Ivan Kouzmine

**ECE 50863 – Sring 2020**

**ABR Algorithms with Dash.js**

**Introduction:**

This is an exploration of available ABR algorithms that are currently out in the world. As the backbone, two formats are used for file encoding, DASH and HLS. We will be exploring both in the upcoming experiments. The tool that will help accomplish this task is Dash.js, an open-source project for playing videos on the web encoded with DASH/HLS. Besides allowing various implementations of existing ABR solutions, it allows for some customization as well which will hopefully be the very final task accomplished in this experiment.

**Implementation:**

The current setup is very simple but working. Using python’s very simple built-in web-server, a single page is hosted along with the content to be viewed. Behind the scenes, there is currently only a simple script that uses ffmpeg to convert an input mp4 video file into the segmented DASH format (valid settings pending). These segments go into a content folder within the server directory. The server can be started with “python -m http.server <port>” and the current sample converted video can be viewed there.

**More to come**

**Playing around with ffmpeg conversion to see effects on streaming quality/speed**

This part is closely tied with network simulation but is up here because I would like to get my conversions correct. I have described several bugs that I have run in to while performing conversions into DASH below

**Implementation of Dash.js ABR algorithms**

**Implementation of Custom ABR algorithms**

**Implementation of network simulation**

This is at the very bottom of the “more to come” stack, but this is because I think the initial setup and network simulation will take the most time while various algorithm implementations shouldn’t consume too much time. Specifically, for network simulation, I might have to stray away from python’s simple web-server which is something that will just take a little more time to setup.

**Plans for future**

Dive into implementing the existing solutions as well as fixing a couple of minor conversion bugs that I ran into while setting up this project. I’d like it to be noted that most of my time was spent on trying to get the conversions working on windows using existing ffmpeg binaries for it. For some reason whenever I would convert the segments, the first half would be overwritten by the second half of the video making only half the video buffer. I made a sort of hot-fix for it, but this issue is also one that I will be tackling sooner rather than later.