J Inf Process Syst, Vol.18, No.6, pp.729~740, December 2022
https://doi.org/10.3745/JIPS.04.0260

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Netflix, Amazon Prime, and YouTube: Comparative Study of Streaming Infrastructure and Strategy

Suman Pandey[1], Yang-Sae Moon[2,3], and Mi-Jung Choi[2,3,*]

Abstract

Netflix, Amazon Prime, and YouTube are the most popular and fastest-growing streaming services globally. It is a matter of great interest for the streaming service providers to preview their service infrastructure and streaming strategy in order to provide new streaming services. Hence, the first part of the paper presents a detailed survey of the Content Distribution Network (CDN) and cloud infrastructure of these service providers. To understand the streaming strategy of these service providers, the second part of the paper deduces a common quality-of-service (QoS) model based on rebuffering time, bitrate, progressive download ratio, and standard deviation of the On-Off cycle. This model is then used to analyze and compare the streaming behaviors of these services. This study concluded that the streaming behaviors of all these services are similar as they all use Dynamic Adaptive Streaming over HTTP (DASH) on top of TCP. However, the amount of data that they download in the buffering state and steady-state vary, resulting in different progressive download ratios, rebuffering levels, and bitrates. The characteristics of their On-Off cycle are also different resulting in different QoS. Hence a thorough adaptive bit rate (ABR) analysis is presented in this paper. The streaming behaviors of these services are tested on different access network bandwidths, ranging from 75 kbps to 30 Mbps. The survey results indicate that Netflix QoS and streaming behavior are significantly consistent followed by Amazon Prime and YouTube. Our approach can be used to compare and contrast the streaming services' strategies and fine-tune their ABR and flow control mechanisms.

Keywords

Amazon, Netflix, QoE, QoS, Streaming, YouTube

1. Introduction

Streaming media is flocking the IP network with video, audio, game, live TV, and educational streaming services. With the present pandemic brought about by coronavirus disease 2019 (COVID-19), social distancing could be a norm, and streaming media consumption could further increase. An easy subscription-based model and high-access network bandwidth combined with cloud and CDN technologies have made these services extremely successful. Apart from these, the adaptive bit rate (ABR), supporting MPEG-DASH (Dynamic Adaptive Streaming over HTTP) and advancing encoding technologies such as MPEG-4, has contributed to providing streaming services at reasonable costs. The streaming industry is highly competitive, a quality-of-service (QoS) measurement and benchmarking from a client's point of

Manuscript received March 19, 2021; first revision September 15, 2022; accepted October 6, 2022.
* **Corresponding Author:** Mi-Jung Choi (mjchoi@kangwon.ac.kr)
[1] School of Electrical Engineering and Computer Science, Gwangju Institute of Science and Technology, Gwangju, Korea (suman17@gist.ac.kr)
[2] Dept. of Computer Science, Kangwon National University, Chuncheon, Korea (mjchoi@kangwon.ac.kr, ysmoon@kangwon.ac.kr)
[3] IGP. in Medical Bigdata Convergence Kangwon National University, Chuncheon, Korea (mjchoi@kangwon.ac.kr, ysmoon@kangwon.ac.kr)

view can give an edge to the service provider. A model to compare the QoS of streaming services is deduced in this study. The model is further used to compare the most prominent streaming service providers. Their infrastructure and delivery model are also surveyed extensively in this research.

This paper does a comparative investigation of QoS for existing streaming services, including Netflix, Amazon Prime, and YouTube. To deliver a QoS measurement from the client machine, the system requires an accurate traffic classification and identification of the stream they are interested in, and then a calculation of QoS based on the different parameters, including throughput, buffering, startup delay, and TCP errors. For accurate classification of the traffic, we require to study the infrastructure and origins of the stream first. Hence, the first part of the paper surveys the general infrastructures, protocols and static page's features of these services. Several standards and protocols involved in end-to-end stream delivery were first analyzed, then the cloud and CDN infrastructure of these three services were compared to ascertain their core and edge caching mechanisms. An infrastructure study helps one identify the correct stream and map the IP addresses of these service providers. We briefly explain the Open Connect, CloudFront, and Google Peering infrastructures behind these services, apart from protocols and service models.

The second part of the paper focuses on overall streaming behavior and QoS. There are two different approaches to quality assessments. First is a subjective approach also called quality-of-experience (QoE). QoE is measured using mean opinion score (MOS) [1] in the range of 1 to 5. This is a subjective measure and requires user perception-related matrix. The second approach is the objective approach also called quality-of-service, QoS [2,3], measured using the application, network, and transport parametric. Several standards have been developed to translate a QoS model to MOS-based QoE models using mathematical equations [4,5]. Even if the standards are developed and implemented by several providers, it is still a challenging task to deduce the QoS of service and requires a lot of measurement data.

Measuring QoS depending totally on application layer metrics related to video players is impractical, as it will require different techniques for different service providers. Application layer metrics could be startup delay, rebuffering events, playback buffer, etc. Media presentation and description (MDP) files [6] or player API such as YouTube iframe API [7] can be used to collect these data, however, MDP files are mostly encrypted and most of service providers do not provide player APIs. Hence designing a generic QoS model based on application layer metrics becomes a challenging task.

TCP throughput, video packet delay, jitter, etc., can be used for network parametric models, however, in ABR technology these parameters are ineffective as the video encoded at different resolutions and bit rates are stored in the server and the bitrate changes with the network condition resulting in almost no jitter and packet loss. Also, the playout buffer at the application layer manages the delay to a significant level. As the streaming technology changes and adopts the ABR and rate-controlled streaming, our parametric models need to evolve. Also, the measurement technique to collect QoS metrics should be generic to all the applications. Hence in this paper, we depend totally on streaming patterns to extract QoS metrics. The goal of this paper is to provide feature extraction techniques based on patterns and characteristics of streaming. We first compare the QoS for static pages followed by the QoS of streaming at the granular level.

This paper's major QoS matrices are rebuffering time, average media bit rate, progressive download ratios, and standard deviation of the On-Off cycle. These streaming services usually download the content at extremely high speed initially, known as the "buffering state," which was slowly known later as the "steady state" [8]. The progressive download ratio is the ratio of these two phases. In the steady state phase, the videos are downloaded in the On-Off cycle. On cycle represents the chunk size also known as

the block size. Off-cycle represents the waiting time, during which data is not downloaded keeping the bandwidth free. The standard deviation of the On-Off cycle shows if the download bit rate during the session is consistent or jittery. The QoS in the bandwidth range of 75 kbps to 30 Mbps for 50 different videos of 180-second duration were compared. These tests contributed to the thorough analysis of the HTTP-DASH streaming and ABR behavior of these services. Such detailed comparative studies help a new service provider in requirement analysis. This study clarifies the infrastructure requirements of the streaming services and streaming strategies of different streaming service providers. The QoS measurement model adopted by this paper is carefully chosen from ITU-T standard [1,4] and Streaming Media Alliance [9]. Our work can act as a base case for classifying and labeling streaming test cases for developing a QoS model based on supervised learning in the future. The proposed approach can help service provider compare their services with their competitors without application-level details and player APIs. Service providers can also do feature analysis and fine-tune these features for better ABR strategy and flow control mechanism.

Rest of the paper is organized in the following manner. Section 2 elaborates on some of the related work and their shortcomings. Section 3 explains the CDN and cloud infrastructure of these services. Section 4 elaborates on the method used to measure the QoS. Section 5 discusses the results of our measurement studies and Section 6 concludes our work.

2. Related Work

QoE focuses on the entire service experience; however, QoS is a network performance measure that is closely related to network deployments, such as caching, traffic engineering, and resource reservation. QoE and QoS are closely related to each other and there is a great number of standards from the Internet Engineering Task Force (IETF) that define them [5]. This paper's measurement studies are limited to QoS calculations. A great deal of QoE models have been already surveyed in past [10]. QoE modeling is categorized into four types, including signal-based model, parametric model, bitstream model, and hybrid model. This study's QoE model is a parametric type. A subset of QoE features forming the parametric model was also recommended by the Streaming Media Alliance [9], whose QoE parameter was highly simplistic and was utilized in this paper. Gathering QoE parameters for parametric model is another challenge. Several researchers depend on MPD file to gather these matrixes as per the previous survey [11]. When a first request for the video file is made by the client, the server sends the corresponding manifest file, also called MPD which consists of the details about the video file, such as the video duration, segment size, available representation levels, and codec. The client's rate adaptation logic depends on the QoE calculation from these MPD files. The rate of adoption broadly depends on available throughput or buffer size. However, these MPD files are encrypted, and a third party cannot manipulate them. There is a need for a model which could compare different streaming services from the premises without needing any specific information from the service providers. In this research, we solely depend on network and transport layer information gathered from packet streams to collect the QoS parameters. Previous researchers used similar approaches comparing Google Cloud CDN, Azure CDN, and Amazon CloudFront using PlanetLab at different locations, concluding that Amazon CloudFront had a slightly better QoE than Google Cloud and Microsoft Azure [12]. The drawback with their work is that they

deployed their own apache servers in the computing instances in various clouds to host their own website for dash. The videos are generated with a fixed bit-rate and their Python-based client collects data from different PlantLab locations. Our paper's measurement studies are not dependent on any such deployments. It rather compares these three services based solely on network and transport layer information gathered from packet streams directly served from their servers. There is a list of researches following a similar approach as ours [13,14]; however, these studies are limited to YouTube traffic in particular. None of these researches have compared Amazon Prime with Netflix and YouTube. Furthermore, their test beds have not covered an Asian region, such as South Korea.

3. Comparing Cloud and CDN Infrastructure

Before measuring the QoS we analyzed the cloud and CDN infrastructure of these services. This study helps us identify the list of IP addresses and domains associated with each service. It also helps us in classifying actual content streams from third-party services such as advertising, analytics, social networking plugins, or AI stream from the same service provider.

Some service providers have their own CDN and cloud infrastructures; however, others depend totally on third party CDNs. For example, YouTube depends on Google-peering CDN and Google Cloud services. Netflix uses Amazon Web Services (AWS) for compute services and OpenConconect CDN for content cache. Amazon prime uses AWS for compute services and CloudFront and Akamai for content cache.

3.1 Open Connect, CloudFront and Google Peering

While analyzing HTTP requests and responses, it was found that Netflix ties each user to a preferred CDN. Netflix uses the Amazon AWS platform for compute caches, such as XML, Javascript, and HXR files; and Netflix uses OpenConnect for video and audio streaming contents. It deploys its own caching servers on the IXPs and ISPs through its OpenConnect services. Open Connect Appliances (OCA) are the building blocks of OpenConnect. OCAs are connected to each other through settlement-free public or private peering (SFI). ISPs provide the power, space, and connectivity to these OCAs. OCAs work together with the AWS control plane and client devices. OCAs interact with AWS for the control logic and with client devices to serve the contents optimally based on network conditions. Netflix uses Peer filling and Tier filling. If contents are available to a peer, they are taken from the peer through Peer filing. If the contents are not available at a peer, the contents are located at different sites and clusters, and then the contents are filled through a BGP connection between two clusters. OCAs are deployed with a FreeBSD OS and NGINX Web server, while a BIRD BGP daemon is used. Netflix applies Edge Caching combined with a clustering approach for heavy tail contents.

CloudFront is Amazon's Content Distribution Network (CDN). Amazon S3 bucket and EC2 instances are configured as the origin server for static and dynamic contents, while computing contents and CloudFront are configured as a CDN for edge locations [5]. First requests for the content are served by S3 services and then the requests are cached at CloudFront edge. It takes lesser hops to reach the edge location using the CloudFront edge network for further requests. For Amazon Prime video HTTP request-

response, videos are noticeably served chiefly by the CloudFront; however, in the case of CloudFront servers' unavailability, they are sometimes served from Akamai CDN, llnw, level 3 communication CDNs in South Korean premises. Akamai edge network is stronger than CloudFront and it is a fallback strategy for Amazon. Amazon CloudFront currently has edge location with 188 Points of Presence (PoP) edge locations in 70 cities across 31 countries as per July 2017. However, Akamai has 1,700 edge locations across 130 countries.

Similar to Netflix, Google also deploys its caching infrastructure through peering technologies. If an ISP expresses an interest in hosting Google's edge cache, then Google usually ships its servers to be attached in the ISP's facilities. After the deployment of these servers, they become part of the Google Edge Node (GGC) infrastructure and part of the AS151169. Apart from GGC, Google has a core data center at the core of the network and an edge PoP network as a middle layer. Google's edge software-defined networking (SDN) architecture is called "Expresso," and the core SDN architecture is called "Jupiter." Usually, Google does not allow peering with its core network infrastructure. GGC infrastructure serves 60% to 80% of the total traffic, saving the backbone and transit bandwidth. Google also has the Google Cloud Interconnect (GCI) service, which is basically a Google Cloud platform. In South Korean premises, Google rents a cache in KT network and also uses Google Cache in Japan as a backup.

3.2 Comparative Analysis of Services

Most streaming service providers use third-party services for advertising, analytics, social networking plugins, or AI on their web pages. All these requests are served from different domains and sometimes different clouds and CDNs to provide a satisfactory overall experience.

From a caching point of view, these different domains and their services were broadly classified into two different categories: compute caching and content caching. The caching requirement for compute services could be different from content services. There is a need for separate strategies for compute cache and content cache for several reasons. These requirements would influence the deployment of these caches. Content cache, such as actual video and audio, advertisement, and images should be deployed nearer to the user as they require high bandwidth. Compute cache, such as CSS, Javascript, HTML, HXR, fonts, and server logic could be lightweight but could require a high amount of computing resources such as CPU and RAM. It would also require a frequent update of the compute logic, and the same logic will be used by the users by the same user again and again, unlike content that does not get updated and is usually watched just once.

One can also see that all these three services use HTTP DASH in the application and TCP for the transport layer, except for YouTube which uses Google QUIC on top of UDP on Chrome and Firefox. Netflix usually binds the users to preferred CDNs. However, Amazon and YouTube bind users with multiple CDNs. If the main Amazon Prime domain does not respond, then the requests are served by Akamai and llnw and then level 3 communication domains. Similarly, for YouTube, if the requests cannot be served from Google Cache at KT, then they are served by Google US domains. Chrome DevOps network blocking features were utilized as they can block a particular domain, server, or list of servers. This feature is useful for measuring if the users are attached to a dedicated server or CDN, or in the case of failure if they can be switched to other servers and CDNs. For logging in pages, Amazon connects to fewer domains, and hence its DNS lookup and SSL.

4. Experimental Results

After analyzing the CDN and cloud infrastructure of these services we collected 50 video streams each 180 seconds on bandwidth range of 75 kbps to 30 Mbps for all these three services using the Chrome browser.

Fig. 1 provides an overview of this study's QoS inference approach.

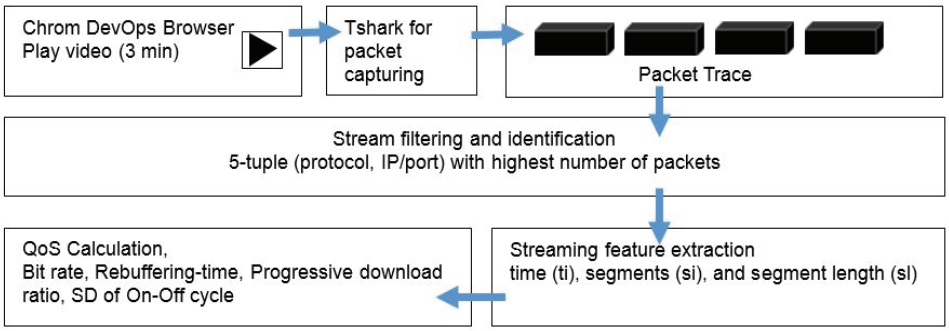

Fig. 1. QoE measurement and calculation.

(1) Chrome browser DevOps tools have bandwidth configuration options, which were used to configure client bandwidth and then start capturing video stream using Tshark tool for a total of 180 seconds.

(2) While streaming, there are several types of HTTP response sent from different non-origin servers to the browser. These requests include audio, video, advertisement, streaming logic, and other types serving other AI requests. All these responses give a holistic video watching experience. The first step is to filter the video stream among several other streams from various non-origin servers. It was identified by capturing the successive IP packets having 5-tuple flow information, including protocol (TCP/QUIC), source address (streaming server IP address), source port (433), destination address (host address), and destination port (port on which the browser is running; for chrome browser, mostly there are multiple ports for streaming) with the highest amount of data. To classify the right stream, IP address was matched with the list of domains of the service provider, and a source port number to 443. Automatic "HTTP stream classification" can be another topic of research, but currently that is not this paper's focus.

(3) After the stream is identified, only three features including time (ti), segments (si), and segment length (sl) were extracted from the packet trace.

(4) These three features are used for the calculations of this paper's main QoS matrices, which are rebuffering time, bitrate, progressive download ratio and standard deviation of the On-Off cycle. Here, the parameters are elaborated in detail.

Playtime (t): This is the actual length of the video. This duration does not include rebuffering time. All the test cases have a total playtime of 180 seconds.

Actual play time (tn): This is the duration of a video that is being played with jitter and rebuffering. For a 180-second video, an application can take more than 180 seconds to play completely, especially if network conditions are not good.

Rebuffering time: This is the time in which a viewer experiences rebuffering issues (i.e., when a video stop playing because of buffer underflow and not due to user interventions such as scrubbing or pausing).

Rebuffering ratio: This is a calculation of total rebuffering time divided by the sum of total playtime.

Bitrate: This metric shows the average bit rate at which content was received. It is calculated as the number of bits received and decoded during a play, divided by the total playing time.

Buffering phase: Buffering phase is when the data transfer rate is limited by the end-to-end available bandwidth. In Fig. 2(a), one can see that Netflix downloads more than 7,000 segments (sb) of size 1,541 bytes (sl) each in just the first 5 (tb) seconds. This 5 second is a buffering phase.

Steady state phase: In the steady state phase, the average download rate is slightly larger than the video encoding rate. In Fig. 2(a), one can see that Netflix downloads 6,061 (ss) segments of size 1,541 (sl) bytes in 166 seconds (ts).

Progressive download ratio: It is the ratio of the average bitrate and the bitrate at a steady state. If enough segments have been buffered then, this ratio will result in lower values. A lower progressive download ratio means that the bit rate at buffering state is higher, resulting in enough downloaded segments for playback. A progressive download ratio of 1 indicates there is no buffering done at the buffering state.

SD of ON cycle: It is a standard deviation of the On-Off cycle of a stream in its steady state phase. During the On cycle, a chunk of data is downloaded. It's also called a block size in streaming. Low SD means that the stream is downloaded with a constant bit rate. If the SD is high the bit rate has been varying during the download, and it represents the jittery stream. Higher SD of the On-Off cycle represents a very jittery stream.

5. Measurement Results and Discussion

The streaming behaviors of 50 different videos on all three service providers were compared. We carefully choose some common videos for the tests across these platforms. The basic ABR characteristics of all these services are the same. This means, there are two phases of the video download. The video stream begins with the buffering phase followed by a steady state phase. There is a cycle of the On-Off period in the steady state phase, which is used to limit the download rate. Here we will compare these three services to the four main QoS features of our model.

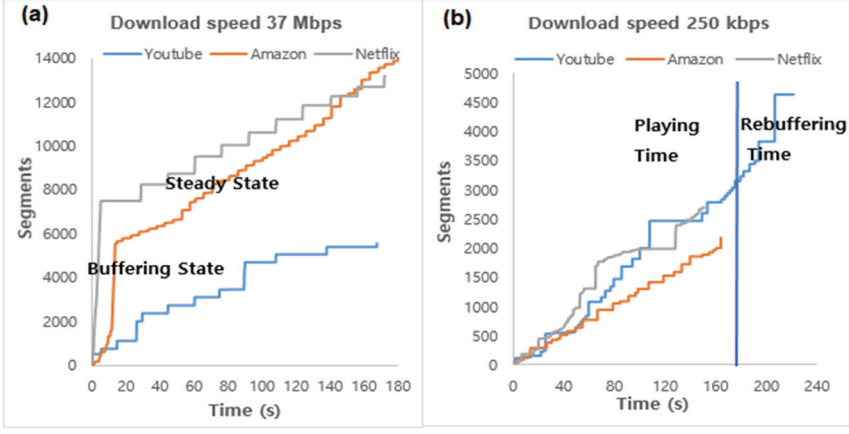

Fig. 2. Number of segments downloaded per time stamp for Peppa Pig, Episode 1: (a) download speed of 37 Mbps and (b) download speed of 250 kbps.

Bitrate: The maximum segments are downloaded by Amazon (16,000), followed by Netflix (14,000) and YouTube (6,000), for the same content. The best bit rate result is shown by Amazon and then Netflix followed by YouTube (Fig. 2(a)). For lower bandwidth, the bitrates were adjusted well for Netflix and Amazon based on the ABR protocol HTTP-DASH (Fig. 2(b)). For this particular video, YouTube does not have the video encoded at lower bit rates, hence one can see that bitrate is still high for YouTube (Fig. 2(b)). Fig. 3 shows that bitrate is directly proportional to the available bandwidth. Even if the available bandwidth is high, some videos are not encoded with high resolutions and hence they still acquire a less amount of bandwidth and eventually a lower bitrate. One can see this behavior in Fig. 4, which shows the results of different videos of different qualities and resolutions for each service provider. Although the available bandwidth is high (37 Mbps), some videos show a lower bitrate. This is because those videos are encoded at low bit rates and are not available in high resolutions. Hence merely a bit rate cannot quantify the quality of the stream. We further compare the "rebuffering time" and "progressive download ratio" with the SD of the On-Off cycle to obtain the quality of the video.

Rebuffering time: We tested 50 different videos with different bandwidth configurations at the client on all these platforms to calculate rebuffering time. The rebuffering time of one of the videos on different bandwidths is shown in Fig. 3. Rebuffering time is represented in the graph with the blue line. The time beyond the blue line represents the rebuffering time. For higher bandwidth, rebuffering time for all these services is null, showcasing a good QoS; however, for lower bandwidth, the rebuffering time for YouTube video is higher compared to Netflix and Amazon as shown in Fig. 3. For Netflix, one could not measure the bandwidth below 250 kbps, as Netflix does not provide services for such low bandwidth. For Amazon and YouTube, one could, however, test bandwidth as low as 75 kbps. One can see that the rebuffering of Netflix is the lowest among these service providers. In some incidents, they even download their video before time, even for lower bandwidth. This showcases the good overall quality of Netflix streams.

Progressive download ratio: The ratio of steady-state bit rate and overall bit rate gives the progressive download ratio. The lesser the progressive download ratio is, the better the QoE of the video will be. We tested 50 different videos with the highest bandwidth capacity of 30 Mbps to test the progressive download ratio, as shown in Fig. 4. For clarity figure only shows the results of 20 videos. The slope of the buffering phase as well as the steady-state depends on the available bandwidth. For lower bandwidth, all these services do not have buffering phase hence progressive download ratio cannot be calculated. One can observe that the Netflix buffering phase downloads a higher number of segments in less time compared to Amazon and YouTube. Netflix has the lowest average progressive download ratio of 0.49. This shows that their replay buffer accumulates enough data for playing resulting in the best QoE.

SD of On-Off cycle: The other interesting characteristic which separates these three services is their On-Off cycle. We observe that Netflix and YouTube have a bigger On cycle, which means a bigger chunk of data is downloaded at a time, as compared to Amazon. Netflix and YouTube also have a lesser number of blocks of the On-Off cycle as compared to Amazon. A bigger On cycle will be helpful for filling the replay buffer but require higher bandwidth. Amazon compensates for this by having a greater number of On-Off cycles as compared to Netflix and YouTube. So even though their cycle size is small they download an equal amount of data for the same time period by having a greater number of cycles. We also observe that in general if the standard deviation of the On-Off cycle is high the quality of the video is bad. In our overall experience, the SD of the On-Off cycle of Netflix and Amazon is much lower than

YouTube. Hence Netflix and Amazon show consistency in bit rate as compared to YouTube. This results in lesser jitter and better quality of the video.

Netflix performs the best among these three service providers by displaying a high bitrate for lower bandwidths, less rebuffering (Fig. 3), better progressive download ratio (Fig. 4), and lower SD of the On-Off cycle. Even though one carefully chooses the common contents available from all these service providers, the encoding rates of the original contents could be different, and one cannot know the exact encoding-related matrix from these service providers; hence, one just relies on the bitrate received at the end user client. Also, the results could vary for different geographical locations, depending on the edge infrastructure. This study is a demonstration of how one can compare different streaming services from a client's machine and choose the services best suitable to him/her.

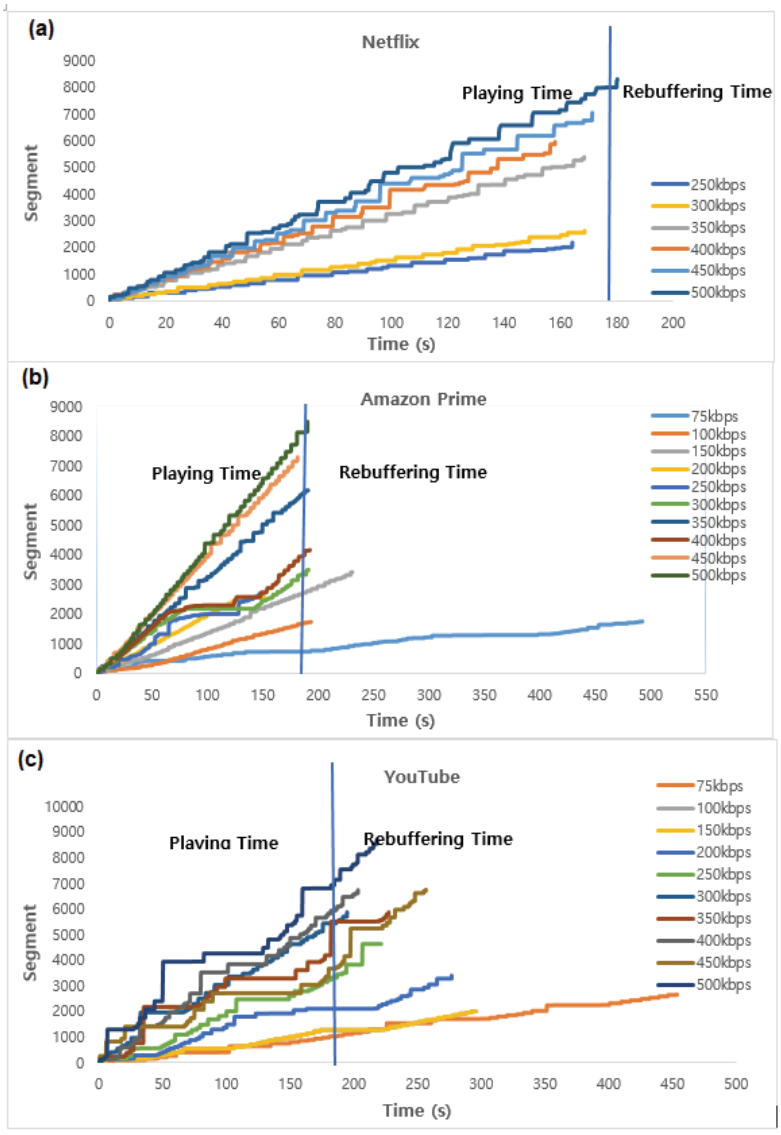

Fig. 3. Rebuffering: (a) Netflix, (b) Amazon Prime, and (c) YouTube.

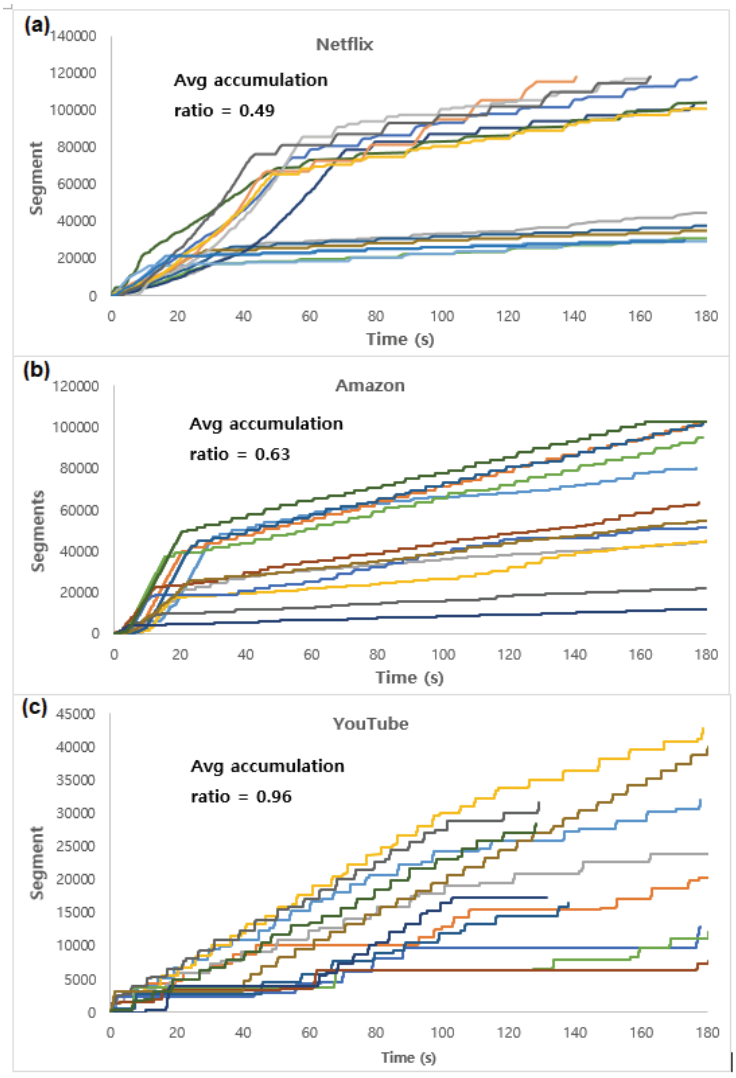

Fig. 4. Progressive download ratio: (a) Netflix, (b) Amazon Prime, and (c) YouTube.

6. Conclusion and Future Work

In this paper, the most popular global streaming service providers, including Netflix, Amazon Prime, and YouTube were surveyed. Their infrastructure and protocols were studied, their services were compared, and methodologies for doing QoS analysis were derived. QoS analysis was conducted for these popular service providers by means of calculating the bitrates, rebuffering time, progressive download ratio, and SD of the On-Off cycle on different bandwidths for 50 videos each. Based on these calculations, it was concluded that Netflix had the best performance; however, the QoS of Amazon Prime was also reasonably good.

This paper demonstrates an approach to do a quality analysis based on the flow of information. The approach can also be used to compare and contrast the importance of different features of these services.

Our analysis can be used by the service providers to analyze their competitors and fine-tune their streaming strategies. In the future, a machine learning approach to identify good and bad quality video services should be devised.

Acknowledgement

This research is a basic research project carried out with the support of the National Research Foundation of Korea with funding from the government (Ministry of Science and ICT) in 2020 (No. NRF-2020R1A2C1012117).

References

[1] International Telecommunication Union, "ITU-T Recommendation P.800.1: Mean opinion score (MOS) terminology," 2016 [Online]. Available: https://www.itu.int/rec/T-REC-P.800.1-201602-S/en.

[2] International Telecommunication Union, "Quality of service regulation manual," 2017 [Online]. Available: https://www.itu.int/pub/D-PREF-BB.QOS_REG01-2017.

[3] N. Seitz, "ITU-T QoS standards for IP-based networks," *IEEE Communications Magazine*, vol. 41, no. 6, pp. 82-89, 2003.

[4] International Telecommunication Union, "ITU-T Recommendation P.1203.1: Models and tools for quality assessment of streaming media," 2019 [Online]. Available: https://www.itu.int/ITU-T/recommendations/rec.aspx?rec=13845&lang=en.

[5] N. Barman and M. G. Martini, "QoE modeling for HTTP adaptive video streaming: a survey and open challenges," *IEEE Access*, vol. 7, pp. 30831-30859, 2019.

[6] ISO/IEC 23009-1:2014, *Information technology - Dynamic adaptive streaming over HTTP (DASH) - Part 1: Media presentation description and segment formats* [Online]. Available: https://www.iso.org/standard/65274.html.

[7] Google, "Youtube Player API," 2022 [Online]. Available: https://developers.google.com/youtube/iframe_api_reference.

[8] D. Pal and T. Triyason, "A survey of standardized approaches towards the quality of experience evaluation for video services: an ITU perspective," *International Journal of Digital Multimedia Broadcasting*, vol. 2018, article no. 1391724, 2018.

[9] Streaming Media Alliance, "Key Network Delivery Metrics," 2016 [Online]. Available: https://www.svta.org/product/key-network-delivery-metrics/.

[10] C. Wang, A. Jayaseelan, and H. Kim, "Comparing cloud content delivery networks for adaptive video streaming," in *Proceedings of 2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, San Francisco, CA, 2018, pp. 686-693.

[11] M. H. Mazhar and Z. Shafiq, "Real-time video quality of experience monitoring for HTTPS and QUIC," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, Honolulu, HI, 2018, pp. 1331-1339.

[12] M. J. Khokhar, T. Ehlinger, and C. Barakat, "From network traffic measurements to QoE for Internet video," in *Proceedings of 2019 IFIP Networking Conference (IFIP Networking)*, Warsaw, Poland, 2019, pp. 1-9.

[13] J. Kua, G. Armitage, and P. Branch, "A survey of rate adaptation techniques for dynamic adaptive streaming over HTTP," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1842-1866, 2017.

[14] Netflix, "Netflix Open Connect Appliance Deployment Guide," 2015 [Online]. Available: http://oc.nflxvideo.net/docs/OpenConnect-Deployment-Guide.pdf.

Suman Pandey **https://orcid.org/0000-0002-4079-2933**

She is an assistant professor in the School of Electrical Engineering and Computer Science at Gwangju Institute of Science and Technology since September 2021. She received her MS in Computer Science from Pohang University of Science and Technology in 2009 and PhD degree from Kangwon National University in 2020. She worked as Post Doc at Pohang University of Science and Technology between 2020 to 2022. She has served on the post of assistant professor in Computer Science Dept. at Daegu Catholic University between 2011–2019. Prior to her journey in academia, she worked in multinational software companies in India between 2004–2007. Her area of interest includes network management, Artificial Intelligence and streaming.

Yang-Sae Moon **https://orcid.org/0000-0002-2396-0405**

He received the B.S., M.S., and Ph.D. degrees from the Korea Advanced Institute of Science and Technology in 1991, 1993, and 2001, respectively, all in computer science. He is currently a professor with Kangwon National University.

Mi-Jung Choi **https://orcid.org/0000-0002-9062-4604**

She received the B.S. degree in CS from Ewha Womans University in 1998 and the M.S. and Ph.D. degrees from the Department of CSE, POSTECH, in 2000 and 2004, respectively. She is currently an associate professor with the Department of Computer Science, Kangwon National University, South Korea. Her research interests include malware analysis and SDN/NFV management.