A Failsoft Scheme for Mobile Live Streaming by Scalable Video Coding

Hiroki OKADA\textsuperscript{1,\textdagger(a), Nonmember, Masato YOSHIMI\textsuperscript{1(b),}, Celimuge WU\textsuperscript{1(c),}, Members, and Tsutomu YOSHINAGA\textsuperscript{1(d),} Senior Member

SUMMARY In this study, we propose a mechanism called adaptive failsoft control to address peak traffic in mobile live streaming, using a chasing playback function. Although a cache system is available to support the chasing playback function for live streaming in a base station and device-to-device communication, the request concentration by highlight scenes influences the traffic load owing to data unavailability. To avoid data unavailability, we adapted two live streaming features: (1) streaming data while switching the video quality, and (2) time variability of the number of requests. The second feature enables a fallback mechanism for the cache system by prioritizing cache eviction and terminating the transfer of cache-missed requests. This paper discusses the simulation results of the proposed mechanism, which adopts a request model appropriate for (a) avoiding peak traffic and (b) maintaining continuity of service.

KEYWORDS live video streaming, mobile cooperative caching, peak avoidance, traffic reduction, D2D, scalable video coding

1. Introduction

Traffic in mobile networks will drastically increase by seven times within six years from 2017 \cite{1}. In 2023, traffic related to video content will occupy approximately 73\% of mobile communications. Advancements in mobile network technologies have facilitated the live streaming of sport games and concerts. The audience at local sites can watch live streams captured by cameras distributed in space on mobile devices such as tablets and smartphones. Each mobile device receives live streaming content broadcast via a base station (BS) to avoid network congestion. However, a feature that replays the broadcasted video backward, called chasing playback, can be harnessed via the individual requests of devices. Traffic spikes, which indicate request concentration to burst access owing to the rapid growth of viewing public, sometimes increase due to specific causes such as hitting a home run and scoring a goal \cite{2}. Accordingly, traffic spikes on the BS decrease the quality of service by delaying response \cite{3}. Peak traffic is the request concentration owing to traffic spikes that trigger service crashes and cause delays. To facilitate continuity, next-generation live streaming services require a mechanism that can withstand traffic spikes.

This study proposes adaptive failsoft control (AFC) as a mechanism to circumvent peak traffic while live streaming with a chasing playback function in a mobile network. We developed a failback procedure focused on a hierarchical structure feature for a content delivery network, as well as on the time variation of the number of requests to maintain continuity of services, which allows for video quality degradation even when peak traffic occurs. We also propose a prioritized cache control method, selective cache eviction (SCE), to be adopted in conjunction with AFC. SCE selectively discords low-priority caches when peak traffic occurs.

In this study, we considered an event venue or a sports stadium, where there are multiple BSs and many user devices. We introduced several existing technologies of distributed cooperative cache techniques to reduce traffic. This study contributes to the suppression of peak traffic by improving the efficiency of the cache network between users via device-to-device (D2D) communication.

2. Related Work

2.1 CDN: Content Delivery Network

A content delivery network (CDN) is used to reduce internet traffic and improve responses. Enterprises such as Akamai and Amazon provide worldwide CDNs to contribute to reducing internet traffic \cite{4, 5}. Because cloud CDN providers usually locate cache nodes on the Internet service provider (ISP) upstream, massive traffic flows occur between users and the cloud. A BS that receives requests from a considerable number of users within its communication area has to forward content from upstream, which create a heavy load on the network traffic. Recent studies have extended the adoption of D2D communication \cite{6}, a feature of the next-generation mobile communication standard, to develop a cache network among users. The objective of these studies was to accommodate the cached content and reduce the BS load. In the near future, as 5G cellular networks become more widespread, D2D communication will also become available at a lower cost.

2.2 Cooperative Cache

In our previous work, we proposed a cooperative caching...
method [7]. The mechanism allocates a tag attribute called spectrum into the nodes and content, called the node spectrum and content spectrum. The spectrum is represented as a bit vector to control the substance-cache capacity in the network. The nodes store content into their cache only when the spectra of both nodes and content coincide partially or entirely. Controlling the spectrum allocation configures the effective cache capacity based on the request volume while maintaining load balance.

The D2D shared network is featured by a variation in the number of devices within a time series. We modeled an optimization problem to minimize traffic and obtain a strategy for storing content on each device [7]. In this approach, every device has a unique spectrum to store independent cache contents. To determine the cache placement that minimizes traffic, we determined the appropriate content spectrum via optimization calculations. However, because this approach is a caching strategy that assumes static content, it cannot store newly generated content such as live streaming. Therefore, we need to consider a novel approach.

2.3 Live Video Streaming

Video streaming services require adaptive control of the quality of content, based on the current capacity of the network. This is realized by two methods. One method involves providing an appropriate bitrate video source from multiple-prepared content with different bitrates [8], [9]. The other provides a video source of the layered structure, which plays content with steadily increasing higher quality. According to [11], the latter method is more efficient than the former. The authors of [12], [13] also proposed adaptive streaming methods for the latter. These adaptive methods predict the network capability based on control theory [14], machine learning [15], and client buffer information [16].

As with our proposed method in the paper, several studies have proposed adaptive quality control mechanisms for video streaming work on gateway nodes.

The authors of [17] proposed a mechanism to improve QoE (stands for Quality of Experience) by dropping layers of video streaming at the eNB, stands for evolved Node B, which is a base station for LTE networks. Their idea is dropping SVC layers to improve average QoE according to simple static or adaptive metrics such as content characteristics or physical resources. Current scenario in [17] supposed that the eNB takes all layers from streaming servers and drops layers to provide each client device to reduce traffic in the LTE network. On the other hand, our approach reduces traffic at the higher network tier by introducing caching scheme and D2D network as a premise that local mobile network after 5G becomes reasonable and high-bandwidth with low-latency.

The authors of [18] tried to improve QoE by HAS (HTTP Adaptive Streaming) delivery by combining TCP congestion control and traffic-shaping. Reference [18] showed that to control bandwidth for each client and network congestion at the gateway has impact to QoE. The approach of [18] is unrestricted by the content characteristics such as SVC and practical evaluations with technical implementation and real but small environment. Our approach is introducing caching scheme adopting content characteristics to achieve both QoE and efficient traffic from wide and coarse point of view such as event venues.

Prior to this paper, the authors of [19] supposed similar use case and solution with our assumption. The authors of [19] proposed a cache allocation mechanism to reduce latency by formulating integer programming problem for adopting the layered structure of SVC. The primary difference between them and us is a priority to store cache for each video quality. Our SCE is prone to holds lower video quality to emphasize service continuity while [19] grants higher priority to higher quality to reduce delay to acquire video data. As the algorithm of AFC can be apply whether cache allocation algorithms such as [19] or our SCE, we adopted the algorithm of [19] for performance comparison in Sect. 7.2.

3. Adaptive Failsoft Control

Figure 2 shows components proposed in this paper and relationship between them. This study proposes a mechanism called adaptive failsoft control (AFC) to address peak traffic. First, AFC considers the prediction of emerging requests to measure, depending on the magnitude of traffic. The BS derives the prediction of traffic by increasing and decreasing the last several units of time read from logs, as well as recording the number of requests per unit time. We define the “congestion factor” as the ratio of predicted requests upstream of the BS, which can limit the bandwidth in the hierarchical network, as illustrated in Fig. 3. The congestion factor is calculated based on the variation in the number of requests during the last three-unit time and the total number of requests that can be delivered from the origin server. We define req as the number of requests during the last unit of time, req’ as the derivative between the last two units of time, and req” as the second derivative between the last three units of time. From the sum of req, req’, and req”, we determine the number of upcoming requests based on past trends. The traffic volume can be predicted by multiplying the predicted number of requests req+req’+req” by the data.
Fig. 2 Components in BS and relationship between three functions; AFC, SCE and D2D

Fig. 3 Example of a three-level CDN environment

chunk size per request. The congestion factor is expressed as:

\[
\text{Congestion factor} = \frac{\text{req} + \text{req'} + \text{req''}}{\text{bandwidth}} \cdot \text{chunksize} \quad (1)
\]

Here, we consider an audiovisual continuity of users in peak traffic. Users expect the video streaming service to continue, which allows for the degradation of video quality, rather than having intermittent video streaming and suspending the service. AFC introduces a scalable data structure that ensures the continuity of video streams. Layered video coding, such as scalable video coding (SVC) [10], improves video quality by laying over the base layer with enhancement layers. User devices can play at least a low-quality video stream as they acquire the base layer. Moreover, the device can play a high-quality video stream if it can acquire enhancement layers. Conversely, depending on the dependence between layers, playing videos with higher quality requires the base layer, as well as all the lower-quality enhancement layers.

It is wasteful to transfer enhancement layers to user devices that have not acquired the base layer, which triggers network congestion and negatively influences audiovisual continuity. To avoid peak traffic, the BS with AFC suspends forwarding requests for enhancement layers in the descending order of highest quality. Even when there is a drastic increase in requests, user devices can continue playing video streams at low quality, which is the least base layer that maintains user experience. When the network has spare bandwidth, the BS mitigates the suspending requests to improve the video quality.

Next, we explain the enhancement layer suspending mechanism using the congestion factor \( CF \). The congestion factor is given by Eq. (1) and indicates the occurrence of congestion when it exceeds 1. It can be broken down for each quality layer, and the congestion factor for a number of layers from 0 to \( N \) (0 and \( N \) represent the base and highest quality layers, respectively) is \( CF_N \). The AFC decreases the value of \( N \) by one until the congestion factor \( CF_M < 1 \), and then suspends forwarding requests for layers other than 0 to \( M \).

4. Selective Cache Eviction

AFC introduces an algorithm called selective cache eviction (SCE) to control cached contents and restrain peak traffic. Figure 4 illustrates an SCE concept with three levels of partitioned cache areas. SCE adopts an LRU-based cache eviction algorithm that is suitable for the chasing playback function. A reason for this adaptation is that recently broadcasted content tends to be requested more frequently [20]. We assumed that the downward trend of requests over time agrees well with Zipf’s law. Several studies have analyzed the request propensity and approximation of the Zipf distribution [21], [22]. A viable mechanism must prioritize the base layer and enhancement layers of lower quality rather than the enhancement layers of higher quality, including an LRU cache replacement policy that evicts the least recently used content in the cache.

SCE splits the cache area into sub-areas to be stored for different layers by prioritizing the cache replacement from sub-areas of higher-quality enhancement layers. Each sub-area’s capacity stretches in the range of the cache’s total capacity in the BS, based on the congestion factor.

When the congestion factor expects peak traffic, the AFC suspends request forwarding to cache nodes in an upper stream or the origin server from the enhancement layers with higher quality. Because layers with lower quality are more important than wasteful layers with suspended forwarding, SCE alters the separators of capacities for sub-areas based on the congestion factor. If the separator is changed, SCE will selectively replace the cache from older, higher-quality layers, and manage the contents of the sub-areas. SCE can (1) protect the important base layer and lower enhancement layers even in the presence of a frequently requested enhancement layer, and (2) maintain the...
user’s audiovisual quality without degradation.

Next, we discuss the mechanism of SCE’s separator decision. SCE determines the capacity of the sub-area by referring to the congestion factor obtained using Eq. (1). If the congestion factor is less than 1, the capacity of the sub-area is determined based on the ratio of the data size of each layer. The capacity of each sub-area at this point was retained as the basic value. If the congestion factor is greater than 1, the capacity of each sub-area is increased to the value obtained by multiplying the basic value by the congestion factor. Because the multiplied value is larger than the total cache capacity, SCE reduces the capacity from the sub-area that stores the high-quality layers to make it smaller than the total cache capacity.

5. Determination of Optimal Spectrum for D2D Network

D2D caching can reduce the traffic at the BS, thus suppressing the increase in the AFC’s congestion factor. In addition, if devices store content that AFC has suspended from forwarding, they deliver that content to the user.

In this section, we discuss a mathematical approach to determine the optimal spectrum factor for allocating content based on [7]. The cooperative cache of the spectrum influences its effectiveness, depending on the number of spectrum bits and their allocation.

5.1 Trade-Off between Cooperative Approach and Cache Hit Ratio

Figure 5 presents a schematic diagram of mobile networks covered by BSs. From the device perspective, the cache capacity’s effectiveness is maximized when devices in a local D2D network have different spectra. However, in an actual case, “node spectrum not found (also called as spectrum mismatch)” frequently occurs, which indicates that no device has the same node spectrum with the content spectrum to require users movement to exchange devices in the local D2D network. Because spectrum mismatch enables devices to send a request repeatedly to the BS to obtain contents, fine spectrum allocation increases the load of both the BS and the network. A challenge inherent in this approach is obtaining the equilibrium point of the spectrum length. In other words, the bit length of the bit vector expresses the spectrum, which minimizes the traffic, thereby balancing the trade-off between effective cache capacity and spectrum mismatch.

5.2 Method to Find Optimal Spectrum Length

We explain how to obtain the optimal spectrum length $S$ to minimize the expected traffic $T$. The formulation herein is an alternative to [7] without an optimized calculation.

The objective function $f$ derives $T$, which is an injective function with arguments $S$ and other parameters that express the environment applied to the D2D-based cooperative cache; the total number of devices under the BS $N$, BS communication range $R$, D2D communication range $r$, device cache capacity $c$, and content request propensity $p$.

$$T = f(N, R, r, c, p, S)$$

Coincidentally, $T$, which is the sum of sub-traffic $T'$, denotes the traffic for contents with each content spectrum bit in $S$, and in the case of the content spectrum, it is the “one-hot” that stores a cache of different devices exclusively.

$$T = S T'$$

$T'$ represents the total traffic between devices and BS, and it satisfies two conditions: (1) spectrum mismatch and (2) cache miss. If these conditions are not met, the traffic can be contained within the D2D level. Suppose that $P_s$ is the probability of the existence of a device with a spectrum that matches the target content spectrum in the D2D communication range, and $P_c$ is the probability of a cache hit of the device in the cache; the occurrence probabilities for each condition is denoted as $(1 - P_s)$ and $(1 - P_c)$, respectively. Note that $P_c$ is the cache hit rate, assuming all the devices in the neighborhood. $T'$ is obtained by multiplying $(1 - P_s)$ and $(1 - P_c)$ by $T_0 = T_0/S$, that is, the total traffic $T_0$ of the BS for each spectrum bit without D2D communication.

$$T' = [(1 - P_s) + P_s (1 - P_c)] \frac{T_0}{S}$$
\( n = \frac{N}{R^2} \) \( \text{(5)} \)

The expected value of \( n \) derived from \( N \), and the communication range, are given by:

\( n = \frac{N r^2}{R^2} \) \( \text{(6)} \)

\( P_s \) can calculate the probability from the binomial distribution of whether there are one or more devices with the desired node spectrum within the D2D communication range.

\[
P_s = 1 - \left( \binom{n}{0} \left( \frac{1}{S} \right)^0 \left( 1 - \frac{1}{S} \right)^n \right)
= 1 - \left( 1 - \frac{1}{S} \right)^n \quad \text{(7)}
\]

Although \( P_c \) cannot be expressed as a mathematical formula, its probability can be obtained from the simulation results with parameters \( S \), \( c \), and \( p \). In this study, we obtain \( P_c \) from a cache simulation implemented on an LRU.

The request propensity \( p \) varies, as expressed by \( S = 1 \) in Fig. 6, which presents the request probabilities for each contents. The graph fixed two parameters \( c \) for cache capacity in each device and the Zipf parameter \( \alpha \).

Therefore, we can rewrite Eq. (4) as Eq. (8).

\[
T' = \frac{1}{S} \left[ 1 - P_c \left( 1 - \left( 1 - \frac{1}{S} \right)^n \right) \right] T_0 \quad \text{(8)}
\]

According to Eq. (3) and Eq. (8), the traffic \( T \) introduced Eq. (9).

\[
T = \left[ 1 - P_c \left( 1 - \left( 1 - \frac{1}{S} \right)^n \right) \right] T_0 \quad \text{(9)}
\]

\( P_c \) is the integrated value in Fig. 6 from the most popular or most recent content (called Rank 1) to Rank \( c/S \), according to \( S \).

As \( T \) is the unimodal value calculated by multiplying the monotonically decreasing value of \( P_c \) and the monotonically increasing value of \( S \), according to \( S \), we can determine the minimum value of \( T \) by a binary search from a comparatively large \( S \).

5.3 Examples for Optimizing Spectrum Length

This subsection introduces some examples of optimizing the spectrum length to influence traffic, based on the above sub-section. Figures 7 and 8 illustrate the relationship between traffic and devices and spectrum length in terms of device density and request propensity, respectively.

Figure 7 presents the relationship between the number of devices and the spectrum length of traffic. The number of neighboring devices is expected to increase with an increase in spectrum length, and this reduces the probability of spectrum mismatch. Figure 8 presents the relationship between the deviation of request propensity by time passed after delivery and the spectrum length of traffic. Chen [22] analyzed the length and genre of a video by characterizing its request propensity. Because the risk of increased traffic owing to spectrum mismatch increases as the deviation of request propensity increases, the spectrum length is ex-
expected to decrease.

6. Experimental Setup

6.1 Environment

We considered a CDN environment to evaluate the effectiveness of AFC via a simulation. Figure 3 presents a CDN environment comprising three levels of hierarchical structure, an origin server located at the cloud level, multiple BSs at the BS level, and several devices at the device level. The CDN environment should be applied to event venues and sports stadiums to gather crowds with mobile audiovisual devices to watch chasing playback for entertainment. To demonstrate the effectiveness of AFC for peak traffic, we set up a scenario in which network congestion occurs.

Figure 3 also illustrates the flow of a device issuing requests to retrieve contents for chasing playback as follows:

1. A device issues a request for contents with the spectrum derived by the content ID for the neighboring devices with the device spectrum is the same as the node spectrum of the desired contents via D2D communication.
2. The device receives the request and returns the contents when the cache is hit. Otherwise, when the cache is missed, the device retrieves the content by forwarding the request to the BS with the node spectrum, as well as with the desired content spectrum.
3. The BS receives the requests, determines their cache, and returns the content when the cache is hit. When the cache is missed, the BS issues a request to the origin server.

We assume that the video content based on SVC comprises a base layer (BL) and two enhancement layers (EL1 and EL2). We define BL only, BL+EL1, and BL+EL1+EL2 as low quality, medium quality, and high quality, respectively. All devices request the highest-quality video, that is comprising all layers (BL+EL1+EL2). Furthermore, we divide all data into chunks of uniform data size based on low quality. Chunk division is an excellent way to utilize the cache capacity, although users need to obtain all the necessary chunks to make the data available [23].

6.2 Configuration

We implemented the simulator-based experimental environment described in Sect. 6.1. Table 1 presents the evaluation parameters. In the experimental CDN environment, the network bandwidth and traffic are configured, such that the network between the BS and the origin server poses a challenge. In the experimental scenario, we assumed a single live streaming environment that supports chasing playback. Every user can access the live streaming; however, the latest stream will be broadcast simultaneously. Therefore, we assume that the request during the experiment is solely for chasing playback. We generated traffic corresponding to Fig. 9 every unit of time and then repeat the process of acquiring the data. Patterns A and B are based on [2], and Pattern [24], respectively. In addition, these patterns (A and B) are based on actual traffic patterns. Figure 9 illustrates three traffic patterns: Patterns A and B based on [2], [24], respectively, as well as Pattern C, which we generated. Users interrupt the request when they return to the scene they were watching before starting the chasing playback. The streaming data consists of a total of 8192 video chunks. We set the bitrate of the video content based on [25], [26], and divided it into 10-second segments based on Apple’s live streaming standard [9]. In addition, we assume that the downward trend of requests over time agrees well with Zipf’s law [21], [22].

At each level of the CDN environment, devices and BSs exhibit spectra on different bitfields to store different contents. We considered each spectrum length for the number of BSs in the BS level and spectrum length mentioned in Sect. 5 in the device level. Each BS and device has one random bit for maximizing the effective cache capacity. Furthermore, each content has one bit from the BS level spectrum and one bit from the device level spectrum in chronological order.

The other parameters are based on the experimental settings of provided by the authors of [7].

Table 1 Evaluation parameters

| Parameter                                | Value       |
|------------------------------------------|-------------|
| Network bandwidth from origin            | 31.25 Gbps  |
| Number of contents                       | 8192        |
| Number of chunk divisions                | 10          |
| Layer data size (BL, EL1, EL2)           | 4, 12, 24 Mbps |
| Zipf’s bias parameter $\alpha$           | 0.68        |
| Radius of BS                             | 600 m       |
| Radius of D2D                            | 100 m       |
| Number of BS                             | 4           |
| Number of devices                        | 4096        |
| Cache capacity of BS                     | 6.25 GB     |
| Cache capacity of device                 | 625 MB      |
| Spectrum length in BS                    | 4 bits      |
| Spectrum length in devices               | 62 bits     |

Fig. 9 Number of users to request
7. Evaluation

7.1 Breakdown Traffic from the Origin Server

To clarify the effect of AFC in terminating request forwarding based on congestion, we simulated and measured the content type (BL, EL1, EL2) forwarded from the origin server. Figure 10 presents the obtained results. The traffic was measured from the origin server because it is a limitation. We normalized the forwarded content type and presented it as a stacked area graph with a limited path bandwidth of 1.

The AFC terminated forwarding requests based on congestion, and the bandwidth did not exceed one. From Fig. 10(a), it can be observed that suspending the EL2 transfer ensures that the BL is returned from the origin server.

In contrast, in the absence of AFC, the bandwidth exceeded one, thus triggering congestion, and content beyond one was not provided to the BS. As illustrated in Fig. 10(b), all content types exhibit a flat response shape during peak traffic. This indicates that all content types could not provide the content that they should have provided. The introduction of AFC selectively stops requests for high-quality content, thus circumventing peak traffic.

7.2 Breakdown of Quality Users Could Play

We measured the content received by the user device and the video quality that could be played back during the simulation in the previous section. For high-quality video playbacks, the user needs to receive all the dependent lower layers. This is an evaluation of the quality and continuity of the service itself. We evaluated service quality as the percentage of users who can receive and play the content of each quality.

Figure 11 shows the quality of the videos that the users played, and the percentage of users that played them over time. The blanks indicate the percentage of users who cannot receive even low-quality video and cannot play anything. In the presence of AFC, all users were provided with better than low-quality videos. In contrast, in the absence of AFC, several user devices failed to retrieve content owing to an increase in requests.

Next, we compare the effects of AFC, SCE, and D2D and present their evaluation in three different traffic patterns. Figure 12 shows the percentage of users when the time variation is discorded from Fig. 11 and summed. We discarded the time variation from Fig. 11 and only aggregated the time periods when one or more users could not watch videos at the highest quality. The obtained aggregated results are presented in Fig. 12. For comparison, we also included experimental results from a caching scheme based on [19]. This caching scheme determines the cache placement that minimizes the average data acquisition time. From left to right, the experimental results are as follows: (1) without both AFC and SCE, that is a simple LRU, (2) with AFC and SCE, that is our proposed method, (3) SCE without AFC, (4) the
8. Discussion

8.1 Effectiveness of AFC

This section demonstrates the fallback function provided by AFC, which is realized by avoiding peak traffic to provide services to all users based on the total amount and breakdown of traffic compared to the case without AFC. Figures 10(a) and 10(b) present the traffic from the origin server to BS, with and without AFC, respectively. In each case, Figs. 11(b) and 11(a) identify the proportion of users that can access the video quality.

In the case without AFC, as illustrated in Figs. 10(b) and 11(b), the occurrence of peak traffic congests to fill the capacity of the network, thus reducing the video quality. In addition, approximately 40% of users cannot play the content even at low quality in the time interval of the most drastic increase in requests. This case could not be considered as exhibiting continuity. Figure 10(b) shows that a negligible proportion of users play high-quality content, although BS forwards several requests for high-quality content. This discrepancy is derived from significantly useless content, by not gathering all content to play at high quality.

In contrast, as illustrated in Figs. 10 and 11, AFC avoids congestion in terms of peak traffic, thereby maintaining the quality provided to devices. The BS suspends request forwarding for the enhancement layers by detecting the increase in the congestion factor before peak traffic, as illustrated in Fig. 10. All users can play at least a base layer for low quality without any lack of content, and the rate of users obtaining higher quality is also higher than that without AFC.

An AFC feature includes suspending request forwarding for numerous and large contents of higher quality that utilize the capacity of the network between the BS and the origin server by selectively forwarding lower quality. This comparison indicates that the AFC introduces a service with higher quality and continuity, even in peak traffic.

8.2 Effectiveness of D2D Cache Network

This section discusses the effectiveness of the cooperative cache in a D2D network. Figure 13 demonstrates the effectiveness of the D2D cache network. The D2D cache network contributes to improving the quality provided to users, especially in peak traffic. Without the D2D cache network, the number of users who can retrieve the enhancement layer decreases. In traffic pattern B, the video quality of approximately 10% of the users was reduced, less than the case with D2D. Without an AFC, the effectiveness of the D2D network is negligible for peak traffic. Although the D2D cache network contributes to improving the quality, many users cannot retrieve any content in peak traffic. Therefore, a failsoft approach such as AFC is required to maintain service continuity.

8.3 Comparison with Existing Method

Figure 11 shows a service quality compared to existing method of [19] with combination of AFC. AFC can be applied independent from cache allocation techniques such as SCE and [19]. Therefore, AFC can be combined to [19] (4). By introducing AFC, both our method (2) and prior work [19] (5) improve the quality of service compared to BS without any device (1). In our method (2) compared to (1), all users can not only take some kind of all layers but also higher quality. The results of (2) and (4), that all users could take some layers, contribute continuity of service quality compared to (4) and (5), respectively. In (3) and (5) without AFC, peak traffic causes a situation that 30% to 50% of users could not take several or all layers. SCE, which changes the capacity for each layer following to con-
gestion factor, contributes to reducing number of users who take only lower layers compared to [19], as shown in results of (2) and (4). Sudden and rapid peak traffic such as Pattern B and C, SCE overcomes cache allocation optimization method of [19].

9. Conclusion and Future Work

In this study, we proposed a mechanism to avoid peak traffic under a mobile live streaming scenario where chasing playback is possible. The experimental evaluation verifies that the video quality fallback during peak traffic allows users to continue watching videos without interruptions. We have also demonstrated that the application of D2D can reduce the load on the BS and improve users’ viewing experience. Future work will consider the impact of the congestion factor and introduce a highly efficient cooperative cache among BS-D2D levels.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Number JP20H00592. In addition we were partly supported by TIS Inc. with the collaborative research project for highly efficient network technologies for a super-smart society.

References

[1] A. Ericsson, “Ericsson mobility report june 2018.” Recuperado de: https://www.ericsson.com/assets/local/mobility-report/documents/2018/ericsson-mobility-report-june-2018.pdf (Accedido el 09/09/2018), pp.3–13, 2018.
[2] Z. Li, M.A. Kaafar, K. Salamatian, and G. Xie, “Characterizing and modeling user behavior in a Large-Scale mobile live streaming system,” IEEE Trans. Circuits Syst. Video Technol., vol.27, no.12, pp.2675–2686, Dec. 2017.
[3] D.V.C. d. Silva, G. d. M.B. Domingues, P.B. Velloso, and A.A. d. A. Rocha, “Analysis of Mobile-Live-Users of a large CDN,” 2018 IEEE Symposium on Computers and Communications (ISCC), pp.00946–00951, June 2018.
[4] B.M. Maggs and R.K. Sitaraman, “Algorithmic nuggets in content delivery,” SIGCOMM Comput. Commun. Rev., vol.45, no.3, pp.52–66, July 2015.
[5] H. Li, L. Zhong, J. Liu, B. Li, and K. Xu, “Cost-Effective partial migration of VoD services to content clouds,” 2011 IEEE 4th Int. Conf. Cloud Computing, pp.203–210, July 2011.
[6] 3gpp, “LTE-Advanced: 3GPP release 12.” http://www.3gpp.org/specifications/releases/68-release-12, 2015. Accessed: 2017-10-19.
[7] T. Shiroma, C. Wu, and T. Yoshinaga, “A Template-Based Sub-Optimal content distribution for D2D content sharing networks,” 2018 Sixth Int. Symp. Computing and Networking (CANDAR), pp.167–173, Nov. 2018.
[8] “ISO/IEC 23009-1:2012,” https://www.iso.org/standard/57623.html, 2014. Accessed: 2021-1-7.
[9] R. Pantos and W. May, “HTTP live streaming,” https://tools.ietf.org/html/rfc8216, Aug. 2017. Accessed: 2021-1-5.
[10] H. Schwarz, D. Marpe, and T. Wiegand, “Overview of the scalable video coding extension of the H.264/AVC standard,” IEEE Trans. Circuits Syst. Video Technol., vol.17, no.9, pp.1103–1120, Sept. 2007.
[11] H. Kalvu, V. Adzic, and B. Furht, “Comparing MPEG AVC and SVC for adaptive HTTP streaming,” 2012 IEEE Int. Conf. Consum. Electron. (ICCE), pp.158–159, IEEEexplore.ieee.org, Jan. 2012.
[12] Y. Sanchez, T. Schierl, C. Helgile, T. Wiegand, D. Hong, D. De Vleeschauwer, W. Van Leekwijk, and Y. Le Louédëc, “Efficient HTTP-based streaming using scalable video coding,” Signal Processing: Image Communication, vol.27, no.4, pp.329–342, April 2012.
[13] Z. Zhang, D. Liu, and Y. Yuan, “Layered hierarchical caching for SVC-Based HTTP adaptive streaming over C-RAN,” 2017 IEEE Wireless Communications and Networking Conference (WCNC), pp.1–6, March 2017.
[14] K. Miller, D. Bethanabhotla, G. Caire, and A. Wolisz, “A Control-Theoretic approach to adaptive video streaming in dense wireless networks,” IEEE Trans. Multimedia, vol.17, no.8, pp.1309–1322, Aug. 2015.
[15] M. Claeys, S. Latré, J. Famaey, and F. De Turck, “Design and evaluation of a Self-Learning HTTP adaptive video streaming client,” IEEE Commun. Lett., vol.18, no.4, pp.716–719, April 2014.
[16] T.Y. Huang, R. Johari, M. McKeown, M. Trunnell, and M. Watson, “A buffer-based approach to rate adaptation: evidence from a large video streaming service,” Proc. 2014 ACM Conference on SIGCOMM, SIGCOMM ’14, New York, NY, USA, pp.187–198, Association for Computing Machinery, Aug. 2014.
[17] B. Fu, D. Staehle, G. Kunzmann, E. Steinbach, and W. Kellerer, “QoE-Based SVC layer dropping in LTE networks using Content-Aware layer priorities,” ACM Trans. Multimedia Comput. Commun. Appl., vol.12, no.1, pp.1–23, Aug. 2015.
[18] C.B. Ameeur, E. Mory, and B. Cousin, “Combining traffic-shaping methods with congestion control variants for HTTP adaptive streaming,” Multimedia Systems, vol.24, no.1, pp.1–18, Feb. 2018.
[19] C. Zhan and Z. Wen, “Content cache placement for scalable video in heterogeneous wireless network,” IEEE Commun. Lett., vol.21, no.12, pp.2714–2717, Dec. 2017.
[20] H. Abrahamsson and M. Björkman, “Caching for IPTV distribution with time-shift,” 2013 Int. Conf. Computing, Networking and Communications (ICNC), pp.916–921, Jan. 2013.
[21] A. Finamore, M. Mellia, M.M. Munafò, R. Torres, and S.G. Rao, “YouTube everywhere: impact of device and infrastructure synergies on user experience,” Proc. 2011 ACM SIGCOMM Conference on Internet Measurement Conference, IMC ’11, New York, NY, USA, pp.345–360, Association for Computing Machinery, Nov. 2011.
[22] Y. Chen, B. Zhang, Y. Liu, and W. Zhu, “Measurement and modeling of video watching time in a large-scale internet video-on-demand system,” IEEE Trans. Multimedia, vol.15, no.8, pp.2087–2098, Dec. 2013.
[23] N. Anjum, Z. Yang, H. Saki, M. Kiran, and M. Shikh-Bahaei, “Device-to-Device (D2D) communication as a bootstrapping system in a wireless cellular network,” IEEE Access, vol.7, pp.6661–6678, 2019.
[24] H. Abrahamsson and M. Nordmark, “Program popularity and viewer behaviour in a large TV-on-demand system,” Proc. 2012 Internet Measurement Conference, IMC ’12, New York, NY, USA, pp.199–210, Association for Computing Machinery, Nov. 2012.
[25] C. Kreuzberger, D. Posch, and H. Hellwagner, “A scalable video coding dataset and toolchain for dynamic adaptive streaming over HTTP,” Proc. 6th ACM Multimedia Systems Conference, MMSys ’13, New York, NY, USA, pp.213–218, Association for Computing Machinery, March 2015.
[26] “Choose live encoder settings, bitrates, and resolutions.” https://support.google.com/youtube/answer/2853702?hl=en&ref_topic=9257892. Accessed: 2020-8-26.
Hiroki Okada recived his B.E. degree from Kanagawa Institute of Technology in 2017 and his M.E. degree from the Graduate School of Informatics and Engineering, the University of Electro-Communications in 2019. He is currently working in the doctoral course of the Graduate School of Informatics and Engineering, the University of Electro-Communications.

Masato Yoshimi recived the B.E., M.E., and Ph.D. degrees from Keio University, Japan, in 2004, 2006, and 2009, respectively. He was Assistant Professor in Doshisha University since 2012 and in University of Electro-Communications since 2014. He currently works for TIS Inc. His research interests include the areas of reconfigurable computing and parallel processing.

Celimuge Wu received his M.E. degree from the Beijing Institute of Technology, China in 2006, and his Ph.D. degree from The University of Electro-Communications, Japan in 2010. He is currently an associate professor with the Graduate School of Informatics and Engineering, The University of Electro-Communications. His current research interests include computer networks, IoT, and mobile cloud computing. He has been a track co-chair or workshops co-chair for a number of international conferences including IEEE PIMRC 2016, IEEE CCNC 2017, ISNCC 2017 and WICON 2016.

Tsutomu Yoshinaga received his B.E., M.E., and D.E. degrees from Utsunomiya University in 1986, 1988, and 1997, respectively. From 1988 to July 2000, he was a research associate of Faculty of Engineering, Utsunomiya University. Since August 2000, he has been with the Graduate School of Information Systems, UEC, where he is now a professor. His research interests include computer architecture, interconnection networks, and network computing. He is a member of ACM, IEEE, and IPSJ.