-cap as ECN) to the client for adjusting the bitrate while local congestion is detected. In addition, the router gathers the video content information in streaming and selects the video contents to be notified of the bitrate-cap during congestion. From the information of the router, the client enables to avoid bitrate overestimation by playing the bitrate smaller than or equal to the notified bitrate-cap, and consequently avoid QoE degradation. In the simulation experiments, we implement CAAS with ECN in two representative implicit bitrate adaptation methods for congestion status estimation, the rate-based...
algorithm (RBA) and the rate and buffer (hybrid)-based algorithm (HBA) [20]. In addition, we define the bitrate-cap strategy for all the video contents and that for only the video contents of the highest bitrate, and then, we compare the effect of each bitrate-cap strategy. Finally, we evaluate an effectiveness of these bitrate-cap strategies for client’s QoE and QoS improvement.

This paper is organized as followings. Section 2 describes backgrounds and related work for composing the design of CAAS with ECN in Sect. 3. We show the evaluation of CAAS with ECN from the point of the client’s QoE metrics and QoS metrics based on throughput in Sect. 4, and conclude CAAS with ECN in Sect. 5. Finally, we discuss the CAAS with ECN for other functions of ICN in Sect. 6.

2. Backgrounds and Related Works

2.1 Adaptive Streaming Over ICN

Figure 1 shows an overview of adaptive streaming over ICN [16]. In Fig. 1, the server stores the AS contents composed of a media presentation description (MPD) and N video segments of short play time with multiple bitrate levels, and the server and routers transfer the AS contents to the AS client over ICN. MPD describes the information for initialization of adaptive streaming (e.g., audio information and available multiple bitrate levels prepared for adaptive streaming). Each video segment is encoded in multiple bitrate levels described in MPD. In the AS client, the AS application starts adaptive streaming with the video player engine and the bitrate adaptation. The video player engine includes a fixed-length video buffer which stores video segments acquired in adaptive streaming, and the AS client starts the play after the video buffer length becomes larger than a predefined threshold. The bitrate adaptation is processed according to the following procedures for AS contents download through the ICN transport with congestion control.

In the following subsections, we describe the details of the bitrate adaptation in the AS application and congestion control in the ICN transport.

In Fig. 2, the AS application first initializes information necessary by downloading MPD of the AS contents ((0) in Fig. 2). To download MPD (0-0) and (0-1) in Fig. 2, the AS client first downloads the manifest of MPD (0-0-0) and (0-0-1) in Fig. 2 and then downloads all the data chunks of MPD ((0-1-0) to (0-1-k) in Fig. 2) using a name address composed of the content information of MPD (e.g., video/MPD/manifest). As soon as completing the MPD download, the AS client starts downloading N video segments sequentially ((1) to (N) in Fig. 2). To download the 1st video segment, the bitrate adaptation in the AS application first selects the bitrate for 1st video segment (Bitrate1) based on throughput of the MPD download ((1-0) in Fig. 2). Then, the AS client starts downloading 1st video segment of Bitrate1 ((1-1) to (1-2) in Fig. 2) as the same with procedures of the MPD download using a name address composed of the content information of 1st video segment (e.g., video/1st_segment/100_kbps/manifest). In this way, the bitrate adaptation selects the bitrate based on throughput of the previous video segment download, and the AS client sequentially downloads the video segment of the bitrate as long as the video buffer is not occupied. In addition, since the Data packet for the AS content delivery may cause congestion, congestion control is installed into the ICN transport.

In the following subsections, we describe the details of the bitrate adaptation in the AS application and congestion control in the ICN transport.
2.2 Implicit Bitrate Adaptation for Congestion Status Estimation

Most of the bitrate adaptation methods process the following procedure at a video segment request: (1) estimating congestion status from the client, (2) selecting the highest possible bitrate of the video segment from given multiple bitrate levels according to estimated congestion status and (3) calculating the time interval length to request the video segment according to the remaining video buffer length.

There are two representative algorithms in the implicit bitrate adaptation for congestion status estimation: the rate-based algorithm (RBA) and the rate and buffer (hybrid)-based algorithm (HBA) [20]. RBA refers to only the average throughput of the previous segment download at the client for congestion status estimation in bitrate adaptation. Dash.js [21], which is the open source project of dynamic adaptive streaming over HTTP (DASH) [3], implements RBA as a ThroughputRule.

Algorithm 1 shows the procedures of RBA when requesting the $n$-th video segment ($segment_n$). For the bitrate adaptation of $segment_n$, RBA actually refers to the smoothed average throughput at the previous video segment download ($T_{n-1}$) using an exponentially weighted moving average (EWMA) (lines 6-7 in Algorithm 1), and selects the highest possible bitrate of $segment_n$ ($bitrate_n$) according to $T_{n-1}$ from the multiple bitrate levels described in MPD ($availableBitrateLevels$) (line 8 in Algorithm 1). After the bitrate adaptation, the time interval length to request the $segment_n$ ($l_n$) is calculated according to the remaining video buffer length ($B_n$) (line 9 in Algorithm 1).

HBA refers to both the smoothed average throughput of the previous video segment download and remaining video buffer length at the client for congestion status estimation in bitrate adaptation. Dash.js implements HBA as InsufficientBufferRule [21].

Algorithm 2 shows the procedures of HBA when requesting the $segment_n$. HBA refers to the value obtained by multiplying the $T_{n-1}$ by the buffer filling factor ($\frac{B_n}{T_{n-1}}$) and the insufficient buffer safety factor ($\alpha$) for bitrate adaptation (lines 8-14 in Algorithm 2). The high $\alpha$ leads to more aggressive HBA. $\alpha$ is 0.5 predefined at InsufficientBufferRule. As compared with RBA, the larger remaining video buffer length, the more easily $bitrate_n$ increases.

Since the reference to throughput at the start of each video segment streaming is implicit for congestion status estimation, RBA and HBA may select the overestimated bitrate for available bandwidth, and cause QoE degradation due to congestion.

2.3 Implicit/Explicit Congestion Control in the ICN Transport

The ICN transport controls Interest packet transmission for congestion control. In other words, the ICN transport reduces the transmission amount of Interest packet according to congestion detection. In ICN research fields, the methods of congestion detection are also classified into two approaches, an implicit approach and an explicit one. The implicit approach is implicit congestion detection determining the cause of congestion based on packet loss at a client, which means the client does not receive the Data packet even after a predefined time length passed since sending the Interest packet as described in Sect. 2.1. The earliest TCP-like ICN transport design, ICP [22], is the implicit approach. On the other hand, the explicit approach is ECN-like explicit congestion detection where the router explicitly notifies congestion status on the communication path to the client. Since the explicit approach enables it to notify congestion status of the network explicitly with short communication delay between the client and the router, communication performance is better than that of the implicit approach [17]–[19].

However, despite the implicit bitrate adaptation causes QoE degradation due to congestion as described in 2.2, adaptive streaming using explicit feedback of congestion status is not well considered. Then, our proposal stands for
the idea of the adaptive streaming for congestion detection using feedback of explicit congestion status based on the router’s packet queue length [18], [19] by introducing the upper band of the selectable bitrate to the bitrate adaptation in the client.

3. A Congestion-Aware Adaptive Streaming over ICN Using Explicit Congestion Notification

3.1 Overview

To avoid client’s QoE degradation, we propose the congestion-aware adaptive streaming over ICN combined with explicit congestion notification (CAAS with ECN). CAAS with ECN follows the prescribed operation of the adaptive streaming over ICN, and further introduces an upper band of the selectable bitrate (bitrate-cap) to the bitrate adaptation for adjusting the bitrate during congestion. Figure 3 shows an overview of CAAS with ECN. In Fig. 3, the server stores the AS contents composed of MPD and N video segments, and the server and routers transfer the AS contents to the AS client over ICN. In the AS client, the AS application starts adaptive streaming according to the following procedures; as soon as the initialization for adaptive streaming is successful ((0) in Fig. 3), the AS application starts streaming of the video segments with the video player engine, the bitrate adaptation and the explicit bitrate capper for congestion ((1) to (N) in Fig. 3). In addition to the prescribed procedure of the video player engine and the bitrate adaptation, the explicit bitrate capper for congestion introduces the following procedure for the bitrate adaptation.

As soon as the AS application starts adaptive streaming of N video segments, every time a Data packet arrives at a network interface of the router, the ECN control in the router checks congestion status of each equipped network interface. Then, the ECN control notifies ECN Data with the bitrate-cap for the content. Thus, the ECN control finally notifies the bitrate-cap to the bitrate adaptation according to the selection policy with the ContentList. As soon as ECN Data with the bitrate-cap arrives at the AS application layer, the explicit bitrate capper for congestion forces the bitrate adaptation to select a bitrate smaller than or equal to the notified bitrate-cap.

In the following subsections, we describe the details of the ECN control in the routers and the explicit bitrate capper for congestion in the AS client, which are the functions unique to CAAS with ECN in Fig. 3.

3.2 The ECN Control in the Router on Communication Path

In CAAS with ECN, the ECN control works on each network interface in each router on a communication path. The network interfaces are equipped with a packet queue and transfers Interest/Data packets from the packet queue. To start the ECN control for each network interface, the router processes the following procedure for initialization of the ECN control.

The initialization of the ECN control is the MPD acquisition to get original multiple bitrate levels as a source of the bitrate-cap selection. The router gets MPD from the server in accordance with the initialization by MPD acquisition in the AS client ((0) in Fig. 3).

As soon as the initialization of the ECN control is successful, the router starts the ECN control for Interest/Data communication. The ECN control consists of the update of ContentList and the ECN procedure according to congestion detection in the following interest/Data communication.

Figure 4 shows the packet sequence with the ECN control between the AS client, the router with the ECN control and the server.

In Fig. 4, the AS client first downloads MPD for the initialization ((0) in Fig. 4). As soon as getting MPD ((0) in Fig. 4), the AS client downloads 1st video segment with the bitrate adaptation. In addition, the update of ContentList is processed on each network interface when the Interest/Data packet arrives at the router ((1-1-1), (1-1-3), (1-2-1) and (1-2-3) in Fig. 4). In detail, the ECN control first collects the set of the video name, the video segment number, and the bitrate from the name address of the arrived Interest/Data packet. Then, the ECN control updates the ContentList with the set newly collected from the Interest/Data packet. If the newly collected set is unique in the ContentList, the ECN control updates the ContentList with the newly collected set and the timestamp. If the newly collected set is not unique, the ECN control only updates the timestamp of the existing set in the ContentList that overlaps the newly collected set. In addition, after the above update procedures, the ECN control deletes the expired set passing a fixed time interval.
Fig. 4 Interest/Data communication in congestion-aware adaptive streaming over ICN combined with explicit congestion notification (CAAS with ECN).

from its timestamp in the ContentList.

The ECN procedure according to congestion detection is scheduled after the update of ContentList every time the Data packet arrives on the network interface ((1-1-3) and (1-2-3) in Fig. 4). Algorithm 3 describes the ECN procedure according to congestion detection.

In Algorithm 3, the router notifies the bitrate-cap when congestion is checked by comparing the packet queue length $Q_{m}$ and congestion threshold $C_{th}$ (lines 8-11 in Algorithm 3). If congestion is detected, $bitrate_{cap}$ is selected with reference to the arrived Data packet’s $bitrate_{data}$ according to the selection policy with the ContentList$_{m}$ for explicit congestion notification (lines 12-29 in Algorithm 3). In addition, we define the following three selection policies for bitrate-cap notification for clients downloading the content.

- **Keep policy (lines 12-14 in Algorithm 3):** This policy sets a keep flag to the Data packets as a bitrate-cap to quickly avoid increment of the bitrate during congestion for all the video contents. In this policy, all the clients are forced to select a bitrate smaller than or equal to the previous bitrate for congestion avoidance.

- **Top-reduction policy (lines 15-29 in Algorithm 3):** This policy selects a bitrate-cap from the multiple bitrate levels in MPD (acquired at the initialization for ECN procedure) to quickly reduce only the highest bitrate in streaming contents during congestion. In detail, while congestion is detected (line 11 in Algorithm 3), this policy selects the bitrate-cap for only the highest bitrate of streaming contents (ContentList) (line 16-17 in Algorithm 3), which is decremented by 1 level bitrate from the highest bitrate. For instance, if MPD records multiple bitrate levels (line 2 in both Algorithm 1 and Algorithm 2) of {100kbps, 200kbps, 300kbps} and the highest bitrate in the ContentList is 300kbps, the bitrate-cap is 200kbps. After $bitrate_{cap}$ selection, if another bitrate-cap has already been set in the Data packet, the lower bitrate-cap is set to the Data packet for the other router’s network interface with heavier congestion on communication path (lines 18-23 in Algorithm 3). Finally, the selected bitrate-cap is set to the Data packet of the highest bitrate (line 27 in Algorithm 3). In this policy, the clients playing the highest bitrate are forced to select a bitrate smaller than the highest bitrate for congestion avoidance at the next play of video segment.

- **Top-reduction + Keep policy (lines 12-29 in Algorithm 3):** This policy is a combination of the top-reduction policy and the keep policy. In this policy, the clients playing the highest bitrate of contents follows the top-reduction policy, and the other clients follow the keep policy.

After the bitrate-cap/keep flag is selected, the ECN control finally sends ECN Data with the bitrate-cap/keep
flag to the AS client ((1-1-4) and (1-2-k) in Fig. 4) according to congestion detection. In this way, the AS client sequentially downloads N video segments with the ECN control ((1) to (N) in Fig. 4).

3.3 The Explicit Bitrate Capper for Congestion in the AS Client

After the AS client starts first video segment streaming, the explicit bitrate capper for congestion updates the availableBitrateLevels in bitrate adaptation (line 2 in Algorithm 2 and Algorithm 3) with the notified bitrate-cap before the bitrate adaptation for next video segment.

When ECN Data with the bitrate-cap/keep flag arrives at the client, the availableBitrateLevels is immediately updated according to the notified bitrate-cap before congestion detection. In this way, the AS client sequentially downloads N video segments with the ECN control ((1-1-4) and (1-2-k) in Fig. 4) according to the notified bitrate-cap. If the regular Data arrives, the availableBitrateLevels is immediately updated with the original availableBitrateLevels in MPD acquired at the initialization of adaptive streaming.

After the update of availableBitrateLevels, the AS client adjusts the bitrate for congestion avoidance with short communication delay as compared to the implicit control. Thus, the explicit bitrate capper for congestion prevents the AS client from increasing the bitrate while the router detects congestion.

4. Evaluation

We evaluate how the bitrate adjustment during congestion in congestion-aware adaptive streaming over ICN combined with explicit congestion notification (CAAS with ECN) improves QoE degradation due to congestion and behavior on the QoS metrics based on throughput in the adaptive streaming through amus-ndnSIM, the event-driven ICN simulator for the adaptive streaming [23].

Figure 5 shows the evaluation topology composed of 5 (white square) servers, 20 routers (black circle) and 25 clients (white circle) [5]. The link speed between the router and the client is 10 Mbps, and the link speed between the router and the router is 100 Mbps. Each server stores the adaptive streaming contents (video segments in 20 bitrate levels and MPD) for 100 kinds of a video. Each router is equipped with basic functionalities in ICN static routing. The ECN control (Cth = 0.1, update time interval of ContentList = 250 ms) is installed on all routers for CAAS with ECN. The adaptive streaming application in all the clients employ a video player engine with a video buffer with 15 video segments, and selects one of the bitrate adaptations in the play and the explicit bitrate capper during congestion. Each client starts playing video as soon as there is more than one video segment in the video buffer.

In the simulation, we set up an adaptive streaming scenario with 100 kinds of video content with playing time length of 600 seconds, composed of 300 video segments (a time length of each video segment is 2 sec) encoded in 20 bitrate levels (from 45 kbps up to 4.3 Mbps) [24]. A client repeats playing one video content selected from 100 kinds of video content according to the popularity of Zipf’s law (α = 1.0) for 5 hours (18000 sec). A video content request interval in each client follows an exponential distribution with an average of 600 seconds.

For comparison, we evaluate 8 methods based on the aforementioned adaptive streaming methods, adaptive streaming over ICN employing RBA (Normal RBA), CAAS with ECN employing RBA and the top-reduction policy (CAAS RBA, top-reduction), the keep policy (CAAS RBA, keep), and the top-reduction + keep policy (CAAS RBA, top-reduction + keep) as a bitrate-cap selection policy, adaptive streaming over ICN employing HBA (Normal HBA), CAAS with ECN employing HBA and the top-reduction policy (CAAS HBA, top-reduction), the keep policy (CAAS HBA, keep), and the top-reduction + keep policy (CAAS HBA, top-reduction + keep).

4.1 Metrics for Evaluation

To evaluate QoE, we use QoELin scoring [25]–[27]. QoELin allows the quantitative assessment of QoE based on log information of an adaptive streaming application in a client. QoELin is a linear combination of three QoE metrics, a bitrate metric, a bitrate magnitude metric and a stall time metric defined in Eq. (1).

\[
\text{QoELin} = \sum_{n=1}^{N} q(R_n) - \lambda \sum_{n=1}^{N-1} |q(R_{n+1}) - q(R_n)| - \mu \sum_{n=1}^{N} b_n - \mu_D D
\]

\[(QoELin) < QoELin \leq (4.3 * N)\] is set to the comprehensive QoE score for a play of video content consisting of N video segments.

\[q(R_n) \{ 0.045 \leq q(R_n) \leq 4.3 \} \] of the first term in QoELin is a utility function q of the bitrate (Mbps) of n-th video segment, \[R_n \{ 0.045 \leq R_n \leq 4.3 \} \]. The first term represents a positive QoE metric, “the bitrate metric.” The utility function q is an identity function according to a definition in Ref. [26].

\[q(R_{n+1}) - q(R_n) \{ 0 \leq q(R_{n+1}) - q(R_n) \leq (4.3 - 0.045) \} \] of the second term in QoELin is the difference between the current bitrate metric: \(q(R_{n+1})\) and the previous bitrate: \(q(R_n)\). The second term represents a negative QoE metric, “the bitrate magnitude metric.”
{b_n} | 0 ≤ b_n < ∞ of the third term in QoE_{lin} is a stall time length due to video buffer exhaustion (sec) at a n-th video segment play. In addition, {D} | 0 ≤ D < ∞ is a sum of stop time length (sec) due to the initialization of adaptive streaming. The third and fourth terms together represent a negative QoE metric, “the stall time metric.”

{λ, μ, μ_D} are non-negative weighting factors for the bitrate magnitude metric and the stall time metric set against the bitrate metric. We use the predefined values, {λ = 1, μ = 4.3, μ_D = 4.3} in [26].

In addition, to evaluate a QoE fairness, we use the Jain’s Fairness Index [28] based on QoE_{lin} (FIQoE). FIQoE is defined in Eq. (2).

\[
FI_{QoE} = \left( \frac{\sum_{n=1}^{N} QoE_{lin,normalized}^{\text{normalized}}}{N \times \sum_{n=1}^{N} (QoE_{lin,normalized})} \right)^2
\]

\{FI_{QoE} | 0 ≤ FI_{QoE} ≤ 1\} is set for measuring the fairness of \{QoE_{lin,normalized} | 0 ≤ QoE_{lin,normalized} ≤ 1\} that is the QoE_{lin} scaled to [0, infinity] range by adding a positive constant. The QoE_{lin,normalized} is the QoE_{lin,normalized} of n-th play of video content.

To evaluate behavior on the QoS metrics based on throughput, we use the following metrics, a percentage of congested video segments (Percent_c), efficiency metrics of bitrate versus throughput, and fairness of throughput per video content. The Percent_c is defined in Eq. (3) [29].

\[
Percent_c = \frac{\text{Num of congested video segments}}{\text{Num of video segments}} \times 100
\]

The congested video segments are video segments where the selected bitrate is greater than the experienced throughput.

\{Inefficiency_n | 0 ≤ Inefficiency_n < ∞\} for n-th video segment is defined in Eq. (4) [29].

\[
Inefficiency_n = \frac{|R_n - W_n|}{W_n}
\]

\{W_n | 0 ≤ W_n < ∞\} represents the n-th video segment’s experienced throughput (Mbps). Since Eq. (4) shows how far the bitrate departs from the experienced throughput of the video segment, the inefficiency metric represents the accuracy of bitrate adaptation control.

4.2 Evaluation of QoE Metrics

In this subsection, we evaluate the effect of the bitrate-cap strategies in CAAS with ECN on the QoE metrics of the bitrate metric, the bitrate magnitude metric, the stall time metric, and QoE_{lin}. Table 1 and Fig. 6 show the comparison of an average and a cumulative distribution function (CDF) of QoE metrics in QoE_{lin} per video content in 8 methods.

First, we evaluate a comprehensive QoE metric, QoE_{lin}. In Table 1, CAAS increases 38.71% in average QoE_{lin} in RBA with the top-reduction policy, 141.61% in

| Table 1 | Average QoE metrics per video content. |
|---------|----------------------------------------|
| Method             | QoE_{lin} | Bitrate | Bitrate magnitude | Stall time |
| Normal RBA         | 188.65    | 731.39  | 256.72            | 286.02     |
| CAAS RBA, top-reduction | 261.67    | 673.9   | 250.12            | 162.05     |
| CAAS RBA, keep      | 455.8     | 637.54  | 146.93            | 34.8       |
| CAAS RBA, top-reduction + keep | 429.8 | 626.47  | 166               | 30.67      |
| Normal IBA         | 368.04    | 730.94  | 296.87            | 66.03      |
| CAAS IBA, top-reduction | 358.22    | 703.88  | 291.46            | 54.21      |
| CAAS IBA, keep      | 543.81    | 716.72  | 152.73            | 20.18      |
| CAAS IBA, top-reduction + keep | 520.87 | 713.1   | 174.62            | 17.61      |

Fig. 6 Comparison of CDF for QoE metrics per video content. (a) QoE linear combination metric (QoE_{lin}), (b) bitrate metric, (c) bitrate magnitude metric, (d) stall time metric.

RBA with the keep policy, and 127.83% in RBA with the top-reduction + keep policy as compared to that of the normal RBA, respectively. Accordingly, in RBA, while all the policies improve QoE_{lin} due to congestion avoidance, the top-reduction policy is inferior in QoE_{lin} as compared to
the keep policy and the top-reduction + keep policy. Therefore, the bitrate adjustment for only the highest bitrate client is small for congestion avoidance because the other clients (for lower than the highest bitrate) increase the bitrate due to the implicit bitrate adaptation and takes time to mitigate congestion. As a result, the CAAS RBA, top-reduction method reduces $QoE_{lin}$ as compared to the CAAS RBA, keep method (in Fig. 6(a)), which forces all the clients to quickly avoid increment of the bitrate during congestion. In addition, in Fig. 6(a), the CAAS RBA, keep method is better where $QoE_{lin}$ is around 500 as compared to the CAAS RBA, top-reduction + keep method. This is because the top-reduction + keep policy reduces $QoE_{lin}$ by increasing the variation of bitrate due to the highest bitrate reduction (in Fig. 6(c)). On the other hand, in Table 1, CAAS reduces 2.67% in average $QoE_{lin}$ in HBA with the top-reduction policy, and increases 47.76% in average $QoE_{lin}$ in HBA with the keep policy, and 41.53% in HBA with the top-reduction + keep policy compared to that of the normal HBA, respectively. Therefore, in HBA, the top-reduction policy does not affect $QoE_{lin}$ due to a small effect of the highest bitrate adjustment to avoid congestion. In addition, the top-reduction + keep policy slightly reduces $QoE_{lin}$ due to the large bitrate variation caused by the highest bitrate reduction as compared to the keep policy (in Fig. 6(c)).

For a deeper analysis of $QoE_{lin}$, we evaluate each effect of QoE metrics, the bitrate metric, the bitrate magnitude metric, and the stall time metric in $QoE_{lin}$.

First, we evaluate the bitrate metric. In Table 1, CAAS reduces 7.86% in the average bitrate metric in RBA with the top-reduction policy, 12.83% in RBA with the keep policy, and 14.35% in RBA with the top-reduction + keep policy compared to that of the normal RBA, respectively. Therefore, in RBA, all the policies reduce the bitrate metric due to the bitrate adjustment during congestion by the selection policy. On the other hand, in Table 1 and Fig. 6(b), all the methods of CAAS HBA result in almost the same behavior of the bitrate metric as compared to the normal HBA because the video buffer length affects the bitrate adaptation of each video segment in HBA (line 13 in Algorithm 2).

Second, we evaluate the bitrate magnitude metric. In Table 1, CAAS reduces 2.57% in the average bitrate magnitude metric in RBA with the top-reduction policy, 42.77% in RBA with the keep policy, and 35.34% in RBA with the top-reduction + keep policy compared to that of the normal RBA, respectively. For HBA, Table 1 shows CAAS reduces 1.82% in the average bitrate magnitude metric with the top-reduction policy, 48.55% with the keep policy, and 41.18% with the top-reduction + keep policy as compared to that of the normal RBA, respectively. Accordingly, in both RBA and HBA, the top-reduction policy does not affect the bitrate magnitude metric because it enables the clients to select lower than the highest bitrate to increase the bitrate even during congestion and takes time to mitigate congestion. As a result, the top-reduction policy causes large bitrate variation due to congestion as compared to the keep methods (in Fig. 6(c)). In addition, in Fig. 6(c), since the keep policy does not change the bitrate of all the clients during congestion, the bitrate magnitude metric in the keep policy always results better than that of the keep + top-reduction policy in both RBA and HBA.

Third, we evaluate the stall time metric. In Table 1, CAAS reduces 43.34% in the average stall time in RBA with the top-reduction policy, 87.83% in RBA with the keep policy, and 89.28% in RBA with the top-reduction + keep policy compared to that of the normal RBA, respectively. For HBA, CAAS reduces 17.9% in the average stall time metric in HBA with the top-reduction policy, 69.44% in HBA with the keep policy, and 73.33% in HBA with the top-reduction + keep policy compared to that of the normal HBA, respectively. Therefore, in both RBA and HBA, all the policies are effective for the stall time metric because the bitrate adjustment enables the clients to avoid the stall time with ECN under all the policies. In addition, in Fig. 6(d), the top-reduction + keep methods are always slightly better in the stall time metric compared to the keep policy in both RBA and HBA. This is because the highest bitrate clients avoid the increase of stall time under the top-reduction + keep policy by reducing the bitrate more than that of the keep policy in both RBA and HBA.

Consequently, CAAS with the keep policy, which all the clients select lower than or equal to the current bitrate during congestion, improves QoE (the bitrate magnitude metric and the stall time metric) in both RBA and HBA as compared to the top-reduction policy because all the clients effectively avoid congestion with ECN. In the comparison of the keep policy and the top-reduction + keep policy, since the top-reduction + keep policy increases the variance of bitrate during congestion, it is excessive control for congestion due to the increase of the bitrate magnitude metric as compared to the keep policy. However, since the top-reduction + keep policy is most effective for the stall time metric in all the policies, the bitrate reduction combined with the keep policy is considerable in case of streaming to minimize the stall time.

In addition, Table 2 shows the fairness of $QoE_{lin}$ for each video content using the fairness index, the $FI_{QoE}$. CAAS slightly improve the $FI_{QoE}$ in the both of RBA and HBA. Especially, the keep policy is effective for the $FI_{QoE}$. Since CAAS reduces the variance of $QoE_{lin}$ as in Fig. 6(a), the CAAS with the keep policy and the top-reduction + keep policy reduce the number of video contents in negative QoE for the both RBA and HBA. As a result, the $FI_{QoE}$ is improved.

| Method                        | $FI_{QoE}$ |
|-------------------------------|------------|
| Normal RBA                    | 0.94       |
| CAAS RBA, top-reduction       | 0.96       |
| CAAS RBA, keep               | 0.99       |
| CAAS RBA, top-reduction + keep| 0.99       |
| Normal HBA                    | 0.96       |
| CAAS HBA, top-reduction       | 0.97       |
| CAAS HBA, keep               | 0.98       |
| CAAS HBA, top-reduction + keep| 0.99       |
Table 3  Percentc and average efficiency metrics of bitrate versus throughput per video segment.

| Method                  | Percentc | Inefficiency | Inefficiencyc | Inefficiencync |
|-------------------------|----------|--------------|---------------|---------------|
| Normal RBA              | 45.34    | 0.43         | 0.22          | 0.22          |
| CAAS RBA, top-reduction | 39.67    | 0.45         | 0.18          | 0.27          |
| CAAS RBA, keep          | 30.97    | 0.42         | 0.11          | 0.3           |
| CAAS RBA, top-reduction + keep | 30.24 | 0.44         | 0.1           | 0.33          |
| Normal HBA              | 40.91    | 0.50         | 0.23          | 0.28          |
| CAAS HBA, top-reduction | 40.74    | 0.50         | 0.23          | 0.28          |
| CAAS HBA, keep          | 39.79    | 0.54         | 0.24          | 0.3           |
| CAAS HBA, top-reduction + keep | 39.8  | 0.55         | 0.25          | 0.3           |

4.3 Evaluation of QoS Metrics Based on Throughput

In this subsection, we evaluate the effect on QoS metrics based on throughput in CAAS with ECN, the Percentc, and the efficiency metrics of bitrate versus throughput. Table 3 and Fig. 7 show the comparison of the Percentc and the average efficiency metrics per video segment, and CDF of the efficiency metric. First, we evaluate a percentage of congested video segments, the Percentc. Table 3 shows that CAAS reduces 12.51% in the Percentc in RBA with the top-reduction policy, 31.69% in RBA with the keep policy, and 33.3% in RBA with the top-reduction + keep policy as compared to that of the normal RBA, respectively. Therefore, in RBA, the keep policy is better in congestion avoidance compared to the top-reduction policy. This is because the keep policy applies the ECN control to all the clients and all the clients quickly adjust the bitrate for congestion avoidance with short communication delay between the clients and the router. On the other hand, all the methods of CAAS HBA do not affect the Percentc as compared to that of the normal HBA. In addition, in Fig. 6 (d), all the methods of CAAS HBA avoid the stall time as compared to normal HBA. This is because the video buffer length is increased due to stall time avoidance and the increased video buffer causes the client to aggressively select a higher bitrate than the throughput (line 13 in Algorithm 2). As a result, all the methods of CAAS HBA do not affect the bitrate as shown in Fig. 6 (b).

Second, we evaluate the efficiency metric, the Inefficiency per video segment, which represents the accuracy of bitrate adaptation control. In Table 3 and Fig. 7 (a), CAAS does not affect the Inefficiency in both RBA and HBA.

For a deeper analysis of Inefficiency, we divide the efficiency metrics in the two cases, the efficiency metric during congestion, an Inefficiencyc, and an Inefficiencync during non-congestion, and evaluate each efficiency metric. The

\[
\text{Inefficiency}_n = \begin{cases} \frac{|R_n - W_n|}{W_n} & \text{if } R_n > W_n \\ \frac{|R_n - W_n|}{W_n} & \text{if } R_n \leq W_n \end{cases}
\] (5)

First, we evaluate the efficiency metric during congestion, the Inefficiencyc. In Table 3, CAAS reduces 50% in RBA with the keep policy and the top-reduction + keep policy, and 18.18% in RBA with the top-reduction policy as compared to that of the normal RBA, respectively. Therefore, all the policies improve the accuracy of RBA during congestion by quickly adjusting the bitrate during congestion. In addition, in RBA, the keep policy reduces the Inefficiencyc compared to the top-reduction policy because the keep policy forces all the clients to quickly adjust the bitrate during congestion under the ECN control and avoid excessive high bitrate streaming for throughput. On the other hand, in Table 3 and Fig. 7 (b), all the methods of CAAS HBA have a small effect on the Inefficiencyc as compared to the normal HBA because these clients properly select the higher bitrate during non-congestion due to the video buffer length increased by stall time avoidance (in Fig. 6 (d)).

Second, we evaluate the efficiency metric during non-congestion, the Inefficiencync. Table 3 shows that CAAS increases 50% in RBA with the keep policy and the top-reduction + keep policy, and 22.73% in RBA with the top-reduction policy as compared to that of the normal RBA, respectively. Therefore, all the policies in CAAS RBA un-
Table 4 Fairness index of throughput per video content ($F_{\text{Throughput}}$).

| Method                  | $F_{\text{Throughput}}$ |
|-------------------------|-------------------------|
| Normal RBA              | 0.83                    |
| CAAS RBA, top-reduction | 0.84                    |
| CAAS RBA, keep          | 0.83                    |
| CAAS RBA, top-reduction + keep | 0.84            |
| Normal HBA              | 0.84                    |
| CAAS HBA, top-reduction | 0.84                    |
| CAAS HBA, keep          | 0.84                    |
| CAAS HBA, top-reduction + keep | 0.84            |

derestimates the bitrate for the throughput because the bitrate adjustment in the ECN control forces RBA to select the smaller bitrate to avoid congestion even if the video buffer length is sufficient. As a result, the clients reduce the bitrate during non-congestion and reduces the $\text{Percent}_c$ in Table 3. In addition, the keep policy increases the $\text{Inefficiency}_c$ by selecting the smaller bitrate as compared to the top-reduction policy. On the other hand, Table 3 and Fig. 7 (c) show that all the methods of CAAS HBA do not affect the $\text{Inefficiency}_c$ as compared to the normal HBA. This is because the bitrate is increased due to the video buffer length increase by stall time avoidance. As a result, all the methods in CAAS HBA do not affect the $\text{Percent}_c$ in Table 3.

In addition, we evaluate the fairness index [28] of throughput per video content, $F_{\text{Throughput}}$. Table 4 shows $F_{\text{Throughput}}$, and we can see that all the methods in CAAS do not affect on $F_{\text{Throughput}}$. In other words, all the policies improve QoE without the biased use of network resources.

5. Conclusion

In this paper, we propose the Congestion-Aware Adaptive Streaming over ICN combined with ECN (CAAS with ECN) to improve QoE by adjusting the bitrate during congestion. The client enable to avoid congestion by quickly adjusting the bitrate with ECN. Through the simulation experiments, the bitrate adjustment for all the video contents is effective for congestion avoidance because all the clients avoid congestion with ECN. On the other hand, the bitrate adjustment for the highest bitrate of contents is not enough because congestion is caused by the clients below the highest bitrate because congestion status is not explicitly notified by ECN. As a result, the bitrate adjustment for all the video contents improves QoE and QoE fairness without reducing the accuracy of bitrate adaptation control. Therefore, we claim that the bitrate adjustment during congestion for all the clients is effective for congestion avoidance in adaptive streaming.

6. Discussion and Future Work

In this section, we discuss the effect of routing and caching that are not well investigated in this paper and show the future direction of CAAS with ECN for those features.

6.1 Adaptive Name-Based Routing

ICN natively supports multipath name-based forwarding of Interest/Data packet [30] and enables the router to adaptively route the Interest/Data packet according to congestion [31]. This behavior of adaptive name-based routing makes the following point of CAAS with ECN considerable because CAAS with ECN assumes the static name-based routing. In CAAS with ECN, the ECN control must acquire the media presentation description (MPD) at the initialization of adaptive streaming ((0) in Fig. 3) for the source of bitrate-cap. In other word, the ECN control does not work when the Interest packet for the uninitialized video content arrives due to adaptive name-based routing. Accordingly, when such Interest packet arrives at the router, the ECN control needs to immediately acquire MPD for initialization of the newly requested video content. Therefore, we are planning to introduce a method of the ECN control initialization for the newly requested video content at the router.

6.2 In-Network Caching

ICN natively supports in-network caching of Data packet [30] and enables the router to avoid congestion by reducing redundant traffic for the same content [32]. In-network caching enables the router to temporarily cache chunks for video contents and returns the cached chunks to the client when requested. In other words, the client often downloads the video using a part of downstream path from the cached router. Therefore, cache hit on the router makes the effect of ECN control incorrect because the ECN control does not always feedback the end-to-end bottleneck information. To support the ECN control for in-network caching effectively, we need to design the ECN control when the router receives the Interest packet of the excessive high bitrate for end-to-end bottleneck due to cache miss in downstream.

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 J. Sel. Areas Commun., vol.34, no.8, pp.2130–2140, 2016.
[4] ISO-IEC 23009-1:2012 Information Technology, Dynamic Adaptive Streaming over HTTP (DASH).
[5] H.T.T. Tran, N.P. Ngoc, C.T. Pham, Y.J. Jung, and T.C. Thang, "A Subjective Study on QoE of 360 Video for VR Communication," Proc. IEEE 19th International Workshop on Multimedia Signal Processing, pp.1–6, 2017.
[6] K. Sipeti, R. Urgaonkar, and R.K. Sitaraman, “BOLA: Near-optimal Bitrate Adaptation for Online Videos,” Proc. IEEE International Conference on Computer Communications, INFOCOM, pp.1–9, 2016.
[7] Z. Li, X. Zhu, J. Gahn, R. Pan, H. Hu, A.C. Begen, and D. Oran, “Probe and adapt: Rate adaptation for http video streaming at scale,” IEEE J. Sel. Areas Commun., vol.32, no.4, pp.719–733, 2014.
[8] X. Yn, A. Jindal, V. Sekar, and B. Sinopoli, “A Control-Theoretic Approach for Dynamic Adaptive Video Streaming over HTTP.”
[9] R. Houdaille and S. Gouache, “Shaping http adaptive streams for a better user experience,” ACM MMSys, pp.1–9, 2012.

[10] S. Akhshabi, S. Narayanaswamy, A.C. Begem, and C. Dovrolis, “An experimental evaluation of rate-adaptive video players over http,” Signal Processing: Image Communication, vol.27, no.1, pp.271–287, 2012.

[11] 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, 2011.

[12] 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, 2012.

[13] D. Nguyen, J. Jin, and A. Tagami, “Cache-Friendly Streaming Bitrate Adaptation by Congestion Feedback in ICN,” Proc. 3rd ACM Conference on Information-Centric Networking, pp.71–76, 2016.

[14] C. Mueller, S. Lederer, and C. Timmerer, “A proxy effect analysis and fair adaptation algorithm for multiple competing Dynamic Adaptive Streaming over HTTP clients,” Proc. Visual Communications and Image Processing, pp.1–6, 2012.

[15] D.H. Lee, C. Dovrolis, and A.C. Begen, “Caching in http adaptive streaming: Friend or foe?,” Proc. NOSSDAV, pp.31–36, 2014.

[16] M. Coniti, 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, 2018.

[17] 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.

[18] K. Schneider, C. Yi, B. Zhang, and L. Zhang, “A Practical Congestion Control Scheme for Named Data Networking,” Proc. ACM ICN, pp.21–30, 2016.

[19] 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, 2019.

[20] J. Kua, G. Armitage, and P. Branch, “A survey of rate adaptation techniques for dynamic adaptive streaming over HTTP,” IEEE Commun. Surveys Tuts., vol.19, no.3, pp.1842–1866, 2017.

[21] DASH Industry Forum. https://github.com/Dash-Industry-Forum/dash.js.

[22] 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.

[23] B. Rainer, C. Kreuzberger, D. Posch, and H. Hellwagner, Democ: amus-ndnSIM – adaptive multimedia streaming simulator for ndn.

[24] MPEG-DASH at ITEC/AAU, http://dash.itec.aau.at

[25] 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, 2011.

[26] Z. Akhter, et al., “Oboe: Auto-tuning Video ABR Algorithms to Congestion status,” Proc. ACM SIGCOMM, pp.44–58, 2018.

[27] H. Mao, R. Netravali, and M. Alizadeh, “Neural Adaptive Video Streaming with Pensieve,” Proc. ACM SIGCOMM, pp.197–210, 2017.

[28] D.-M. Chiu and R. Jain, “Analysis of the increase and decrease algorithms for congestion avoidance in computer networks,” Computer Networks and ISDN Systems, vol.17, no.1, pp.1–14, 1989.

[29] W.U. Rahman, C.S. Hong, and E. Huh, “Edge Computing Assisted Joint Quality Adaptation for Mobile Video Streaming,” IEEE Access, vol.7, no.1, pp.129082–129094, 2019.

[30] K. Pentikousis, B. Ohlman, D. Corujo, G. Boggia, G. Tyson, E. Davies, A. Molinaro, and S. Eum, “Information-Centric Networking: Baseline Scenarios,” RFC 7476, 2015.

[31] R. Yongmao, L. Jun, S. Shanshan, L. Lingling, W. Guodong, and Z. Beichuan, “Congestion control in named data networking – A survey,” Computer Communications, vol.86, pp.1–11, 2016.

[32] D. Nguyen, K. Sugiyama, and A. Tagami, “Congestion price for cache management in information-centric networking,” Proc. INFOCOM WKRSHP, pp.1–6, 2015.

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/IM2M 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 1998, 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.