Mobile Netw Appl (2017) 22:98–112
DOI 10.1007/s11036-016-0694-8

# Fast-Start Video Delivery in Future Internet Architectures with Intra-domain Caching

**Zhi Liu[1] · Mianxiong Dong[2] · Bo Gu[4] · Cheng Zhang[1] · Yusheng Ji[3] · Yoshiaki Tanaka[1]**

Published online: 31 March 2016


**Abstract** Current TCP/IP based network is suffering from the tremendous usage of IP. Recently, *content centric network* (CCN) is proposed as an alternative of the future network architecture. In CCN, data itself, which is authenticated and secured, is a name and can be directly requested at the network level instead of using IP and DNS. Moreover, routers in CCN have caching abilities. Then end users can obtain the data from routers instead of remote server if the content has been stored in the routers, thus the overall network performance could be improved by reducing the transmission hops. Orthogonally, video plays a more and more important role nowadays and dominates the network traffic. Response time of each video request greatly affects the *quality of user experience* (QoE), users may even abandon the requested video service if they have to wait for long time before the video playback. Hence how to provide fast-start video delivery in CCN is critical. In this paper, we target to provide users fast-start video delivery in CCN. Specifically, we design a new caching policy for popularity-aware video caching in topology-aware CCN. And we propose to encode the video using *scalable video coding* (SVC) for fast-start video delivery and cache each video layer separately following the designed caching policies. Given an assigned weight by users, the tradeoff between the waiting time and received video quality is studied. Simulations are conducted to verify the performances and the results show that the proposed scheme outperforms state-of-the-art schemes significantly in typical scenarios.

**Keywords** Scalable video coding (SVC) · CCN · ICN · Caching · Fast start · Quality of experience (QoE)

✉ Mianxiong Dong
mx.dong@ieee.org

Zhi Liu
liuzhi@aoni.waseda.jp

Bo Gu
bo.gu@cc.kogakuin.ac.jp

Cheng Zhang
cheng.zhang@akane.waseda.jp

Yusheng Ji
kei@nii.ac.jp

Yoshiaki Tanaka
ytanaka@waseda.jp

[1] Waseda University, 3-4-1 Ohkubo, Shinju-ku, Tokyo 169-8555, Japan

[2] Muroran Institute of Technology, 27-1 Mizumotocho, Muroran, Hokkaido 050-8585, Japan

[3] National Institute of Informatics, 2-1-2, Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan

[4] Department of Information and Communications Engineering, Kogakuin University, 2665-1 Nakano, Hachioji, Tokyo 192-0015, Japan

## 1 Introduction

Current TCP/IP based network is suffering from the tremendous usage of IP especially in the era of *Internet of things* (IoT), where everything could be connected into the existing TCP/IP networks. CCN [1] (which is similar with *information centric networking* (ICN) [2], *named data networking* (NDN) [3], and *data oriented architecture* [4], etc.)[1] was

[1] Please note that the proposed scheme also works in other ICN architectures, where the key ideas of these network architectures are more or less the same.

proposed by Van Jacobosen [1], as an alternative of the current TCP/IP-based network. Comparing with the traditional networks, CCN focuses on 'what' instead of 'where', i.e. content itself is more important than where the content is ('where' is represented using IP addresses in the traditional TCP/IP networks). In CCN, content is a primitive, data itself is a name and can be directly requested at the network level, which means IP is not necessary and there is no more DNS. Anybody with the data can answer the data request. The data itself is authenticated and secured instead of securing the connections it traverses.

Another difference between CCN and traditional networks is that CCN's routers have the caching abilities. If the requested data has been stored in the routers, the end users can obtain the data from the routers directly instead of from the remote server. This greatly reduces the total transmission hops (or transmission time or network traffic) needed. The inherent problem is what should be cached in the routers and how to replace the cached content when a new to-be-cached content arrives at a full cache. The caching decision policy [5, 6] decides whether the new arriving video content should be stored in a particular router or not, and the caching replacement (such as *least recently used* (LRU))[2] policy decides what content will be moved out when the new arriving content is decided to be cached in the cache-size-limited full router. The caching policy greatly affects the overall network performance, hence it is an important issue for CCN and the focus of this paper.

On the other hand, video plays a great important role nowadays [7, 8], where it dominates the current network traffic and this trend will last at least for a couple of years. According to *Cisco Visual Networking Index* (VNI)[3], IP video traffic will be 79 % of all consumer Internet traffic in 2018, up from 66 % in 2013. How to provide high-quality video service while at the same time reduce the network burden (traffic) becomes an important issue to be investigated for both the current network and the future network architecture. One character of video is that each video has an associated popularity, which indicates the average demanded times among all the video requests during a time window. Zipf [9] is commonly applied to help calculate the video popularity. Given some video are requested more often compared with the rest video, caching the more popular video content in the routers close to the users will help reduce the total network transmission hops. How to cache the video given video's different popularity values has been studied in literature [5, 10, 11]. A recent study [5] addressed the popularity-aware video caching in CCN with better performance comparing with other competing schemes. But this scheme has the 'redundancy' problem, where the same content might be stored multiple times in the routers along the path from server to users, thus leading to the caching performance degradation.

Orthogonally, a recent study [12], which is based on 23 million views by 6.7 million unique users, shows that users will start abandoning 'short' videos (which is less than 30 minutes) after two seconds, and that 20 percent have moved on after five seconds. This shows how important the response time of the video service is. Hence how to shorten the video start time becomes critical. This paper targets how to reduce the waiting time of the video service in CCN, which could make the video streaming service more attractive to users. The waiting time here is defined as the period between sending the video request and beginning the video playback.

To reduce the initial response time, H.264 SVC [13] is applied in this paper. H.264 SVC is an extension of the *H.264 Advanced Video Coding (AVC)* standard [14], which is widely used in the current video streaming scenarios [15–17] for high quality video transmission. H.264 SVC encodes the *group of pictures* (GOP, composed of a number of frames) into one base layer and multiple enhancement layers. The base layer is encoded with low quality and can be decoded by itself. Unlike the base layer, the enhancement layer $i$ cannot be decoded without correct decoding of all its lower layers (base layer and the enhancement layers up to layer $i-1$). The more enhancement layers are decoded, the higher video quality could be achieved. If the video is encoded using H.264 SVC and users are willing to watch the video at a lower quality, the users can receive the lower layers only and then start the video playback. The waiting time can then be reduced greatly given the number of packets needs to be delivered becomes smaller. But as far as we understand, what should the tradeoff between the video quality sacrificing and waiting time reduction be, and what should the corresponding video caching policies be remain unsolved.

In this paper, we first propose a new caching scheme for the popularity-aware video distribution over CCN with known topology (or router levels).[4] Then we propose to encode the video content using H.264 SVC and cache each video layer separately following the designed caching policies. The tradeoff between the video quality sacrificing and waiting time reduction are mathematically formulated in

[2]This paper mainly discusses the video caching policy, and we use the existing cache replacement policies instead. Moreover, we have simulation results applying various state-of-the-art caching replacement methods to further verify the performances.

[3]Cisco Visual Networking Index: Forecast and Methodology, 2013∼2018

[4]Part of this work has been published in [18], where only a new caching algorithm to solve the cache redundancy problem is presented and this journal is an non-trivial extension by applying the H.264 SVC for fast-start video delivery.

this paper. Then given an assigned weight by the users, the system can help calculate how the video layers should be cached in each router and decide how many layers will be delivered to each user. When less layers are chosen (meanwhile the popular video and lower video layers will be placed closer to users), the waiting time becomes shorter, which enables quicker response. Extensive simulations are conducted to verify the proposed scheme's performances and the experimental results show that the proposed scheme can outperform the state-of-the-art solutions greatly. The detailed contributions of this paper are listed as follows:

1. We propose to use H.264 SVC as the video encoding tool to encode the video and discuss the tradeoff between the video quality sacrificing and waiting time reduction.
2. We design a new efficient caching policy for video distribution in CCN, which depends on the video popularity, the sizes of the video, the sizes of the caches, users' assigned weight parameter and the video layers' differences.
3. Extensive simulations are conducted to verify the performances by comparing with the state-of-the-art competing schemes.

The rest of this paper is organized as follows: Section 2 discusses the existing work about CCN, CCN caching and H.264 SVC in CCN. The detailed system model, video popularity model and H.264 SVC coding methodology are introduced in Section 3. How we design the new caching policy, how we formulate the unsolved fast-start video delivery problem, and the algorithm to solve the formulated problem are explained in detail in Section 4. The simulation results are shown in Section 5 and we conclude this paper in Section 6.

## 2 Related work

The related work section is divided into two parts. Section 2.1 introduces the CCN and the CCN caching. Section 2.2 discusses the H.264 SVC and H.264 SVC in CCN. The difference between this paper and the related work is explained at the end of this section.

### 2.1 CCN and CCN caching

CCN, as an alternative of the future network architecture, is similar with ICN [2], NDN [3], and *data oriented architecture* [4]. CCN emphasizes the content by making the content itself directly addressable and routable, and the communication between the endpoints are based on named data instead of IP addresses. CCN can help solve the problems raised by the IP addresses as happened in the traditional networks, and could be an alternative of the future network structure especially in the IoT era. Routers in CCN have the caching abilities. If the requested content has been stored in the router, the user can directly retrieve the demanded content from the router instead of from the remote server, i.e. the transmission hops are reduced. Therefore caching the video in the router can help reduce the network traffic and improve the network performance.

There are various caching strategies proposed in literature [10, 11, 19–25]. Specifically, [22] proposed an implicit coordinate chunk caching location and searching scheme in CCN hierarchical infrastructure. Li and Simon [23] introduced a cooperative caching strategy that has been designed for the treatment of large video streams with on-demand access, where it needs the cooperation between routers that are not along the same paths.

The video popularity also has been considered in the caching policy design. Li et al. [24] discussed the popularity-driven coordinated caching by formulating this into an optimization problem. Given the computation complexity is too large, the authors also proposed an online algorithm. Wu et al. [25] also handled the 'redundancy' problem, but the authors took the globe routers into consideration to generate the weight for caching. A recent related paper is [5], where the popular video are scheduled to be placed close to the end users to reduce the expected *round-trip time* (RTT) and improve the server hit rate. But this work has the 'redundancy' problem as introduced in Section 1, hence the system performance is affected. Suksomboon et al. [26] discussed the cache orchestration through network function visualization, which heavily depends on the RTT of the interest packets.

### 2.2 H.264 SVC and H.264 SVC in CCN

H.264 SVC is proposed and widely used in video communication systems to provide ubiquitous video transmission to variable mobile devices due to its scalability in terms of providing different source encoding rates. Then an optimal layer could be chosen and the corresponding packets are sent to the end users to maximize their received video quality [15–17, 27] according to each device's specifications and network conditions. H.264 SVC has also been used in CCN. For example, [28] discussed how to use H.264 SVC in CCN, where their proposed scheme could avoid video freeze but without reducing the high hit rate on the CCN router or affecting the video quality. Hwang et al. [29] discussed how to use H.264 SVC for video streaming in CCN, and the proposed scheme could prevent video freeze thereby provide better video streaming quality than the existing non-hybrid streaming technologies.

As mentioned in the introduction section, video service's response time greatly affects the QoE. But as far as we

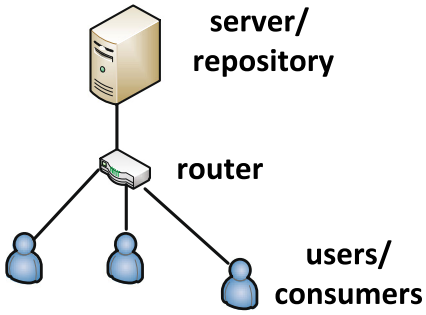



**Fig. 1** Illustration of the CCN

understand, how to provide users smaller response time in CCN remains unsolved. Different from these related works, this paper first solves the 'redundancy' problem in CCN caching, where the same content might be stored multiple times in the routers along the path from server to users. Meanwhile, this paper also focuses on how to utilize the SVC to balance the video quality sacrificing and the waiting time reduction. By sacrificing some video quality, users could enjoy shorter response time.[5]

## 3 System overview

This section overviews the CCN system and introduces the H.264 SVC encoding method. The Zipf popularity model is introduced at the end of this section.

### 3.1 CCN overview

As an initial study of the video caching in CCN, we use a simple CCN network to introduce the CCN architecture and explain the principles of the proposed scheme for fast-start video delivery.

Figure 1 is a simple illustration of the CCN. This CCN system is composed by a server/repository, a router and many users/consumers. In CCN, routers can buffer data and content store (CS) of routers plays a role of a buffer memory. The caching in CCN involves 1) *caching decision* and 2) *cache replacement*. The caching decision mainly helps decide where to cache the 'new' data chunks if they are not in the cache currently. We can also say the video caching decision helps decide whether or not to cache a video content when it arrives at a router, where it is not cached. When a new data chunk is decided to be stored in a fully occupied CS, some data chunk needs to be moved out. Cache replacement scheme discusses which existing data chunk should be replaced.

In Fig. 1, if a data chunk requested by the user is not stored in the routers' CS, server will send the data chunk to the user. When the data chunk forwarded from the server arrives at the router, the router will decide whether to cache the chunk or not according to the caching decision policy. If the decision outputs a 'yes' but the corresponding CS' memory space is fully occupied, the new chunk will replace the CS's existing data chunk in accordance with the cache replacement scheme. For instance, if LRU is the chosen cache replacement scheme, the fully occupied CS will replace the least recently used chunk using the newly arrived chunk. The caching decision policy and cache replacement scheme affect the caching efficiency and the overall network performance. This paper talks about the caching decision policy in a router of the CCN, with the aim to provide the fast-start video delivery service using H.264 SVC.

### 3.2 Scalable video coding

H.264 SVC encodes the frames into one base layer and multiple enhancement layers with the scalability in temporal, spatial and quantization domain. The base layer is encoded with low quality, thus the base layer's source encoding rate is low. According to the adopted encoding methodology, the base layer can be decoded by itself. Each enhancement layer is based on the corresponding previous layers, i.e. before capable of decoding one enhancement layer, the user needs to decode all its previous layers first. The more enhancement layers are received and decoded, the higher video quality could be achieved. Without loss of generality, we assume each video is encoded into $L$ layers, but please note that the proposed method also works in the scenario where different video may have different numbers of video layers.

Figure 2 shows the H.264 SVC encoding with one base layer and two enhancement layers. The more layers lead to larger required data rates, but the received video quality is

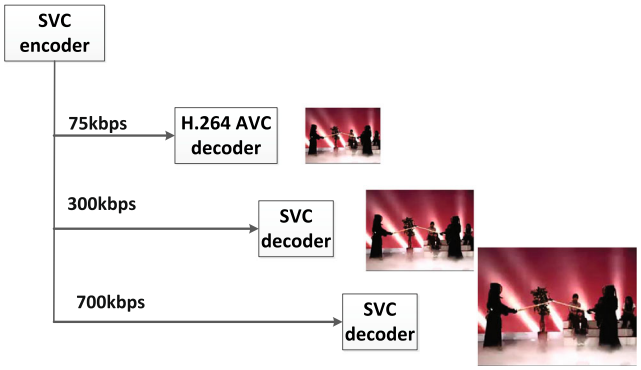



**Fig. 2** Illustration of the SVC encoding method

[5]Please note that encoding the same video at different rates and storing each encoded version in the caches/server can also provide the fast start, where the smaller-size version leads to shorter waiting time. But this needs more storage comparing with using H.264 SVC. Note that the amount of data generated outpaces decline of the disk drive's cost [30] therefore storage cost is non-trivial. Also the routers' caches are size-limited, hence this solution is not discussed in this paper.

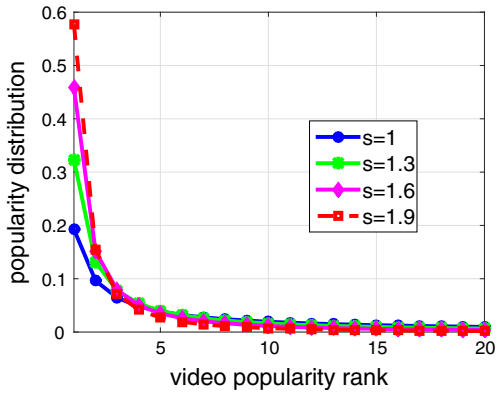



**Fig. 3** Zipf popularity distribution for the 20 most popular video contents with different s when N = 100

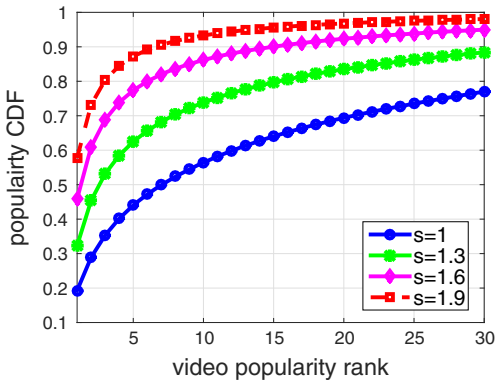



**Fig. 4** Zipf popularity CDF for the 30 most popular video contents with different s when N = 100

better in terms of higher frame rate, larger resolution, and better quantization level. This could be partially observed from the frames shown in Fig. 2.

### 3.3 Zipf popularity model

Different video contents have different popularity values, where video $i$'s popularity is defined as the ratio of video $i$'s request number to the number of total video requests. Since popular content will be subscribed more often than the other video contents, video contents that are more popular should be placed closer to users to reduce the network cost (traffic) and shorten the RTT. In this paper, state-of-the-art *Zipf* popularity distribution is adopted.[6] We assume the total number of video contents is $N$ and $k_i$ is $i$th video item's popularity rank. Smaller rank instance number indicates higher popularity. Then the Zipf's law predicts that out of a population of $N$ video elements, the frequency of element $i$ with rank $k_i$ is as follows:

$$f_{i,k_i,s,N} = \frac{1/{k_i}^s}{\sum_{n=1}^{n=N} 1/n^s} \quad (1)$$

where $s$ denotes the value of the exponent characterizing the distribution and is referred to the skewness of the popularity distribution. Figure 3 shows the popularity distribution with different $s$. The x-axis is the popularity rank and y-axis is the corresponding popularity value. We can observe that large $s$ leads to highly right-skewed histogram, representing high diversity among video contents in terms of video popularity. On the other hand, small $s$ leads to a flat-skewed histogram, and represents less video popularity diversity.

To better show the Zipf popularity distribution, we also calculate the *cumulative distribution function* (CDF) of the popularity distribution as shown in Fig. 4. From the figure, we can see that different $s$ lead to different popularity CDF distributions. More importantly, it can be noticed that the first several most popular video can satisfy a large proportion of the total requests. For example, the 5 most popular video serve 87.2 % video requests when $s = 1.9$. Then by placing the popular video close to users, the network cost could be greatly reduced. This motivates designing caching decision policies to help improve the network video distribution.

Next we show how we utilize video popularity and SVC to design the caching decision policies to improve the video distribution over the networks and provide fast-start video delivery.

## 4 Caching decision policy

This section introduces the objective and the designed caching policy for fast-start video delivery. Specifically, we first introduce the caching policy without considering fast start of the video delivery service, which has been discussed in [18]. And then we show how we rely on this to design a new caching and transmission scheme to provide the fast-start video delivery service. Some practical issues related with this scheme are discussed at the end of this section.

### 4.1 Objective

Video QoE measures customers' experience regarding the video streaming service. The waiting time, which is defines

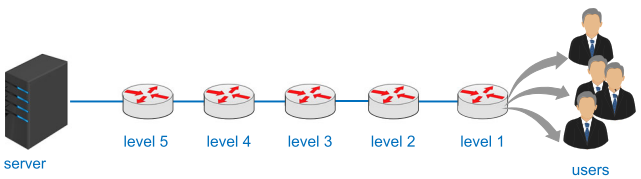



**Fig. 5** Illustration of the Cascade topology with five levels. Level 1 router is the most close to users

[6] Our scheme also works with other popularity distribution, and here we use the Zipf as an example.

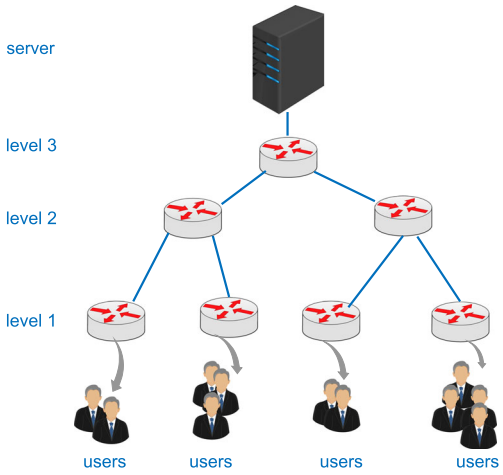



**Fig. 6** Illustration of the Binary Tree topology with three levels. Level 1 router is the most close to users

as the period between sending the video request and beginning the video playback, affects the QoE greatly. A quick load time (smaller waiting time) can improve the QoE. The objective of this paper is to fully utilize the routers' caches under the assumption that the video popularity and network topology (or router levels) is known, and meanwhile provide better QoE. Recall that Zipf popularity model indicates that the several most popular video contents could satisfy a large proportion of the video requests. By placing the popular video in the routers closer to users, users can retrieve the video from routers instead of from the remote server, which significantly reduces the transmission hops needed. This motivates us to take the video popularity into consideration when designing the caching decision policy.

To provide the fast-start video delivery, video is encoded into multiple video layers using H.264 SVC. By sending less number of layers if the users would like to tolerate the corresponding received video quality degradation, the waiting time could be reduced. Meanwhile, the caching policy will be redesigned for this fast-start video delivery, where the lower layers will be cached closer to users to help further reduce the waiting time to some extend.

### 4.2 Caching policy without fast start

There are multiple routers along the path from the server to users possibly. The routers are labeled with different levels according to their distances from the users. Figs. 5 and 6 show the *Cascade network topology* and the *Binary Tree topology*, respectively. These two topology models are also used in the simulations. In Fig. 5, there are five routers which are denoted using level instance numbers from number 1 to number 5. Level 1 router is the most close to users, and level 5 is the furthest from the users. Figure 6 illustrates the Binary Tree topology network structure with 3 levels. The same as the Cascade network topology, level 1 router is the nearest to users and the level 3 is the furthest. $i$th level router' CS has a cache size of $x_i$ chunks. Video $i$'s size is $\delta_i$ with its popularity rank to be $k_i$. Before discussing the caching policy, a router index number $I_{i,j}$ is introduced. Since router $j$ has limited cache size, i.e. $x_j$ chunks, it can only store limited number of video contents. But the several most popular video serves a large proportion of the total video requests, therefore the video content with highest ranks should be allocated to the routers that are the closest to users. We first calculate the index number $I_{i,j}$ to indicate whether video $i$ should be placed at router $j$ or not considering the video ranks and the sizes of the caches. Assuming along the path from user to server, there are $M$ level routers, denoted as 1, 2, ..., $M$, we can use the following equation to calculate $I_{i,j}$;

$$I_{i,j} = \begin{cases} 1 : if \sum\limits_{\forall p \in N, k_p \le k_i} \delta_p > \sum\limits_{q=1}^{j-1} x_q \,\&\, \sum\limits_{\forall p \in N, k_p \le k_i} \delta_p \le \sum\limits_{q=1}^{j} x_q \\ 0 : o.w. \end{cases} \tag{2}$$

From this equation, we could find that $I_{i,j} = 0$ or $I_{i,j} = 1$. $I_{i,j} = 1$ means video $i$ should be placed at the $j$th level router according to its popularity rank $k_i$ and the caching abilities of the corresponding routers. The caching abilities are determined by the sizes of the caches and the sizes of the video contents. Recall that the principle of the caching is to place the video contents with highest ranks in the router nearest to the users. By comparing the cache sizes of routers up to level $j$ and the size of the video, whose popularity instance number is less than $k_i$, we can know whether video $i$ should be placed in router $j$ or not.

Then whether video $i$ should be cached in router $j$ or not can be decided by $I_{i,j}$. The caching policy is: if $I_{i,j}$=1, video $i$ will be cached in router $j$, and if $I_{i,j}$=0, video $i$ will not be cached in router $j$. This means only when the video should be placed in the corresponding router, the video will be cached. The video will not be cached in the router if its popularity rank is too high or too low for this corresponding router. Hence the 'redundancy' problem in the current existing schemes could be solved. The disadvantage is that the system needs time to cache the video content since the caches are empty at first, and we will show the related simulation results in Section 5.2.

### 4.3 Cache policy for fast-start video delivery

To provide users fast-start video delivery, we propose to encode the video into different layers using H.264 SVC and then assign different layers with different priorities according to video's popularity and the layer instance number. The caching policy is then designed based on the new defined

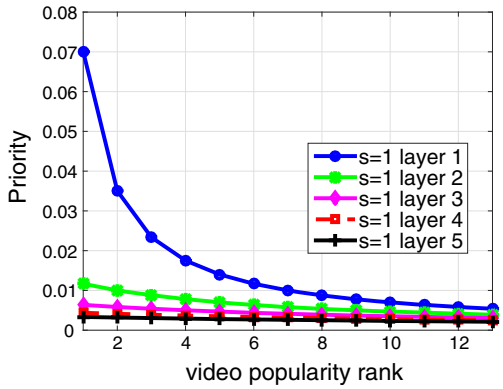



**Fig. 7** Defined priority with different layers when s=1 and $\beta$=5

*priority*, and the new priority plays a similar role with the video popularity rank in the caching policy as introduced in Section 4.2.

### *4.3.1 Priority design*

The priority takes both the video popularity and the layer instance number into consideration. And we aim to put the popular video and lower video layers closer to users to benefit the video distribution and provide the fast start. The priority value of video layer $l_i$ ($i$ denotes the video) is defined using the following equation:

$$F_{i,l_i} = \frac{\frac{1}{(k_i+\beta(l_i-1))^s}}{\sum_{n=1}^{n=N}\sum_{m=1}^{m=L}\frac{1}{(n+\beta*(m-1))^s}} \tag{3}$$

The new priority as shown in Eq. 3 is a modified version of the Zipf function as introduced in Eq. 1, where a weighted layer instance number $\beta(l_i - 1)$ with weight parameter $\beta$ is added to the popularity rank value. For the same video, lower layer will have smaller priority value and hence will be placed closer to users. At the same time, the unpopular video content's lower layer may have larger priority value than the popular video content's higher layer, hence these layers will be placed closer to users. By assigning the layer with smaller instance number smaller priority value, these layers could be placed closer to users, which could reduce the waiting if these layers alone can satisfy the users. $\beta$ balances the weight of the video popularity value and the layer instance number, and indicates how much the users care about the waiting time. In this paper, $\beta$ is assumed to be pre-assigned by users.

Figure 7 shows the new defined priority value (y-axis) with $\beta = 5$, where x-axis is the popularity rank. Similar trend could be observed with the Zipf function as introduced in Eq. 1. We can observe that different layers of the same video have different priorities from this figure. The larger layer instance number leads to smaller priority value. We can also find that the lower layer of higher popularity rank video may have larger priority value comparing with the higher layer of smaller popularity rank video, which depends on $s$, $\beta$ and the popularity rank. This means the lower layer could be cached in routers closer to users. We also tested the priority value with smaller $\beta$ as shown in Fig. 8.

Comparing Figs. 8 and 7, we could observe that smaller $\beta$ leads to larger priority value for the layers with larger instance number, resulting in different decisions. A larger $\beta$ leads to higher priority value of the base layers and layers with smaller instance numbers, hence they will be placed closer to users, which could provide shorter waiting time.

### *4.3.2 Caching probability*

Similar with Section 4.2, we first generates a index number $I_{i,l_i,j}$ to denote whether layer $l_i$ of video $i$ should be placed in the $j$th level router or not.

$$I_{i,l_i,j} = \begin{cases} 1 : if \ \ (C1) \ and \ (C2) \ are \ both \ satisfied \\ 0 : o.w. \end{cases} \tag{4}$$

where (C1) and (C2) are expressed as follows:

$$\sum_{\forall p\in N, \forall l_p, F_{p,l_p}\le F_{i,l_i}} \delta'_{p,l_p} > \sum_{q=1}^{j-1} x_q \tag{C1}$$

$$\sum_{\forall p\in N, \forall l_p, F_{p,l_p}\le F_{i,l_i}} \delta'_{p,l_p} \le \sum_{q=1}^{j} x_q \tag{C2}$$

$\delta'_{i,l_i}$ denotes the source encoding rate of video $i$'s layer $l_i$, i.e. $\delta'_{i,1}$ is the source encoding rate of the base layer and

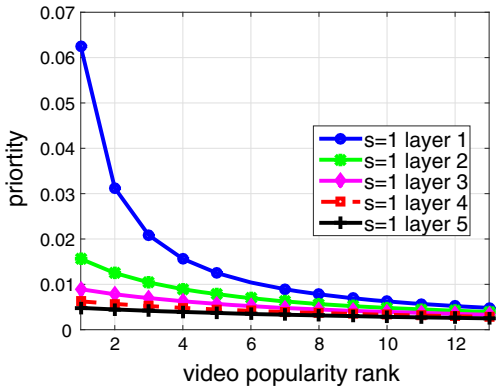



**Fig. 8** Defined priority with different layers when s=1 and $\beta$=3

$\delta'_{i,l_i}(l_i > 1)$ is the total source encoding rate up to layer $l_i$ minus the total source encoding rate up to layer $l_i - 1$. The condition (C1) and (C2) indicate whether $l_i$ layer of video $i$ should be placed in the $j$th level router or not by checking its priority and the routers' cache sizes, which is similar as conducted in Section 4.2.

Then the caching policy could be defined as: if $I_{i,l_i,j} = 1$, layer $l_i$ of video $i$ should be cached in router $j$, and if $I_{i,l_i,j} = 0$, layer $l_i$ of video $i$ should not be cached in router $j$.

#### *4.3.3 Layer selection for fast-start video delivery*

Upon a view request and a weight parameter $\beta$, how many H.264 SVC encoded video layers should be transmitted is the next question. Here we adopt a simple method,[7] i.e. the number of video layers to be sent is decided by the weight parameter $\beta$, and different $\beta$ lead to different layer numbers. Assume user requests video item $i$, the number of video layers sent $r_i$ could be calculated using the following equation:

$$r_i = min(\lceil \frac{\sigma L}{\beta} \rceil, L) \tag{5}$$

In the above equation, $\beta$ is this video request's weight parameter and $\sigma$ is a real constant. The video is encoded into $L$ layers using SVC. From Eq. 5, we could notice when the user assigns a large weight parameter $\beta$, i.e. the user gives higher priority to the waiting time, smaller number of video layers will be delivered to users. By sending users less number of layers in the same network scenario, the waiting time could be shortened, hence the fast-start video service could be provided. If the users care the video quality more, smaller $\beta$ will be assigned. Then the number of video layers need to be delivered is larger, resulting in higher video quality received and longer waiting time before the video playback.

#### *4.3.4 Caching and transmission policy*

The caching and transmission policy tell what to store in each CS and upon a user's view request, what should be sent. What to store in each CS is based on the priority value introduced in Section 4.3. The number of layers sent could be calculated according to Eq. 5 as explained in Section 4.3.3. The caching and transmission policy is shown in Algorithm 1.

[7]Other layer selection method might be designed considering more factors such as video popularity.

**Algorithm 1** Caching and transmission policy flow

1: Encoding the video into L layers using H.264 SVC.
2: Calculating the video layer priority value $F_{i,l_i}$ for each video layer.
3: Upon a video request, calculating the number of layers to be sent according to Eq. 5.
4: If the video layers are stored in the router along the path from users to server, the router sends the video layers to users. Otherwise, the server will send the video layers, and the video layers will be cached in the corresponding router according to the policy.

Specifically, the system will first calculate the priority value for each H.264 SVC encoded video layer. Then upon a video request, the system calculates the number of layers to be sent according to Eq. 5. If the router caches the corresponding video layers, the user will retrieve the video from the router directly, otherwise, the video will be sent to user from the remote server. If the video is delivered from the server, at each visited router, the caching policy will help decide whether to cache the video or not.

Please note that the proposed method for fast-start video delivery also works for the scheme with other caching probability design such as the methodology used in [5] with some modifications.

### 4.4 Practical issues

Comparing with H.264, H.264 SVC sacrifices the encoding efficiency. But the sacrifice is limited and only the beginning part of the long video clips needs to be encoded using H.264 SVC to provide quick response. Hence the overhead is trivial comparing with the massive network traffic.

To implement this caching decision scheme, the routers need to know video's rank table in order to decide whether they should cache the video or not. But please note that the communication between the routers is not that expensive given the table is not that big. Exchanging this information among routers periodically is realistic. On the other hand, the popularity value is based on video request statistics. The video requests come from the end users and if the requests can not be satisfied at router $i$, the video request will be forwarded to level $i + 1$ router. Therefore level 1 routers can know all the request information, and the most popular video contents can be cached. Level 2 routers can know the most popular video contents except the video contents that have been cached in router 1, since the corresponding requests will not be forwarded to level 2 router. Similarly, level $i$ router can obtain the necessary information for its decision making. The caching decision policy itself is with low computation complexity as could be observed easily. Given a users tradeoff weight value $\beta$ of the video quality

and the waiting time, the layers to be sent could be decided quickly according to Eq. 5. Hence the proposed algorithm is feasible.

This paper requires that the known network topology should have routers with clear levels. This is actually reasonable if we take a look at the last mile of the network. Moreover, CCN works in backbone network and the nationwide (or other large scale) backbone network' topology is usually known.

## 5 Simulation

This section introduces the simulation setup and the simulation results are shown comparing with the state-of-the-art competing schemes.

### 5.1 Simulation setup

To evaluate the proposed scheme's performances, two different network topologies are used including Cascade model and Binary Tree model as shown in Figs. 5 and 6, respectively. Both network topologies are with five levels. Each video content's delivery time is composed of the transmission time, propagation delay and the queueing delay. Since the propagation delay and the queueing delay are very small comparing with the video transmission time, these two terms are assumed to be zero. The transmission time can be denoted using the number of hops the video has traveled. The server hit rate is defined as the ratio that the requested video is retrieved from the remote server. The server hit rate is an important parameter for the CCN system since it indicates the frequency the server is accessed. Therefore, we use the number of hops the video has traveled and server hit rate as the evaluation metrics.

The parameters used in the simulations are shown in Table 1. Given the channel bandwidth (10Gbps) is much larger than the source rates of the requested video contents (average total size is 120*6.9MB), the requests are assumed to be satisfied within one second in this paper. Each content lasts for roughly 10 seconds, and we encode all the video using H.264 SVC, but normally only the first several GOPs need to be encoded using H.264 SVC. Please note that this is just one typical scenario, and our scheme is not picky in terms of the network and video parameters.

**Table 1** Detailed network parameters

| Parameters | Values |
|---|---|
| Total request rate: $\lambda$ | 120 content items/s |
| # of different content items: M | $2 \times 10^4$ items |
| Average content size: $\sigma$ | 690 chunks (6.9 MB) |
| Cache size of node 1: X(1) | $2 \times 10^5$ chunks (2GB) |
| # of content classes: $K$ | 100 classes |

**Table 2** Video encoding parameters

| Layer#. | Resolution | Freq (fps) | Data rate | PSNR |
|---|---|---|---|---|
| 1 | 256 × 192 | 8 | 74.4680 | 28.15 |
| 2 | 256 × 192 | 16 | 93.5760 | 30.61 |
| 3 | 512 × 384 | 32 | 297.6520 | 37.27 |
| 4 | 1024 × 768 | 16 | 552.2440 | 38.54 |
| 5 | 1024 × 768 | 32 | 710.4040 | 39.21 |

In the simulation, *Kendo*[8] is used and it is encoded using JSVM [31]. The original resolution of kendo is 1024 × 768 with the frequency to be 32 *frame per second* (fps). The GOP size is set to be 32 frames. We encode the video with 5 SVC layers. Table 2 shows the detailed settings of the data rates and the PSNRs for all the 5 SVC layers.

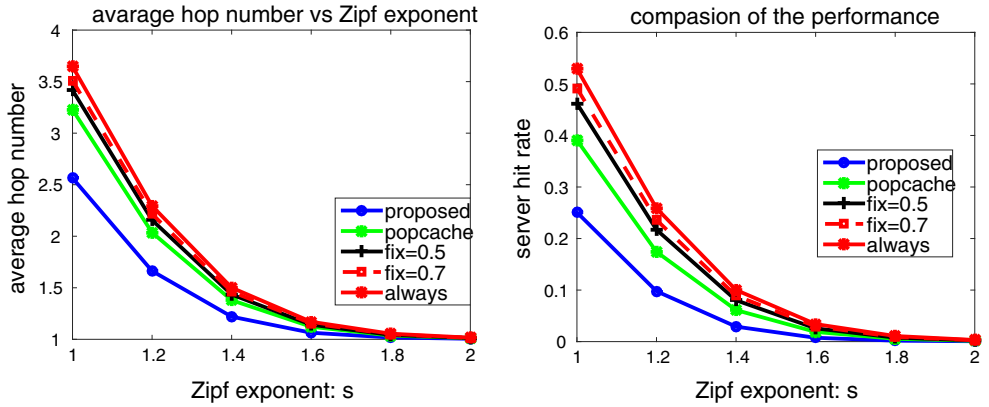
### 5.2 Simulation results without fast start

To show the performance of the proposed scheme, we compare our scheme with *popcache*, *always* and $fix = x$. *popcache* [5] is a scheme dedicated for popularity-aware video caching with good performance in terms of reducing the round-trip time and improving the server hit rate. *always* means when a new video content arrives at a router, it will be stored. $fix = x$ means when a new video content arrives at a router, it has a probability $x$ to be cached. We could find that *always* is an extreme case of $fix = x$, where $x = 1$. The caching probabilities of the *popcache* are obtained from reference [5] directly. The cache replacement policy is LRU, unless there is specific explanation about the adopted cache replacement method. The first 7000s are the 'adjusting' period, where the cache will be updated following the caching policy and the replacement policy. The results shown are the average results from 7001s to 10000s.

Figures 9 and 10 show the performances in terms of the average transmission hops needed and the server hit rates in the Cascade topology and the Binary tree topology, respectively. From Fig. 9a, we can observe that *always* performs the worst and $fix = x$ is better than *always*. *popcache* is better than *always* and $fix = x$ since it considers the video popularity. But all these schemes have the 'redundancy' problem, where the same content might be stored multiple times in the routers along the path from server to users. The proposed scheme is much better in terms of the transmission hops needed.

Figure 9b talks about the server hit rate, which indicates the percentage that the requested video needs to be sent

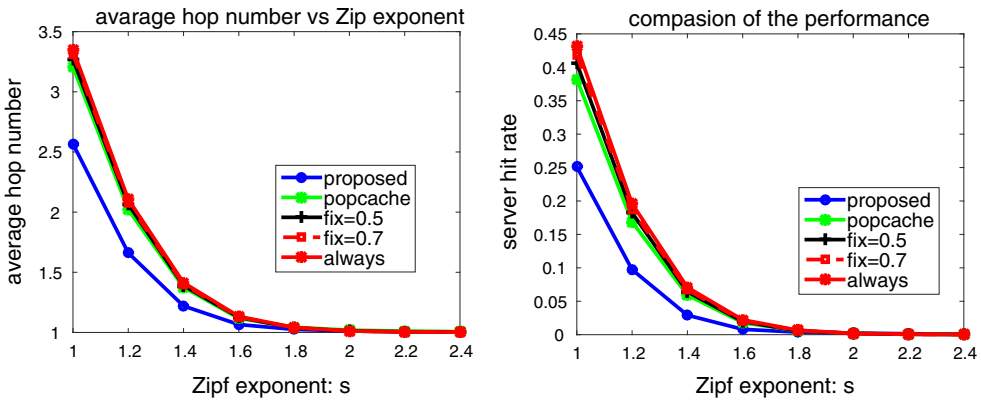
[8] http://www.fujii.nuee.nagoya-u.ac.jp/multiview-data/




(a) average transmission hops (b) server hit rates

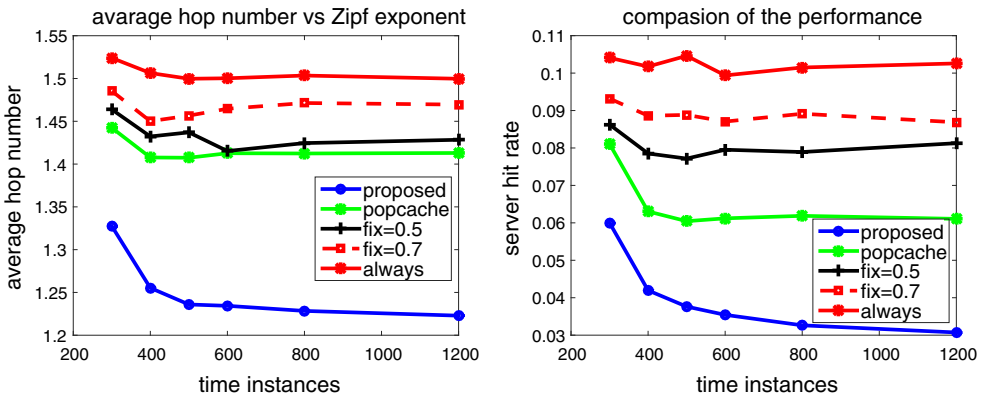
**Fig. 9** Performance evaluation of the CCN with Cascade network topology

**Fig. 10** Performance evaluation of the CCN with Binary tree network topology

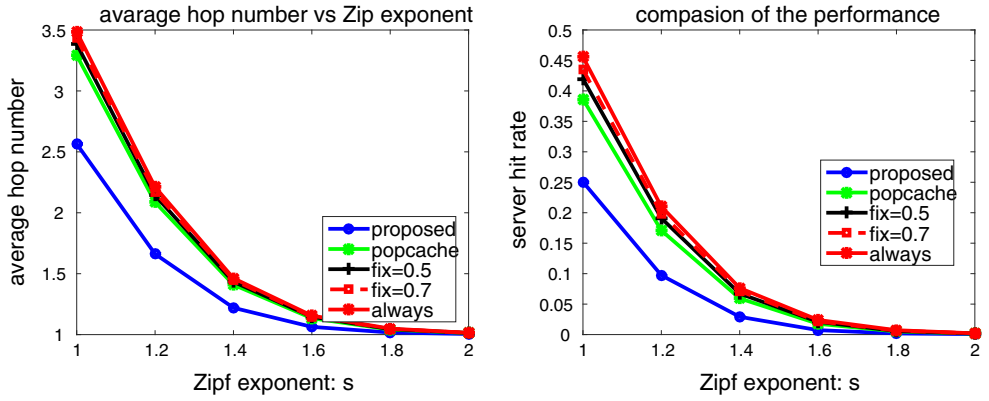



(a) average transmission hops (b) server hit rates

**Fig. 11** Performance evaluation of the caching policy at different time instances

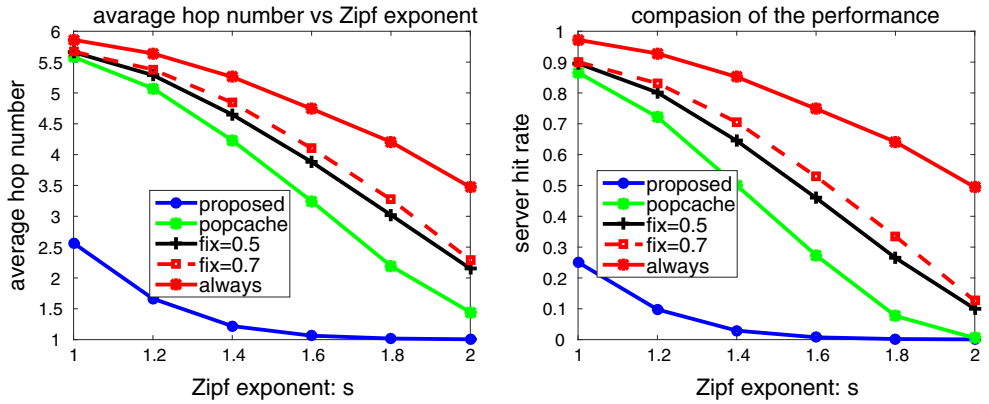



(a) average transmission hops (b) server hit rates




(a) average transmission hops (b) server hit rates

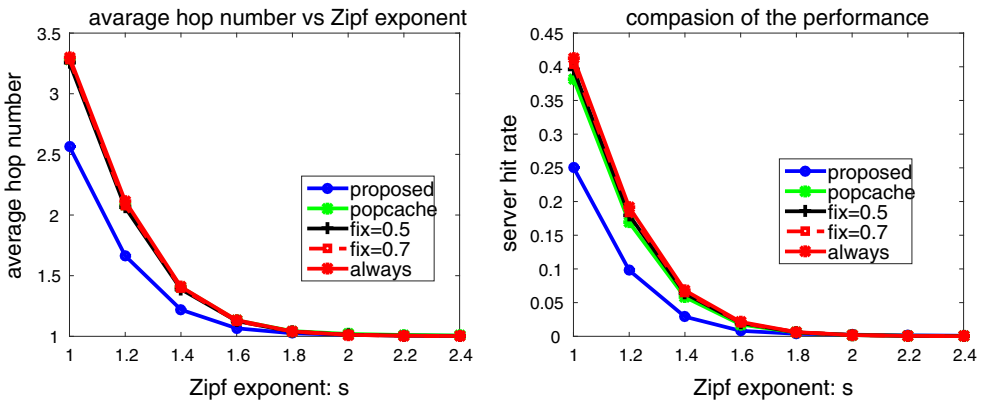
**Fig. 12** Performance evaluation of the CCN with Cascade network topology with RR



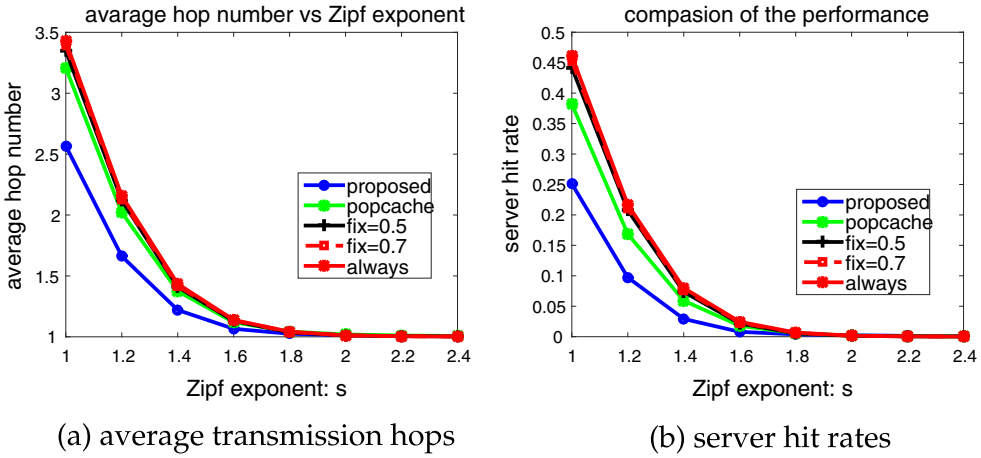

(a) average transmission hops (b) server hit rates

**Fig. 13** Performance evaluation of the CCN with Cascade network topology with MRU

**Fig. 14** Performance evaluation of the CCN with Binary tree network topology with RR

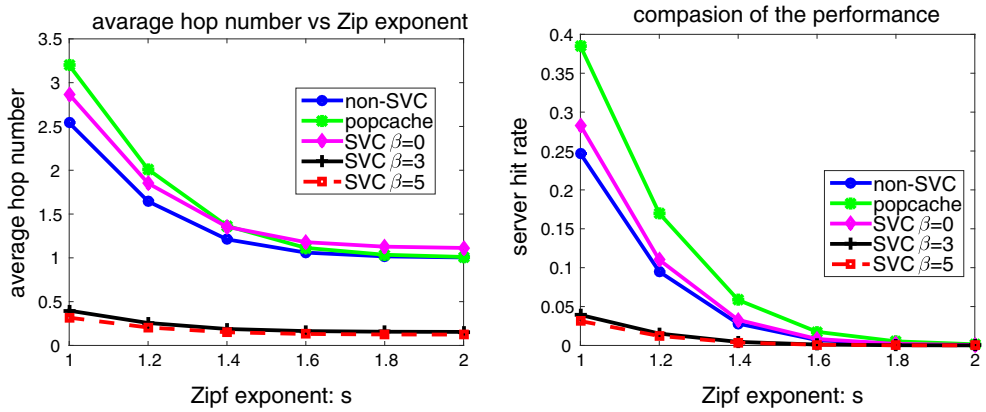



(a) average transmission hops (b) server hit rates



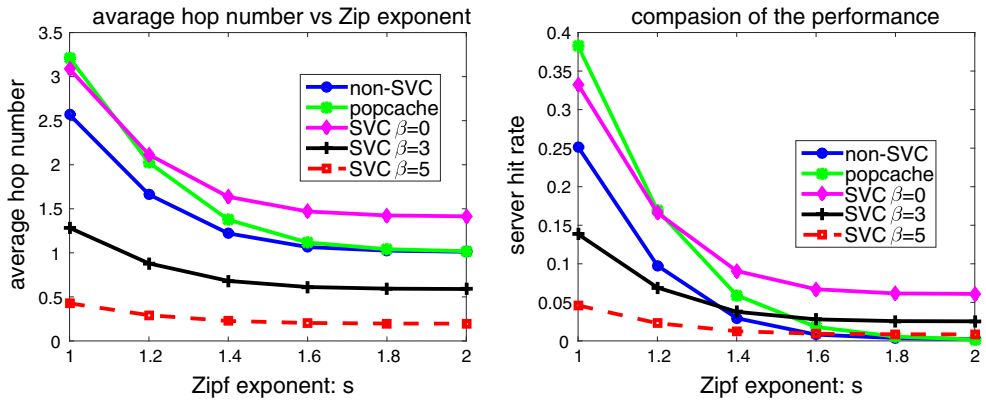

(a) average transmission hops (b) server hit rates

**Fig. 15** Performance evaluation of the CCN with Binary tree network topology with MRU




(a) average transmission hops (b) server hit rates

**Fig. 16** Performance evaluation of the CCN with Cascade network topology for fast-start video delivery




(a) average transmission hops (b) server hit rates

**Fig. 17** Performance evaluation of the CCN with Binary Tree network topology for fast-start video delivery

**Table 3** The average received video quality vs $\delta$ and $\beta$

| PSNR (dB) | $\beta = 0$ | $\beta = 3$ | $\beta = 5$ |
|---|---|---|---|
| $\delta = 1$ | 39.2 | 30.7 | 28.1 |
| $\delta = 2$ | 39.2 | 38.5 | 30.6 |

from the server. We can observe from the results that *always* performs the worst and $fix = x$ is better comparing with *always*. The *popcache* is better comparing with *always* and $fix = x$ but worse comparing with our proposed scheme. The advantage of the proposed scheme is due to the caching redundancy reduction, which leads to the caching of more contents.

Figure 10a, b show the average transmission hops needed and the server hit rates in the scenarios with the Binary Tree topology. We could observe similar performances as shown in Fig. 9, where the proposed scheme greatly reduces the transmission hops needed and the server hit rates.

We also investigate the performances of the proposed scheme at different time instances, as shown in Fig. 11. In this figure, the topology used is the Cascade with 5 levels and $s$ is set to be 1.4. The x-axis is the total running time of the proposed scheme, and the data to calculate the corresponding y-axis value is from time $x - 300$ to time $x$. For example, when x = 500, the system runs for 500s and the results shown are calculated based on the data from 200 to 500 s. The system updates the routers' caches before $x - 300$. From the figures, we can observe that the proposed scheme's performances become stable as the system runs and the 'adjusting' period is quite short. The results regarding the server hit rate are quite similar.

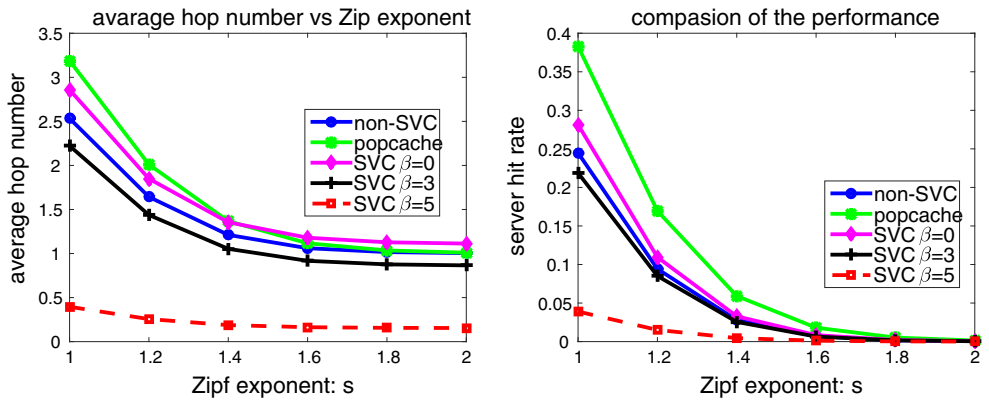
Mostly recently used (MRU) and random replacement (RR) are also investigated as the replacement schemes. Figure 12 shows the performance of the Cascade network topology with RR and Fig. 13 shows the performance of the Cascade network topology with MRU as replacement strategy. We could observe similar performance as the cases where LRU is used. Comparing with LRU, the RR and MRU do not effect the performance of our proposed scheme due to that the proposed scheme only caches the content that should be cached in that router and hence there is no replacement. But the cache replacement schemes do affect the competing schemes since these schemes have redundancy problem. Among the three discussed cache replacement schemes, LRU and RR perform similarly, and MRU performs the worst.

Figure 14 shows the performance of the Binary Tree network topology with RR replacement strategy and Fig. 15 shows the performances of the Binary Tree network topology with MRU replacement strategy. We could also observe similar performances as the results in the Cascade topology. From the results, we observe that our proposed scheme is not affected by the cache replacement schemes. While the competing schemes are affected by the cache replacement schemes.

### 5.3 Simulation results for fast-start video delivery

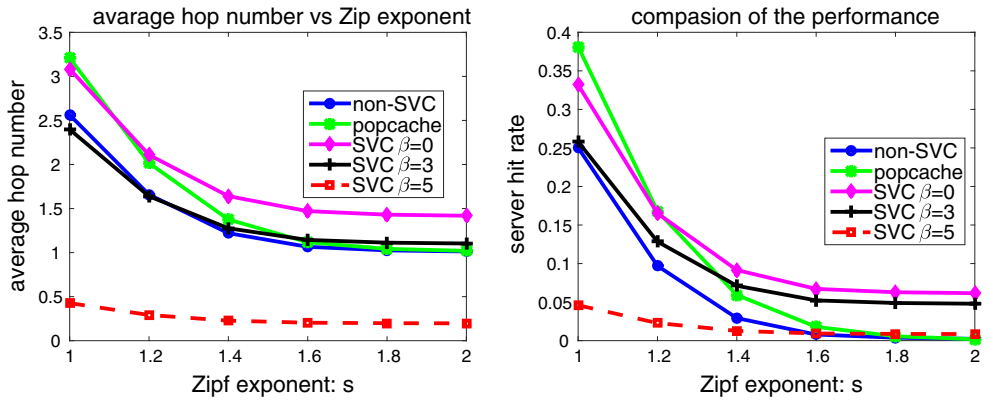
In this section, we show the performance of the proposed fast-start video delivery scheme. *always* and $fix = x$ are not shown in the figures to save space. Instead, we have *non-SVC* which stands for the scheme that does not provide the fast start (i.e. the caching scheme is used but the H.264 SVC encoding method is not applied). *SVC* $\beta = x$ stands for the scheme we proposed with $\beta$ to be $x$. When $\beta = 0$, we set $r_i = L$.

We first let $\delta = 1$ and LRU to be the caching replacement policy, the corresponding results of the Cascade topology and the Binary Tree topology are illustrated in Figs. 16 and 17, respectively. When we count the number of hops, the sizes of the video layers are taken into consideration.




(a) average transmission hops (b) server hit rates

**Fig. 18** Performance evaluation of the CCN with Cascade network topology for fast-start video delivery with $\delta = 2$




(a) average transmission hops (b) server hit rates

**Fig. 19** Performance evaluation of the CCN with Binary Tree network topology for fast-start video delivery with $\delta = 2$

Specifically, we assume the requested video's size is $X$ when it is encoded using H.264 (without using H.264 SVC), and we count the number of hops needed as $\frac{h*X'}{X}$, where here $h$ stands for the number of hops to the nearest server/router that stores the video layer to be delivered and $X'$ stands for the source rate of the corresponding video layer chosen.

From Fig. 16, we can observe that as $\beta$ increases, the average number of transmission hops needed decreases. This is due to that less number of video layers are sent, also because lower layers are assigned with higher priority and hence cached closer to users when $\beta$ is larger. Compare with *non-SVC*, the average number of hops of *SVC* $\beta = 0$ increases, this is due to the scarifies of the video encoding efficiency, where the same video encoded by H.264 SVC is slightly larger than the video encoded using H.264. By delivering the requested video using fewer number of transmission hops, the fast-start video delivery is provided. At the same time, different $\beta$ lead to different video quality degradation. From Table 3, we can see the resulted video qualities are different. Smaller $\beta$ means more video layers will be delivered and the received video quality is better. When $\beta$ is fixed, increasing $\delta$ may enlarge the number of layers transmitted, hence the video quality can be improved as well.

As to the server hit rate, since larger $\beta$ will lead to fewer delivered video layers and these layers are assigned higher priority according to our scheme (they are placed closer to users), the server hit rate decreases with larger $\beta$. Please note that a steadily varying $\beta$ is also possible in this system, where the user could enjoy a steady video quality improvement during the playback.

Figure 17 illustrates the simulation results when Binary Tree topology is used. From Fig. 17, we can observe similar performances with the results shown in Fig. 16, where the proposed scheme can let users receive the requested video with fewer transmission hops. Hence fast-start video delivery is provided. Consequentially, the server hit rate is smaller.

We also test the performance with different $\delta$ by changing $\delta$ to be 2. The corresponding results are shown in Figs. 18 and 19 as the simulation results with the Cascade and Binary Tree topology, respectively. We can observe that when $\delta$ increases, the number of layers need to be delivered increases. The number of hops needed increases (the start time is delayed) and the server hit rate also becomes larger. Meanwhile, the quality of the received video becomes higher.

## 6 Conclusion

In this paper, we propose to encode the video content using SVC for the fast-start video delivery in CCN. The tradeoff between the video quality sacrificing and waiting time reduction are mathematically formulated in this paper. According to the designed protocol, given an assigned weight by users, the system can help calculate how each video layer should be cached in the routers and how many layers will be delivered to each user. Extensive simulations are conducted to verify the performances. The experimental results show that the proposed scheme can outperform the state-of-the-art schemes significantly.

**Acknowledgments** This work is partially supported by JSPS KAKENHI under Grant 15K21599, 26730056 and JSPS A3 Foresight Program

## References


1. Jacobson V, Smetters DK, Thornton JD, Plass MF, Briggs NH, Braynard RL (2009) Networking named content. In: Proceedings of the 5th international conference on emerging networking experiments and technologies, coNEXT. New York, USA: ACM, pp 1–12
2. Ahlgren B, Dannewitz C, Imbrenda C, Kutscher D, Ohlman B (2012) A survey of information-centric networking. IEEE Commun Mag 50(7):26–36
3. Zhang L, Estrin D, Burke J, Jacobson V, Thornton JD, Smetters DK, Zhang B, Tsudik G, Massey D, Papadopoulos C, et al. (2010) Named data networking (ndn) project, Relatório Técnico NDN-0001 Xerox Palo Alto Research center-PARC
4. Koponen T, Chawla M, Chun B-G, Ermolinskiy A, Kim KH, Shenker S, Stoica I (2007) A data-oriented (and beyond) network architecture. In: ACM SIGCOMM Computer Communication Review, vol 37, no 4. ACM, pp 181–192
5. Suksomboon K, Tarnoi S, Ji Y, Koibuchi M, Fukuda K, Abe S, Motonori N, Aoki M, Urushidani S, Yamada S (2013) Popcache: cache more or less based on content popularity for information-centric networking. In: IEEE 38Th conference on local computer networks (LCN), 2013, pp 236–243
6. Psaras I, Clegg RG, Landa R, Chai WK, Pavlou G (2011) Modelling and evaluation of ccn-caching trees. In: NETWORKING 2011. Springer, pp 78–91
7. Liu Z, Cheung G, Ji Y (2013) Optimizing distributed source coding for interactive multiview video streaming over lossy networks. IEEE Trans Circuits Syst Video Technol 23(10):1781–1794
8. Liu Z, Cheung G, Chakareski J, Ji Y (2015) Multiple description coding and recovery of free viewpoint video for wireless multipath streaming. J Sel Top Sign Proces 9(1):151–164
9. Cha M, Kwak H, Rodriguez P, Ahn Y.-Y., Moon S (2007) I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system. In: Proceedings of the 7th ACM SIGCOMM conference on internet measurement. ACM, pp 1–14
10. Carofiglio G, Gallo M, Muscariello L, Perino D (2011) Modeling data transfer in content-centric networking. In: Proceedings of the 23rd international teletraffic congress. International teletraffic congress, pp 111–118
11. Suksomboon K, Ji Y, Koibuchi M, Fukuda K, Abe Nakamura Motonori S, Aoki M, Urushidani S, Yamada S (2012) On incentive-based inter-domain caching for content delivery in future internet architectures. In: Asian Internet Engineering Conference. ACM, pp 1–8
12. Krishnan SS, Sitaraman RK (2013) Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs. IEEE/ACM Trans Networking 21(6):2001–2014
13. Schwarz H, Marpe D, Wiegand T (2007) Overview of the scalable video coding extension of the h. 264/avc standard. IEEE Trans Circuits Syst Video Technol 17(9):1103–1120
14. Wiegand T, Sullivan GJ, Bjontegaard G, Luthra A (2003) Overview of the h. 264/avc video coding standard. IEEE Trans Circuits Syst Video Technol 13(7):560–576
15. Wien M, Cazoulat R, Graffunder A, Hutter A, Amon P (2007) Real-time system for adaptive video streaming based on svc. IEEE Trans Circuits Syst Video Technol 17(9):1227–1237
16. Schierl T, Hellge C, Mirta S, Gruneberg K, Wiegand T (2007) Using h. 264/avc-based scalable video coding (svc) for real time streaming in wireless ip networks. In: IEEE international symposium on Circuits and systems, ISCAS 2007. IEEE, pp 3455–3458
17. An R, Liu Z, Ji Y (2014) Video streaming for highway vanet using scalable video coding. In: Vehicular technology conference, 2014 IEEE 80th. IEEE, pp 1–5
18. Liu Z, Dong M, Gu B, Zhang C, Ji Y, Tanaka Y (2015) Inter-domain popularity-aware video caching in future internet architecture. In: 11Th international conference on heterogeneous networking for quality, reliability, security and robustness (Qshine)
19. Rossi D, Rossini G (2012) On sizing ccn content stores by exploiting topological information. In: IEEE Conference on computer communications workshops (INFOCOM WKSHPS), 2012, pp 280–285
20. Chai WK, He D, Psaras I, Pavlou G (2012) Cache "less for more" in information-centric networks. In: Proceedings of the 11th international IFIP TC 6 conference on networking - Volume Part I, ser. IFIP'12. Springer, Berlin, pp 27–40
21. Psaras I, Chai WK, Pavlou G (2012) Probabilistic in-network caching for information-centric networks. In: Proceedings of the second edition of the ICN workshop on information-centric networking, ser. ICN '12. New York, NY, USA: ACM, pp 55–60
22. Li Y, Lin T, Tang H, Sun P (2012) A chunk caching location and searching scheme in content centric networking. In: IEEE ICC. IEEE, pp 2655–2659
23. Li Z, Simon G (2011) Time-shifted tv in content centric networks: The case for cooperative in-network caching. In: IEEE ICC. IEEE, pp 1–6
24. Li J, Wu H, Liu B, Lu J, Wang Y, Wang X, Zhang Y, Dong L (2012) Popularity-driven coordinated caching in named data networking. In: Proceedings of the eighth ACM/IEEE symposium on architectures for networking and communications systems. ACM, pp 15–26
25. Wu H, Li J, Pan T, Liu B (2013) A novel caching scheme for the backbone of named data networking. In: 2013 IEEE international conference on Communications (ICC). IEEE, pp 3634–3638
26. Suksomboon K, Fukushima M, Hayashi M, Ji Y (2014) Pending-interest-driven cache orchestration through network function virtualization. In: IEEE GLOBECOM. IEEE, pp 1867–1872
27. Jin J, Ji Y, Zhao B, Hao Z, Liu Z (2011) Error-resilient video multicast with layered hybrid fec/arq over broadband wireless networks. In: Global telecommunications conference (GLOBECOM), 2011 IEEE. IEEE, pp 1–6
28. Lee J, Hwang J, Choi N, Yoo C (2013) Svc-based adaptive video streaming over content-centric networking. TIIS 7(10):2430–2447
29. Hwang J, Lee J, Choi N, Yoo C (2014) Havs: hybrid adaptive video streaming for mobile devices. IEEE Trans Consum Electron 60(2):210–216
30. Liu Z, Feng J, Ji Y, Zhang Y (2014) Eaf: energy-aware adaptive free viewpoint video wireless transmission. J Netw Comput Appl 46:384–394
31. Reichel J, Schwarz H, Wien M (2005) Joint scalable video model jsvm 0. In: Joint video team of ITU-t VCEG and ISO/IEC MPEG, Doc. JVT N, vol 21, pp 723–726