Comparision of Energy Efficiency of YouTube Streaming in YouTube Application and Chrome

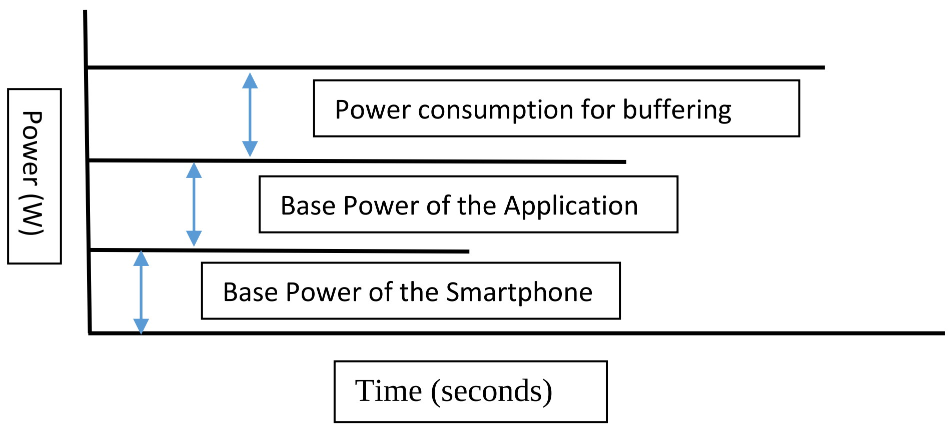
Md. Sakib Nizam Khan (546713)
Shiva Prasad Thagadur Prakash (547903)
Viswanathan Manihatty Bojan (547039)

Overview of the Applications

Currently, video streaming consumes the major proportion of the overall internet traffic. This is primarily because of the large number of multimedia content being produced and also due to the popular video-on-demand services that are providing these contents in an on-demand basis to the viewers. With the advancement in the technologies, such as, Wi-Fi and mobile data networks (i.e. 4G, 5G) aiding the wider availability of the internet to the larger portion of the society, the video streaming traffic is anticipated to increase to a greater extent in the coming future. YouTube is one of the most popular video-on-demand service since its launch in 2005. YouTube has played a prominent role in transforming the way how people perceived entertainment through media content a decade back. YouTube basically acts a platform where the users can watch and upload the videos. Apart from this, it also acts as a social networking platform where the users can share, comment and rate the videos. In this way, YouTube has proved to be more successful than its competitors in holding on to its customers.

The expansion of the smartphone industry helped YouTube in gaining additional users through the dedicated YouTube mobile applications built for various platforms, such as, Android, iOS and customized site (m.youtube.com) for mobile browsers apart from the existing desktop site. The smartphones are getting more powerful every single day and with every new model release. However, it has limitations when considering its battery life. Even though there was a tremendous growth in the smartphone hardware architecture, there were not many developments in improving the battery life. Therefore, studying the energy consumption behaviour of smartphones has been a subject of research interest for many researchers for a long time. The applications residing in the smartphone contributes largely towards the energy drain and YouTube is no different. In the case of YouTube application, whenever a user requests for a video, the application requires to perform a fair number of network related and other functionalities, such as, establishing network connections, transfer of packets and decoding. These functionalities in turn require the work of the smartphone hardware and thereby contribute its share towards the battery drain. Accessing the YouTube website from the mobile browsers also shows a similar practise. Hence, it becomes essential to study the energy consumption behaviour of the mobile YouTube which includes both mobile application and website.

The purpose of this paper is to compare the power consumption behaviour of video streaming between YouTube application and Google chrome browser on an android smartphone. Different test cases were designed with the key intention of identifying the major differences in power consumption pattern while streaming a video on YouTube application and Google chrome browser. In addition, we also tried to find out the causes for these differences in power consumption behaviour. The two main activities performed by mobile YouTube for video streaming are **buffering** and **local processing**. Buffering includes delivery of video from servers and local processing includes decoding and local playback. Hence, it is important that an energy consumption study is well grained to cover each activity's energy consumption including the interactions between them. We initially measured the base power consumption of the smartphone. Then, we measured the base power consumptions of the YouTube application and the Google chrome application. To study the difference in power consumption behaviour for buffering a video on both these applications, we measured the energy consumption for buffering on both the applications and subtracted the corresponding measurements from the sum of base power consumptions of the smartphone and their respective applications. The below diagram gives an overview of the same.



Similar steps were followed to study the difference in power consumption for local processing on both the applications. In order to identify the different phases of the video streaming activity, we have captured the network packets using Wireshark. In addition, we also compared the energy consumptions for streaming the videos of different resolutions (i.e.

high and low resolution videos) and also using different network access technologies (i.e. 4G and Wi-Fi). In the below table, each row depicts the different use cases and each column symbolizes different test variables.

Use Cases	Network	Video Quality
YouTube loading on Chrome	Wi-Fi + 4G	NA
YouTube Application loading	Wi-Fi + 4G	NA
Streaming of YouTube video on Chrome	Wi-Fi + 4G	High Quality + Low Quality
Streaming of video on YouTube Application	Wi-Fi + 4G	High Quality + Low Quality
YouTube video playing on Chrome after buffering is complete	Wi-Fi + 4G	High Quality + Low Quality
Video playing on YouTube Application after buffering is complete	Wi-Fi + 4G	High Quality + Low Quality

Table 1

Equipment Used for the Experiment:

Smartphone:

Samsung Galaxy S4 smartphone was used as a part of the experiment. It runs on Android OS and has a Quad Core processor. It has a 5 inch display with multi touch feature. The smartphone is equipped with a removable Li-Ion 2600 mAH battery. It supports both Wi-Fi and 4G technologies for internet communication. The smartphone also houses a dedicated YouTube application and Google chrome browser. The YouTube application is designed in an optimized manner providing multiple features such as viewing, uploading and sharing of videos.

PowerTool and Power Monitor:

PowerTool is software and Power Monitor is hardware. Both the tools work together in measuring the power of the mobile devices. The capacity of the Li-Ion battery residing in the smartphone is subjected to drop depending on the type of functionality. Experimenting with Li-Ion battery in the smartphone will lead to results that are uncertain. Hence, we used a power monitor to provide a consistent power supply to the smartphone. Of the three channels (Main, Aux and USB) available in the Power Monitor for power supply, we have used the Main Channel to connect to the smartphone. The USB connector at the back of the Power Monitor is connected to the USB of the workstation where the PowerTool is installed. This helps in the power measurement of the smartphone in a synchronised manner.

Experimental Setup

Initially, the smartphone battery was removed and the Voltage (+) and Ground (-) terminals are identified and these terminals are insulated using the cellophane tape. Two thin lengthy copper strips were then connected to the insulated terminal so that they in turn connect to the phones power supply terminals. It was made sure that the remaining two terminals of the battery are left completely free. The battery was then placed inside the smartphone. The other free ends of the copper strips were connected to the corresponding knobs of the main channel using the j-hooks. Then we started the PowerTool software to set the voltage according to the requirement of the phone and finally enabled the power supply. The smartphone was then switched on. The PowerTool interacted with the Power Monitor and displayed the interface for power measurement and recording. The scale dialog box of the PowerTool was then modified as per the requirement. At this stage, the PowerTool began to display the graphs based on the power recorded by the Power Monitor. The test cases were then run in a sequential manner to collect the recordings under each scenario.

Experimental Results:

Based on the test cases discussed in the above table, we performed the experiments to measure the energy consumption of YouTube in its dedicated application as well as launching it in the chrome web browser. We will also be discussing the obtained results and the reason for the behaviour of the application in the specific test environment.

Base Power of the phone:

The smartphone consumes a considerable amount of energy even during its idle state. This includes the energy consumed to keep the display on, keeping the network connections, etc which cannot be avoided. This energy also has a significant impact on the overall energy consumption of the smartphone. Hence, it becomes mandatory to measure the base power of the smartphone in idle state. However, having the mobile in the idle state is not always possible as there are some background services that are always functioning which are invisible as well as unavoidable. Hence, to maintain the idle state, we have kept the following parameters constant.

- The brightness of the phone was set to maximum and the automatic brightness was switched off.
- The volume of the smartphone was brought down to its minimum.

- All the applications were closed.

The following table gives the details of the base power recorded for the smartphone under different network connections.

Network connection enabled	Average Base Power (W)
4G LTE	1.03
Wi-Fi	0.89

Table 2

Base power of the application:

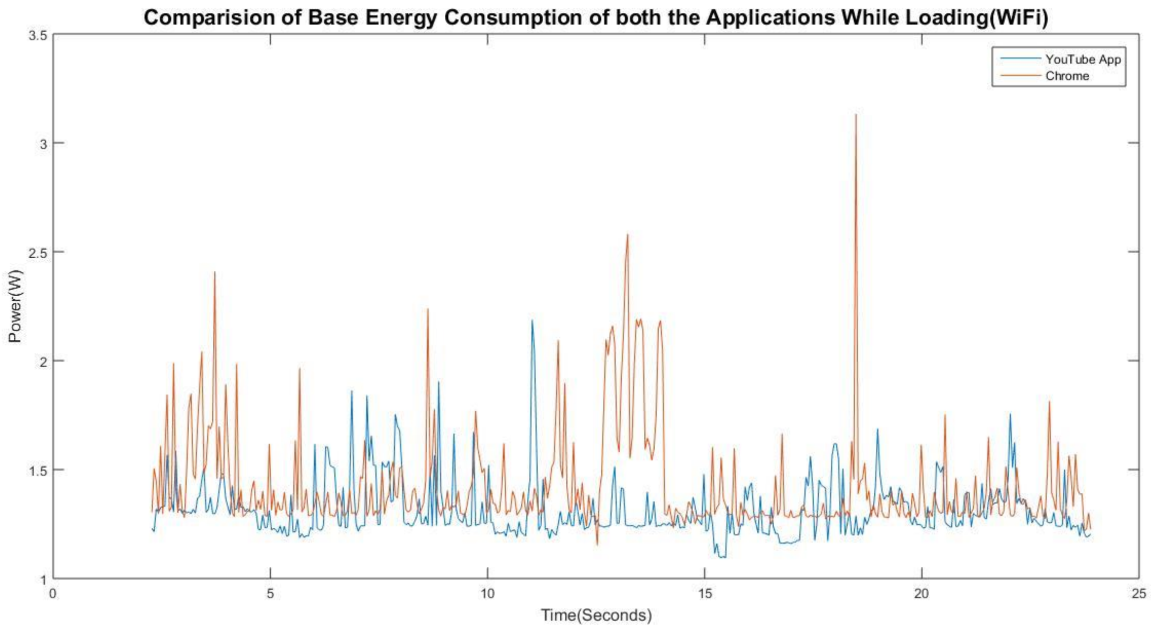


Figure 1

Both YouTube and the Chrome applications independently require certain amount of energy to load. Here, we refer to it as the base energy consumption. It is essential to measure this base energy associated with both the applications as it acts as the threshold point beneath which the energy never falls back even during the application's idle state. In addition, in order to measure the streaming power of the application, which is the prime purpose of the experiment, it is mandatory to identify the base energy consumption of the application as the actual streaming power will be the difference between this base power and the total power consumed during the streaming. The below table gives the details of the base power recorded for the applications under Wi-Fi and 4G connections.

Application	Average Base Power (W) with Wi-Fi connection	Average Base Power (W) with 4G connection
YouTube Application	1.31	1.32
Chrome	1.42	1.52

Table 3

The graph (Figure 1) depicts the base energy consumption of both the application under WiFi network connection. From the graph, it is evident that the energy consumed by the chrome browser is more than that of the YouTube application. This is because chrome is not optimized for specific applications such as YouTube. On the other hand, the YouTube application is a dedicated smartphone application which is designed focussing on energy optimization.

In addition, we also measured the base energy of the applications under 4G connection and it showed a similar behaviour to that of the Wi-Fi connection. However, we can observe that, under both the network connections, the YouTube application recorded very similar power. But, in the case of the chrome browser, the energy consumed while loading in a 4G connection was even more than that recorded under the Wi-Fi connection.

Loading power of the applications:

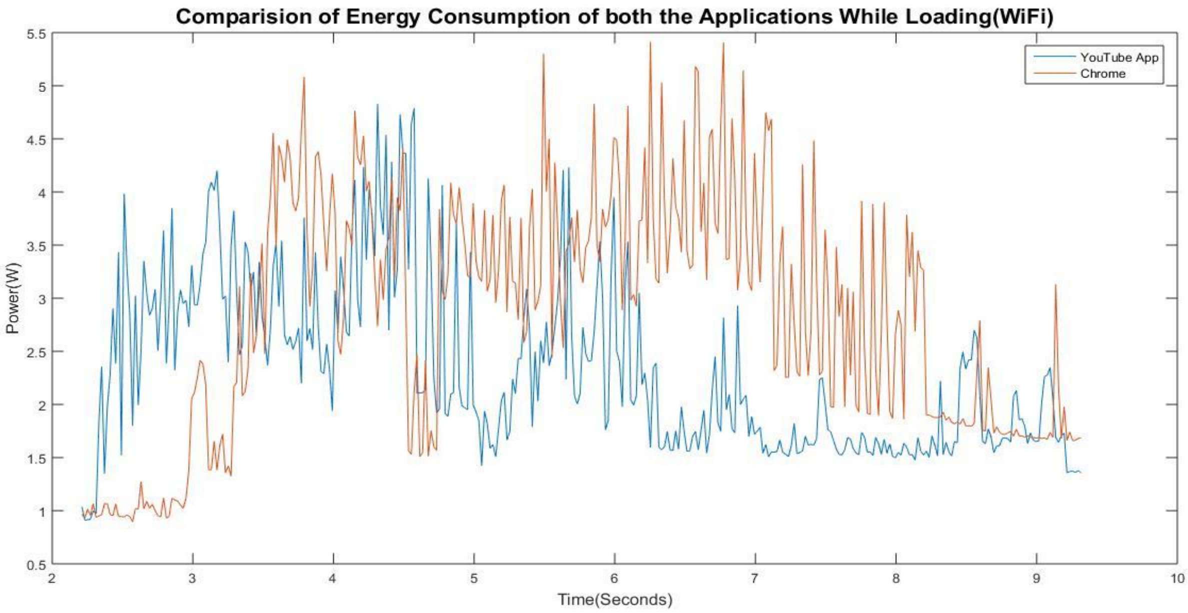


Figure 2

Here, we refer to the loading power as the energy consumed when the application gets loaded. Studying the loading power is important since they contribute substantially towards the total energy consumed for video streaming.

Once the YouTube application is launched, the processor requires some processing to initialize the application and make it ready for the user. In addition, initially when the application loads, it provides recommendations based on the user's viewing history and subscriptions which requires a fair amount of network functionalities. All the aforementioned tasks contribute towards the overall energy consumption of the YouTube application. When the YouTube site is launched in the chrome browser, it also involves in establishing network connections and processing similar to the YouTube application and thereby consuming energy.

However, when comparing the average energy recorded for the both the applications as displayed in the below table, we can clearly understand that the loading consumes higher energy in the case of chrome browser than in YouTube application.

Application	Average Loading Power (W) with Wi-Fi connection	Average Loading Power (W) with 4G connection
YouTube Application	2.32	2.89
Chrome	2.81	3.03

Table 4

Power Consumption during Video Streaming:

The video streaming includes the video buffering phase and the local processing phase (decoding and local playback).

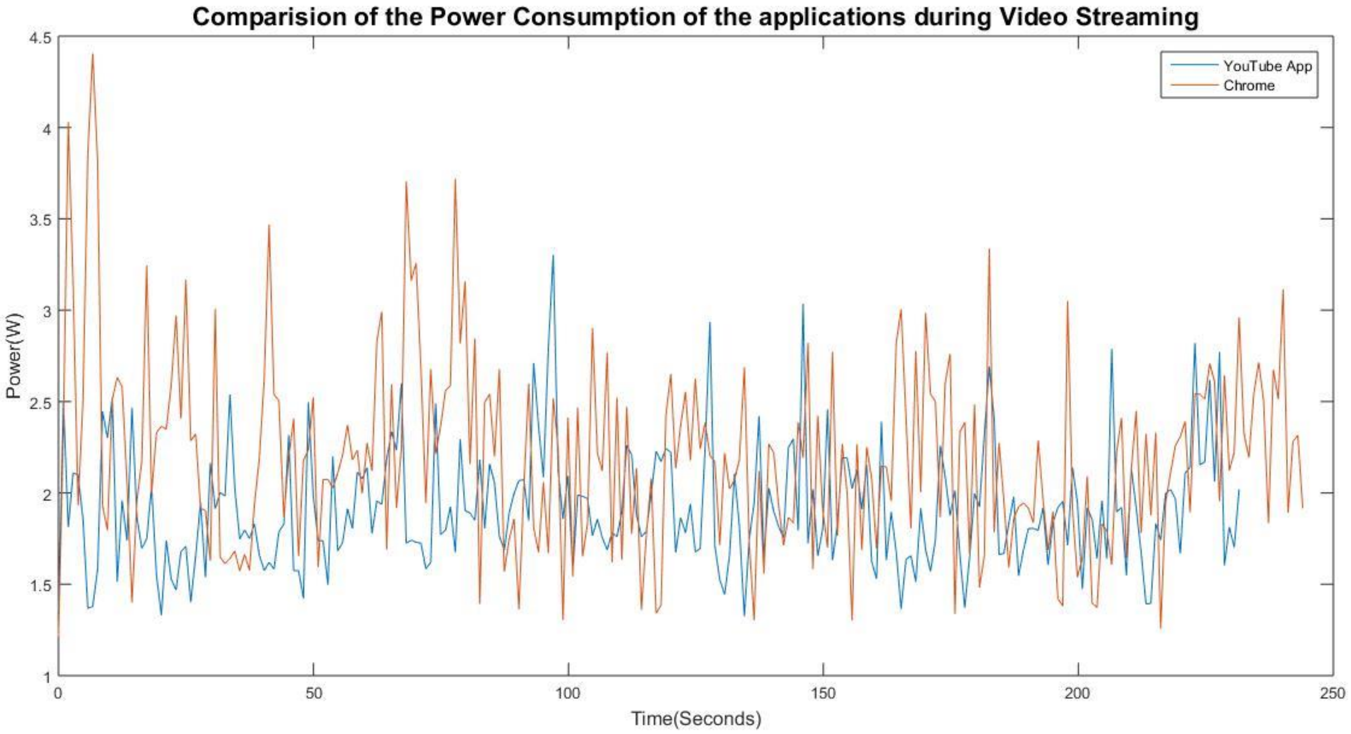


Figure 3

The energy consumed once after the video begins to play in the YouTube video player is referred to as the Streaming power. The streaming involves operations such as video codec decoding, downloading and the playback at the YouTube application or browser's end from where the request is made.

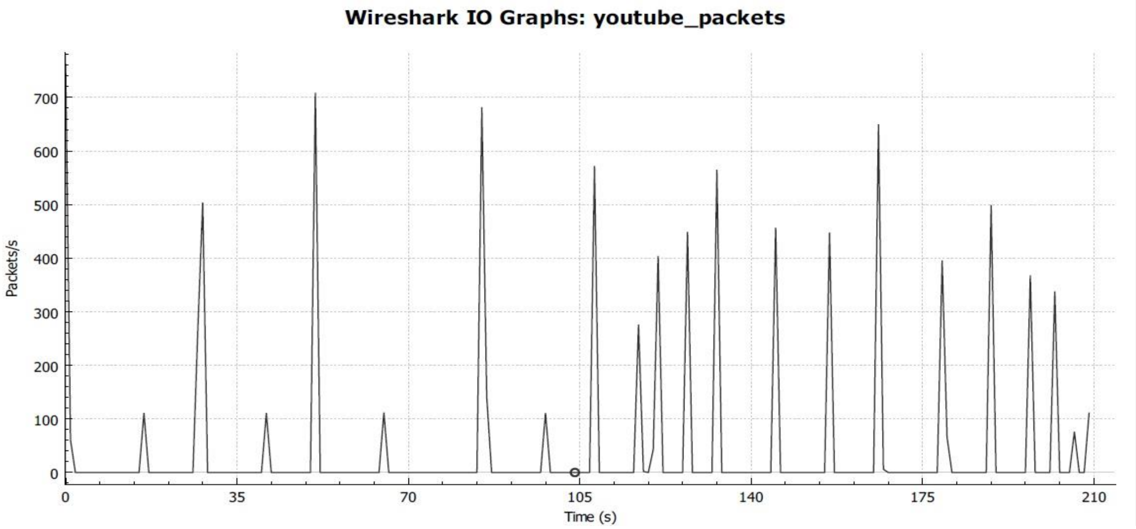
The vast traffic of YouTube video streaming is delivered to the users using progressive downloading technique. Under this technique, there are two phases:

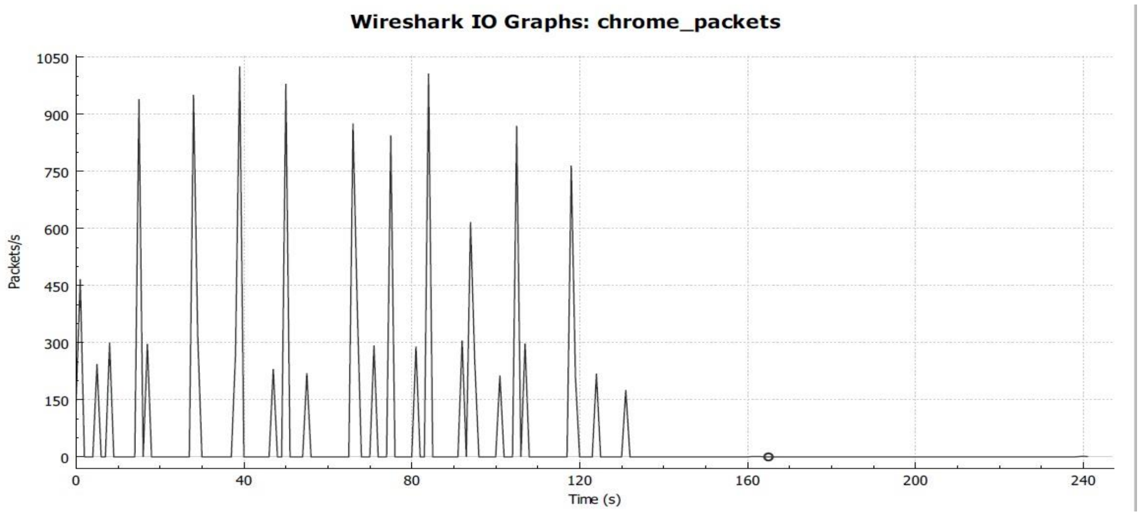
- Startup Phase
- Throttling phase

In the startup phase, YouTube's application server sends the first 30-40 seconds of video data as a bulk TCP transfer. This helps in building the initial playback buffer. The below pic displays the same where the initial chunk of data is already downloaded and ready for play. This behaviour can be seen in both the YouTube application and in any browser.

In the throttling phase, YouTube's application server delivers small chunks of video data to the client's video player at specific intervals of time. This can be observed clearly when a video is getting played where the small progressive 'white line' (as seen in the above image) begins to progress in an incremental fashion. During every incremental step, there is a burst TCP data packets that is being sent to the client machine.

We observed the behaviour of streaming of a high quality video (1080p HD) on both YouTube application and the chrome browser under WiFi connectivity and their respective energy consumption was studied and depicted in graph (Figure 3). Here, we can observe that at the start of the video playback, the energy is much higher in both YouTube application and chrome browser. This refers to the startup phase. Since that, at this point, there is a considerable amount of TCP packet transfer happening between the YouTube application server and the client, the smartphone's hardware components demand energy for conducting the network operations and the same can be seen reflected in the graph. The intermediate spikes resemble the periodic TCP bursts. The TCP traffic for a video streaming (buffering and local processing) operation in YouTube application and in chrome browser was collected using the Wireshark and the same is plotted. The graphs correspond to behaviour as discussed above.





Now, from the graph (Figure 3), we can also understand that the energy consumed in the YouTube application under the WiFi connectivity is considerably less compared to that of the chrome browser. This is because of the energy optimization characteristics of the YouTube application. The initial part of the graph corresponds to the startup phase and the later part of the graph corresponds to the local processing phase. We have also confirmed the different phases by mapping the energy graph with the Wireshark graph (plotting the packet transfer activity) at different time intervals. The average power consumed for the streaming operation in both the applications has been recorded.

The streaming power of the videos under the WiFi connection was observed in both the YouTube application and in the chrome browser for multiple rounds and their corresponding standard deviation was also noted.

Test Case	Round	Average Power during Startup Phase (W)	Average Power during Throttling Phase (W)	Standard Deviation of Average Power (Startup phase, Throttling phase)
Streaming Power- YouTube application	1	2.01007	1.79569	0.031721, 0.01869
	2	2.06036	1.7584	
	3	2.06871	1.77951	
Streaming Power- YouTube in Chrome	1	2.95548	2.14726	0.10461, 0.03235
	2	2.78976	2.16232	
	3	2.98324	2.20929	

Table 5

A study was then conducted over a low quality (240p) video and a medium quality (480 p) video on both the applications under different connections. We observed that they displayed a similar behaviour of startup phase and throttling phase. In all the test cases, the chrome browser consumed more energy than the YouTube application.

Application	Video Quality	Connection	Average Streaming Power(W)
YouTube Application	1080 p	WiFi	1.91
Chrome			3.56
YouTube Application		4G	2.34
Chrome			3.63
YouTube Application	240 p	WiFi	1.74
Chrome			3.23
YouTube Application		4G	2.13
Chrome			3.38
YouTube Application	480 p	WiFi	1.88
Chrome			3.32
YouTube Application		4G	2.21
Chrome			3.47

Table 6

The above recorded values are the average values taken after three runs of the test cases.

Additional Experimental Statistics

All the above experiments were repeated three times and the values were recorded. We also measured the standard deviation of the recorded values. The below table gives the details of the same.

Test Case	Connection	Rounds	Average Power(W)	Standard Deviation of Average Power
Base Power of the phone	WiFi	Round 1	1.02619	0.01134214
		Round 2	1.01199	
		Round 3	1.00377	
	4G	Round 1	0.98028	0.006637148
		Round 2	0.98272	
		Round 3	0.9702	
Base Power of the application – YouTube	WiFi	Round 1	1.34745	0.007421395
		Round 2	1.33292	
		Round 3	1.34281	
	4G	Round 1	1.32498	0.027534776
		Round 2	1.29321	
		Round 3	1.34805	
Base Power of the application – Chrome	WiFi	Round 1	1.26124	0.00375576
		Round 2	1.25388	
		Round 3	1.25626	
	4G	Round 1	1.25356	0.00519824
		Round 2	1.26165	
		Round 3	1.26326	

Table 7

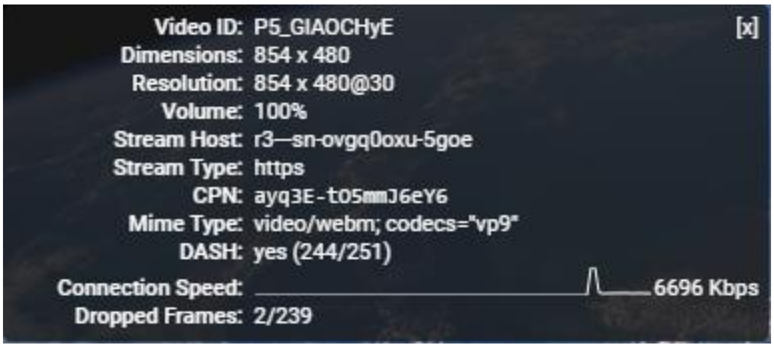
From all the above obtained data, we can clearly understand that there is a very little deviation from the recorded power when running the experiment for multiple times.

Impact of Video Codecs on Power Consumption:

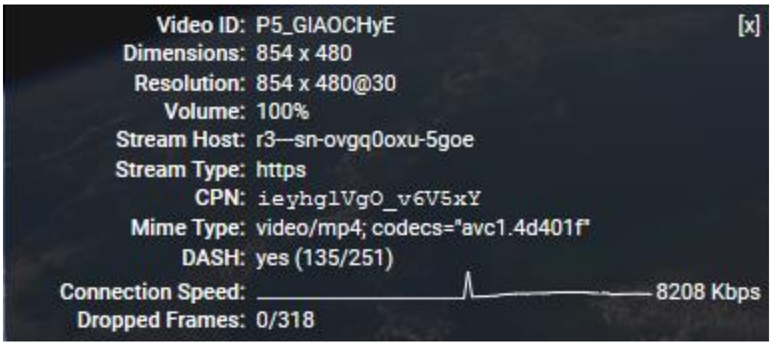
Every raw video that is uploaded in YouTube gets compressed through an encoding operation. The video encoder helps in the compression operation with the help of a video codec. Similarly, when it is received at the client end, the video has to be decoded before and during the video playback operation happens. The video decoder helps in the decompression operation with the help of the video codec.

Studying the video decoding is important because of the energy consumed during the decoding operation at the client end. Depending on the type of video codec used for encoding, the energy consumption differs. There are different types of codecs such as H.262, H.263, H.264, VP8, VP9, etc. Of all the codecs currently in use, VP9 and H.264 is the most widely used codecs for encoding and decoding operations. H.264 codec has been in use for a long time now and is supported by all the major web browsers. H.264 decoding at the client browser is hardware accelerated. This in turn means that playing the H.264 video is carried out by the Graphics Processor (GPU). They are highly efficient and consume less CPU time. As a result, the energy consumption is also less. All the smartphones are hardware accelerated. However, when we consider the VP9 video file decoding, there are not a lot of manufacturers who help in hardware accelerated decoding operation. As a result, it requires the work of CPU to decode the file and this in turn demands more energy.

Now, when we look at the Google products like Chrome and YouTube application, they decode the video file using the VP9 codec. Hence, they demand more work from CPU and consume more energy. And in the case of browsers like Firefox, the decoding is carried out by H.264 codec. GPU is responsible for decoding here and demands less CPU work and thereby consumes less energy. The below screenshots show the video codecs used in chrome browser and in Firefox browser.



Chrome Browser



Firefox Browser

Mime Type provides the details regarding codecs used in the file and the type of video format that gets downloaded. The ‘avc1’ codec in Firefox browser corresponds to the H.264 codec.

Summary

In this Section we will discuss the findings of the experiment. One of the initial observations of the experiment was that even in the ideal state to maintain the network connection and keeping the display on consumes significant amount of energy. In addition, It was also observed that initial loading of applications also require a good amount of energy. Moreover, when the application is in idle state, an increase in the base power was observed which is not negligible. The reason behind this increase is that keeping the application running in foreground requires some extra task for CPU which in turn requires extra energy. The experiment also revealed that the network transmissions are the major source of energy consumption during streaming. Whenever there was a network activity a substantial rise in power consumption was observed. In addition, the experiment also showed that there is no major difference in energy consumption between Wi-Fi and 4G as well as between High and Low quality videos.

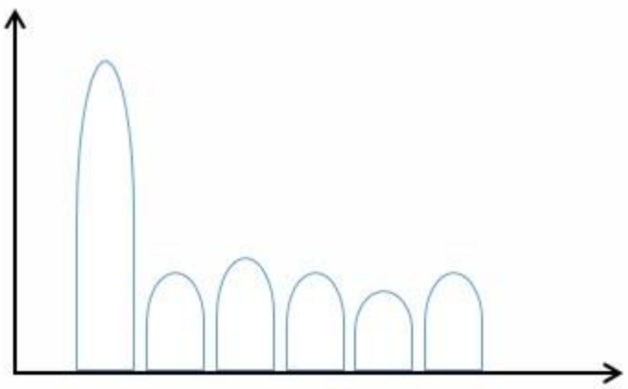
One of the major findings of the experiment was the YouTube application has better energy optimization technique than chrome as it consumes less power. Though it is hard to guess why YouTube application consumes less energy than chrome, but what we understood is that chrome is more resource intensive than YouTube. Furthermore YouTube uses smarter technology for traffic shaping which involves less network transmissions and in turn less energy consumption than in the case of chrome. Lastly, we also observed that the video codec used in the file also has an impact on the overall energy consumption. Video file encoded with H.264 video codec consumes less energy while the video files encoded with VP9 codecs consume more energy.

Power Saving Mechanisms

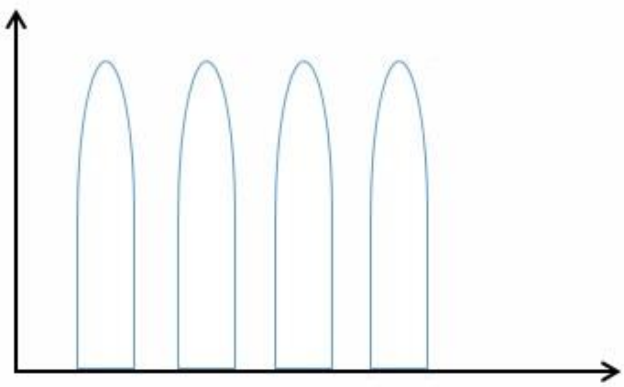
As per the studies done in different research works, the most energy efficient way of streaming a video is to download the whole video at once. Nonetheless, there is no certainty that the user would watch the whole video and if he quits watching the video in between it would result in unnecessary download of data which may incur monetary loss to the user.

Therefore, in order to combat this unnecessary loss, the standard technique used for streaming the video is to download the media content in chunks in a progressive manner.

Through the collective study of traffic pattern captured using Wireshark and the energy traces collected using the power meter, we observed that in the case of progressive download, if the interval between the bursts is very small, it would either make the wireless radio to be active throughout the video streaming or it will let the radio to have small sleep times between the bursts. This substantially impacts the battery consumption of the smartphone. In order to make the energy consumption efficient in progressive download, we propose a technique to shape the traffic in such a way that the smartphone's radio gets sufficient sleep time between the appropriate large bursts. The traffic pattern we observed is similar to the diagram shown below.

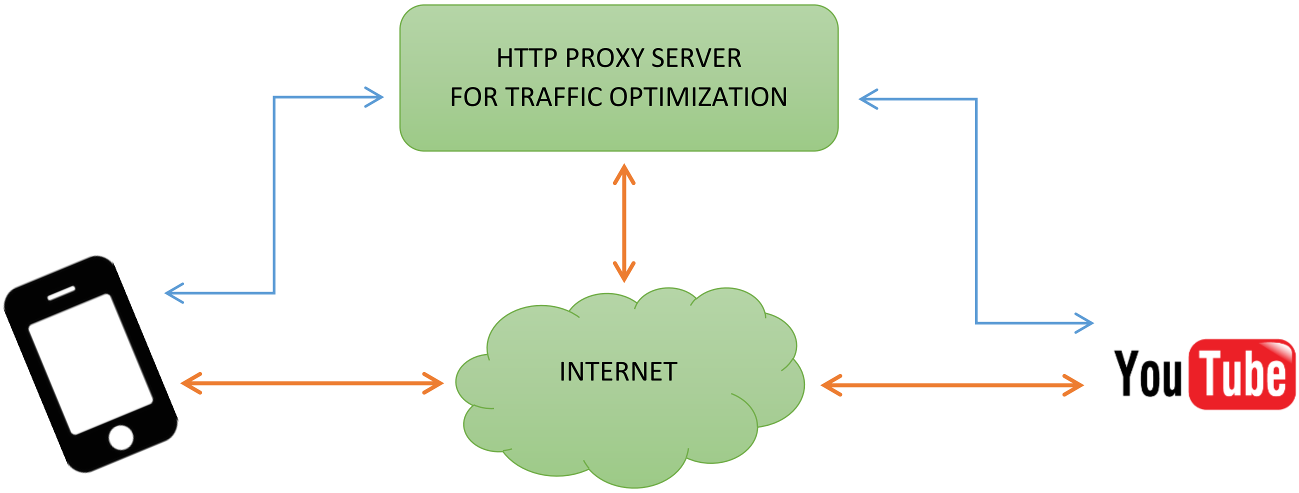


It is evident that the smartphone's radio is getting less sleep time between the bursts i.e. the interval between the traffic bursts is very small. The traffic pattern of the proposed traffic shaping technique looks similar to the one in below diagram.



This can be achieved by using a HTTP proxy. The proxy forwards the client's requests to server. It then receives and buffers the media stream from the server. Finally, it sends the buffered media repeatedly to the client in suitable big bursts so that the client's radio gets sufficient sleep time between the bursts. This in turn saves the energy. The main challenge of this proposed technique is to choose the proper burst size. One way to choose the burst size is to choose it exactly same as the receive buffer size. The receive buffer referred here corresponds to the TCP receive buffer.

During playback, the player decodes content from the playback buffer at the encoding rate and the corresponding space is freed from the playback buffer. As a result the player transfers data from TCP receive buffer to playback buffer at the encoding rate. Hence, the maximum burst size can be sum of receive buffer size and playback buffer size. In addition, since the proxy server doesn't know about the client's TCP receive buffer size, one way to probe the right size is to use binary search i.e. initially choose a random size and then increase or decrease the size in the subsequent transfer based on the intensity and presence of packet loss. For the implementation of the proposed technique we can have a proxy server that performs the functions as mentioned above and for the evaluation we can compare the power measurements taken with and without the presence of the proxy server. The below diagram depicts the architecture of the proposed technique.



Also, as per our study Wi-Fi consumes comparatively less energy than 4G. Hence, whenever the user starts streaming video we can prompt the user to check for any available Wi-Fi connections so that phone battery is saved. Additionally, since we observed that the YouTube application consumes less energy than Chrome, we can suggest the user to use the YouTube application for video streaming. These can be implemented as a part of android application.