Final Year Project

Project Plan

Performance Analysis for Different Congestion Control Algorithms on Video Streaming over LTE

Submitted By :
Xue Kaiwen 55199704

Contents

1 Motivation

Congestion control, aiming at reducing network congestion situations that cause too many packet retransmissions by controlling the sending rate from the sender, is one of the major parts in design of the transport layer in computer networks[1]. Internet standards of the set of TCP congestion control algorithms are regulated in RFC2001[2] as TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms. Current state-of-art congestion control algorithms includes CUBIC[3], BBR[4], CQIC[5], Sprout[6], etc. Each algorithm is designed to achieve certain benefits.

Adaptive Bitrate Streaming (ABR) is a modern video streaming technique. ABR algorithms feature changing the bitrate of a video during playback dynamically to ensure least rebuffering events while maintaining best video quality as far as it can. This requires proper video encoding standards and implementation of ABR players. MPEG-DASH[8] is an HTTP-based ABR implementation with APIs of creating ABR players.

Video streaming currently occupies over half of network traffics[9]. For this reason, finding methods to improve its overall quality is of great benefit. As different congestion control algorithms favor different situations, it is essential to measure and analyze the performance of different congestion control algorithms on video streaming.

2 Problem Statement

Existing works on congestion control mentioned above design and implement congestion control algorithms and test these algorithms under various networking conditions. On the other hand, some works proposes specific ABR algorithms[10][11]. Testings on these works focuses on the comparison of different ABR algorithms implemented on client side. Transport protocol is seldom considered. Furthermore, LTE[12] networks, which is used widely for video streaming on mobile, has some distinct features other than normal cable links. Fluctuation of LTE traces makes the internet condition unstable.

Our project aims at design a complete experimental toolkit of testing different different ABR algorithms on different congestion control algorithms over different LTE links. Note that here congestion control algorithms are not limited to TCP congestion control. Implementations of congestion control algorithms on QUIC[13] instead of TCP will also be considered.

3 Technical Components

3.1 LTE Network Emulation

We expect to emulate LTE networks using Mahimahi, a command line tool that provides accurate measurements when recording and replaying HTTP content over emulated network conditions[14]. Various LTE traces from different provider and real conditions will be used under the emulation.

3.2 Automated Testing

We will combine several scripts to automate the testing. These scripts will invoke the mahimahi, connect to server, play the video, and output data needed.

3.3 Video Server

We will implement a video server that supports playing several videos under MPEG-DASH standards. This may include modification of dash.js[15] and pensieve[10] code. Other tools could be added for testing on QUIC.

3.4 Plotting Tool

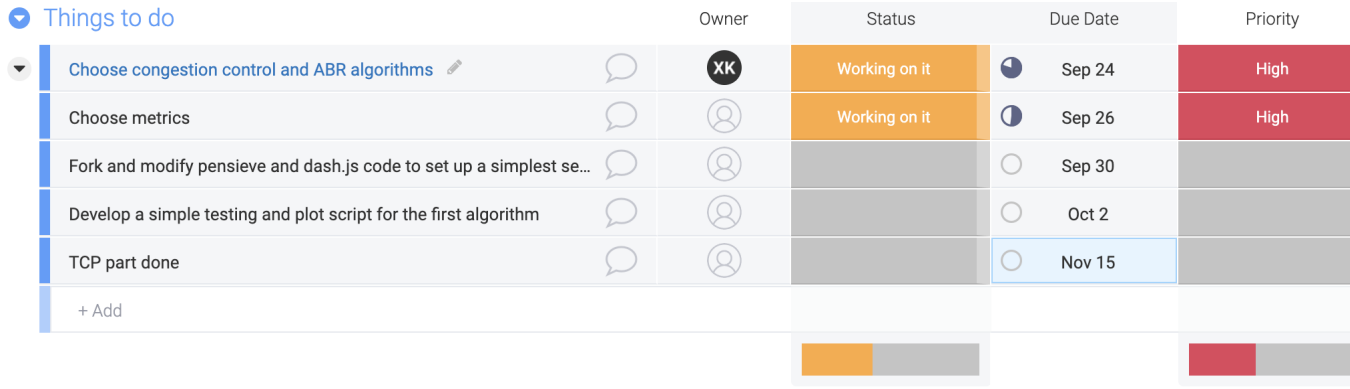
After getting all results needed, data visualization is very important. We will implement plotting tools that can be used for the analyzed results and reproduced results.

4 Results and Delieverables

Upon completion, this project will produce detailed results and analysis on different congestion control algorithms and ABR algorithms. The experiment will be implemented as a toolkit with guide on how to build testbeds. It will be a starting point for designing new congestion control protocols that are beneficial to most video streaming algorithms over LTE networks. Metrics used in this project can also be reused.

5 Project Schedule

This project is expected to adapt iterated developing procedure. Testing on different congestion control algorithms will require different emulations, testings, servers, and plotters. As these implementations may be similar, we will consider integrating it to as few lines of code as possible once several tests are done. The schedule will be roughly split into preparation, TCP testing, and QUIC testing.



Note that this gantt graph is very rough as the future work plan depends on the current progress.

Bibliography

[1] Kurose, James; Ross, Keith (2012). *Computer Networking: A Top-Down Approach (4th ed.).* Pearson. ISBN 978-0-13-285620-1

[2] TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms. (n.d.). Retrieved from https://tools.ietf.org/html/rfc2001

[3] Ha, S., Rhee, I., Xu, L. (2008). CUBIC: a new TCP-friendly high-speed TCP variant. Operating Systems Review, 42, 64-74.

[4] Neal Cardwell, Yuchung Cheng, C. Stephen Gunn, Soheil Hassas Yeganeh, Van Jacobson, BBR: Congestion-Based Congestion Control

[5] Feng Lu, et al. CQIC: Revisiting Cross-Layer Congestion Control for Cellular Networks

[6] Keith Winstei, et al. Stochastic Forecasts Achieve High Throughput and Low Delay over Cellular Networks

[7] A. Bentaleb, B. Taani, A. Begen, C. Timmermer, and R. Zimmermann, "A Survey on Bitrate Adaptation Schemes for Streaming Media over HTTP", In IEEE Communications Surveys Tutorials (IEEE COMST), Volume 1 Issue 1, pp. 1-1, 2018.

[8] I. Sodagar, *The MPEG-DASH Standard for Multimedia Streaming Over the Internet,* in IEEE MultiMedia, vol. 18, no. 4, pp. 62-67, April 2011. doi: 10.1109/MMUL.2011.71

[9] Sandvine, *The Global Internet Phenomena Report*, October 2018

[10] *Neural Adaptive Video Streaming with Pensieve* Hongzi Mao, Ravi Netravali, Mohammad Alizadeh Proceedings of the 2017 ACM SIGCOMM Conference

[11] *A Buffer-Based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service* August 2014ACM SIGCOMM Computer Communication Review

[12] The MobileBroadband Standard. (n.d.). Retrieved from https://www.3gpp.org/DynaReport/36-series.htm

[13] Adam Langley, et al. *The QUIC Transport Protocol: Design and Internet-Scale Deployment*

[14] Netravali, R. (n.d.). Mahimahi. Retrieved from http://mahimahi.mit.edu/

[15] Dash-Industry-Forum. (2019, September 13). Dash-Industry-Forum/dash.js. Retrieved from https://github.com/Dash-Industry-Forum/dash.js