International Journal of Future Generation Communication and Networking
Vol. 7, No. 6 (2014), pp. 165-180
http://dx.doi.org/10.14257/ijfgcn.2014.7.6.16

Opportunities and Challenges of HTTP Adaptive Streaming

Hongyun Yang[1], Xuhui Chen[2]*, Zongkai Yang[3], Xiaoliang Zhu[3] and Yi Chen[4]

[1]College of Education, Hubei University, Hubei, 430062, China
[2]College of Mathematics & Computer Science, Wuhan Textile University, Hubei, 430070, China
[3]National Engineering Research Center for E-learning, Central China Normal University, Hubei, 430079, China
[4]Computer School, Central China Normal University, Hubei, 430079, China

yhy_cxh@126.com, {yhycxh, wh.chxh}@gmail.com

Abstract

HTTP-based Adaptive Streaming (HAS) has emerged as the prominent technology for the delivery of audiovisual content over the Internet in recent years and has a major impact on network traffic. Although traditional stateful session-based streaming solution based on UDP was used initially for media content delivery, researchers and practitioners soon realize that HAS technology, due to get through firewalls friendly , transfer NAT easily ,effectively utilize the existing networking infrastructure and provide uninterrupted video streaming service to users with dynamic network conditions and heterogeneous devices, has the potential to improve the Quality of Experiments compared with traditional streaming technologies. Consequently, various HAS media streaming solutions have been proposed and deployed successfully. This paper reviews the state-of-the-art of HAS technology and discoveries achieved by numerous researchers. We describe the basic taxonomy of HTTP adaptive streaming systems and summarize the major issues associated with HAS systems' design. Then we outline the key challenges and open problems and highlight possible avenues for future directions.

Keywords: *Adaptive Streaming over HTTP, Live, Video-on-demand*

1. Introduction

Over the past few years, with the emergence of convenient and advanced digital multimedia capture and production technologies, a large number of applications, including Internet video, distance education and online game etc., have been created and deeply stepped into people's daily life. As the result, the amount of available media contents has increased tremendously. The video streaming is a huge and growing fraction of Internet traffic and occupied the most of the total traffic. A recent Visual Networking Index report by Cisco [1] predicts that video will constitute 80%~90% of the total Internet traffic by 2017. Netflix, which is the leading subscription service provider for online movies and TV shows in the US [2] has consumed 29.7% of peak downstream traffic in the US and Canada in 2011 and 32.25% in the US in 2012. Internet video has become the single biggest Internet traffic generator [3] and with the ten times faster than other applications traffic growth [4]. This trend significantly challenges content providers as well as Internet service providers (ISP) as to assure users to receive a high-quality experience.

In order to provide a high-quality user experience with uninterrupted video streaming service under dynamic network conditions and heterogeneous devices, HTTP-based Adaptive Streaming (HAS) technology was first proposed by Move Network company in 2006[5] to provide live or on-demand service to a large number of users. Then the majority of media content providers and streaming service providers, such as Microsoft, Apple, Akamai and Netfflix etc., quickly adopted the technology. Several HAS solutions have been proposed and widely used, such as Microsoft's Smooth Streaming, Adobe's http dynamic streaming, Apple's Http live streaming, Akamai's HD as well as the players used by Netflix. HAS technology has launched the latest shift in video streaming technologies. However HAS uses HTTP/TCP as transport protocol. Traditionally convention wisdom believed that TCP protocol doesn't fit for video streaming transmission because of the throughput variations caused by TCP's congestion control and the potentially large retransmission delays [6]. Although in the recent few years, some researchers [7] systematically investigate the performance of TCP for both live and stored media streaming and point out that TCP generally provides good streaming performance when the achievable TCP throughput is roughly twice the media bitrate with only a few seconds of startup delay. However since HAS uses dynamic rate selection algorithms to adapt video quality, the complex interactions between TCP's congestion control and the application's rate-adaptation mechanisms create a "nested double feedback loop[8]" and the dynamics of such interacting control systems can be notoriously complex and hard to predict. What's more, some experiments have done to evaluate existing commercial HAS players' performance [9, 10] and the results show when multiple HAS players compete for bandwidth ,it will cause the downward spiral effect phenomenon [9].And research [11] further indicates that all these are caused by current heuristic rate adaption methods. As the video traffic volume rises, rate selection inefficiency becomes more prominent and is raising considerable challenges for the design of HAS solutions.

In this paper, we review the state-of-the-art of HAS technologies, especially the key algorithms, such as rate adaption algorithms and server selection algorithms, present the taxonomy of various solutions that have emerged. We examine typical HAS components and key technologies. We then outline future challenges that must be addressed to make Internet video using HAS technologies more reality.

The remainder of this paper is organized as follows. Section 2 we briefly discuss the technology choice for Internet video. In Section 3 we introduce the HAS technology and the research status of some key technologies problems. We then present current technical challenges and open issues in Section 4. Section 5 concludes this paper.

2. Technology Choices for Internet Video

From the viewpoint of media streaming protocol we first review the evolution of video delivery technologies.

2.1. Stateful Session-based Proprietary Streaming Technologies

In the Internet environment, the primary issue for media distribution is how to provide the highest quality of user experiments, such as short start-up delay, smooth playback and high bit-rate streaming services. Taking into account the real-time requirements of media distribution, traditional streaming technologies adopt session-based proprietary streaming protocol, such as RTSP (Real-Time Streaming Protocol) [12], RTMP (Adobe's Real Time Messaging Protocol) [13] and Microsoft's MS-RTSP [14] protocols. In these protocols, the client connects to the streaming server from the first time and the streaming server keeps track of the client's state until the time the client disconnects from the server. And during the session, the client communicates its state to the server by issue commands such as PLAY, PAUSE or TEARDOWN in RTSP protocol.

Since session-based proprietary streaming technologies have been proposed, they have been widely used from pure audio conferences to multicast multiparty low delay video sessions applications for short startup latency, low control overhead, good user interaction performance and smooth audio and video playback experience. However, there are some disadvantages due to the protocols' implementation and system deployment hindering its future deployed in today's Internet environment. First, the realization of these technologies requires a pre-configured and specialized server .These servers require specialized skills to set up and maintain, and in a large-scale deployment this can be costly. Second, these protocols generally based on UDP as transport protocol and UDP traffic is often not permitted by default firewall and NAT settings. Third, the server has to keep track of the state of every streaming session, which will cost a large of servers' resources and limit the system's scalability. Finally, in conventional stateful session-based proprietary streaming protocols, the bitrate the server transmitting content to clients equals to the media encoding bitrate, which equals to the client's media playback bitrate. Under normal circumstances, this will ensure that the client buffer level remains stable over time and optimize the use of network resources [15]. However if the network environment becomes terrible, packet loss or transmission delays occur, the client's media consumption rate is greater than the available transmission bandwidth. Once the client's buffer filling rate is less than the consumption rate, it is likely that the client's buffer drained out and causes the playback pause.

2.2. HTTP Progressive Download

In order to reduce the complexity of system's deployment and effectively utilize the existing networking infrastructure, http progressive download technology, which originally is used for web file transfer was proposed. The technology uses a standard HTTP web server rather than a streaming server to transmit the media file. And the video is encoded as one big chunk, the client can playback once the first few seconds' content loaded in its buffer while the download process is still in progress. Popular video sharing sites on the web today, including YouTube, Youku, Vimeo, MySpace, and MSN Soapbox, almost exclusively use progressive download.

HTTP progressive download technology uses existing HTTP caching servers that are specialized for HTTP packet delivery, which makes media delivery systems based on this technology can be deployed quickly. And since it is based on TCP as transport protocol, the technology is simplicity and friendly to get through firewalls and transfer NAT. However there are still existing two major shortcomings which hamper its further usage. One is that the technology can't change video quality (bit rate) to adapt network congestion .Using the technology, all clients will receive the same encoding of the video, despite the large variations in the underlying available bandwidth both across different clients and across time for the same client. The other is bandwidth unnecessary waste. For instance, if 20 seconds into a fully downloaded 20 minute video, the viewer decides that he doesn't like it and quit the video, both the viewer and his content provider have to waste 19 minutes and 40 seconds worth of bandwidth. Although some strategies have been proposed, such as slowing down the speed of video loading so media player doesn't keep loading video in the background or bitrate throttling provided by Microsoft IIS 7.0 web services[14] ,which reduce unnecessary delays and give the user the choice to abort the streaming before the whole content is downloaded, thus reducing bandwidth waste. However even if the client or server would limit the download speed to be “just fast enough”, once the client starts downloading, it still downloads as fast as possible, wasting a lot of bandwidth when the video is not watched fully.

2.3. HTTP Adaptive Streaming

With the development and widely deployment of Content Delivery Network (CDN), the Internet infrastructure has evolved to efficiently support HTTP. For instance, CDNs provide localized edge caches, which reduce long-haul traffic. HTTP/TCP protocol has become the mainstream for content delivery on the Internet. What’s more, Considering the Internet based on IP protocol only provides “Best-effort” delivery service, it is essential to provide uninterrupted video streaming service to heterogeneous devices (display resolution, CPU and memory resources) under the dynamic networking conditions, especially for devices in wireless and mobile network environment. Driven by these motivations, in 2006 HTTP-based Adaptive Streaming (HAS) was originally proposed by Move network Company [5] and further proved its feasibility in 2008 to provide live and on-demand Internet video streaming service to a large number of users. In October 2008 Microsoft announced that Internet Information Services (IIS) 7.0 would feature a new HTTP-based adaptive streaming extension: Smooth Streaming and applied it during the coverage of the 2008 Beijing Summer Olympic Games [14]. From then on, the technology is quick put to use by many streaming service providers, such as Apple, Adobe, Akamai and Netflix.

HAS refers to a set of novel streaming approaches. It is a hybrid delivery method that acts like streaming but is based on HTTP progressive download [16]. In contrast to conventional RTP/UDP-based video streaming, HAS uses HTTP/TCP -the protocol stack traditionally used for web traffic to deliver streaming media content. Just like HTTP progressive download, the technology is firewall friendly and doesn’t need NAT transfer. And by using CDN and standard HTTP optimization techniques, the technology can reduce the server-side cost and improve server-side scalability. In addition to these advantages, the most important feature of the technology is rate selection algorithms to adapt video quality based on the available bandwidth capacity of the network path between the server and the client so as to provide uninterrupted video streaming service.

But since everything has two sides, so does HAS. First, the basic disadvantage of this approach is the increased storage requirements and the fact that adaptation is characterized by a coarser granularity since video bitrates only belong to a discrete set of levels. Second, although the client’s playback buffer can reduce skips, freeze, and stutter and smooth the playback process, it introduces a little longer startup delay than stateful based technology but faster than progressive download technology because the buffer has to reach a certain threshold before playing. Another issue is fairness; stability and effectiveness when multiple http streamings complete the bottleneck network. We will discuss these issues in Section 4 in detail.

Table 1 summarizes the differences among the above three streaming technologies. In the table, we use the technology based-on RTSP/RTP protocols (abbreviated as RTSP/RTP) as the typical representation of stateful session-based proprietary streaming technologies.

Table 1. The Comparison of Three Streaming Media Technologies

Performance	RTSP/RTP	HTTP progressive download	HAS
Service-side implementation	Proprietary streaming media	Common web server	Common web server

	server		
Client-side implementation	Difficult	Easy	Easy
System Installation and Configuration	Complex	Simple	Simple
Support application type	VOD, Live streaming	VOD	VOD, Live streaming
Startup delay	<2s	>30s	approximation 30s(under certain configuration)
Client buffer	Memory, the smallest	Hard disk, file size	Memory, smaller
Support VCR operation	Support, seek with high accuracy	Support partly	Support, seek with general accuracy
Network bandwidth adaptation	Support partly	Not support	support
Bandwidth usage	Likely more efficient	Likely the least efficient	Likely less efficient
Server Fault Protection	Not support	Not support	Support
Support DRM	Good	Bad	Better
Friendly to firewall	Bad	Good	Good
Multicast Support	Yes	No	No
Application Scope	Large-scale, low latency, real-time streaming media system	Short low bit rate video, such as advertisement, Trailers	Heterogeneous, dynamic media streaming systems, such as mobile media streaming ,fit for embedded devices

3. State-of-the-Art of HAS Technology

In the section, we discuss the category of HTTP adaptive streaming technology and then discuss some key components and related algorithms adopted in a typical HTTP adaptive streaming system.

3.1. The Category of HTTP Adaptive Streaming

According to the adaptive strategies, HAS can be classified into three main categories [17]: transcoding-based, scalable encoding-based and stream-switching.

Transcoding-based method can fine-granularly throttle frame rate, compression and video resolution by means of on-the-fly transcoding of the raw content to match a specific bitrate. However, due to the per-client processing load, this method is poor scalability and difficult to be deployed in CDN network. Scalable encoding-based method employs scalable codecs such as H264/SVC [18] and exploits spatial and temporal scalability to adapt picture solution and frame rate. Comparing to transcoding-based method, it can reduce the processing load and improve the scalability. However the method needs to deploy specialized servers implementing the adaptation logic. In Stream-switching method (also called Multiple Bit Rate (MBR) method), the raw video is encoded into different bitrates and quality levels (also called profiles) and storage, maintained by the web server. Further the video profile is partitioned in short segments of a few seconds long. Meantime the server maintains a manifest file with necessary information. After downloading the manifest file, the client then depends on corresponding rate adaption mechanisms to select the video bitrate of the segment on-the-fly, downloads the segments sequentially at different encoding bitrates using plain HTTP GETs. Figure 1 shows a typical HTTP adaptive streaming system.

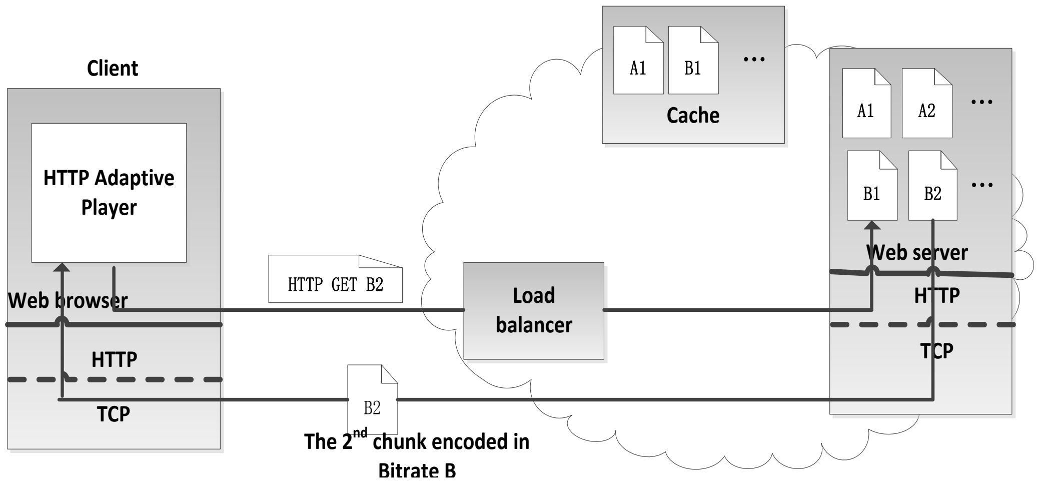

Figure 1. A Typical HTTP Adaptive Streaming System [19]

Comparing to transcoding-based and Scalable encoding-based methods, stream-switching method is coarser granularity, codec-agnostic and needn't special server to support. What's more, the method minimizes the processing costs because once the video is encoded, no further processing is required in order to adapt the video to the variable bandwidth. The method is relative simple and quick deployment. But the disadvantage of this approach is the increased storage requirements since the same video content has several profiles with different bitrate.

Nowadays leading commercial media service providers are preferring the stream-switching approach to the others and a large number of related streaming solutions ,such as Microsoft Smooth Streaming(SS),Apple's HTTP Live Streaming(HLS),Adobe's HTTP Dynamic Streaming (HDS) and Akamai's HD streaming are proposed and widely deployed. The paper focuses on stream-switching HTTP streaming, referred to as HAS without particular description. However with the development of video codec technology and Coud media services, it is becoming reality to adopt scalable encoding-based method in HTTP streaming. So we will introduce such method in our discussion in Section 4.

3.2. Key Design Issues of HAS

In a typical HAS systems (Figure 1), there exist two basic parts: the client and the server. The server saves the video content encoded into multiple versions at different rates and each encoded video is further fragmented into small video chunks. The client requests chunks using HTTP Get command. To construct and maintain an efficient Internet video delivery system, mainly three questions should be answered. The first relates to the video segmentation and chunks management, *i.e.*, whether dose the chunk length affect the performance of clients received? How to index all the chunks in the server so as to be quickly retrieved and easy managed? The second concerns bitrate selection, *i.e.*, once the segmentations are organized, how to select bitrate to adapt the dynamic network throughput?

And what are the import criteria? The third is server selection, *i.e.*, in order to support plenty of clients to simultaneously watch video, many HAS systems employ multiple servers hosting the same set of video contents. Under the environment, whether the server selection methods affect the performance? What are the server selection methods proposed in recently literature?

Since HAS has been proposed in 2006, several HAS systems and algorithms have been proposed to address the above issues. Following from the view of the main components of a typical HAS, we outline a brief survey .And because we focus on the video delivery technology and the video encodec methods are out of the scope of the paper.

3.2.1. Content Segmentation and Organization: In a typical HAS implementation, a video/audio content is encoded into multiple versions at different rates. Each encoded video/audio is further cut into many short segments ("chunks"), each of which contains seconds or tens of seconds worth of video/audio. At the video codec level, the chunks are carefully encoded and cut along video GOP (Group of Pictures) boundaries without any gaps or overlaps between them, so the chunk will be decoded independently on past or future chunks. HAS systems can provide seamless video playback.

The encoded chunks are hosted on a HTTP web server. And there are two chunks organization approaches. One is the one-file-per-chunk approach and the other is one-file-all chunks of the same encoded bit rate approach (abbreviated as one-file-per-bitrate). One-file-per-chunk approach is used in prototype implementation of Microsoft's Smooth Streaming solution and Apple's HLS. This approach is very straightforward. However since each chunk will be uploaded to CDN's web servers ,CDN operators lose many hours managing the millions of tiny files. So the approach using one file to save the chunks of the same bitrate was proposed in subsequent products of Microsoft.

Nowadays several solutions have been deployed among real systems and used by a large number of users, such as Microsoft Smooth Streaming (SS), Apple Http Live Streaming (HLS), Adobe Http Dynamic Streaming(HDS) and MPEG-DASH standard which has been released as ISO/IEC DIS 23009-1.2[20] by MEPG—the International Standardization Organizations in 2012. Different solutions have their own file format. In table 2 we use these commercial solutions as representative examples and compare the segmentation and organization of them.

Table 2. Compares the Segmentation and Organization of Four Typical Solutions

File Format and Extension	Microsof's Smooth Streaming	Apple HTTP Live Streaming	Adobe's HTTP dynamic streaming	MPEG-DASH
Container	MP4	MP2 TS(MPEG-2 Part 1)	MP4(MPEG-4 Part 14 and Part 12)	MP4,MP2 TS
File storage unit	One-file-per-bitrate	One-file-per-chunk	One-file-per-bitrate	One-file-per-chunk
Segment size	2s	10s	depending on the particular implementation	flexible
Data File extensions	.isma, ismv	.ts	.f4f	.mp4,.m4s
Server	.ism	.m3u8	.f4x	.mpd

manifest file extension				
Client manifest file extension	.ismc	.m3u8	.f4m	.mpd
Codec	V: H.264 or VC1 A: AAC or WMA	V：Mpeg4,H.264 A：AAC,mp3	V: H.264 or VP6 A: AAC or MP3	open

3.2.2. Rate Adaption Algorithms: In streaming delivery systems, in order to overcome the variable network bandwidth resource of the Internet and maximize the user's quality of experience of multimedia streaming services, the video rate needs to be adjusted on-the-fly to adapt the varying network capacity. The mechanism is called rate adaption. Depending on whether the adaption is controlled by the client or the server, these mechanisms can be mainly classified into two categories [21]: sender-driven [17, 22] and receiver-driven [8, 23-24].

Some works focuses on server-side solutions. YouTube experiments with server-side pacing to ensure full utilization of the link. [22] proposed a method to estimate client-side buffer occupancy in the server and to adapt media bitrates to maintain the client-side buffer occupancy above a certain threshold in TCP-based streaming. In [17] based on the results obtained by an experimental investigation of Akamai HD Video Streaming, the authors pointed out that receiver-driven rate control method caused a forward connection delay and a backward connection delay, which increased the response time when the link's available bandwidth changed. In order to maximize the QoE by delivering the best quality that was possible given the network available bandwidth while minimizing playback interruptions, the authors proposed Quality Adaptation Controller (QAC) centralized at the server. The controller took the error of pre-defined threshold value to sender buffer size as input and chose a proportional integral (PI) controller to ensure that the video level matched the available bandwidth on average. The advantages of these methods are since the server knows all the requests sending from the clients, it can assign the bandwidth more fairly. However, the servers have to always control the bitrate and thus suffer from large workload which reduces the system's scalability greatly. What's more, current sender-driven method [17] tries to maintain the sending buffer at a target level without estimating the bandwidth. However, maintaining the sending buffer cannot provide the same rebuffer guarantee same as at the client side. So the most researchers believe that the client is in the best position to detect and respond to overall dynamics of the system.

The bitrate selection in Microsoft's smooth streaming (SS) is a typical receiver-driven method. In SS's video streaming service, different bit rate segments are encoded with configurable bit rates and video resolutions at servers and clients dynamically switch among different bit rate segments by requesting videos based on conservative available-bandwidth estimation of links. And the conservation bandwidth evaluation means that it prefers to estimate the availble reliably by using several per-fragment throughput measurements instead of acting opportunistically based on the latest fragment throughput measurement [8].what' more, in order to avoid annoying the user with sudden quality transitions and providing a dynamic but smooth watching experience, the clients in SS avoid large jumps in the requested bitrate (more than two successive bitrates). Adobe and Apple also develop client-side HTTP adaptive live streaming solutions operating in the similar manner. Akamai's High Definition (HD) video Streaming for Flash over HTTP [25] is another video solution which adopts receiver-driven rate adaption strategy. In Akamai's HD solution, the video client monitors its buffer size, received video frame rate and received good put to estimate the link's available bandwidth and implements a proportional controller which takes the error of received buffer size and the target buffer size as the input and the throttle percentage as the output to ensure that the player buffer length tracks a desired buffer length. The client period decides the bitrate for requesting the next chunk that matches the available bandwidth and sends commands with those parameters to the server and. In [21], Chenghao Liu *et al.* proposed a receiver-driven rate adaption method for HTTP/TCP streaming that used the ratio of media segment duration to the latest single segment fetch time to measure the HTTP throughput and used a step-wise increase/aggressive decrease method to switch up/down between the different representations of the content that were encoded at different bitrates. Different from the authors' previous paper [21] where segments are requested and received sequentially, in [24] the authors proposed parallel segment fetching method to fetch different segment or portion of a segment from different edge servers in CDN and used a sliding window to measure the latest multiple rate adaptation metrics to determine switch-up/switch-down. These above algorithms are merely based on bandwidth estimation and still adopted by several mainly commercial HAS systems [8]. In [23] Guibin Tian and Yong Liu proposed client-side video adaption algorithms to strike the balance between the responsiveness and smoothness in HAS system. As a control system always benefits from more accurate measurements, the authors introduced the client-side buffered video time as feedback signal and designed the algorithm as a PID controller in conjunction with the video rate selector which uses a prediction of TCP throughput as an input to the ABR algorithm. Since the client is in the best position to detect and respond to overall dynamics of the system [26],the advantages of receiver-driven rate selection algorithms are flexible strategies, system's scalability for reducing the servers' control workload. And recent works suggest the need for cross-CDN optimizations that implies the need for keeping minimal state in the network or servers [27-28] .However since every client has its own rate adaption method, it is difficult to fairly share the networking bandwidth and sustain the playback stability, especially when multiple players share the same network bottleneck [29].

In order to overcome two or more adaptive streaming players compete for bandwidth causing instability problem, several solutions have been proposed to combine the client with the server to shape the bitrate [29-30] or more fine-grained client selection methods [11, 19, 31-32]. We will discuss the problem and current solutions in Section 4.

3.2.3. Server Selection Algorithms: In a large scale, fast growing video streaming platform, such as Netflix [33], it is basic to employ multiple servers(CDN) hosting the same set of video contents, each client is assigned to one server [33] or simultaneously connected to several servers [23] to download different video chunks. So in multiple servers environment, it is important to select one or multiple optimal servers. Current there are two types of server selection methods. One is static server selection and the other is dynamic server selection. In [33], the authors performed active measurements of Netflix and found that Netflix statically assigned a CDN to user for extended period of time. This means that Netflix's players stay attached to a fixed CDN even when the other CDN can offer better video quality. And only when the selected CDN can't support even the very low quality, the player switches to the second CDN. The selected CDN is associated with its rank, which is based upon the user account, independent of available bandwidth from each CDN and remained unchanged for at least several days. These statistical assignment will reduce the quality of users receive. To improve the performance, the authors proposed average bandwidth measurement based CDN selection strategy. In [23], the authors proposed dynamic server selection strategy based on Support Vector Regress (SVR) TCP throughput estimate model. While a client downloads from its current server, it constantly monitors its throughput to other candidate servers by using the SVR TCP throughput estimation model and switches to a new server only if the estimated throughput to that server is at least 20% higher than the achieved throughput with the current server.

4. Technical Challenges and Open Issues

Nowadays HAS technology has taken over as the dominant technology for video delivery over the Internet. The main commercial companies have picked up the technology and built their own adaptive streaming solutions [14, 34-35]. However recent studies have found some important performance issues with this technology including unfairness, instability, and underutilization among the clients [8-11, 19, 30-31]. What' more, although many researcher are conducting early inventions to find ways to answer many questions about adaptive streaming, because it is a new technology and the lack of field data required to conduct a rigid analysis, these questions had yet not to be adequately answered. In this section we review some of these issues as well as the proposed solutions and point out some open issues further.

4.1. Issue of Fairness, Efficiency, and Stability

HAS has been adopted as the technology of video delivery over IP networks. With abundant video content and increasing bandwidth demands, it is becoming commonly that two or more adaptive streaming players have to compete for available bandwidth of a network bottleneck [10, 19, 26]. This competition can lead to three performance problems: player instability, unfairness between players and bandwidth under-utilization. In [10], based on the experiment evaluation of two typical adaptive streaming players, the authors point out that there are two typical phases in a streaming session: buffering-state and steady-state. And in the steady-state phase, it includes activity periods (On periods) followed by inactivity periods (OFF periods). The players can't estimate their fair share bandwidth correctly during Off periods which is the main root cause behind the problems.

To solve the problem, researchers have proposed several solutions. An example is shaping the bitrate, such as [29-30]. In [30], Houdaille and Gouache observed instability and unfairness with competing adaptive streaming players and proposed a traffic shaping method at home gateways, which first determined desirable target bitrates to be reached by each stream and then constrained the clients to stay within their limits to reduce the extent of these problems. Akhshabi and Begen [29] analyzed the instability problem and identified the root cause of the instability problem was that the player overestimated the available bandwidth through an ON-OFF activity pattern in Steady-State phrase and they proposed a server-based traffic shaping method. Through detecting the oscillation during the streaming session, the shaper on the server was activated and dynamically adjusted the shaping rate so as to assure the playback bitrate stably. However since the proposed mechanism does not require the players' cooperation and it is entirely implemented at the server, it introduce overhead for the server.

Another direction is a more elaborate and systematical approach to divide the rate selection process into several components or sub-processes. The typical examples are FESTIVE [26]. In [26], Junchen Jiang *et al.* proposed an adaptive streaming player involved three components: scheduler, adaptation and B/W estimation. And after formally defined the metrics of inefficiency, unfairness and instability and proposed a family of adaptation algorithms that varied in the tradeoff across these metrics. The adaption algorithms include: (1) randomized chunk scheduling to avoid synchronization biases in sampling the network state; (2) a stateful bitrate selection that compensates for the biased interaction between bitrate and estimated bandwidth; (3) a delayed update approach to tradeoff stability and efficiency; (4) a bandwidth estimator that used the harmonic means of download speed over recent chunks to be robust to outliers.

However, based on the measure of three popular video streaming services – Hulu, Netflix, and Vudu [9], Yuan Huang *et al.* [11] found that inaccurate bandwidth estimation would trigger a feedback loop which leads to unnecessary rebuffering events and suboptimal video quality. They called this phenomenon the downward spiral effect. Further they proposed a broad class of video rate selection methods which were only based on the player's current buffer occupancy and avoided estimating bandwidth at all. They formulated the buffer dynamics as a simple differential equation and descripted two models of the streaming buffer. Based on the buffer-rate map plane to constant bit rate (CBR) videos and chunk-rate map plane [32] to variable bit rate (VBR) videos, they described a baseline algorithm. The basic principle of the algorithm was that the player stayed at the current video rate as long the rate suggested by the rate map didn't pass either the next higher available video rate or the next lower available video rate. If either of these barriers were hit, the rate was switch up or down to a new discrete value suggested by the rate map. Besides [11], Zhi Li *et al.* proposed PANDA [31] solution to avoid the pitfall of inaccurate bandwidth estimation. The solution was based on a "probe-and-adapt" principle. In the approach, when the network was congested, the off-intervals were absent, the TCP downloading throughput was taken as an accurate indicator of the fare-share bandwidth. However when the off-intervals was presented, the algorithm constantly probed the network bandwidth by incrementing its sending rate, and prepared to back off once it experiences congestion.

The issue of fairness, efficiency, and stability has become urgent for it effects the performance of Internet video delivery. Although several researches have explore its root cause and several solutions have been proposed, the research is still in its early stage. Currently previous study lacks the analytical and computational model for rate selection. And the performance of different players with different adaptation logic competing bottleneck bandwidth is not discussed. These will be the future work we plan to study.

4.2. Buffer Bloat Effect

Buffer bloat is a phenomenon in packet-switched networks, in which excess buffering of packets causes high latency and packet delay variation [36].It was initially described as far back as in 1985, and gained more widespread attention starting in 2009. In 2012 Gettys and Nichols in [37] collected evidence to show that the Internet can suffer from significant congestion due to the existence of large buffers at different network devices. As a result, traffic can experience very high queuing delays that can reach several hundreds of milliseconds and sometimes even more than a second. High delays can be very harmful to many applications on the Internet such as VoIP, interactive games, e-commerce and video transmission. In [38] the authors used measurements on a test bed to demonstrate and quantify the buffer bloat effect of HAS. Their measurements results showed that in a typical residential setting a single video stream could easily cause queuing delays up to one second and even more hence seriously degrading the performance of other applications sharing the home network. The root cause of the problem was the way TCP flow control. In order to achieve the best throughput, TCP keeps a send buffer of approximately the bandwidth delay product (BDP) of the path between the source and the destination. This means that the maximum number of bytes in flight TCP can have is equal to the BDP. In addition, TCP uses packet losses to detect congestion. When TCP detects packet loss, it realizes that the path is congested and backs off to a lower transmission rate. While it is important for TCP to detect packet loss in a timely manner, large network buffers can store a large number of packets before loss can occur and hence loss detection is significantly delayed. This causes TCP to over-estimate the BDP and consequently send larger bursts of data that fill the large buffers and cause high delays. The maximum burst TCP can send at any point is equal to min (cwnd, rwnd). The value of rwnd is a function of the empty space at the receive socket buffer at any point in time. To resolve the problem they proposed SABRE (Smooth Adaptive Bit RatE) scheme. In order to reduce data bursts, the scheme introduced HTTP pipeline to request multiple video segments to reduce the rwnd size. What' more, the author proposed at application layer using video bitrate to limit the download rate so as to avoiding large data bursts all the time. And in order to overcome the rate control method decreased the bandwidth efficiency, the authors proposed back off/refill mode of operation to tune the players' download bit rate.

However, since buffer bloat phenomenon is widely existed in today Internet, it is not adequate to solve the problem only on client side. A further research direction is to combine the server with the client together as well as network devices to reduce the excess buffer effect and investigate HAS-aware middle box to decrease the effect of On/Off behavior of adaptive video players to other delay-sensitive applications.

4.3. SVC-based HAS

Multi-rate switching based HAS has been evolving into a major mechanism for video delivery for its simplicity to incorporate for content delivery network, friendly to get through firewall and cache efficient. However the technology exists a number of disadvantages such as content redundancy in different quality levels, which require additional storage and bandwidth; a large client play out buffer of several tens of seconds to absorb network impairments; and non-optimal quality selection and bandwidth efficiency under fluctuating network conditions. These disadvantages made researchers introduce SVC (Scalable Video Coding) into HAS [39-43] to resolve part of these problems.

SVC is defined as an amendment for MPEG 4 Advanced Video Coding (MPEG 4-AVC), which uses layered coding including a base layer that is AVC compatible and one or more enhancement layers to support access to an encoded video stream at different quality levels such as resolution, frame rate, or fidelity in order to support diverse terminal equipment and address varying network conditions. Raf Huysegems *et al.* [39] have shown that comparing with traditional AVC HAS, video with SVC encoding would reduce storage overhead greatly with little encoding overhead increased. And an equal amount of bandwidth can be saved on the server/cache link. At the same time SVC-based HAS can resume faster with a next segment at a lower quality to avoid a substantial dip in the buffer filling level and easily to introduce different transport strategies. However a number of challenges have been identified in the application of SVC-HAS [39, 42]. First is svc coding and bandwidth penalty. Paper [39] has shown through simulation that svc coding can introduce roughly 10 percent additional video data per extra layer comparing with avc coding and need more bandwidth, which could lead to reduced viewing quality for the end user if the last mile is the network bandwidth bottleneck. Second is RTT penalty. In HAS system, between the last byte of the previous segment and the first byte of a new segment, there exists an idle-time during which request/response signal are sent. The idle time equals to RTT between the server to the client. Since the fine segment granularity in SVC-based HAS solution, it is more sensitive to large RTT than an AVC solution. Some papers [39, 42, 44] have suggested to use HTTP pipeline management technology to reduce the idle time so as to solve the problem. Third is the HTTP message overhead due to the increasing number of segments in SVC-based HAS.

In [44] Sanchez *et al.* discuss the benefits of using SVC for HAS delivery in terms of web caching and saved uplink bandwidth and propose a scheduling algorithm for live HAS deliver. In [42] the authors compared the performance of AVC-based HAS and SVC-based HAS and summary that AVC performed better under high latencies, while SVC more easily adapted to sudden and temporary bandwidth fluctuations when using a small buffer. In SVC-based HAS, it is essential to decide to either download the next segment or increase the quality of previously downloaded segment. Andelin *et al.* [43] proposed a heuristic algorithm, which used a slope to define the trade-off between downloading the next segment and upgrading a previously downloaded segment. The slope can be configured in the heuristic to give priority to either prefetching (downloading for future segments) or backfilling (downloading for the current segments). If the steeper the slope, the more backfilling will be chosen over prefetching. Similarly, the flatter the slope, the more additional base layers of new segments will be downloaded.

Nowadays SVC-based HAS has aroused researchers interesting. However since the complexity of svc coding and the penalty it brought, the problems need be further study before the technology could be deployed in realistic systems. Some possible alternative solutions include using multiple SVC streams, or using specific hierarchical encodings to create a multitude of bandwidth operating points with a limited number of enhancement layers and encoding overhead [39].

The main advantages of SVC are the possibility of serving a great number of users with different equipment capabilities with a single bit stream and the fact that it facilitates coping with congestion by applying on-the-fly adaptation, performed by adding or subtracting layers to match the capabilities of the network at every time instant.

4.4. Combile with Peer-to-Peer Media Delivery

HAS technology has been the main approach for video delivery. However one of the main challenges of systems based on HAS technology is capacity issue. The unicast nature of the HTTP protocol creates a potential bottleneck at the source of the stream with a linear increase in bandwidth demand as the number of viewers increases. The primary solution to the problem is to use Content Distribution Networks (CDNs) as content cache for the open Internet [45]. However for private networks, which interconnect multiple network segments representing geographically distributed offices with fixed VPN links and interconnect with public Internet through gateway link, it is challenging to deploy efficiently. In these networks, it is hard to deploy and manage private CDNs to assist streaming for its' expensive and specialized hardware.

Peer-to-Peer streaming technology (abbreviated as P2P streaming) has been widely deployed to provide on-demand or live video streaming services over the Internet for nearly ten years. There are two key characters which make the technology more attractive. One is in a typical P2P streaming systems, a peer not only gets data from the network, but also uploads the downloaded data to other users. Consequently, such an approach has the potential to scale with group size, as greater demand also generates more resources. The other is that the technology does not require support from Internet routers and network infrastructure, and consequently is extremely cost-effective and easy to employ to quickly disseminate data [48].

To solve the capacity issue in the Internet, several research efforts have focused on delivery videos using P2P streaming technology with scalable video coding. And in the private networks, some researchers have proposed software-based CDN [45-47] for HAS. An example is Peer2View platform [47], which uses consumer machines to cache date and not dedicated servers. The consumer machines self-organize them into a mesh-based overlay network and use proxy to redirect the http get request to the promoted nodes which have the date. Promoted nodes were selected using absolute ranking based on metrics such as computational load, bandwidth and connectivity and its tasks were pre-fetching content ahead of all other peers. This peer-to-peer distributed caching approach has greatly reduced the traffic streamed from the source while providing the same quality of user experience of a CDN.

5. Conclusions

In the latest ten years, Internet video has come of age. And now video may become the dominant type of traffic over the Internet, dwarfing other types of traffic. HAS solutions represent the most promising technical approaches for Internet video due to its simple, reusing the deployment of CDN infrastructure and its dynamic rate selection algorithms to adapt video quality based on the available network bandwidth and clients' capacity.

In this paper, we reviewed the state-of-the-art of HAS. On one hand, HAS solutions have shown great promise in media delivery, as witnessed by their increasingly widespread deployments. On the other hand, there are a number of key technical challenges that need to be overcome before HAS solutions can approach the service quality of users expect, such as high bitrate, smooth playback of high definition video,3D video. What' more, in the near term, most of the challenges have to do with the limited amount of access capacity in the Internet. Nowadays the majority researches consider the underlying network as a black box and only use the application layer information to select bitrates. However the network itself would be the first to know about a congestion or failure, helping servers and clients adapt faster and more accurately. So how to combile the underlying network information with the application layer to improve the user's quality of experiments is an interesting future research direction. Since there are many HAS solutions proposed, it is essential to develop instrumentation tools with friendly user interface to assess their effectiveness, performance and provide adequate information for diagnostics and fault isolation.

Acknowledgements

This work is supported by China Postdoctoral Science Foundation (2013M531711), the National Natural Science Foundation of China (No. 61202470), Technical Supporting Programs Funded by Ministry of Science & Technology of China (2013BAH18F02), Wuhan Science and Technology Project (No.2013010501010148).

References

[1] Cisco, Cisco Visual Networking Index: Forecast and Methodology, 2012–2017, http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html.
[2] V. K. Adhikari, Y. Guo, F. Hao, M. Varvello, V. Hilt,M. Steiner and Z.-L. Zhang, "Unreeling Netflix: Understanding and Improving Multi-CDN Movie Delivery", Proceedings of the IEEE INFOCOM 2012, **(2012)** March 25-30, Orlando, FL, USA, pp. 1620–1628.
[3] H. Zhang, V. Sekar and I. Stoica, "Internet Video: Past, Present, and Future, sigcomm 2013, Internet Video Tutorial", http://conferences.sigcomm.org/sigcomm/2013/ttliv.php,2013.
[4] Y. Hao, Z. Tongyu and L. Chuang, "J. Chinese Journal of Computers", vol. 35, no. 6, **(2012)**, pp. 1120-1130.
[5] History of Move Networks. Available online:http://www.movenetworks.com/history.html.
[6] S. Akhshabi, S. Narayanaswamy, A. C. Begen and C. Dovrolis, "An experimental evaluation of rate-adaptive video players over HTTP", J. Image Commun., vol. 27, no. 4, **(2012)**, pp. 271-287.
[7] B. Wang, J. Kurose, P. Shenoy and D. Towsley, "Multimedia streaming via TCP: An analytic performance study", J. ACM Trans. Multimedia Comput. Commun. Appl., vol. 4, no. 2, **(2008)**.
[8] S. Akhshabi, A. C. Begen and C. Dovrolis, "An experimental evaluation of rate-adaptation algorithms in adaptive streaming over HTTP", Proceedings of the second annual ACM conference on Multimedia systems (MMSys '11), **(2011)** February 23-25, San Jose, California, pp. 157-168.

[9] T.-Y. Huang, N. Handigol, B. Heller, N. McKeown and R. Johari, “Confused, timid, and unstable: picking a video streaming rate is hard”, Proceedings of the 2012 ACM conference on Internet measurement conference (IMC '12), **(2012)** November 14-16, Boston Massachusetts USA, pp. 225-238.

[10] S. Akhshabi, L. Anantakrishnan, A. C. Begen and C. Dovrolis, “What happens when HTTP adaptive streaming players compete for bandwidth?”, Proceedings of the 22nd international workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV '12),**(2012)** June 7-8, Toronto, Canada, pp. 9-14.

[11] T.-Y. Huang, R. Johari and N. McKeown, “Downton abbey without the hiccups: buffer-based rate adaptation for HTTP video streaming”, Proceedings of the 2013 ACM SIGCOMM workshop on Future human-centric multimedia networking (FhMN '13), **(2013)** August 12-16, Hongkong, China, pp. 9-14.

[12] H. Schulzrinne, A. Rao and R. Lanphier”, “Real Time Streaming Protocol (RTSP)”, RFC 2326, Standard track, (**1998**), April.

[13] Adobe Systems Inc. Real-Time Messaging Protocol (RTMP) Speciation, (**2009**).

[14] A. Zambelli, “IIS smooth streaming technical overview”, Microsoft Corporation, 2009.http://download.microsoft.com/download/4/2/4/4247C3AA-7105-4764-A8F9-321CB6C765EB/IIS_Smooth_Streaming_Technical_Overview.pdf.

[15] A. Begen, T. Akgul and M. Baugher, “Watching Video over the Web: Part 1: Streaming Protocols”, J. IEEE Internet Computing, vol. 15, no. 2, **(2011)**, pp. 54-63.

[16] A. C. Begen, Watching Video over the Web: Adaptive Streaming over HTTP, http://ali.begen.net,2011.

[17] L. D. Cicco, S. Mascolo and V. Palmisano, “Feedback control for adaptive live video streaming”, In Proceedings of the second annual ACM conference on Multimedia systems (MMSys '11)), **(2011)** February 23-25, San Jose, California, pp. 145-156.

[18] R. Kuschnig, I. Koer and H. Hellwagner, “An evaluation of TCP-based rate-control algorithms for adaptive internet streaming of H. 264/SVC”, Proceedings of ACM SIGMM conference on Multimedia systems, **(2010)** August 30-September 03, New Delhi, India, pp. 157-168.

[19] J. Jiang, V. Sekar and H. Zhang, “Improving Fairness, Efficiency, and Stability in HTTP-Based Adaptive Video Streaming With Festive”, J.IEEE/ACM Transactions on Networking, vol. 22, no. 1, **(2014)**, pp. 326-340.

[20] ISO/IEC DIS 23009-1.2.Information Technology —Dynamic Adaptive Streaming over HTTP (DASH) — Part 1: Media Presentation Description and Segment Formats.2012.

[21] C. Liu, I. Bouazizi and M. Gabbouj, “Rate adaptation for adaptive HTTP streaming”, In Proceedings of the second annual ACM conference on Multimedia systems (MMSys '11)), **(2011)** February 23-25, San Jose, California, pp. 169-174.

[22] L. S. Lam, J. YB Lee, S. C. Liew and W. Wang, “A transparent rate adaptation algorithm for streaming video over the internet”, Proceedings of 18th International conference on advanced information networking and applications, **(2004)** March 29- 31, Fukuoka, Japan.

[23] G. Tian and Y. Liu, “Towards agile and smooth video adaptation in dynamic HTTP streaming”, Proceedings of the 8th international conference on Emerging networking experiments and technologies (CoNEXT '12), **(2012)** December 10-13, Nice, France, pp. 109-120.

[24] C. Liu, I. Bouazizi, M. M. Hannuksela and M. Gabbouj, “Rate adaptation for dynamic adaptive streaming over HTTP in content distribution network”, J. Image Commun., vol. 27, no. 4, **(2012)**, pp. 288-311.

[25] L. D. Cicco and S. Mascolo, “An experimental investigation of the Akamai adaptive video streaming”, Proceedings of the 6th international conference on HCI in work and learning, life and leisure: workgroup human-computer interaction and usability engineering (USAB'10), **(2010)** November 4-5, Klagenfurt, Austria, pp. 447-464.

[26] J. Jiang, V. Sekar and H. Zhang, “Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive”, Proceedings of the 8th international conference on Emerging networking experiments and technologies (CoNEXT '12), **(2012)** December 10-13, Nice, France, pp. 97-108.

[27] H. Liu, Y. Wang, Y. R. Yang, A. Tian, and H. Wang, “Optimizing Cost and Performance for Content Multihoming”, Proceedings of SIGCOMM 2012, **(2012)** August 13-17, Helsinki, Finland.

[28] X. Liu, F. Dobrian, H. Milner, J. Jiang, V. Sekar, I. Stoica and H. Zhang, “A Case for a Coordinated Internet Video Control Plane”, Proceedings of. SIGCOMM 2012, **(2012)** August 13-17, Helsinki, Finland.

[29] S. Akhshabi, L. Anantakrishnan, C. Dovrolis and A. C. Begen, “Server-based traffic shaping for stabilizing oscillating adaptive streaming players”, Proceeding of the 23rd ACM Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV '13), **(2013)** February 27, Oslo, Norway, pp. 19-24.

[30] R. Houdaille and S. Gouache, “Shaping HTTP adaptive streams for a better user experience”, In Proceedings of the 3rd Multimedia Systems Conference (MMSys '12), **(2012)** February 22-24, Chapel Hill, North Carolina, pp. 1-9.

[31] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. C. Begen and D. Oran, “Probe and Adapt: Rate Adaptation for HTTP Video Streaming At Scale”, J. IEEE Journal on Selected Areas in Communications, **(2014)**, In Print.

[32] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell and M. Watson, “Using the Buffer to Avoid Rebuffers: Evidence from a Large Video Streaming Service”, Networking and Internet Architecture (cs.NI), arXiv:1401.2209v1, **(2014),** http://arxiv.org/abs/1401.2209

[33] V. K. Adhikari, Y. Guo, F. Hao, M. Varvello, V. Hilt, M. Steiner, and Z.-L. Zhang, “Unreeling Netflix: Understanding and Improving Multi-CDN Movie Delivery”, Proceedings of the IEEE INFOCOM 2012, **(2012)** March 25-30, Orlando, FL, USA, pp. 1620–1628.

[34] Apple Inc. HTTP Live Streaming Overview, https://developer.apple.com/library/ios/documentation/networkinginternet/conceptual/streamingmediaguide/StreamingMediaGuide.pdf,2014.

[35] Adobe Inc. HTTP Dynamic Streaming-Part 1: An Introduction to Streaming Media,http://www.realeyes.com/blog/2011/07/27/http-dynamic-streaming-part-1-an-introduction/,2011.

[36] Bufferbloat, http://en.wikipedia.org/wiki/Bufferbloat,2014.

[37] J. Gettys and K. Nichols, “Bufferbloat: dark buffers in the internet”, J. ACM Commun., vol. 55, no. 1, **(2012)**, pp. 57–65.

[38] A. Mansy, B. Ver Steeg and M. Ammar, “Sabre: A client based technique for mitigating the buffer bloat effect of adaptive video flows”, Technical report, Georgia Tech, (**2012**).

[39] R. Huysegems, B. De Vleeschauwer, T. Wu and W. Van Leekwijck, “SVC-based HTTP adaptive streaming”, J. Bell Labs Technical Journal, vol. 16, no. 4, **(2012)**, pp. 25–41.

[40] Y. S´anchez de la Fuente, T. Schierl, C. Hellge, T. Wiegand, D. Hong, D. De Vleeschauwer, W. Van Leekwijck and Y. Le Lou´edec, “iDASH: improved dynamic adaptive streaming over HTTP using scalable video coding”, Proceedings of the second annual ACM conference on Multimedia systems (MMSYS '11), **(2011)** February 23-25, Santa Clara, CA, USA, pp. 257–264.

[41] N. Bouten, S. Latre, J. Famaey, F. De Turck and W. Van Leekwijck, “Minimizing the impact of delay on live SVC-based HTTP adaptive streaming services”, Proceedings of 2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013), **(2013)** May 27-31, Ottawa, Canada, pp. 1399-1404.

[42] J. Famaey, S. Latre, N. Bouten, W. Van de Meerssche, B. De Vleeschauwer, W. Van Leekwijck and F. De Turck, “On the merits of SVC-based HTTP Adaptive Streaming”, Proceedings of 2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013), **(2013)** May 27-31, Ottawa, Canada, pp. 419-426.

[43] T. Andelin, V. Chetty, D. Harbaugh, S. Warnick and D. Zappala, “Quality selection for Dynamic Adaptive Streaming over HTTP with Scalable Video Coding”, In Proceedings of the 3rd Multimedia Systems Conference (MMSys '12) , **(2012)** February 22-24, Chapel Hill, North Carolina, pp. 149-154.

[44] Y. Sanchez, T. Schierl, C. Hellge, T. Wiegand, D. Hong, D. D. Vleeschauwer, W. V. Leekwijck and Y. L. Lou´edec, “Efficient http-based streaming using scalable video coding”, J. Signal Processing: Image Communication, vol. 27, no. 4, **(2012)**, pp. 329-342.

[45] R. Roverso, S. El-Ansary and M. Högqvist, “On HTTP live streaming in large enterprises”, Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM (SIGCOMM '13), **(2013)** August 12-16, Hongkong, China, pp. 489-490.

[46] R. Roverso, S. El-Ansary and S. Haridi, “SmoothCache: HTTP-Live streaming goes peer-to-peer”, Proceedings of the 11th international IFIP TC 6 conference on Networking - Volume Part II (IFIP'12), (**2012)**, pp. 29-43.

[47] R. Roverso, S. El-Ansary and S. Haridi, “Peer2View: A peer-to-peer HTTP-live streaming platform”, Proceedings of P2P, **(2012)** September 3-5, Tarragona, Spain, pp. 65-66.

[48] L. Jiangchuan, S. G. Rao, L. Bo and H. Zhang, “Opportunities and Challenges of Peer-to-Peer Internet Video Broadcast”, J. Proceedings of the IEEE, vol. 96, no. 1, **(2008)**, pp. 11-24.