Proactive Content Caching for Mobile Video Utilizing Transportation Systems and Evaluation Through Field Experiments

Kenji Kanai, *Member, IEEE*, Takeshi Muto, Jiro Katto, *Member, IEEE*, Shinya Yamamura, Tomoyuki Furutono, Takafumi Saito, Hirohide Mikami, *Member, IEEE*, Kaoru Kusachi, Toshitaka Tsuda, *Fellow, IEEE*, Wataru Kameyama, *Member, IEEE*, Yong-Jin Park, *Member, IEEE*, and Takuro Sato, *Fellow, IEEE*

***Abstract*— In order to provide high-quality and highly reliable video delivery services for mobile users, especially train passengers, we propose a proactive content caching scheme that uses transportation systems. In our system, we place content servers with cache capability [e.g., content centric networking/named data networking (CCN/NDN)] in every train and station. Video segments encapsulated by MPEG-Dynamic Adaptive Streaming over HTTP (MPEG-DASH) are distributed and pre-cached by the station servers before the trains arrive at the stations. The trains receive content via high-speed wireless transport, such as wireless LANs or millimeter waves, when they stop at the stations. We developed prototype systems based on hypertext transfer protocol and CCN/NDN protocol, evaluate their performance through two field experiments that uses actual trains, and compare with traditional video streaming over cellular networks. Such evaluations indicate that our system can achieve high-quality video delivery without interruption for up to 50 users simultaneously.**

***Index Terms*— Proactive content caching, mobile video, MPEG-DASH, CCN/NDN, transportation systems.**

I. INTRODUCTION

PROVIDING robust video delivery services with efficient wireless resource usage is important for mobile users and carriers. According to Cisco's forecast [1], mobile data traffic in 2019 is expected to increase approximately 7 times more compared with the data for 2014. In particular, mobile video will have the highest growth rate than any other mobile applications.

However, video delivery over current cellular networks is challenging because bandwidth is limited and fluctuates depending on cross traffic and radio interference [2]. These lead to serious impact on the Quality of Experience (QoE), such as re-buffering (i.e., video freezing) and decreasing video quality [3], [4].

To alleviate this problem, we propose a proactive content caching scheme that utilizes transportation systems to provide high quality and highly reliable video delivery with efficient wireless resource use [5]. We aim at turning transportation vehicles into radio base stations and mobile cache servers. This system can contribute to reducing cellular traffic because users can connect to the cache server inside a vehicle via wireless LANs. Our approach consists mainly of three steps. First, the requested content is divided into several segments which are delivered separately to relay points, such as train stations and/or bus stops, according to the vehicle's schedule. Second, the transportation vehicle receives the distributed content segments at relay points via high-speed wireless transport schemes, such as wireless LANs or millimeter waves. Finally, the transportation vehicle streams the received content segments to mobile users inside the vehicle via wireless LANs. The key function is a delivery scheduler that we call a "smart scheduler." This scheduler determines content quality and the amount of content segments encapsulated by MPEG-DASH [6], and also selects delivery locations and timing.

We firstly developed two prototypes using HyperText Transfer Protocol (HTTP) and Content Centric Networking (CCN) [7]. We also performed the first field experiment using a train in a factory railroad in February 2014. The evaluations indicated that our system can provide higher quality and more reliable video delivery than traditional video streaming over cellular networks [5]. However, these evaluations were limited to a small scale (i.e., a single user was assumed) and some operations were performed manually. In addition, a link utilization of the first prototypes is insufficient.

Manuscript received May 29, 2015; revised December 4, 2015; accepted February 16, 2016. Date of publication June 6, 2016; date of current version August 11, 2016. This work was supported in part by the Ministry of Internal Affairs and Communications through the Promotion of Advanced Communication Applications Development Project and in part by the European Union Seventh Framework Programme within the National Institute of Information and Communications Technology of Japan through the Green Information Centric Networking Project. *(Corresponding author: Kenji Kanai.)*

K. Kanai, T. Muto, J. Katto, W. Kameyama, Y.-J. Park, and T. Sato are with the Graduate School of Fundamental Science and Engineering, Waseda University, Tokyo 169-8555, Japan (e-mail: kanai@katto.comm.waseda.ac.jp; t_muto@katto.comm.waseda.ac.jp; katto@katto.comm.waseda.ac.jp; wataru@waseda.jp; yjp@ieee.org; t-sato@waseda.jp).

S. Yamamura and T. Furutono are with Fujitsu Kyushu Network Technologies Ltd., Fukuoka 814-8588, Japan (e-mail: yamamura.shinya@jp.fujitsu.com; t.furutono@jp.fujitsu.com).

T. Saito, H. Mikami, K. Kusachi, and T. Tsuda are with the Global Information and Telecommunication Institute, Waseda University, Tokyo 169-8555, Japan (e-mail: tk-saito@cybersoken.com; h.mikami@cybersoken.com; k-kusachi@lsi-j.co.jp; tsuda-toshitaka@nifty.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSAC.2016.2577238

Therefore, we extended our system to work in a fully automatic manner, improve the link utilization, provide robust streaming and performed the second field experiment on a larger scale in February 2015 [31]. We first replaced CCN with Named Data Networking (NDN) [23] because the latest CCNx [8] that is one of implementations of CCN is no longer available as open-source software, and we developed a browser-based implementation written by JavaScript, that we call "DASH-NDN-JS" [9]. Second, we modify the link utilization of DASH-NDN-JS by Interest aggregation and burst Interest transmission. In the second field experiment, we then used actual train vehicles served on a commercial railroad line, built a high-speed backbone network to connect three railroad stations, and installed servers and Wi-Fi access points at each station and train. Then, a maximum of 50 users participated in the experiment to download video content simultaneously. The evaluations indicated that both HTTP and NDN-based systems can achieve high quality video delivery without interruption for up to 50 users simultaneously. Furthermore, in order to improve the robustness of delivery schedule, we introduce robust scheduling by using margin segments and multiple wireless connections.

In this paper, we summarize series of our contributions to the proactive caching in [5], [9], and [31], and introduces new discussions and considerations based on our previous work. The rest of this paper is organized as follows. Section II presents related work. Section III illustrates the proactive content caching scheme. Section IV describes performance evaluations using computer simulations. Section V and VI introduce performance evaluations of the first (small-scale) and second (large-scale) field experiments, respectively. Finally, Section VII provides the conclusions.

II. Related Work

A. Proactive Caching

Many studies have been conducted on proactive caching schemes. An opportunistic content pushing scheme was proposed in [10] that predicts the moving routes of roaming users and pre-locates content to Wi-Fi spots along their routes. The authors of this paper proposed a similar method called comfort route navigation that recommends a moving route that maximizes throughput or minimizes power consumption instead of the conventional shortest path [22]. Lobzhanidze and Zeng [11] proposed a proactive video caching scheme based on video popularity prediction using a topic modeling tool called Latent Dirichlet Allocation and a frequent pattern mining algorithm called Apriori. Vasilakos *et al.* [12] proposed a method for Information Centric Networking (ICN) called Selective Neighbor Caching that enhances seamless mobility in ICN, which selects an optimal subset of neighbor proxies that consider user mobility behavior. Similar to [12], Rao *et al.* [13] proposed a proactive caching approach for seamless user-side mobility support in NDN.

This paper proposes a proactive caching method that uses stations and trains as content caches. Fixed train (and bus) time-tables are used to schedule content delivery, although extension to prediction or navigation is possible. We consider network dynamics and use of MPEG-DASH to support adaptive streaming without interruption. Similar to [12] and [13], in-network caching capability of ICN/CCN/NDN is considered in our system. Content popularity can be used, but this is future work.

B. Throughput Prediction

A delivery scheduler should know communication quality in wired and/or wireless networks. This can be achieved by throughput prediction technologies. There are many ways for predicting throughputs, which can be categorized into two general types. One approach is to estimate the throughput using a formula that expresses TCP behavior [14], [15]. Although it is possible to obtain the predicted values immediately, its accuracy is not so high. The other approach is based on data measurement history [2], [16], [17]. In [16], mobile performance was predicted by constructing regression trees for the performance metrics. In [17], a method that adapts the Auto Regressive Integrated Moving Average (ARIMA) model was proposed. In [2], throughput fluctuation was predicted by constructing a stochastic model of TCP throughput, which is a mixture of stationary and non-stationary states. The authors of this paper proposed similar approaches using throughput history [27], [28]. This paper also uses the throughput history to allocate the amount of delivered contents in a conservative manner.

C. CCN/NDN

In CCN/NDN delivery, two message types called Interest and Content Object are exchanged. Interest messages are used to request data by specifying the content chunk name. Such messages also contain a name prefix to limit the data that is most suitable from the collection of the same prefix. Content Object messages are used to supply data. Such messages are mainly composed of a name, publisher, and chunk of data; they also contain data payload, cryptographic signature, publisher identification, and other information about signing. In communication, a data consumer broadcasts an Interest message over all available connectivity, and any node with the content that satisfies the Interest must respond with at most one Content Object message. In order to satisfy the Interest, the Content name in the Interest message has to be a prefix of the Content name in the Content Object message. One of the key features of CCN/NDN is the router content caching mechanism. When content goes through a CCN/NDN router, it caches itself. This feature provides congestion reduction and fast content delivery because clients can fetch content from the nearest cache rather than the origin content server. Different implementations of CCN and NDN are known as, CCNx [8], NDNx [24] and NDN-JS [25].

D. MPEG-DASH

MPEG-DASH is a streaming technology, standardized by MPEG that is capable of a continuous playback by changing the bitrate dynamically and adaptively while observing the network bandwidth. Video content is encoded into multiple bitrates and resolutions, and is divided into segments. URLs of each segment are written in the

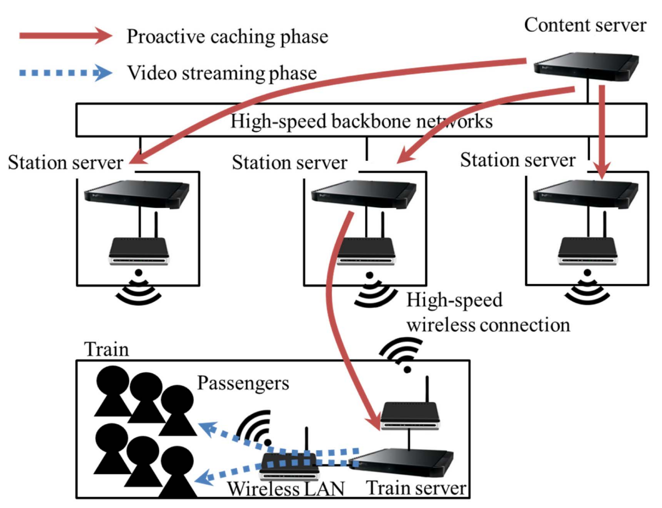


Fig. 1. An overview of our scheme.

Media Presentation Description (MPD), which also has information on encoded bitrates, resolution, minimum buffer time, etc. Clients access this MPD file at the start of the streaming session, and refer to it in order to select the optimal bitrate according to the network conditions. Note that every segment can be accessed individually by the client via HTTP GET requests.

DASH-JS [18] is one of the libraries integrated for the DASH standard. This is a JavaScript-based implementation that uses the Media Source API from Google's Chrome browser, and has no need for further plugins. Furthermore, this feature allows DASH playback on multiple devices and makes it easier to develop mobile applications.

III. Proactive Content Caching Utilizing Transportation System

A. Architecture

Our proactive content caching scheme utilizes transportation systems, especially trains, for robust video delivery along with efficient wireless resource use. In this paper, we focus on trains because their schedules are fixed, and predicting their behavior is easy. An overview of our scheme is shown in Fig. 1. We place servers with cache capability (e.g., CCN/NDN) in every train and station. Every station server connects to content servers via high-speed wired transport scheme. Our approach has two phases: proactive caching and video streaming.

The proactive caching phase indicates how to distribute content to the nearest cache router/server, which is a train server in this case. When the content server receives a request message, the requested content is divided into several segments, and then proactively distributed to station servers according to the train's schedule. Content quality and the amount of content segments should be adaptively changed according to the train's schedule and network conditions. The details of this adaptation are discussed in the next subsection. Once the trains arrive at the stations, they download pre-cached content segments from the station servers via high-speed wireless transport scheme, such as millimeter-waves, until they leave the stations. As a result, video content is stored in the train servers.

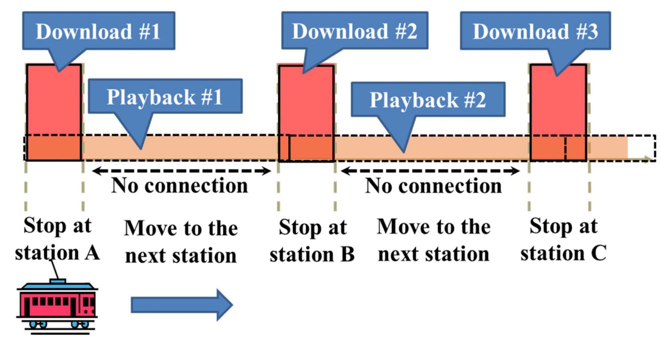


Fig. 2. Transportation vehicle and user behavior.

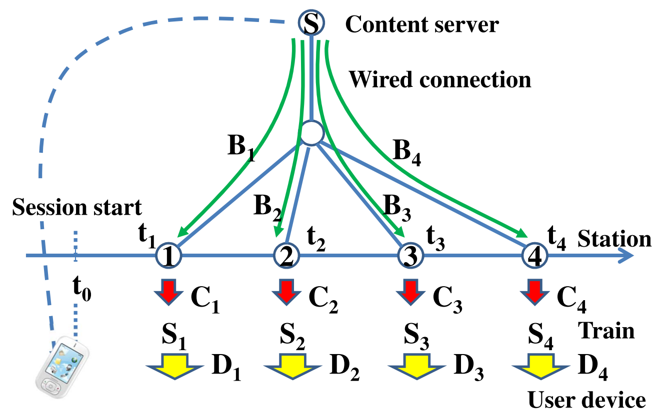


Fig. 3. Delivery model of proactive content caching.

Then, the received video segments are streamed to the users via high-speed wireless access schemes such as wireless LANs. We called this phase the video streaming phase. Train and user behavior are shown in Fig. 2. The main feature of our approach is that the users and train servers do not access the original content servers directly. This can reduce delay and packet losses compared with traditional video streaming on cellular networks.

B. Smart Scheduler

The key function of our system is a delivery scheduler we call smart scheduler. This scheduler determines content quality and the amount of content segments, and also selects delivery locations and timing. Video content is encoded by multiple bitrates and stored according to MPEG-DASH. In MPEG-DASH, one of the layers in the video quality hierarchy is called "representation," and each representation consists of several "segments" that correspond to a few seconds of video content. Thus, in our approach, smart scheduler simply selects one representation per station by referring to the calculated content bitrate.

The content bitrate is calculated considering train's schedule and network conditions. Content delivery model and parameter definitions are shown in Fig. 3 and listed in Table I, respectively. We assume that wired/wireless network bandwidth can be estimated using the bandwidth prediction technologies described in Section II.B. For simplicity, we also assume that the estimated bandwidth can be shared fairly among multiple users. Then, the content bitrate is computed by fulfilling the following three conditions:

TABLE I
PARAMETER DEFINITIONS

Parameter	Definition
n	Station ID
t_n [s]	Arrival time at station n
Δt_n [s]	Stoppage time at station n
B_n [Mbps]	Estimated bandwidth between content server and station n server
C_n [Mbps]	Estimated bandwidth between station n server and train server
D_n [Mbps]	Estimated bandwidth between train server and user device at station n
E_n [Mbps]	Estimated cellular bandwidth between train server and content server until arrival at station n
S_n [Mbps]	Transferred data for train server at station n
R_n [Mbps]	Content bitrate at station n
j_n	Number of users at station n
L_n	Number of segments at station n
k [s]	Playback time per segment
α_n	Margin for delivery timing at station n
β_n [s]	Margin for transferred extra segments at station n

1) Proactive Caching: content segments have to be delivered to station servers before the trains arrive. We define α_n as the margin number of stations for proactive delivery at station n. Delivery timing depends on vehicle location and this margin. For example, when $\alpha_n = 1$, the content server transmits content segments to station n when the train arrives at station $n-1$. Let B_n denote the available backbone bandwidth and S_n denote the data size that the train server can transfer at station n; then the proactive caching condition can be formulated by

$$S_n \leq B_n \cdot (t_n - t_{n-\alpha_n}). \quad (1)$$

2) Continuous Playback: video playback has to be continuous until arrival at the next station to avoid interruption. Let R_n be the content bitrate of MPEG-DASH and j_n be the number of users in a train at station n; then the continuous playback condition can be formulated by

$$\frac{S_n}{j_n} \geq R_n \cdot (t_{n+1} - t_n). \quad (2)$$

3) Smooth Streaming: received video segments should be streamed smoothly to the users inside a train. Because content bitrate R_n should be lower than the available bandwidth between a train server and user device, the smooth streaming condition is formulated by

$$\frac{D_n}{j_n} \geq R_n \quad (3)$$

where D_n represents the available bandwidth in a train.

Although a train server can connect to a content server via cellular networks while moving, we assume a train server downloads content segments at the stations only. Let C_n represent the available bandwidth between station n and train, and Δt_n represent the stoppage time at station n; then, S_n is calculated by

$$S_n = C_n \cdot \Delta t_n \quad (4)$$

By substituting equation (4) into inequalities (1), (2) and (3), content bitrate R_n has to satisfy

$$R_n \leq B_n \cdot \frac{(t_n - t_{n-\alpha_n})}{j_n(t_{n+1} - t_n)} \ (Proactive\ caching) \quad (5)$$

$$R_n \leq \frac{C_n \cdot \Delta t_n}{j_n(t_{n+1} - t_n)} \ (Continuous\ playback) \quad (6)$$

$$R_n \leq \frac{D_n}{j_n} \ (Smooth\ streaming) \quad (7)$$

The smart scheduler finally selects the MPEG-DASH representation that satisfies the three inequalities above.

In particular, R_n strongly depends on Inequality (6) because the wired network bandwidth (B_n) is generally larger than the wireless network bandwidth (C_n), and train travel time ($t_{n+1} - t_n$) is usually larger than train stoppage time (Δt_n).

The smart scheduler also calculates the number of segments L_n that the content server transmits to station n. To provide continuous video streaming, each station server should cache additional segments. We define k as the playback time of each segment in second. Thus, L_n is calculated by

$$L_n = \frac{t_{n+1} - t_n}{k}. \quad (8)$$

C. Robust Scheduling

The smart scheduler needs to be extended for robust scheduling because a current delivery schedule may be collapsed by train delays and users' playback behaviors (e.g., channel zapping).

First, in order to address this fact, the scheduler should distribute required videos with extra segments called "margin" to each station server, and then the train server can obtain these extra segments from each station server. This margin makes the users to keep playback video for a longer time even if the train delays happen. Therefore, the scheduler can provide higher robustness as the margin gets longer.

However, there is a trade-off between this robustness and video quality (R_n) (i.e., Representation of DASH content). We formulate the condition of continuous playback by inequality (6) because this condition is strongly depended on R_n as described in previous sub-section. Let β_n denote the margin for transferred extra segments at station n, inequality (6) can be transformed by

$$R_n \leq \frac{C_n \cdot \Delta t_n}{j_n(t_{n+1} - t_n + \beta_n)} \quad (9)$$

As shown in inequality (9), the DASH representation becomes lower as the margin gets longer because transferred data at each station is mainly affected by the network bandwidth.

Next, in order to alleviate this trade-off, the train server can obtain the content by using multiple wireless connections (e.g., Wi-Fi and cellular). The train server potentially increases the amount of transferred data when the train server constantly communicates with the content server via cellular networks. Let E_n represent estimated cellular bandwidth between a train server and a content server until arriving at station n as shown

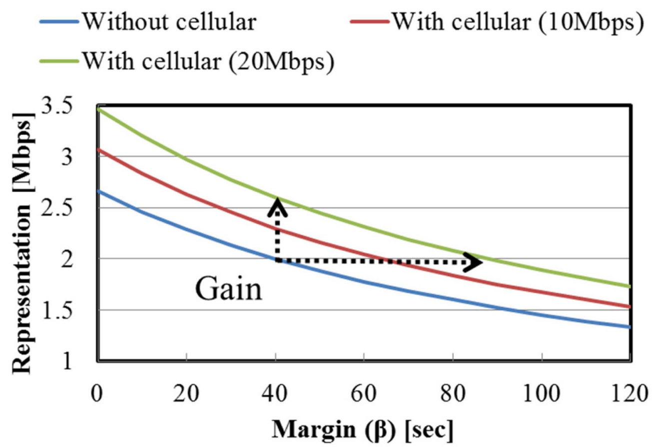


Fig. 4. Example trade-off characteristic between DASH representations and margins. (Without cellular: Inequality (9), with cellular: Inequality (10)).

TABLE II
PARAMETER CONDITIONS

Parameter	Values
$t_n - t_{n-1} [s]$	120
$t_{n+1} - t_n$ [s]	120
Δt_n [s]	20
C_n [Mbps]	400
E_n [Mbps]	10/20
j_n	25

in Table I, and inequality (9) can be transformed by

$$R_n \leq \frac{C_n \cdot \Delta t_n + E_n \cdot (t_n - t_{n-1})}{j_n(t_{n+1} - t_n + \beta_n)} \tag{10}$$

As shown in this inequality (10), the DASH representation becomes higher compared with inequality (9) as the smart scheduler selects the fixed value of the margin (β_n). Similarly, the margin (β_n) becomes longer as the smart scheduler selects the same DASH representation as Inequality (9). A wise use of cellular connections can also handle flexible users' playback behaviors, in particular, channel zapping, but, this is our future work.

In order to confirm the trade-off characteristics, Fig. 4 graphically presents inequalities (9) and (10) under the parameter conditions shown in Table II. As shown in Fig. 4, although the longer margin decreases the video quality, the use of cellular connection can alleviate the quality degradation.

D. Communication Protocols

To implement our scheme, we need to determine the communication protocols. With regards to CCN/NDN architecture, there is no standard protocol for control packets (i.e., signaling). Therefore, we use the IP for control packets, and use IP (HTTP) or CCN/NDN for data packets. The basic procedures of our prototype are demonstrated in Fig. 5, and summarized as follows:

1) Signaling (IP): once a train server or a content server receives a request message from a user, the servers forward the request to the smart scheduler. Then, the smart scheduler collects the train time-table from a database and determines content delivery schedule. This scheduler also collects information of network conditions from stations and trains periodically and stores them into another database.

2) Proactive Caching (IP and/or CCN/NDN): the station servers download video segments by referring to the delivery schedule. Once the station servers receive video segments via IP and/or CCN/NDN networks, they cache the received segments.

3) Video Segment Download (IP and/or CCN/NDN): when the train arrives at a station, the train server download video segments via IP and/or CCN/NDN networks. Therefore, the train server can obtain video segments.

4) Video Streaming (IP and/or CCN/NDN): finally, the user devices receive video data via the IP or CCN/NDN networks.

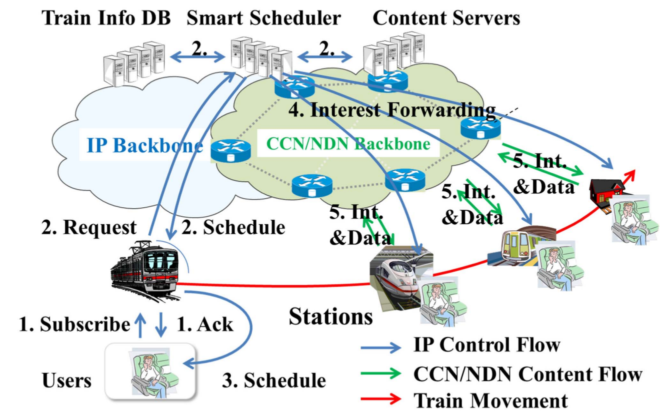


Fig. 5. Basic protocol procedures of our prototype.

E. Implementations

We developed and improved two prototype implementations aiming to field experiments described in Sections V and VI.

For the first field experiment in February 2014, we developed a pure HTTP-based prototype and an HTTP/CCN hybrid prototype. For signaling, HTTP was commonly used via PHP scripts. For content transfer, HTTP or CCNx were used for each prototype. For a streaming application, we used VideoLAN Client (VLC) with DASH and CCN plugins [18]. Because these prototypes were not fully automatic, an operator who rode on a train manually connected to appropriate Wi-Fi APs at each station. Furthermore, the number of accommodated users and video contents were very few.

For the second field experiment in February 2015, we improved the prototypes in several ways. First, CCNx was replaced with NDNx because the latest CCNx is no longer available as open-source software and the NDN-based prototype written in JavaScript was developed, and we call it DASH-NDN-JS [9]. DASH-NDN-JS integrates DASH-JS and NDN-JS to allow playback of DASH content on web browsers instead of VLC. This is convenient for large-scale experiment because participants can use browsers which they are familiar with. Second, Secure Copy Protocol (SCP) was used in the HTTP based prototype, and Interest message aggregation was used in the NDN based prototype, respectively. As explained in Section VI, they contribute to increasing throughput efficiency

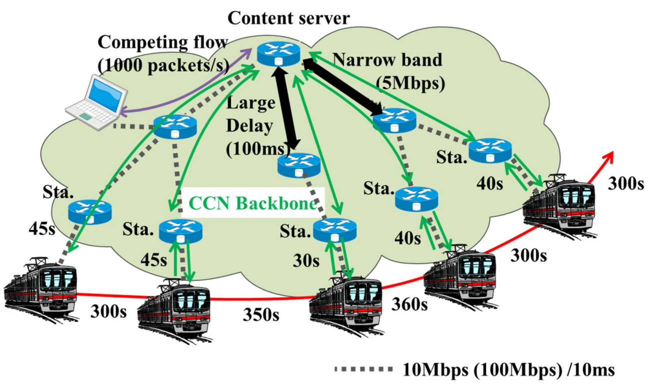


Fig. 6. Simulation environment.

TABLE III
EXPERIMENT SCENARIO

Scenarios	Bandwidth	Delay
(a)	10Mbps	10ms
(b)	100Mbps	10ms
(c)	50Mbps	10ms

by underlying TCP. Third, scripts were developed to switch Wi-Fi connection by checking signal strength and the train time-tables. As a result, fully automatic implementations were performed.

On implementation of the smart scheduler, we managed the train time-tables and network conditions by constructing data based on an HTTP server, which also supports CCN/NDN transport. The smart scheduler itself was written in JavaScript in both the initial and subsequent prototypes.

IV. Simulation Evaluations

A. Performance of Proactive Caching

First, we evaluate the basic performance of our proactive content caching scheme using ndnSIM [19]. The simulation environment is shown in Fig. 6. We set link parameters in two experiment scenarios (a) and (b), as indicated in Table III. Each scenario also has a large delay link, narrowband link, and competing flow as depicted in Fig. 6, and the train server can connect to the content server using a cellular network (1Mbps/10ms) while moving. In the case of proactive caching, the train server obtains content from the content server at the first station only. At the remaining stations, the train server receives pre-cached content from each station server. We also attempt a no-caching case for comparison, where a train node downloads from the content server at every station. When there are Quality of Service (QoS) degradations in the nearest link from the content server, such as larger delay and narrower bandwidth, the gain for our proposal is expected to become larger. The comparison results of the total received data in all four cases (two scenarios and with/without caches) are shown in Fig. 7. In particular, we confirm that great gain can be achieved in the case of large bandwidth and fast link.

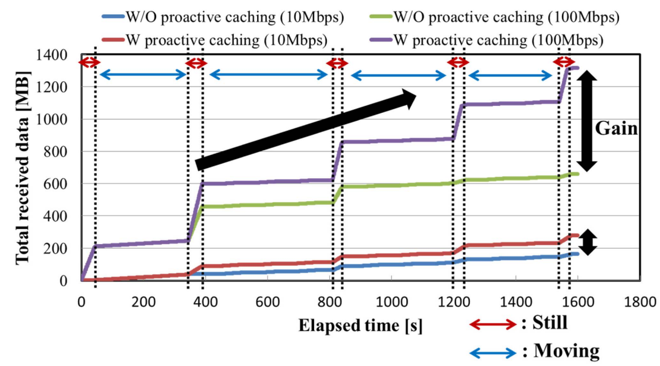


Fig. 7. Results of total received data with/without proactive content caching.

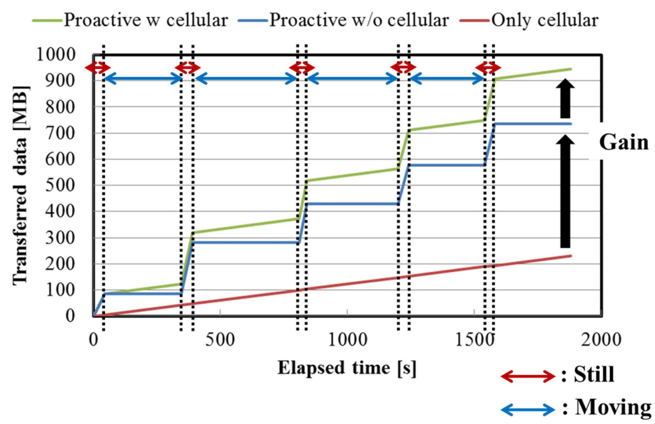


Fig. 8. Comparison results of combination of cellular use and our proactive caching.

B. Performance of the Use of Cellular Connection

Next, we also evaluate the gain of our system with cellular connections using ndnSIM. The simulation environment is mostly same as shown in Fig. 6. We set link parameters in the experiment scenario (c) as indicated in Table III. In this simulation, to evaluate basic contribution of cellular connection to our system, we do not consider large delays, narrowband links, and competing flows as depicted in Fig. 6. The train server can constantly connect to the content server using a cellular network (1Mbps/10ms). The comparison results of combination of cellular use and our proactive caching are shown in Fig. 8. From this figure, we confirm that significant gain can be achieved in our proactive caching system by incorporating cellular connection.

V. Performance Evaluations in a Single User Scenario

A. Field Experiment Environment in February 2014

We evaluate system performances for a single user case by performing field experiment using actual trains. This experiment was supported by the Keikyu Corporation, one of train operating companies in Japan. Details of our experiment environment are shown in Fig. 9. We set three virtual stations: stations A, B, and C. An IEEE 802.11ac Access Point (AP) and server were placed at each station and inside a train. A content server was also placed inside a control room. All servers, with exception of the train server, were

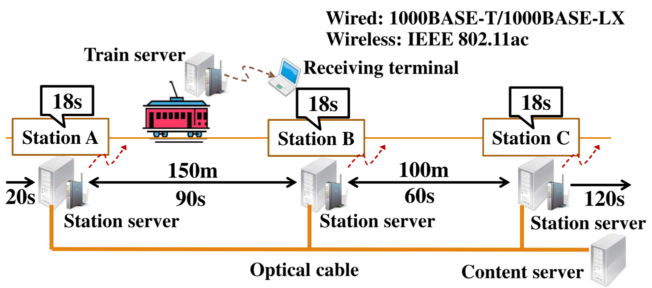


Fig. 9. Details of our experiment environment in the first field experiment.

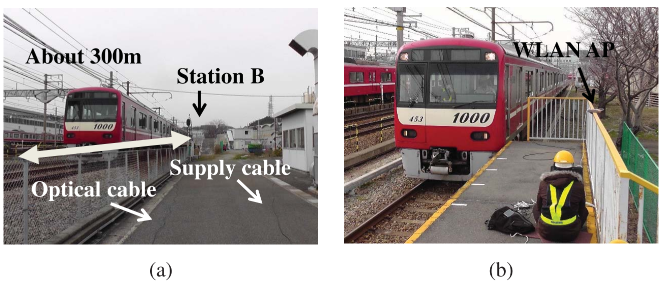


Fig. 10. Experiment snapshots. (a) First field experiment. (b) Virtual station B.

connected by optical cables (1000BASE-LX) and each AP was connected to the nearest station/train server with a Gigabit Ethernet cable (1000BASE-T). Note that each AP is assigned different channels to avoid radio wave interference. Available wired/wireless network bandwidths were observed at approximately 600/100 Mbps using iperf [20] ($B_n = 600$ and $C_n = D_n = 100$ in Table I), respectively. The train moved from station A to station C; it moved for 90 sec between stations A and B and for 60 sec between stations B and C. After the train left station C, we assumed that the train would move to the next station for 120 sec. At each station, the train stopped for 18 sec. This stoppage time is an average value observed for an actual served train line in Tokyo.

After a user sent a request message to the smart scheduler via the wireless network, and waited for the train to arrive for 20 sec. Meanwhile, the smart scheduler determined the delivery schedule, and required content segments were cached to the station servers. Each station server cached them with 10 additional segments as soon as the smart scheduler updates cache information ($\alpha_n = 3$ and $\beta_n = 20$). When the user boarded the train, the user attempted to playback video. In this experiment, the user requested a video of "Elephants Dream," known as one of test sequences used in MPEG standardization [21]. The video resolution was 1080p, and the encoding bitrates were 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, and 5.0 Mbps. For simplicity, we merely evaluated a single user case ($j_n = 1$). All packets were captured by Wireshark. Snapshots of our field experiment environment and station B are shown in Fig. 10.

B. Proactive Caching Performance Between a Train and Stations

Throughput transitions between the train and other servers are shown in Fig. 11. In this evaluation, both the wired and wireless bandwidths are so wide that the smart scheduler could select the highest quality video ($R_n = 5Mbps$). Note that the content server also successfully distributed video segments to each station for 2 - 5 sec. As shown in Fig. 11, at each station, the train server was able to receive requested video segments while stopped. The transmission time at each station was 13, 8, and 16 sec, respectively. Although the available wireless bandwidth was nearly 100 Mbps, the actual throughput at each station was under 50 Mbps mainly because CCNx is applied to a legacy congestion control, such as TCP-Tahoe. Therefore, throughput is saturated and greatly decreased when even a single Interest packet is dropped. We aim to alleviate this problem by conveying CCN/NDN packets over TCP in the next step.

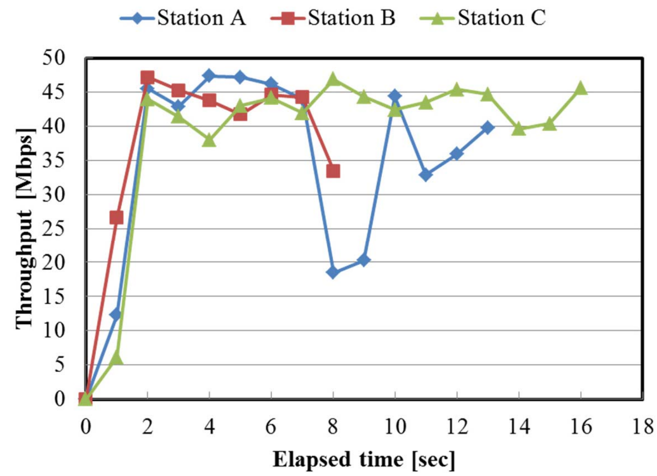


Fig. 11. Observed throughputs between train and station servers.

C. Video Streaming Performance in a Train

To enhance the effectiveness of our proposal, we compared it with traditional video streaming scenarios. Current mobile users, especially younger people, enjoy video streaming services over cellular networks. We employed two scenarios using Long Term Evolution (LTE); the mobile users stayed in place or moving about. Both scenarios were evaluated at our university campus. In the moving case, the mobile user randomly walked about. The experimental conditions were completely the same for the field experiment because we used the same content server, receiving terminal, and video data. Throughput comparison is shown in Fig. 12. Unsurprisingly, the throughput of our prototype was larger than both cellular scenarios. In particular, in the cellular moving scenario, throughput sometimes dropped to nearly 0 Mbps and video playback froze because of severe communication quality degradation. Captured videos of all scenarios are shown in Fig. 13 and published in [26]. As shown in Fig. 13, the video playback is delayed in the cellular cases. Therefore, we can conclude that our system can achieve higher quality and highly reliable video delivery.

VI. Performance Evaluations for 50-User Scenarios

A. Efficient Content Transfer in NDN

When our system sends video data by CCN/NDN, efficient content transfer is important because the train server

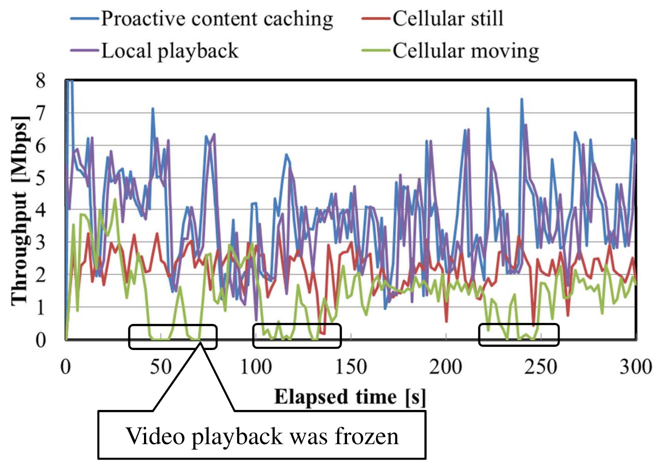


Fig. 12. Comparison of observed video transmission rates under different scenarios.

Fig. 13. Comparison of playback videos under different scenarios.

has to transfer large video data only while stopped at a station. Because current CCN/NDN implementations, such as NDNx and NDN-JS, emulate legacy congestion control, such as TCP-Tahoe, their network bandwidth utilization is insufficient, as demonstrated in the previous section. In this section, we introduce an efficient transmission mechanism of Interest packets to improve throughput efficiency in NDN.

First, we send NDN packets over TCP, specifically CUBIC-TCP, which is implemented into the Linux kernel. It is well known that CUBIC-TCP provides higher-throughput and better link utilization than TCP-Tahoe. Because TCP provides reliable data transport capability by retransmission, we no longer worry about packet drops of Interest and Content Object messages.

Second, to fully utilize CUBIC-TCP's efficient congestion control, Interest messages are aggregated and transmitted in a single burst per station. In the current video streaming application with CCN/NDN capability, Interest messages are transmitted intermittently. However, TCP connection per Interest is inefficient, and Interest aggregation contributes to throughput improvement. This is also true for the Content Object case, and we also attempted video data aggregation in the HTTP based video delivery.

Third, to reduce unnecessary Interest messages, we allow the smart scheduler to calculate the amount of Interest messages that the consumer needs to send. In current NDN,

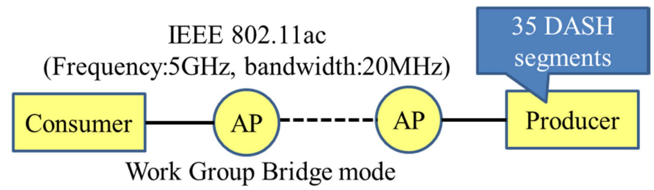


Fig. 14. Experiment setup for bandwidth utilization evaluation.

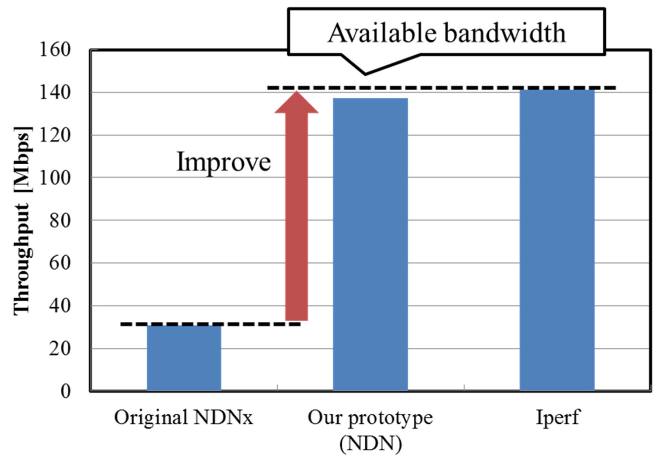


Fig. 15. Throughput comparison for original NDNx and our NDN-based prototype. Results for iperf show available bandwidth.

a consumer does not know how many Interest messages it should send because NDN divides video content into several chunks by its own rule (e.g., chunk size is 4096 Byte in NDN-JS). For this reason, the consumer continues to send Interest messages until it can receive Content Objects. Therefore, our smart scheduler calculates the amount of Interest messages by referring to chunk size and content size in advance. In MPEG-DASH, content size can be managed by the MPD file.

We then evaluated and compared link utilizations between the original NDNx and our system over a wireless network. An evaluation environment is shown in Fig. 14. A consumer was connected to a producer via IEEE 802.11ac wireless LAN. We used two Cisco Aironet 3700 Access Points (APs) that function in work bridge mode. The channel frequency and bandwidth were 5GHz and 20MHz, respectively. We placed 35 DASH segments that equaled 300 MB in total, and the segments were divided into several chunks by NDNx. We also observed the available network bandwidth using iperf. Then, as shown in Fig. 15, we compared the throughputs of original NDNx, our proposal, and the available bandwidth given by iperf. As shown in Fig. 15, our proposal can achieve higher throughput than original NDNx and almost reach the available bandwidth.

B. Field Experiment Environment in February 2015

We evaluated the performance of our subsequent prototypes by performing a larger-scale field experiment using a commercial railroad line, called Keikyu Daishi line, in Kawasaki, Japan as shown in Fig. 16. This line has seven stations and requires 10 min for a one-way trip. We selected

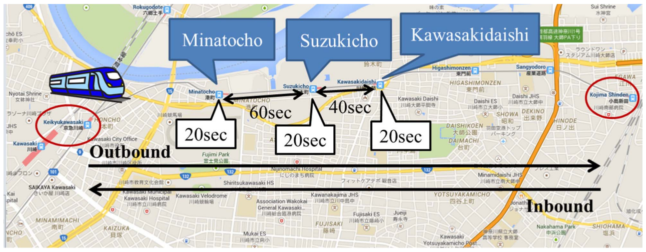


Fig. 16. Train movement map of Keikyu Daishi line in Kawasaki, Japan.

Fig. 17. Snapshots of installed equipment at a station. (Inbound side platform).

three train stations from seven, Minatocho, Suzukicho, and Kawasakidaishi. As shown in Fig. 16, the train stops 20 sec at each station, and requires 40 sec to 1 min to move between stations. As shown in Fig. 17, we set a station server, 802.11ac AP, and backbone router in a box, and installed the box to the inbound side platform of each station. Inside a train, as shown in Fig. 18, we placed a train server, 802.11ac AP, and two 802.11n APs, and connected them by LAN cables. The 802.11ac AP was used to communicate with stations, and the two 802.11n APs were used to deliver video content to train riders. Therefore, in inbound case, the link quality between the station and train could be higher compared with outbound case because the distance between the train and station APs is nearer. Each station server was connected via high-speed optical backbone networks. This backbone network was constructed by NTT East Corporation, and it was also connected to Waseda University to monitor the backbone. The station servers at Minatocho (outbound case) and Kawasakidaishi (inbound case) also played roles as content servers. Each AP was assigned different channels to avoid radio wave interference. We randomly selected 25 different 2 K videos from YouTube. Each video was encoded by H.264/AVC and divided into several segments. The encoding rates were 100, 200, 400, 600, and 800 kbps and 1 Mbps, and the segment length was 2 sec, which is an original DASH parameter. We allocate sufficient cache sizes (4GB), compared with the total video sizes (approximately 1GB), to the train and station servers.

C. Evaluation Scenarios

In this field experiment, a total of 50 participants were separated into two groups; the first group consisted of 25 Android users, and the second group consisted of 25 Windows users. The two groups boarded different trains, each of which had one 802.11n AP. Up to 50 users requested video content simultaneously before the train arrived at Minatocho (outbound) and

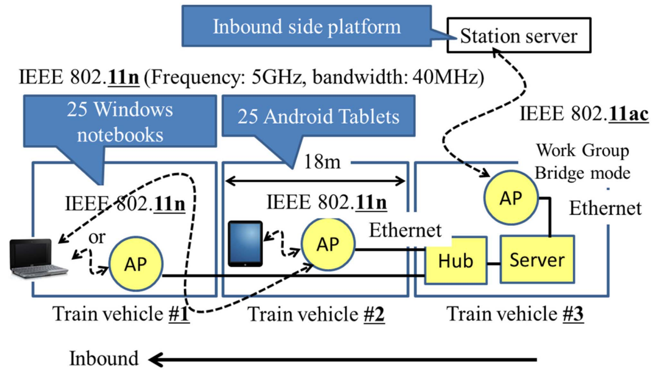


Fig. 18. Experiment setup inside train.

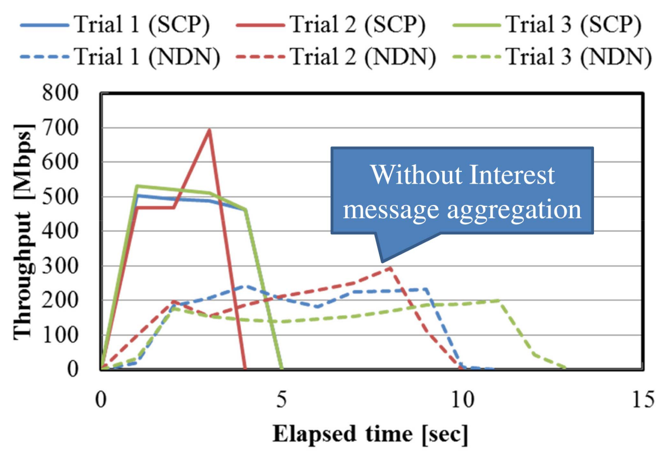


Fig. 19. Example of throughputs on backbone networks observed between content and station servers at Suzukicho station. NDN-based prototype was not applied Interest message aggregation.

Kawasakidaishi (inbound). We employed the following three different evaluation scenarios:

1) 25 Users and One AP: in this scenario, 25 Android users connected to the 802.11n AP located on train #2.

2) 50 Users and Two APs: in this scenario, 25 Android users connected to the AP located on train #2, and 25 Windows users connected to the AP located on train #1.

3) 50 Users and One AP: in this scenario, all users connected to the AP located on train #2.

We were able to conduct six round trips for the field experiment, which required approximately three hours. We developed two prototypes: HTTP-based and NDN-based (DASH-NDN-JS), both of which ran automatically without manual operations. We had three scenarios and two directions (outbound and inbound). Therefore, we attempted twelve ($2 \times 3 \times 2$) measurements in total according to the prototypes, scenarios, and train directions. All packets were captured by Wireshark.

D. Backbone Network Performance

First, we observed throughput transitions of backbone networks between the content and station servers. Fig. 19 shows the results for the Suzukicho case. Though omitted, the results of the three stations show similar characteristics. At this station, the content server had to distribute video segments

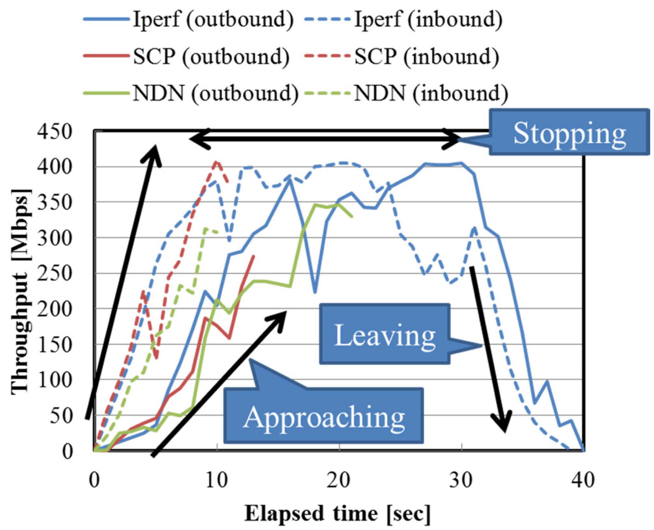


Fig. 20. Example of throughputs for proactive caching observed between train and station servers at Suzukicho station.

within 60 sec for outbound and 40 sec for inbound as shown in Fig. 16. In Fig. 19, SCP represents the HTTP-based prototype, and NDN represents the NDN-based prototype. As explained in Section III.E, SCP was used to send video data after aggregating segments in order to improve TCP performance.

From this figure, we can notice that the throughputs of NDN become less than half of that of SCP. This is because the content server has comparatively enough time to delivery to the station servers and we did not apply Interest message aggregation described in Section VI.A. In fact, both the HTTP and NDN-based prototypes were able to distribute successfully video segments well in advance. In other words, data transfer between a train and stations is most time critical in our system.

E. Proactive Caching Performance Between a Train and Stations

Second, we observed throughput characteristics of proactive caching between the train and stations for our two prototypes. Fig. 20 shows an example of throughput characteristics between the train and station servers. In this figure, throughputs of the outbound case become lower and fluctuate because the distance between the train and station APs is farther than the inbound case. From this figure, we can recognize that both prototypes work well and approach the iperf results which indicate available bandwidth. This achievement is provided by the video data aggregation at the content server for the HTTP-based prototype, and by Interest message aggregation into a single burst for the NDN-based prototype. CUBIC-TCP also contributes to maintaining high bandwidth utilization. We can also notice that most data transfers by the prototypes were completed before a train left the station. This is because our prototype designs were conservative in order to avoid experiment failures. The results encourage us to send more video data in our future design.

We finally note that, among twelve measurement trials, two trials failed to download video segments due to severe wireless communication quality degradation. One reason was insertion of another train between our train and station server in the outbound case. Another case was due to radar wave detection that forced temporal shutdown of outdoor Wi-Fi connection as specified in IEEE 802.11h.

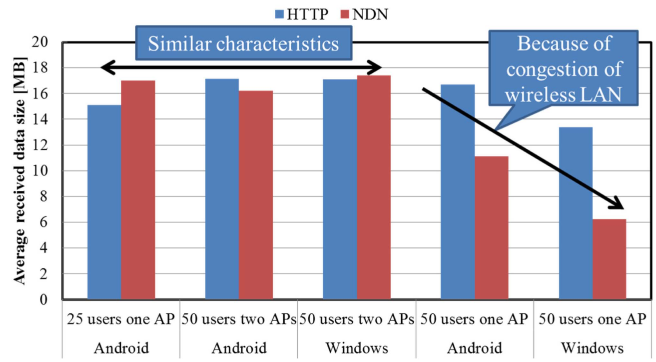


Fig. 21. Averaged received data sizes, with the exception of Ack packets, from train server to client terminals.

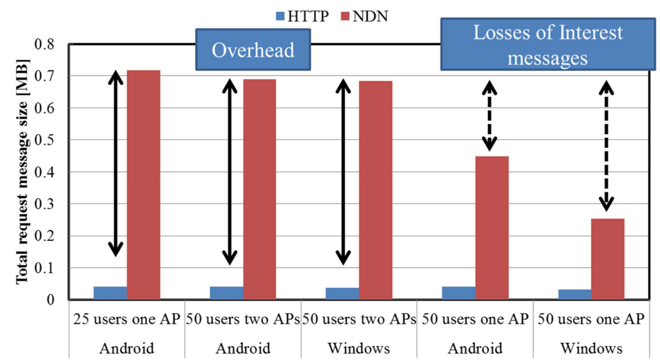


Fig. 22. Results of total successful transmitted request message size. (HTTP: GET message, NDN: Interest message).

F. Video Streaming Performance in a Train

Third, we observed the performance for video streaming in a train. In this field experiment, an initial buffering delay has only a few seconds because the link capacity is sufficient high, and the overhead of smart scheduler is negligible. Fig. 21 shows an average received data size, with the exception of Ack packets, from the train server to the client terminals. As shown in this figure, both HTTP and NDN show similar characteristics in middle traffic load scenarios, such as "25 users one AP" and "50 users and two APs" cases. However, in the high traffic load scenario (50 users one AP), we can notice received data sizes were reduced. This is because the capacity of IEEE 802.11n that we used inside a train is limited. In addition, in this high traffic load scenario, we can recognize that Windows users obtain lower performance than Android users for both prototypes. This is because Windows users attempted to access the AP that was placed at train #2 where Android users were sitting down as shown in Fig. 18. We also notice that the NDN-based prototype degrades more severely than the HTTP-based. This is because the NDN needs more overheads (i.e., massive Interest messages) to get video contents as shown in Fig. 22, and the packet collisions might be triggered many times in wireless LAN. Therefore, as shown in Fig. 22, in the high traffic load scenario (50 users one AP), total request messages in NDN (i.e., Interest messages) are decreased.

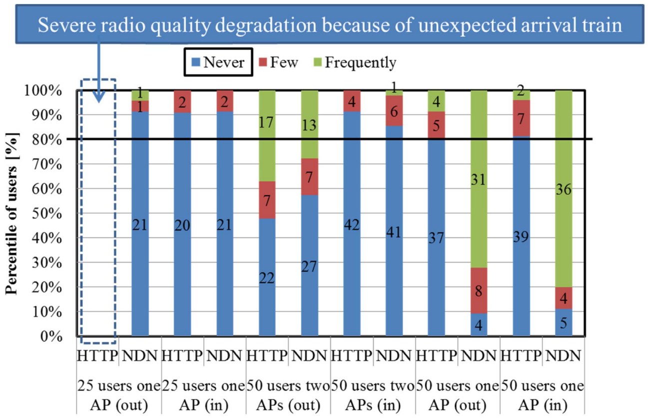


Fig. 23. QoE evaluation results of video playback impression (in: inbound, out: outbound).

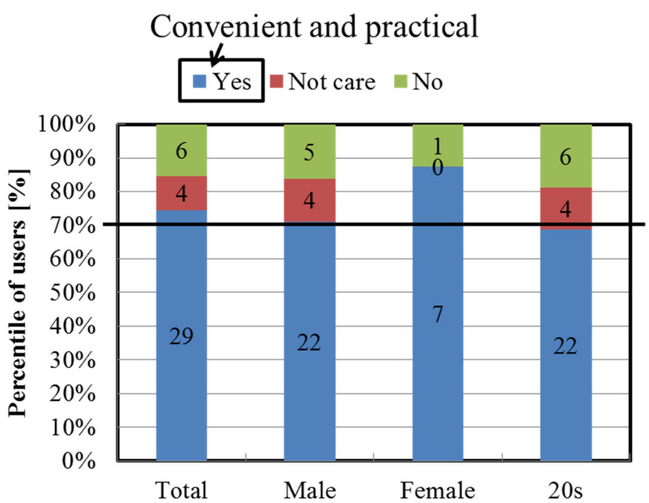


Fig. 24. Overall QoE evaluation results for our systems (Yes: positive, No: negative).

G. QoE Characteristics of Our Prototypes

Finally, to confirm the subjective effectiveness of our systems, we evaluated QoE using questionnaires. Among the many questions we provided, Fig. 23 shows the results of the video playback interruption experienced by users. We employed three choices; "never freezing," "few times," and "frequently freezing." The results show that more than 80 When the train failed to obtain video segments from the stations or the single AP had to accommodate 50 users, some trials indicate that video did not play back smoothly and many users experienced video that froze frequently. In particular, in the middle traffic load case (i.e., 25 users one AP), the video never played back because the train server failed to obtain any video segments. This is because radio quality is severely degraded by unexpected arrival trains between the train and stations. In the high traffic load case (i.e., 50 users one AP), we need more investigations why NDN-based video delivery leads to low quality.

We also presented a questionnaire on our system's overall impression, and the results are shown in Fig. 24. As shown in this figure, more than 70% of users, including 20s young people, answered "Yes," which means that users believed our system to be convenient and practical, regardless of gender.

VII. Conclusions

In this paper, we proposed a proactive content caching scheme that uses transportation systems for robust video streaming and efficient utilization of wireless resources. In our system, caching servers are located at stations and inside trains. They are connected when the trains arrive at the stations, and content is forwarded from the stations to the trains. We developed two prototypes based on the HTTP and CCN/NDN. In the first field experiment [26], we evaluated the performance of a single user scenario with some manual operations. We then extended our system to work in a fully automatic manner and to improve bandwidth utilization by message or data aggregation. In the second field experiment [29], we evaluated larger-scale scenarios, where a maximum of 50 users downloaded video content simultaneously. Evaluations indicated that our system can achieve higher quality and highly reliable video delivery than traditional video streaming on cellular networks. In particular, our system can provide video streaming without interruption for up to 50 users. Furthermore, we confirmed more than 70% of users indicated positive impressions of our system.

For future work, first, we will improve the smart scheduler to achieve more reliability and robustness. Because the users' playback behaviors (e.g., channel zapping) and quality requirements (e.g., variability of bit-rates) are diverse, the smart scheduler should schedule the video delivery in a more flexible manner. We will extend the smart scheduler to fulfill these requirements by incorporating related work such as [30] and considering content popularity.

Next, we will consider the energy consumption of our system and contribute to green video delivery. Development of an efficient congestion control for NDN, which can function even in heavy load, is another future study. We also try to extend our framework to other transportation infrastructures, such as buses and automobiles.

Acknowledgment

The authors would like to thank Keikyu Corporation, a train operating company in Japan, who provided the field experiment environment. They would also like to thank NTT East Corporation, who provided the high-speed backbone network for the field experiment.

References

[1] *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2014–2019 White Paper*, accessed on May 2015. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visuav-networking-index-vni/white_paper_c11-520862.html

[2] H. Yoshida, K. Satoda, and T. Murase, "Constructing stochastic model of TCP throughput on basis of stationarity analysis," in *Proc. IEEE GLOBECOM*, Dec. 2013, pp. 1544–1550.

[3] A. G. Tasiopoulos, I. Psaras, and G. Pavlou, "Mind the gap: Modelling video delivery under expected periods of disconnection," in *Proc. ACM CHANTS*, Sep. 2014, pp. 13–18.

[4] S. Singh, O. Oyman, A. Papathanassiou, D. Chatterjee, and J. G. Andrews, "Video capacity and QoE enhancements over LTE," in *Proc. IEEE ICC*, Jun. 2012, pp. 7071–7076.

[5] K. Kanai *et al.*, "Proactive content caching utilizing transportation systems and its evaluation by field experiment," in *Proc. IEEE GLOBECOM*, Dec. 2014, pp. 1382–1387.

[6] I. Sodagar, "The MPEG-DASH standard for multimedia streaming over the Internet," *IEEE Multimedia Mag.*, vol. 18, no. 4, pp. 62–67, Apr. 2011.

[7] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, “Networking named content,” in *Proc. ACM CoNEXT*, Dec. 2009, pp. 1–12.

[8] *Project CCNx*, accessed on May 2015. [Online]. Available: https://www.ccnx.org/

[9] T. Muto, K. Kanai, and J. Katto, “Implementation evaluation of proactive content caching using DASH-NDN-JS,” in *Proc. IEEE WCNC*, Mar. 2015, pp. 2239–2244.

[10] T. Han and N. Ansari, “Opportunistic content pushing via WiFi hotspots,” in *Proc. 3rd IEEE IC-NIDC*, Sep. 2012, pp. 680–684.

[11] A. Lobzhanidze and W. Zeng, “Proactive caching of online video by mining mainstream media,” in *Proc. IEEE ICME*, Jul. 2013, pp. 1–6.

[12] X. Vasilakos, V. A. Siris, G. C. Polyzos, and M. Pomonis, “Proactive selective neighbor caching for enhancing mobility support in information-centric networks,” in *Proc. ACM ICN*, Aug. 2012, pp. 61–66.

[13] Y. Rao, H. Zhou, D. Gao, H. Luo, and Y. Liu, “Proactive caching for enhancing user-side mobility support in named data networking,” in *Proc. IMIS*, Jul. 2013, pp. 37–42.

[14] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, “The macroscopic behavior of the TCP congestion avoidance algorithm,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 27, no. 3, pp. 67–82, Jul. 1997.

[15] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, “Modeling TCP throughput: A simple model and its empirical validation,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 28, no. 4, pp. 303–314, Oct. 1998.

[16] Q. Xu, S. Mehrotra, Z. Mao, and J. Li, “PROTEUS: Network performance forecast for real-time, interactive mobile applications,” in *Proc. ACM MobiSys*, Jun. 2013, pp. 347–360.

[17] S. Vazhkudai, J. M. Schopf, and I. Foster, “Predicting the performance of wide area data transfers,” in *Proc. IPDPS*, Apr. 2001, pp. 34–43.

[18] *ITEC-Dynamic Adaptive Streaming Over*, accessed on May 2015. [Online]. Available: http://www-itec.uni-klu.ac.at/dash/

[19] *ndnSIM*, accessed on May 2015. [Online]. Available: http://ndnsim.net/2.0/

[20] *Iperf.fr*, accessed on May 2015. [Online]. Available: https://iperf.fr/

[21] *Xiph.Org Test Media*, accessed on May 2015. [Online]. Available: http://media.xiph.org/

[22] K. Kanai, J. Katto, and T. Murase, “Performance evaluations of comfort route navigation providing high-QoS communication for mobile users,” *ITE Trans. Media Technol. Appl.*, vol. 2, no. 4, pp. 327–335, Oct. 2014.

[23] L. Zhang *et al.*, “Named data networking,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 3, pp. 66–73, Jul. 2014.

[24] *Named Data Networking*, accessed on May 2015. [Online]. Available: http://named-data.net/

[25] W. Shang, J. Thompson, J. Burke, and L. Zhang. (Aug. 2013). “Development and experimentation with NDN-JS, a javascript library for named data networking,” Tech. Rep. NDN-0014, accessed on May 2015. [Online]. Available: http://named-data.net/publications/techreports/

[26] (2013). *Proactive Content Caching, Phase 1*, (in Japanese), accessed on May 2015. [Online]. Available: http://www.comm.waseda.ac.jp/ProactiveCaching/

[27] K. Kanai, K. Hidenori, Y. Ishizu, and J. Katto, “A highly-reliable buffer strategy based on long-term throughput prediction for mobile video streaming,” in *Proc. 12th IEEE CCNC*, Jan. 2015, pp. 677–682.

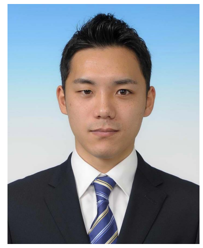
[28] H. Konishi, K. Kanai, and J. Katto, “Improvement of throughput prediction accuracy for video streaming in mobile environment,” in *Proc. 3rd IEEE GCCE*, Oct. 2015, pp. 749–750.

[29] (2014). *Proactive Content Caching, Phase 2*, (in Japanese), accessed on May 2015. [Online]. Available: http://www.comm.waseda.ac.jp/ProactiveCaching2/

[30] V. Joseph and G. de Veciana, “NOVA: QoE-driven optimization of DASH-based video delivery in networks,” in *Proc. IEEE INFOCOM*, Apr./May 2014, pp. 82–90.

[31] K. Kanai *et al.*, “Performance evaluation of proactive content caching for mobile video through 50-user field experiment,” in *Proc. IEEE Globecom Workshops*, Dec. 2015, pp. 1–6.

Kenji Kanai (S’12–M’16) received the B.E., M.E., and Ph.D. degrees from Waseda University, Tokyo, Japan, in 2010, 2012, and 2015, respectively. He is currently a Research Associate with Waseda University. He is a member of IEICE.



Takeshi Muto received the B.E. and M.E. degrees from Waseda University, Tokyo, Japan, in 2013 and 2015, respectively. He is currently with Microsoft Corporation.

Jiro Katto (S’89–M’92) received the B.S., M.E., and Ph.D. degrees from the University of Tokyo in 1987, 1989, and 1992, respectively, all in electrical engineering. He joined NEC Corporation in 1992 and then joined Waseda University in 1999. He is a member of ACM, IEICE, and ITE.

Shinya Yamamura received the B.E. degree in electronics engineering from Oita University in 1988, and the D.E. degree from the Kyushu Institute of Technology in 2013. In 1988, he joined Fujitsu Kyushu Network Technologies Ltd. He has been engaged mainly in research on network architecture.

Tomoyuki Furutono received the B.E. and M.E. degrees in physics from Kyushu University in 1989 and 1991, respectively. In 1991, he joined Fujitsu Kyushu Network Technologies Ltd. He has been engaged mainly in research on network architecture.

Takafumi Saito received the B.E. degree from Hiroshima University in 1973, and the M.E. degree from Osaka University, Japan, in 1975. He is currently an ICT Consultant with Cyber Creative Institute Company Ltd., Tokyo, Japan. He is a Full Member of IEICE.

Hirohide Mikami (M’99) received the B.E. and M.E. degrees from the University of Electro-Communications. He was a Visiting Senior Researcher with Waseda University. He is currently a Senior Consultant with Cyber Creative Institute Company Ltd. He is a member of IEICE and IPSJ.

Kaoru Kusachi joined Hitachi Systems Ltd. in 1971. He is currently with LSI Japan Company Ltd. and also a Visiting Senior Researcher with Waseda University.

Toshitaka Tsuda (F'90) received the B.S., M.S., and Ph.D. degrees from the University of Tokyo in 1970, 1972, and 1975, respectively, all in electrical engineering. He joined Fujitsu Laboratories Ltd. in 1975, where he engaged in the research and development of digital signal processing, video coding, ISDN transmission, optical transmission, packet switching, and microprocessor architecture. From 2012 to 2015, he was a Professor with Waseda University, where he is currently a Guest Professor. He is an Honorary Member and a fellow of the Institute of Electrical, Information, and Communication Engineers (IEICE). He was the Chairman of the IEEE Tokyo Section and the President of IEICE.

Wataru Kameyama (M'86) received the bachelor's, master's, and D.Eng. degrees from the School of Science and Engineering, Waseda University, in 1985, 1987, and 1990, respectively. He joined ASCII Corporation in 1992, and was transferred to France Telecom CCETT from 1994 to 1996 for his secondment. After joining Waseda University as an Associate Professor in 1999, he has been a Professor with the Department of Communications and Computer Engineering, School of Fundamental Science and Engineering, Waseda University, since 2014. He has been involved in MPEG, MHEG, DAVIC, and the TV-Anytime Forum activities. He was a Chairman of ISO/IEC JTC1/SC29/WG12, and a Secretariat and Vice Chairman of the TV-Anytime Forum. He is a member of IEICE, IPSJ, ITE, IIEEJ, and ACM. He received the Best Paper Award of Niwa-Takayanagi in 2006, the Best Author Award of Niwa-Takayanagi in 2009 from the Institute of Image Information and Television Engineers, and the International Cooperation Award from the ITU Association of Japan in 2012.

Yong-Jin Park received the B.E., M.E., and Ph.D. degrees from Waseda University, all in electronics engineering. From 1978 to 2010, he was a Professor with Hanyang University, Seoul. He has been with Waseda University, Tokyo, since 2010, where he is currently a Professor with the Department of Communications and Computer Engineering. He is a Professor Emeritus with Hanyang University.

Takuro Sato (F'13) was with Research and Development Laboratories, Oki Electric Industry Company, Ltd., Tokyo, Japan, where he worked on PCM transmission equipment, mobile telephone, and standardization of mobile data transmission and CDMA system for international standardization committee. In 1995, he became a Professor with the Niigata Institute of Technology. He was a Professor with Waseda University in 2004. His current research are next generation mobile communication system and network, wireless communication algorithm/architecture and communication control. Recently, he has been interested in smart grid technologies cooperated with ICT system, including wireless communication. He established a new academic society named Information Communication and Telecommunication Smart Grid in IEICE. He is also interested in mobile edge computing technologies based on information centric network to apply for 5G mobile communication network. He is a fellow of IEICE.