Regular Paper

A Scalable Video-on-Demand System on Edge Computing Environments

Satoru Matsumoto[*] and Tomoki Yoshihisa[*]

[*]Cybermedia Center, Osaka University, Japan
{smatsumoto, yoshihisa}@cmc.osaka-u.ac.jp

Abstract -Due to the recent increase in the users of video-on-demand (VoD) services, many clients such as smart phones or laptop computers request video data to a video distribution server. Such large-scale VoD systems utilize the edge computing technology to distribute the communication load and the processing load on the video distribution server. In most of the VoD systems utilizing edge computing, the edge servers receive a part of the video data from the video distribution server and caches them for other transmissions. However, the edge servers can receive them before receiving the requests from the clients (pre-cache). Moreover, the edge servers can transmit pre-cached video pieces to other edge servers. Here, the research challenges are: which pieces the edge servers should pre-cache and how to redistribute the pieces among the edge servers to effectively distribute the loads. In this paper, we propose a scalable VoD system on edge computing environments. Our evaluation results revealed that our proposed system reduces the probability that the transmissions overlap with other transmissions and increases the shortest arrival interval of the clients that the system can work by 26% compared with the conventional system in the simulated situation.

Keywords: Streaming media distribution, interruption time, webcasting, Internet broadcasting, cloud computing

1 INTRODUCTION

Recently, video-on-demand (VoD) services such as YouTube or Netflix are widely used. In most VoD services, the clients request the video data to the video distribution server. The video distribution server sends the video data pieces sequentially to the clients so that they can play the video while receiving the pieces. When the clients cannot finish receiving each piece before starting playing it, the video playback is interrupted. A longer interruption time more annoys the viewers. Here, the interruption time means the total time that the playback is interrupted while a client is playing the video. Therefore, various interruption time reduction schemes for VoD services have been proposed.

In large-scale VoD systems, many clients request the video data and thus the transmissions of data pieces frequently overlap with other transmissions. Here, the transmission means a series of distributing the pieces to a client. A transmission starts when a client requests playing a video to the video distribution server and finishes when the client finishes receiving all the pieces of the video data. If the times required for transmissions increase and the transmissions continue to overlap with others, the interruption times also continue to lengthen. Therefore, existing schemes for interruption time reduction aim to reduce the transmission time to avoid the overlapping of transmissions. Major techniques for this are pre-caching ([1]-[4]), redistributions of data pieces ([5]-[7]), etc. and are adopted in various CDN (Contents Delivery Networks) for scalable VoD systems ([8]). Unfortunately, these traditional approaches cause the communication and processing loads on the clients. Such extra loads decline the users' operability of the clients and further consume the batteries if they are mobile devices.

Edge computing is one of the approaches that relief the computational loads on the clients since the edge servers, i.e., the servers on the edge of the network and geometrically close to the clients, are often managed by CDN companies such as Akamai or Cloudflare. In most of the VoD systems utilizing edge computing, the edge servers receive some pieces from the video distribution server and caches them for other transmissions. However, the number of the pieces that the edge servers need to send decreases by adopting the above both techniques (pre-caching and redistributions) to the edge servers. Here, the research challenges are: which pieces the edge servers should pre-cache and how to redistribute the pieces among the edge servers to effectively reduce the communication load and the processing load.

In this paper, we propose a scalable VoD system on edge computing environments. Our proposed system adopts a distributed edge caching scheme. In our proposed system, the edge servers store some pieces before starting the VoD service. When a new client requests a video data, the video distribution server selects an edge server for sending the pieces of the requested video data to the client. The edge server sends its stored pieces to the client. After finishing sending the stored pieces, the video distribution server sends the subsequent pieces to the client. Thus, our proposed system reduces the probability that the transmissions overlap with other transmissions and increases the maximum number of the clients of that interruption time converges. The novelty of the proposed system is the adopting both the pre-caching technique and the redistribution technique to the VoD systems utilizing edge computing. The contributions of the paper are (1) the increase of the number of the clients that the system can accommodate, (2) the proposition of a scalable video-on-demand system, (3) the confirmation of the effectiveness of the proposed system.

The paper is organized as follows. Some related work are introduced in Section 2. The proposed scheme is explained in Section 3, analyzed in Section 4, and evaluated in Section 5. Finally, we conclude the paper in Section 6.

2 RELATED WORK

Many studies focus on fast data reception for VoD services.

2.1 Pre-caching Techniques for VoD Services

Abuhadra et al. proposed a proactive caching technique for mobile devices [1]. The probability that the clients encounter interruptions decreases by sending more video data to the mobile devices while their network connections are available because they sometimes disconnect from the network. The proposed technique reduces in-network transmission delays by caching the video data. Feng et al. found the optimal cache placement for the system with wireless multicasting [2]. My research group proposed a broadcasting method for pre-caching video data pieces predicting the video data that the client will play [3]. Coutinho et al. proposed a proactive caching technique for DASH video streaming [4]. In the proposed technique, the clients select a proxy caching server based on the network conditions. However, these pre-caching techniques for VoD services require the clients' storage capacity.

2.2 Redistribution (Peer-to-Peer Sharing) Techniques for Video-on-Demand Services

Sheshjavani et al. proposed a peer-to-peer data sharing mechanism for VoD services [5]. In the proposed mechanism, the clients manage their buffer map. The buffer maps inscribe the received video data pieces and non-received pieces of each client. In the mechanism, the clients exchange the pieces based on the buffer map to receive the pieces that each client does not have. In the method proposed by Zhang et al., the clients send the pieces considering the bandwidth consumption [6]. Fratini et al. analyzed the efficiency of using replicated video servers [7].

However, similar to the pre-caching techniques, these redistribution techniques cause communication and processing loads on the clients. Such extra loads decline the users' operability of the clients and consume the batteries if they are mobile devices.

2.3 Video-on-Demand Services on Edge Computing

Due to the recent prevalence of edge computing, some researchers focus on edge computing for VoD services. Mehrabi et al. proposed an edge computing assisted adaptive mobile video streaming [9]. The mechanism determines the video resolution and the video data rate based on the network conditions between the clients and the edge servers. The performance of an edge computing enhanced video streaming is investigated by Yang et al. [10].

3 PROPOSED SYSTEM

In this section, we explain our proposed system.

3.1 Proposed System Architecture

Figure 1 shows the assumed system. The system consists of three layers, the top layer, the edge layer, and the client layer. One video distribution server is in the top layer. CDN (Contents Delivery Network) companies or VOD service companies provide the machines in the edge layer. The edge layer includes some edge servers. The clients are in the client layer. Similar to other researches for the VoD systems utilizing edge computing, the networks for these three layers are application layer networks and we assume that the influences of the underlaying session/transport layers are sufficiently small.

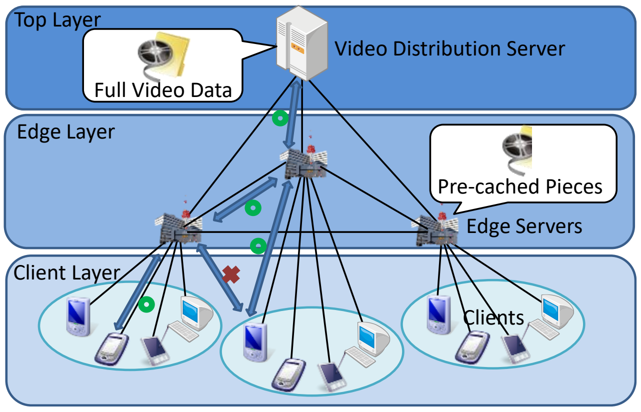


Figure 1: Assumed Environment

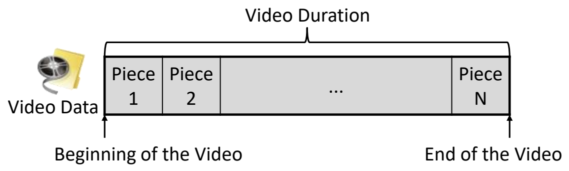


Figure 2: Video Data Division

The single video distribution server has full video data for all the videos and connects to the Internet. The edge servers also connect to the Internet and can communicate with the video distribution server. They can store a part of some video data (pieces). The clients connect to the geometrically closest edge server and can communicate with the edge server. The clients request the video data to the video distribution server. Each video data are divided into some pieces, as shown in Fig. 2. The clients receive the requested video data pieces from the video distribution server and the edge servers.

The assumed system model is general and practical. One of the applications is that the video distribution server provides the videos to the clients in a prefecture, and each edge server serves for each region in the prefecture. For example, there are eight local regions in Osaka, Japan. In this case, the number of the edge servers is eight, and the edge servers provides 100 videos.

3.2 Target Issue

We explain our target issue in this subsection.

3.2.1 Issues in Conventional Systems

In the VoD systems, the video playback is interrupted when the clients cannot finish receiving each piece before starting playing it. A longer interruption time more annoys the viewers. Here, the interruption time means the total time that the playback is interrupted while a client is playing the video. In the VoD system in that a video distribution server distributes the video data to the clients, the communication load and the processing load for the distribution concentrate on the server. Therefore, in the cases that such VoD system hold many clients (scalable) and the clients frequently requests to play the videos to the video distribution server, the server is easy to overload. The overloading results in long video data transmission times and causes long interruption times. In most of the VoD systems utilizing edge computing, the edge servers receive pieces from the video distribution server and caches them for other transmissions.

3.2.2 Pre-Caching Pieces

The edge servers can receive pieces from the video distribution server before the requests for them come from the clients. That is, the edge servers can pre-cache the pieces. The pre-caching can reduce the communication load and the related processing load on the video distribution server and the edge servers. This is because they receive the pieces without the requests for the pieces and can receive them before starting the VoD service. Pre-caching more pieces reduce more loads, but require more storage capacity on the edge servers.

3.2.3 Redistributing Pieces

Moreover, the edge servers can redistribute pieces to other edge servers. The redistribution can distribute the communication load and the related processing load on the video distribution server to the edge servers because the edge servers send the pieces instead of the server. However, if an edge server frequently redistributes the pieces, the loads concentrate on the edge server, results in long interruption time. Therefore, the edge servers need to redistribute the pieces without causing the overloads on it.

3.2.4 Our Objective

Based on the discussion in this subsection, we aim to reduce the interruption time by adopting pre-caching and redistributing pieces on edge computing environments. For this, we propose a scheme which determines which pieces the edge servers should pre-cache and how to redistribute the pieces among the edge servers.

3.3 Proposed Scheme

Our proposed scheme runs on our proposed system architecture explained in Subsection 3.1. In the scheme, the edge servers pre-cache several preceding pieces of popular videos. When a client requests playing a video to the connected edge server, the edge server checks whether it has pre-cashed the requested video already. If the piece that the client is going to receive is pre-cached, the client receives the piece from the connected edge server. If the piece is pre-cached by another edge server, the connected edge server receives it from the edge server and after that sends it to the client. If the pieces is not pre-cached by any edge server, the connected edge server receives it from the video distribution server and after that sends it to the client.

3.3.1 How to Pre-Cache Pieces

To solve the first issue, the edge servers pre-cache preceding several pieces of popular videos, i.e., the pieces that are close to the beginning of the videos. This is because the time to start playing the preceding pieces is early, and the possibility that the clients encounter interruptions is high. The preceding pieces of the videos that are frequently requested by the connected clients are pre-cached. To avoid redundant pre-caching, the edge servers do not pre-cache the pieces that are pre-cached by other edge servers. The edge servers do not cache the pieces that are transmitted to the clients except for the pre-cached pieces to reduce the required storage capacity for caching. The number of the pieces to be pre-cached is a parameter for the scheme.

For example, imagine the case that the VoD system provides 100 videos and owns eight edge servers. When the number of the pieces to be pre-cached is set by 10 per each video, each edge server pre-caches the preceding 10 pieces of the 100/8=25 videos.

3.3.2 How to Redistribute Pieces

To solve the second issue, in the cases that the edge server does not pre-cache the requested video, it requests the redistribution of the preceding pieces of the video to the edge server that pre-caches the requested video. This is because the edge layer and the client layer are separated in our proposed system, and thus, the clients cannot communicate with the edge servers that they do not belong to. Therefore, the edge servers redistribute the pieces to other edge servers, not directly to the clients.

For example, imagine the case that the client 1 directly connects to the edge server 1 and requests the video 2. The preceding pieces of video 2 is pre-cached by the edge server 2. In this case, the edge server 2 redistribute the pre-cashed pieces of the video 2 to the edge server 1 and the edge server 1 sends the received pieces to the client 1.

3.4 Flow of Procedures

In this subsection, we explain the flow of the procedures for the video distribution server, the edge servers, and the clients.

3.4.1 Video Distribution Server

The video distribution server has all the pieces of all the video data. When it receives a request for a piece, it sends the requested piece to the requesting edge server.

Moreover, the video distribution server manages the statistics of the VoD system and measure the popularity of the video data to determine which video data each edge server should pre-cache. This can be performed by calculating popularity of the video data. The video distribution server can calculate this by getting the information about the video requests from all the edge servers.

3.4.2 Edge Servers

Figure 3 shows the procedure flow of the edge servers. In the figure, $p_{j,i}$ indicates the ith piece of the video j. When an

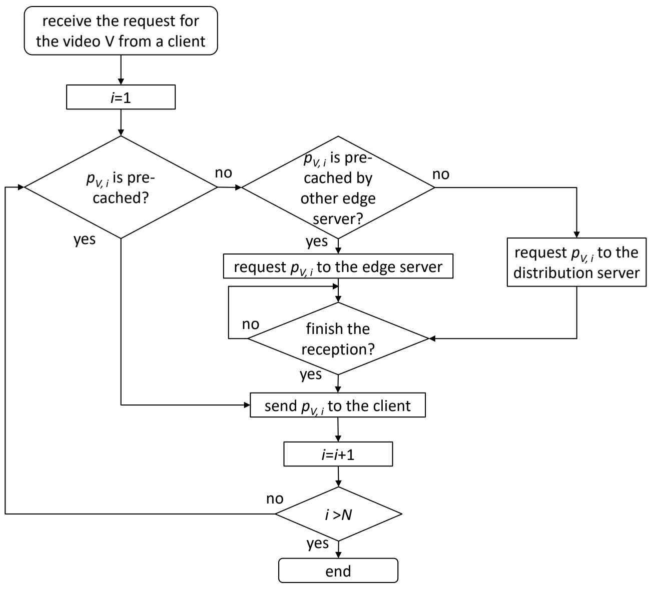

receive the request for the video V from a client
i=1
$p_{V,i}$ is pre-cached?
no
yes
$p_{V,i}$ is pre-cached by other edge server?
no
yes
request $p_{V,i}$ to the edge server
request $p_{V,i}$ to the distribution server
finish the reception?
no
yes
send $p_{V,i}$ to the client
i=i+1
i >N
no
yes
end

Figure 3: The procedure flow of the edge servers

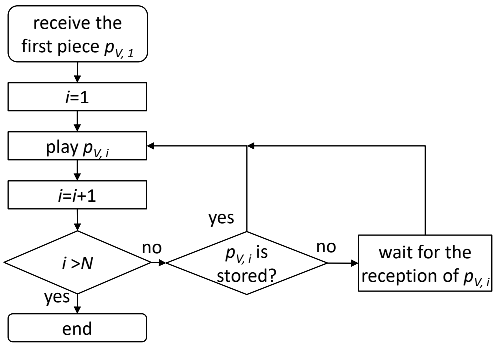


Figure 4: The procedure flow of the clients

edge server receives the request for the video from a client, it checks whether it pre-caches the first piece of the video or not. If the edge server pre-caches the piece, it sends the piece to the client. Otherwise, it finds the edge server that pre-caches the piece. If there is the edge server that pre-caches the piece, it requests the piece reception to the edge server and waits for the reception. When the reception completes, it sends the received piece to the client. If no edge servers pre-cache the piece, it request the piece reception to the video distribution server and sends it to the client when the piece reception completes. The edge server that receives the request of the video continue to this procedure until it sends the last piece. When it completes sending all the pieces of the video to the client, the flow for the request finishes.

3.4.3 Clients

Figure 4 shows the procedure flow of the clients. Similar to Fig. 3, $p_{j,i}$ indicates the ith piece of the video j. When a client receives the first piece, it starts playing the piece. After playing the piece, the client try to continuously play the next piece and checks whether it has stored the next piece or not. If the next piece has already stored in its storage, it starts playing the next piece. Otherwise, it waits for the reception of the next piece. In this case, interruption occurs. The client continue to this procedure until finishing playing all the pieces.

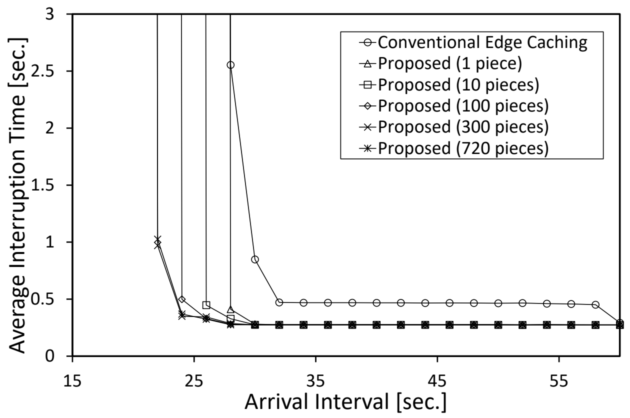


Figure 5: Average arrival interval and the interruption time

4 MATHEMATICAL ANALYSIS

In this section, to find the necessary bandwidth among the video distribution server, the edge servers, and the clients, we analyze the communication situation under the proposed scheme.

In our proposed scheme, each video data is divided into some pieces. The data amount for each piece is the same and is denoted by P. For example, if the number of the pieces of the video i is NP_i, the data size of the video is $P \times NP_i$.

4.1 Necessary Bandwidth Between Clients and Edge Servers

Suppose the case when an edge server receives the new video request before finishing sending all the pieces to the current client. In this case, the interruption time diverges since the piece transmissions overlap. The average arrival interval of the clients for an edge server is given by λE. λ is the average global arrival interval of the request for the videos from the clients, and E is the number of the edge servers. Therefore,

$$\frac{PM}{B_{EC}} < \lambda E\,. \qquad (1)$$

Here, M is the maximum number of the pieces of all the videos. B_{EC} is the bandwidth between the edge server and the clients. For the analysis, I assume that the bandwidth is the same for all the edge servers. From (1), the following inequality should be satisfied to converge the interruption time.

$$B_{EC} > \frac{PM}{\lambda E} \qquad (2)$$

4.2 Necessary Bandwidth Among Edge Servers

We assume that the video data are requested with a same probability. Each edge server has the same number of the videos. Let V denote the number of the videos that the system provides. Then, the probability that a video is requested when a client requests playing a video is $1/V$. Therefore, the probability that the video data that an edge server has are requested is $(V / E) / V = 1/E$. Contrary, the probability that the directly connected edge server does not have the requested video data is $(E - 1)/E$. Therefore, a redistribution occurs with the probability $(E - 1)/E$. The time that the edge server can consume for the redistribution is λE if the edge server does not have the next requested video. The probability that the edge server does not have the next requested video is (E - 1)/E. If the edge server has the next requested video (the probability is 1/E), the time that the edge server can consume for the redistribution is $2\lambda E$. Thus, the average time that the edge server can consume for the redistributions is given by

$$\sum_{i=1}^{\infty} i\lambda E\left(\frac{E-1}{E}\right)^2\left(\frac{1}{E}\right)^{i-1}. \tag{3}$$

Let J the number of the pieces allocated to each edge server. The time needed to send J pieces among the edge servers should be shorter than this average arrival interval to avoid transmission overlapping. Therefore,

$$\frac{PJ}{B_{EE}} < \lambda(E-1)^2\sum_{i=1}^{\infty} i\left(\frac{1}{E}\right)^{i}. \tag{4}$$

Hence, the bandwidth among the edge servers B_{EE} should satisfy the following inequality.

$$B_{EE} > \frac{PJ}{\lambda(E-1)^2\sum_{i=1}^{\infty} i\left(\frac{1}{E}\right)^{i}} \tag{5}$$

4.3 Necessary Bandwidth for Video Distribution Server

The edge servers receive the pieces that are not allocated to any other edge servers from the video distribution server. The time needed to send $M - J$ pieces from the video distribution server to the edge server should be shorter than the average global arrival interval. Therefore,

$$\frac{P(M-J)}{B_{DE}} < \lambda. \tag{6}$$

B_{DE} is the bandwidth between the video distribution server and the edge servers. Hence, the following inequality should be satisfied to converge the interruption time.

$$B_{DE} > \frac{P(M-J)}{\lambda} \tag{7}$$

5 EXPERIMENTAL EVALUATION

To check the performances of the proposed scheme, we measured the interruption time using our developed simulator.

5.1 Evaluation Setting

Based on the application example in Subsection 3.1, the number of the edge servers is eight, and the edge servers provide 100 videos. The bandwidth between each edge server and the clients is 100 [Mbps] considering a realistic situation. The bandwidth between the video distribution server and the edge servers is 600 [Mbps], and that among the edge servers is also 600 [Mbps], considering that these are in the backbone network. I set the same bandwidth to all the edge servers to make the experiments precisely understandable. The video distribution server can communicate with each edge server directly, and the edge servers can communicate with each other. The clients connect to the closest edge server.

The video duration is 60 [min.], and the bitrate is 5 [Mbps] based on the videos provided by practical services. The data amount of a piece is the same as the video data for 5 [sec.] based on HLS (HTTP Live Streaming [15]) and is 3125 [Kbytes]. The number of the pieces in each video data is 720.

The users request playing one of the videos according to a fixed arrival interval to make the results be easily understandable. In the case that the requests non-uniformly arrive, the average interruption time increases because the maximum value increases. The popularity of the video data is given fairly.

We compare the proposed scheme with a conventional edge caching scheme, an often used caching technique for CDN. In the scheme, the pieces are cached at the edge servers, but not redistributed among them. The edge servers receive the pieces that need to be sent to the clients and are not cached from the video distribution server.

5.2 Influence of Arrival Interval

More frequent arrivals of the requests for playing the video data cause more transmission overlaps, and thus, the interruption time can continue to increase with a higher probability. Therefore, we investigate the influence of the arrival intervals of the clients' requests.

5.2.1 Interruption Time

Figure 5 shows the average interruption time under different arrival intervals. The horizontal axis is the arrival interval and the vertical axis is the average interruption time. In the legend, `Conventional Edge Caching' indicates the average interruption time under the conventional edge cashing scheme explained in the previous subsection. `Proposed (J pieces)' indicated the average interruption time under our proposed scheme. In the scheme, each edge server pre-caches J pieces.

We can see that the average interruption times under each scheme are almost the same when the arrival interval is longer than a certain value. This is because the transmissions does not overlap with others and the interruption time does not continue to increase. Under the conventional scheme, the average interruption time when the arrival interval is longer than 30 [s] is a little bit longer than that under our proposed scheme. This is because the bandwidth between the edge servers and the clients are not sufficient and the transmissions sometimes overlap with others. On the other hand, we can see that the average interruption times under all schemes suddenly increases when the arrival interval is shorter than a certain value. This is because the transmissions always overlap with the next transmission and the interruption time continue to lengthen. In such cases, the VoD system abandons and cannot provide their services.

For example, when the arrival interval is 25 [s], the conventional scheme cannot provide the service, but our proposed scheme can provide it and the average interruption time is approximately 325 [ms]. At the shortest, our proposed system

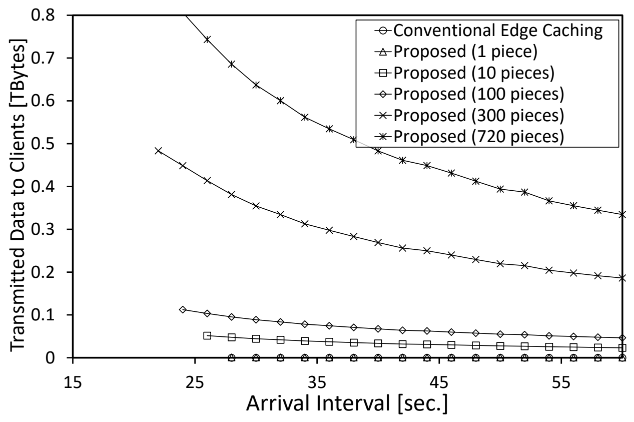


Figure 6: Average arrival interval and the transmitted data to the clients

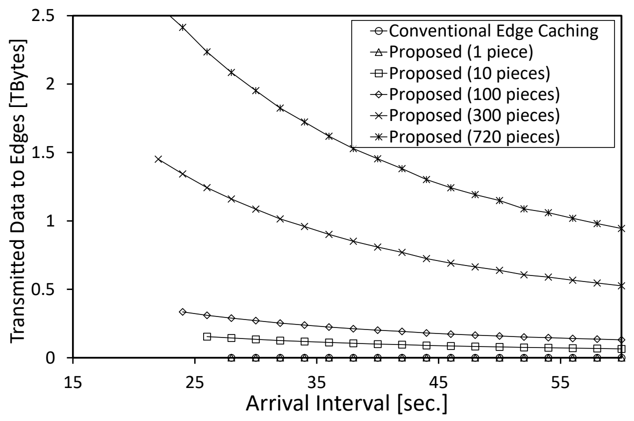


Figure 7: Average arrival interval and the transmitted data to the edge servers

can work even when the arrival interval is 22 [s]. The conventional method can provide service only when the arrival interval is longer than 30 [s] in this situation. Therefore, our proposed system can improve the shortest arrival interval under that the system can work 26% compared with the conventional system.

5.2.2 Edge Servers' Data Transmission

One of the indexes for the communication load and the processing load on the edge servers is the data amount transmitted to others. Therefore, we investigate the amount changing the arrival interval.

Figure 6 shows the total data amount transmitted to the clients by the edge servers. Since the average interruption time diverges when the arrival interval is excessively short, the lines stop at the shortest arrival interval under that the interruption time converges. The data amount decreases as the arrival interval increases because the number of the clients that receive data per time decreases. The data amount increases as the edge servers pre-cache more data because they need to transmit them instead of the video distribution server.

Figure 7 shows the total data amount transmitted to other edge servers. The result is similar to Fig. 6, the total data amount transmitted to the clients. But, the amount differs. The data amount transmitted to other edge servers is generally larger than that to the clients because the edge servers request the data transmissions to other edge servers in proportional to

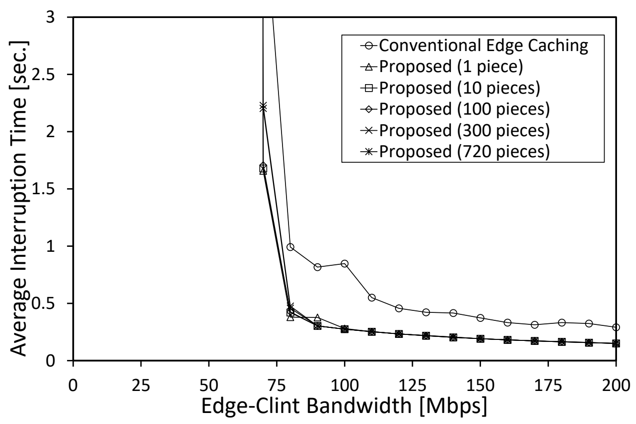


Figure 8: Edge-client bandwidth and the interruption time

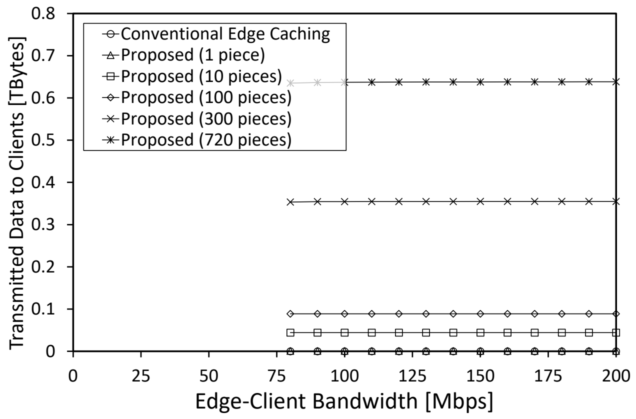


Figure 9: Edge-client bandwidth and the transmitted data to the clients

the number of the clients and each edge server respond to the requests from all the other edge servers.

We only show the total data amount transmitted to the clients because we confirmed that the total data amount transmitted to the edge servers is similar to this.

5.3 Influence of Edge-Client Bandwidth

A less communication bandwidth between the edge servers and the clients (edge-client bandwidth) cause more transmission overlaps, and thus, the interruption time can continue to increase with a higher probability. Therefore, we investigate the influence of the bandwidth between the edge servers and the clients.

Figure 8 shows the average interruption time under different client-edge bandwidth. The arrival interval is 30 [s]. The horizontal axis is the client-edge bandwidth, i.e., the bandwidth between each edge server and the clients. The vertical axis is the average interruption time. We can see that the average interruption times under our proposed scheme are almost the same when the edge-client bandwidth is larger than a certain value. Similar to the discussion for the previous subsection, this is because the transmissions does not overlap with others when the edge-client bandwidth is large. For the same reason as the previous subsection, the average interruption time under the conventional scheme is a little bit longer than that under our proposed scheme. Since the behavior between the edge servers and the clients are the same between

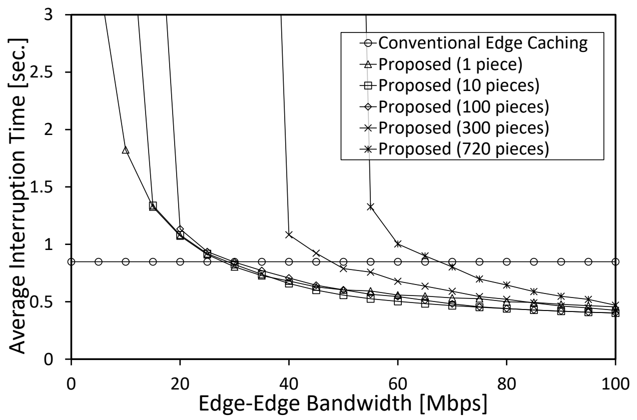


Figure 10: Edge-edge bandwidth and the interruption time

Table 1: Transmitted data to the clients under Figure 10.

Pre-cached pieces	0	1	10	100	300	720
Transmitted Data Amount	0	888 [Mbytes]	44.4 [GBytes]	88.8 [GBytes]	354 [GBytes]	636 [GBytes]

our proposed scheme and the conventional scheme, the edge-client bandwidth under that the interruption time diverges is the same, approximately 75 [Mbps].

Figure 9 shows the total data amount transmitted to the clients. Since the average interruption time diverges when the edge-client bandwidth excessively small, the lines stop at the smallest bandwidth under that the interruption time converges. The total data amount does not change even when the edge-client bandwidth changes because the data amount only depends on the number of the clients and the number of the pre-cached pieces.

5.4 Influence of Edge-Edge Bandwidth

A less communication bandwidth among the edge servers (edge-edge bandwidth) cause more transmission overlaps. Thus, the interruption time can continue to increase with a higher probability.

Figure 10 shows the average interruption time. The arrival interval is 30 [s]. The horizontal axis is the edge-edge bandwidth. The average interruption time under the conventional scheme is constant even when the edge-edge bandwidth changes because the edge servers do not redistribute the pieces in the scheme. The average interruption time under our proposed scheme decreases as the edge-edge bandwidth increases because the edge servers can faster redistribute the pieces to another edge server. Moreover, the average interruption time continue to lengthen when the edge-edge bandwidth is smaller than a certain value because the transmissions overlap.

The total data amount transmitted to the clients and the edge servers do not depend on the edge-edge bandwidth and these have a similar tendency as shown in Figs. 6 and 7. Therefore, we show the total data amount transmitted to the clients for the situation of Fig. 10 in Table 1. As shown in the table, the data amount increases as the number of the pre-cached pieces increases because the edge servers own more data. At the maximum case, i.e., the edge servers pre-cache all the 720 pieces, the data amount is 636 [GBytes] for 100 video data.

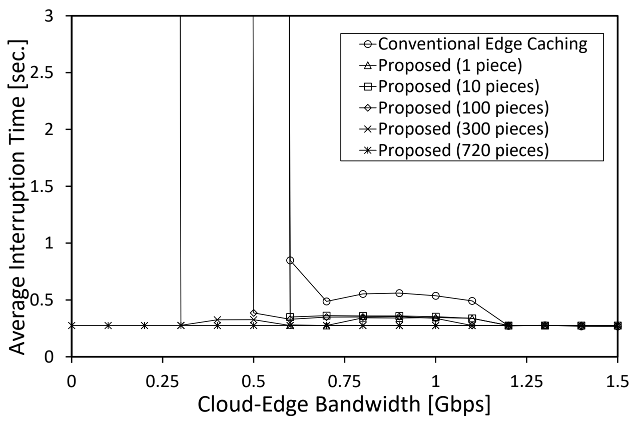


Figure 11: Cloud-edge bandwidth and the interruption time

5.5 Influence of Cloud-Edge Bandwidth

A less communication bandwidth between the video distribution server and the edge servers (cloud-edge bandwidth) cause more transmission overlaps. Therefore, we investigate the interruption time changing the cloud-edge bandwidth.

Figure 11 shows the average interruption time. The arrival interval is 30 [s]. The horizontal axis is the cloud-edge bandwidth. The average interruption time under the conventional scheme is longer than that under our proposed scheme because the bandwidth between the edge servers and the clients are not sufficient and the transmissions sometimes overlap with others as shown in Fig. 5. Similar to previous results, the average interruption time suddenly increases when the cloud-edge bandwidth is smaller than a certain value because the transmissions continue to overlap. The cloud-edge bandwidth under that the average interruption time converges is larger as the number of the pre-cached pieces increases because the edge servers receive less pieces from the video distribution server as the pre-cached pieces increase. Since the edge servers pre-cache all the pieces when they pre-cache 720 pieces, the average interruption time does not change even when the cloud-edge bandwidth changes in this case.

5.6 Influence of Edge Servers

The load on the edge servers can distribute as the system equips with a larger number of the edge servers. However, it takes much monetary cost as the system equips with a larger number of the edge servers. Therefore, we investigate the interruption time changing the number of the edge servers.

Figure 12 shows the result. The arrival interval is 30 [s] and the horizontal axis is the number of the edge servers. We can see that the average interruption time is almost constant under each scheme when the number of the edge servers is large because the transmissions do not overlap if the system equips with a sufficient number of the edge servers. Otherwise, the transmissions overlap and the interruption time continue to lengthen. For example, in the evaluated situation, the interruption time diverges when the number of the edge servers is less than 6. Therefore, it is better for the system to find the appropriate number of edge servers under that the interruption time converges.

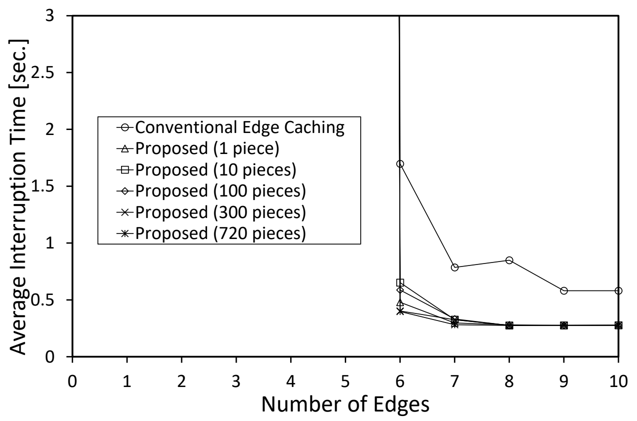


Figure 12: Number of the edges and the interruption time

5.7 Summary of Evaluation Results

We confirmed that the average interruption time did not largely depend on the number of the pre-cached pieces when the communication network was not congested, i.e., a longer arrival interval, a larger bandwidth (edge-client/edge-edge/cloud-edge). Meanwhile, the arrival interval or the bandwidths under that the system could work, i.e., the average interruption time converges, could be improved by pre-caching more pieces on the edge servers. This was because the transmissions did not overlap with others when the communication network was not congested. On the other hand, a larger number of pre-cached pieces causes more computational loads on the edge servers.

6 CONCLUSION

In this paper, we proposed a scalable VoD system on edge computing environments. Our proposed system adopted the pre-caching and the redistribution techniques. Our proposed system reduces the probability that the transmissions overlap with other transmissions and increases the maximum number of the clients of that interruption time converges. Our simulation evaluation revealed that our proposed system can transmit the video data more than the conventional scheme. Our proposed system can improve the shortest arrival interval under that the system can work by 26% compared with the conventional system in the simulated situation.

In the future, we will consider the popularity of the videos and adopt the dynamic caching technique to the edge servers. Moreover, we will consider the communication loads on the edge servers and the distribution server.

ACKNOWLEDGMENTS

This work was partially supported by JSPS KAKENHI Grant Numbers JP21H03429, JP18K11316, and by G-7 Scholarship Foundation.

REFERENCES

[1] R. Abuhadra and B. Hamdaoui, “Proactive In-Network Caching for Mobile On-Demand Video Streaming,” in Porc. IEEE International Conference on Communications (ICC), pp. 1-6 (2018).

[2] H. Feng, Z. Chen, H. Liu, and D. Wang, “Optimal Cache Placement for VoD Services with Wireless Multicast and Cooperative Caching,” in Proc. IEEE Wireless Communications and Networking Conference (WCNC), pp. 1-6 (2018).

[3] Y. Gotoh, T. Yoshihisa, and M. Kanazawa, Y. Takahashi, “A Broadcasting Protocol for Selective Contents Considering Available Bandwidth,” IEEE Transactions on Broadcasting, Vol. 55, No. 2, pp. 460–467 (2009).

[4] R. Coutinho, F. Chiariotti, D. Zucchetto, and A. Zanella, “Just-in-time proactive caching for DASH video streaming,” in Proc. Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net), pp. 1-6 (2018).

[5] A.G. Sheshjavani, B. Akbari, and H.R. Ghaeini, “An Adaptive Buffer-Map Exchange Mechanism for Pull-based Peer-to-Peer Video-on-Demand Streaming Systems,” Springer International Journal of Multimedia Tools and Applications, Vol. 76, No. 5, pp. 7535–7561 (2016).

[6] Y. Zhang, C. Gao, Y. Guo, K. Bian, X. Jin, Z. Yang, L. Song, J. Cheng, H. Tuo, and X.M. Li, “Proactive Video Push for Optimizing Bandwidth Consumption in Hybrid CDN-P2P VoD Systems,” in Proc. IEEE International Conference on Computer Communications (INFOCOM), p. 2555-2563 (2018).

[7] R. Fratini, M. Savi, G. Verticale, and M. Tornatore, “Using Replicated Video Servers for VoD Traffic Offloading in Integrated Metro/Access Network,” in Proc. IEEE International Conference on Communications (ICC), pp. 3438-3443 (2014).

[8] E. Ghabashneh, S. Rao, “Exploring the Interplay between CDN Caching and Video Streaming Performance,” in Proc. IEEE Int’l Conference on Computer Communications (INFOCOM), p. 2555-2563 (2018).

[9] A. Mehrabi, M. Siekkinen, and A. Ylä-Jääski, “Edge Computing Assisted Adaptive Mobile Video Streaming,” in IEEE Transactions on Mobile Computing, Vol. 18, No. 4, pp. 787-800 (2019).

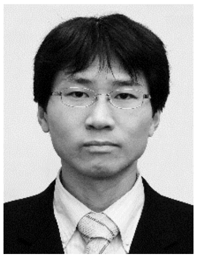
[10] S. Yang, Y. Tseng, C. Huang, and W. Lin, “Multi-Access Edge Computing Enhanced Video Streaming: Proof-of-Concept Implementation and Prediction/QoE Models,” in IEEE Transactions on Vehicular Technology, Vol. 68, No. 2, pp. 1888-1902 (2019).

[11] J.-P. Hong, W. Choi, “User Prefix Caching for Average Playback Delay Reduction in Wireless Video Streaming,” IEEE Transactions on Wireless Communications, Vol. 15, Issue 1, pp. 377-388, 2015.

[12] RFC 8216, HTTP Live Streaming, https://tools.ietf.org/html/rfc8216 (2017).

(Received: November 8, 2021)
(Accepted: December 24, 2021)

Satoru Matsumoto received his Diploma’s degrees from Kyoto School of Computer Science, Japan, in 1990. He received his Master’s degree from Shinshu University, Japan, in 2004. From 1990 to 2004, he was a teacher in Kyoto School of Computer Science. From 2004 to 2007, he was Assistant Professor of The Kyoto College of Graduate Studies for informatics. From 2007 to 2010, he was Assistant Professor of Office of Society Academia Collabo-ration, Kyoto University. From 2010 to 2013, he was Assistant Professor of Research Institute for Economics & Business Administration, Kobe University. From 2015 to 2016, he was a specially appointed assistant professor of Cybermedia Center, Osaka University. From April 2016 to September 2016, he became a specially appointed researcher. Since November 2016, he became an assistant professor. His research interests include distributed processing systems, rule-based systems, and stream data processing. He is a member of IPSJ, IEICE, and IEEE



Tomoki Yoshihisa received the Bachelor's, Master's, and Doctor's degrees from Osaka University, Osaka, Japan, in 2002, 2003, 2005, respectively. Since 2005 to 2007, he was a research associate at Kyoto University. In January 2008, he joined the Cybermedia Center, Osaka University as an assistant professor and in March 2009, he became an associate professor. From April 2008 to August 2008, he was a visiting researcher at University of California, Irvine. His research interests include video-on-demand, broadcasting systems, and webcasts. He is a member of the IPSJ, IEICE, and IEEE.