Jacob Feldman, Michael Buhrer, Lumbardh Halitjaha, Rohan Chaudhari
COSC 465- Computer Networks
Professor Gember-Jacobson
5/3/19

Informative Adaptive Real-time Streaming

ABSTRACT

Adaptive streaming over HTTP is being adopted by major commercial companies that provide video delivery on demand; however, these companies typically use their own proprietary DASH clients instead of leveraging adaptive software. Instead, this paper hopes to describe a tool that has been developed which integrates a flexible and adaptable DASH client, streaming from an Apache server into the open source VLC media player.

We will additionally discuss this program in the context of its role as an informative, educational tool that can help students of Computer Networks better understand how video streaming works and how different parameters affect the user experience. We provide students with the opportunity to make changes to underlying mechanisms and experiment with different bandwidth constraints and bandwidth-determining algorithms that can affect performance, and as a result the quality of experience. Our code can also be used as a framework for more experienced programmers to modify the underlying algorithm which makes decisions regarding the management of resources and the subsequent requests to the Apache server.

Keywords: Video, Adaptive Streaming, Frames, Resolution, HTTP, HTML5.

1. INTRODUCTION

Video streaming is predicted to account for 84% of all internet traffic by 2020 [1]. Most of the video streaming services and platforms such as Netflix and Youtube use their own proprietary players and algorithms to integrate video within the browser. All these systems leverage the same technical advantages over traditional video streaming, i.e., the real time transport protocol (RTP) which is based on the user datagram protocol (UDP) [2]. A DASH client, the underlying technology behind video streaming, is able to adapt the video stream to the varying bandwidth conditions intelligently. This is very important when we consider scenarios where users may be using a limited mobile connections on a smartphone, tablet, or otherwise. As previously mentioned, most of these decisions regarding managing client bandwidth constraints is done by proprietary algorithms which are under wraps. However, we can see tremendous advances in the ability of streaming platforms as they came from pipe dream to reality. In addition entirely new and novel opportunities using streaming algorithms have presented themselves, from driverless cars to internet of things devices which stream video to the cloud for processing.

The decisions that streaming clients make are as simple or as complex as we choose to make them. This being said, it is not wise to downplay the effects that these choices have on user experience. Even small changes in the loading time of a video or number of buffering events could mean millions of dollars in user retention. As a result, we find that understanding how these decisions are made, is a critical issue. However, it is not easy to explain given the number of different factors which are involved. We believe a tool that helps users experiment with the parameters and the decision making process helps ameliorate the understanding of our, rather complex, problem. We hope that the tool we have built provide a meaningful interface that students of networking can use to see the effects of parameter changes in real time.

Our paper begins by discussing the various related works that gave us insight into the techniques and applications of real time streaming which are utilized throughout industry. Next we summarize this research by discussing the process of video streaming. After, we discuss the solutions we developed, namely our educative software. Finally, we discuss the educational value and potential impact of our final product.

2. RELATED WORKS

Akshabi et. al. note that even though there is extensive work done on rate-adaptive streaming over UDP, transport of rate-adaptive video streaming over TCP, and HTTP in particular, presents unique challenges and have not been studied in depth in the past.

Rainer, Lederer, Müller and Timmereret[2] note that Apple's HTTP Live Streaming (HLS) [2] is fully integrated in the web browser using the HTML5 video element and therefore it is possible to define the m3u8 manifest file as source of the video element. That is, the parsing of the m3u8 file, a playlist file used by different audio and video playback programs, and download of the segments is handled within the Web browser. Unfortunately, this functionality works on Apple systems only since Safari versions for Windows or Linux do not support HLS. They further note that other adaptive HTTP streaming systems do not make use of the HTML5 video element but deploy browser independent platforms like Microsoft Silverlight (Smooth Streaming [1]) and Adobe Flash (Adobe HTTP Dynamic Streaming [3]). However, while the video player platform is available for download on demand, these platforms are poorly supported on mobile devices.

3. HOW VIDEO STREAMING WORKS

Industry solutions, as well as the accepted DASH approach, follow the same method regarding chunk, or segment based HTTP streaming [2]. The fundamental idea is that to avoid sending entire files over potentially inconsistent connections videos can be broken into smaller segments and requested sequentially via HTTP GET requests [2]. This simplification also gives the content server the opportunity to present the user with different resolutions for each segment according to network conditions. In MPEG-DASH, the server stores all the segments for each resolution in a framework called a representation. Representations are the powerful tool in streaming because they give the player the opportunity to request different resolutions, languages, ect.

The file which maintains a ledger of these representations is a media presentation description (MPD) file. After the user initially acquires the MPD, they are able to request whichever segments they want. What is required then is an algorithm which decides which segment to request and when. The algorithm which we utilize is a basic available bandwidth scheme which relies on measuring the time which segments take to pull. Based on the size of the next segment to pull in different possible resolution, this scheme selects the highest resolution possible given the size of the previous segment pulled and the time it took to pull it. Put simply, it measures the bandwidth which was available in the last segment and assumes that it will be the best estimate of the bandwidth during the requesting of the next frame. This solution is obviously limited in its predictive powers when bandwidth is irregular. In addition, our solution does not consider the effects of multiple streams or general trends in connections which many more advanced algorithms do.

4. METHODS AND TOOLS

Streaming algorithms use several variables in order to determine the optimal way to stream video. As such, our software allows the client to manipulate these determinants to simulate a wide range of streaming environments. Our user interface gives access to variables such as bandwidth, join time, and buffer capacity while also providing the user with real-time feedback on the video streaming quality. Additionally, as bandwidth is typically complex and varying in a real-world environment, we include several distinct algorithms to allow for artificial bandwidth fluctuations. The algorithms that we provide for testing in our software are Linear Max, Random, Linear Growth, Sawtooth, and Custom. Linear Max is the algorithm we described in section 3, namely we request the best resolution based on the bandwidth observed during the pull of the previous segment. Random simulates random variables for bandwidth which is then used to determine the resolution of the segment we pull. Linear Growth starts with a small bandwidth and increments it linearly, thus improving resolution over time. Sawtooth simulates a fluctuating bandwidth and as a result we observe a frequent increase and decrease of the video quality. In addition to the specific algorithms, our software includes the option for custom functions which determine bandwidth behavior.

We use Python-VLC, a VLC-based open-source python video player (found here: https://pypi.org/project/python-vlc/#description) to output the content.[12] Each segment of video buffered is also rendered into a file to allow for video streaming of the content. We implement Dynamic Adaptive Streaming from "A survey on bitrate adaptation schemes for streaming media over HTTP” by Rainer et. al. as a basic implementation of an adaptive streaming algorithm.[2] This provides functionality for resolution adaptation based on the bandwidth observed from previous segments.

We allow for the implementation of other adaptive streaming algorithms into our software. Users may modify part of our code to replace our streaming algorithm with a choice of their own. This makes our software a framework for testing performance of a variety of algorithms. Our work ultimately resulted in a compelling tool which allows users great freedom to implement their own algorithms and observe their behavior under various parameters using the framework that we created. All the functionality serves the purpose of helping students compare the relationship between video quality and connection bandwidth under numerous conditions.

5. CONCLUSIONS

In this paper, we present an educational tool that we have implemented for students to be able to understand the various trade-offs involved in maximizing quality of experience when streaming video. In addition to serving as a means to understand the factors that affect the video streaming experience, the tool can be used as a medium through which students can compare the results of the various adaptive streaming algorithms currently available. Being able to interactively experiment with fundamental parameters will give students an easy way to visualize the challenges of video streaming and to better understand how one may design an algorithm to improve the overall quality of experience.

We strongly believe that a framework where students and researchers can test potential new algorithms will help further research and education in adaptive streaming algorithms, ultimately improving user experience for everyone. Students can develop solutions that could potentially compare with those which are used today in applications which reach billions of users.

REFERENCES

1. Paper, Cisco White. "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2010-2015." (2011).
2. B. Rainer, S. Lederer, C. Müller, C. Timmerer. “A seamless Web integration of adaptive HTTP streaming,” in Proc. 20th Eur. Signal Process. Conf. (EUSIPCO), Aug. 2012, pp. 1519–1523, https://www.researchgate.net/publication/236594173_A_seamless_Web_integration_of_adaptive_HTTP_streaming/download
3. A. Bentaleb, B. Taani, A. Begen, C. Timmerer. "A survey on bitrate adaptation schemes for streaming media over http", *IEEE Communications Surveys & Tutorials* (2018), https://ieeexplore.ieee.org/document/8424813.
4. S. Akhshabi, A. Begen, C. Dovrolis. “An experimental evaluation of rate-adaptation algorithms in adaptive streaming over HTTP”, *Proc ACM MMSys (2011): 157-168, https://dl.acm.org/citation.cfm?doid=1943552.1943574*.
5. L. De Cicco, S. Mascolo. "An adaptive video streaming control system: Modeling, validation, and performance evaluation", *IEEE/ACM Transactions on Networking (TON)* 22.2 (2014): 526-539, https://ieeexplore.ieee.org/document/6493502
6. D. Wu, Y. T. Hou, W. Zhu, Y. Zhang, J. M. Peha. "Streaming video over the Internet: approaches and directions", *IEEE Transactions on circuits and systems for video technology* 11.3 (2001): 282-300., https://dl.acm.org/citation.cfm?id=2323216.
7. T. Stockhammer. “Dynamic Adaptive Streaming over HTTP- Standards and Design Principles”, *ACM Multimedia Systems*, San Jose, California, USA, 133-143.
8. I. Sodagar. “The MPEG-DASH Standard for Multimedia Streaming Over the Internet”, IEEE Multimedia, vol. 18, no. 4, 62-67.
9. R. Pantos, W. May, ―HTTP Live Streaming ‖ , IETF draft, http://tools.ietf.org/html/draft-pantos-http-live-streaming-07.
10. Adobe HTTP Dynamic Streaming, http://www.adobe.com/products/httpdynamicstreaming/.
11. HTML5 Media Source API, http://code.google.com/p/html5-mediasource-api/.
12. C. Müller, C. Timmerer, ―A VLC Media Player Plugin enabling Dynamic Adaptive Streaming over HTTP, ACM, Scottsdale, Arizona.
13. J. Le Feuvre, C. Concolato, J. C. Dufourd, R. Bouqueau, J.-C. Moissinac, Experimenting with Multimedia Advances using GPAC ‖ , ACM Multimedia, Scottsdale, USA.