

Dissecting Mobile Video Services: An Energy Consumption Perspective

Mohammad Ashraful Hoque*, Matti Siekkinen*, Jukka K. Nurminen*, Mika Aalto[†]

*Aalto University School of Science

Email: {mohammad.hoque,matti.siekkinen,jukka.k.nurminen}@aalto.fi [†]Nokia Siemens Network

Email: mika.aalto@nsn.com

Abstract—Multimedia streaming applications are among the most energy hungry applications in smartphones. The energy consumption mostly depends on the delivery techniques and on the power management techniques of wireless interfaces (Wi-Fi and 3G). In order to provide insights on what kind of streaming techniques exist, how they work on different mobile platforms, and what is their impact on the energy consumption of mobile phones, we have done a large set of active measurements with several smartphones having both Wi-Fi and cellular network access. Our analysis reveals five different techniques to deliver the content to the video players. The selection of a technique depends on the device, player, quality, and service. The results from our power measurements allow us to conclude that none of the identified techniques is optimal because they take none of the following facts into account: access technology used, user behaviour, and user preferences concerning data waste. However, we point out the techniques that provide the most attractive trade-offs in particular situations. Furthermore, we make several observations on the energy consumption of different players, containers, and video qualities that should be taken into consideration when optimizing the energy consumption.

I. INTRODUCTION

The present mobile market is filled with smartphones of Android, iOS, Windows, and other platforms. These devices have full functionality to play multimedia content from different streaming services such as YouTube, Vimeo and Dailymotion. In 2011 YouTube had more than 1 trillion global views and 10% of it was accessed via mobile phones or tablets [1]. Vimeo and Dailymotion are also gaining popularity on mobiles. Therefore, digital video content is increasingly consumed in mobile devices [2]. At the same time battery life of smartphones has become a critical factor in user satisfaction [3]. Consequently, it is essential that mobile video streaming not only provides a good viewing experience but also avoids excessive energy consumption.

Multimedia streaming services consider a number of challenges while sending content to the streaming clients, such as initial playback delay, clients with different kinds of connectivity and the bandwidth between a server and a client, in order to provide smooth playback [4]. With recent evolution in mobile phone industry and mobile broadband networks, energy consumption of smartphones is also considered as an important issue. To that extent, a significant number of research works focused on reducing energy consumption of mobile devices while using streaming applications [5]. The aforementioned streaming services have adopted various techniques to deliver video content to mobile users, such as Bitrate streaming,

Bitrate Throttling, ON-OFF, Dynamic Adaptive Streaming over HTTP (DASH) [6] and Fast Caching. Bitrate streaming is used to deliver content at the encoding rate. Throttling and Fast Caching send video content at a higher rate than the encoding rate. ON-OFF mechanism works based on the playback buffer status of a client and the client receives content from the server only during an ON period. DASH adapts video quality according to the end-to-end bandwidth between a server and the client.

There has been work on analyzing the merits of these streaming techniques from the server performance point of view. For example, Fast Caching reduces startup delay at the client and guards against bandwidth fluctuation, but it also consumes a lot of CPU and memory at the streaming server [4]. Although most of the techniques are understood by research community, a thorough study of these streaming techniques is still required from mobile communication perspective. Even though some studies have looked at video streaming techniques with Android, iOS devices and desktop users [7], [8], [9], at present it is not well understood how the different techniques are chosen, how they compare to each other, and what are the optimal techniques to use in different contexts. Most importantly, the effect of these streaming techniques on Wi-Fi and cellular network usage and on the energy consumption of mobile devices is yet to be studied. Such knowledge is imperative before one can design an energy optimal streaming service.

For this reason, we study three popular video streaming services, YouTube, Dailymotion and Vimeo, in six different smartphones covering five major mobile platforms. We actively captured traffic of approximately 450 video sessions via Wi-Fi and 3G, and measured energy consumption during those sessions. Our main observations are the following:

- In general, Fast Caching and Throttling are applied by the server, whereas video players employ Bitrate and ON-OFF mechanisms by exploiting TCP flow control mechanism over Fast Caching and Throttling respectively. In Bitrate streaming, the player unintentionally triggers TCP flow control because the player has too small playback buffer compared to the amount of content the server offers. In ON-OFF, the player deliberately pauses and resumes download, and this technique is applied only in Android phones. (Section IV)
- Our analysis reveals that in smartphones different techniques are applied with little or no consensus: different techniques are used by different clients to access the same service in the same context. For example, An-

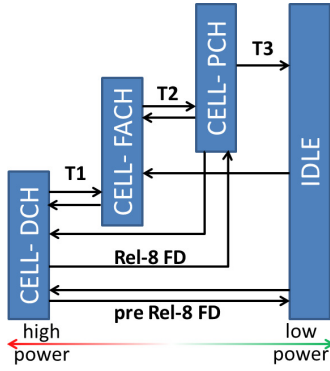


Fig. 1. 3G RRC state machine with CELL_DCH, CELL_FACH, CELL_PCH states.

droid devices use three different techniques for YouTube videos. The selection of those techniques depends on the quality of the video and the player. However, the strategy selection does not depend on the wireless interface being used for streaming and thus, network operators do not play any role. (Section IV)

- When watching a complete video, Fast Caching is the most energy efficient technique for both Wi-Fi and 3G access. If the user is likely to interrupt the video viewing, ON-OFF streaming is more attractive, but the ON period duration should be adjusted to match Fast Start period in order to avoid server rate throttling. However, none of the identified techniques is optimal because they do not adapt to the wireless access technology, user behaviour, and preferences. (Section V-C)
- There is a large variation in playback energy consumption between different types of player and containers on the same device. The differences are due to inefficient player implementation. However, the video quality (resolution) does not seem to have a large impact. (Section V-A and V-B)

We structure our paper as follows. In the next section, we briefly describe the energy consumption characteristics of wireless communication in smartphones and explain how mobile streaming services work. In Section III, we describe our methodology. In Section IV, we investigate the different streaming techniques. Section V is devoted to presenting the results from the energy consumption measurements. In section VI, we outline the potential future work in mobile multimedia streaming. Finally, we contrast our work with earlier research in Section VII before concluding the paper.

II. BACKGROUND

Smartphones allow users to access Internet via Wi-Fi and mobile broadband interfaces. Currently WCDMA/HSPA is the most widely deployed mobile broadband interface in mobile phones. LTE is the upcoming mobile broadband technology which is already available in certain markets. In this paper we focus on WCDMA/HSPA and refer to it as 3G. The power consumption of these interfaces can be very high even though there are already existing standard mechanisms for power saving. The power saving mechanisms of the radio layers use protocols and state changes independently without

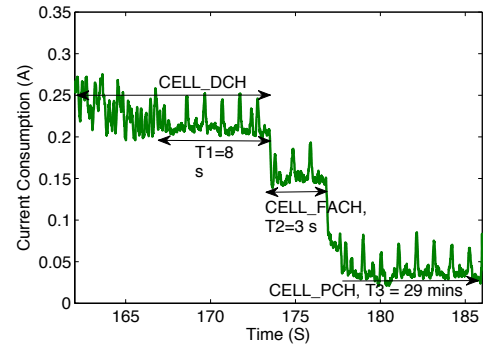


Fig. 2. Current consumption at different states and state transitions with Lumia 800.

any knowledge about the applications being used in mobile devices. In this section, we first review the power consumption characteristics of Wi-Fi and 3G network interfaces that we use in this study. Then, we explain the typical characteristics of mobile streaming services.

A. Power Saving Mechanisms for Wi-Fi

Wi-Fi interface of a smartphone can operate in power saving mode (802.11 PSM). PSM allows the client to switch the Wi-Fi radio into sleep mode. The client periodically wakes up after every 100 ms to receive a traffic indication map (TIM) message, often called beacon, from the access point (AP). The beacon tells the client whether the AP has buffered data for it. If so, the client sends PS-Poll frame to the AP. Otherwise, the client goes back to sleep until the next beacon. Modern devices usually implement a timer which keeps the interface in idle state for a few hundred milliseconds after the transmission or reception of packets, which improves especially the performance of short TCP connections.

B. Power Saving Mechanisms for 3G

3GPP standards specify how to control radio resources in such a manner that mobility and mobile device power consumption can be optimized. From power consumption point of view, the radio resource control and states and transitions between states must be understood. Figure 1 shows the states and the inactivity timers in 3GPP RRC protocol. These timers are used by the 3G radio network to control the transitions among different states. Network configuration of these timers has impact on radio resource usage, power consumption, and user experience. The actual data transmission happens in CELL_DCH state and a mobile phone switches from CELL_DCH to the other states in absence of data transmission. If the mobile device and network both support standard Fast Dormancy (FD) [10], CELL_DCH→CELL_PCH transition happens. For non standard FD, the transition is CELL_DCH→IDLE (Figure 1) which releases the RRC connection altogether.

RRC protocol has a large impact on energy spending and the power consumption in different states varies a lot. Figure 2 shows that average current consumption in CELL_DCH is 200 mA, in CELL_FACH is 150 mA, and in CELL_PCH is 50 mA approximately. The figure also shows that when T1 expires the state transitions to CELL_FACH and so on.

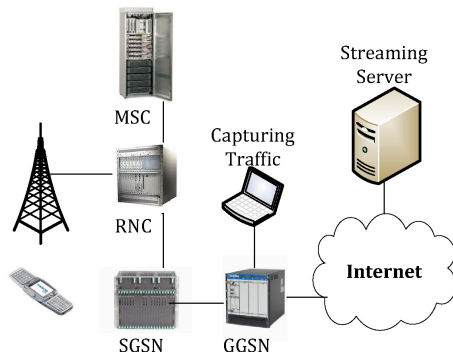


Fig. 3. Collecting video traffic from 3G network.

These timers are operator controlled. They have static values and there is no standard procedure on deciding their values. Network operators use different configurations in their radio network depending also on the capabilities of the network equipment provided by vendors. Inactivity timer settings in live networks vary from few seconds up to tens of seconds. The potential consequence especially with long inactivity timers is high power consumption at the mobile device. To learn more about different cellular network configurations and their effect on energy consumption, readers can follow [11].

C. Mobile Video Streaming

Today mobile streaming services deliver data using HTTP over TCP. Smartphone users can access these services using either a native app or a browser which loads a Flash or HTML5 player in the beginning of a streaming session.

The quality of the video played is often denoted with a p-notation, such as 240p, which refers to the resolution of the video. 240p usually refers to 360x240 resolution. Different services use also low, standard, and high definition (LD, SD, HD) notations but the resolutions that each one refers to varies between services. Therefore, we define 240p videos as LD, 270-480p videos as SD and 720-1080p videos as HD.

A common feature of all streaming services is an initial buffering of multimedia content at the client, which tries to ensure smooth playback in the presence of bandwidth fluctuation and jitter. This buffering is visible to user as start-up delay and referred to as Fast Start. The name comes from the fact that this initially buffered data is typically downloaded using all the available bandwidth, while the rest of the video is downloaded using one of the techniques; i) Bitrate streaming, ii) Throttling, iii) ON-OFF, iv) DASH, and v) Fast Caching. Fast Caching is similar to Fast Start, the only difference is that Fast Caching lasts longer until the whole content is downloaded.

III. METHODOLOGY

We studied video traffic from three popular streaming services, YouTube, Vimeo and Dailymotion, in six different smartphones covering all major mobile platforms. Most of these services have their own apps for the smartphone users. For example, YouTube app exists in five platforms, whereas Vimeo and Dailymotion do not have any app for the legacy Symbian and Meego platforms. In Android phones, Nexus S

and Galaxy S3, we used both the app and browser to access YouTube videos. In browser, the desktop edition of YouTube was used as it provides the opportunity to use both Flash and HTML5 players. We also studied the video services in iPhone 4S and Lumia 800 which run iOS and WP 7.5 respectively.

In Table I, we list the available players, as App, Flash or HTML5, in smartphones for our target video streaming services. The default container for these players are mp4, x-flv and webm respectively. The native app plays also 3gpp videos. Whenever available for the particular smartphone and player, we streamed videos of multiple qualities that range from 240p to 1080p. The average duration of the videos was 10 minutes.

We streamed the videos from our target video services to the different devices via both Wi-Fi and 3G. We used a DLink DIR-300 wireless AP supporting 802.11 b/g (54Mbps), which was connected to the Internet via 100 Mbps Ethernet. We captured Wi-Fi traffic using a PC equipped with a traffic capturing tool called AirPcap [12] in monitoring mode.

We performed 3G measurements in two scenarios. In the first scenario, we used an HSPA test network which was provided by Nokia Siemens Networks. The network was configured according to vendor recommendations. The most relevant parameters for this study are the inactivity timers ($T1 = 8$ s, $T2 = 3$ s and $T3 = 29$ min) and enabled CELL_PCH state. We captured traffic of the streaming clients from the Gn interface i.e. between SGSN and GGSN according to Figure 3. The downlink capacity of the 3G subscription was 6 Mbps. The results of these measurements are presented in Section IV. The second set of measurements were also conducted in the lab environment but with the limited bandwidth and discussed in Section V-E.

Considering all the combinations of video services, video quality, smartphones, wireless interfaces and experiment settings, there were approximately 150 actual streaming sessions. Again every streaming session was repeated atleast three times in order to filter out any measurement anomalies. Consequently, we had measurement results for 450 streaming sessions. Since we are interested in popular video services in the Internet, in this way we also verified that a particular traffic pattern was not the result of an anomalous behavior of the client or server.

In order to understand the energy consumption characteristics, we measured the current draw of the smartphones during the streaming sessions. We used two instruments: Monsoon Power Monitor [13] and another custom power monitor. One of these was attached to the phone to measure the current consumption during a complete video playing period. We detached the phone batteries and powered the phones directly from the power monitor (Monsoon) or from an external power supply. During the power measurements all the devices were in automatic brightness settings.

IV. TRAFFIC ANALYSIS TO INFER STREAMING TECHNIQUES

From traffic traces we inferred manually the type of streaming technique used for each of the different combinations of device, service, stream quality, player type, and access network type. These findings are summarized in Table I and discussed below.

TABLE I

STREAMING TECHNIQUES FOR POPULAR VIDEO STREAMING SERVICES TO MOBILE PHONES OF FIVE PLATFORMS. THE SELECTION OF A STREAMING TECHNIQUE DOES NOT DEPEND ON THE WIRELESS INTERFACE BEING USED FOR, RATHER DEPENDS ON THE PLAYER, VIDEO QUALITY, DEVICE AND THE VIDEO SERVICE PROVIDER.

	N9 (Meego)	Nokia701 (Symbian)	iPhone-4S (iOS 5.0)	Nexus S(Android-2.3.6)		Galaxy S3(Android-4.0.4)		Lumia800 (WP7.5)
YouTube Streaming Wi-Fi&3G	(App) Bitrate	(App) Bitrate	(App) Throttling Factor=2.0	(Flash) Throttling Factor=1.25	(App& HTML5) ON-OFF	(Flash) Bitrate(HD), Throttling(<HD) Factor=1.25	(App& HTML5) ON-OFF-M	(App) Fast Caching
Quality	LD(240p), SD(270p)	LD(240p), SD(270p)	LD(240p), SD(360p), HD(720p)	LD(240p), SD(360,480p)	LD(240p), SD(360p)	LD(240p), SD(360,480p), HD(720,1080p)	LD(240p), SD(360,480p), HD(720p)	LD(240p), SD(270p)
Container	mp4(270p) 3gpp(240p)	mp4(270p) 3gpp(240p)	mp4(360,720p) 3gpp(240p)	xflv	mp4(360p) webm(360p) 3gpp(240p)	xflv	mp4(>240p) webm(>240p) 3gpp(240p)	mp4(270p) 3gpp(240p)
Vimeo Streaming Wi-Fi&3G	(App) Bitrate	(Flash) Fast Caching	(App) DASH	(App) ON-OFF		(App) ON-OFF		(App) Fast Caching
Quality	SD(270p)	SD(270p)	SD(270,480p), HD(720p)	SD(270p)		SD(270p)		SD(270p)
Container	mp4	mp4	mp4	mp4		mp4		mp4
Dailymotion Streaming Wi-Fi&3G	(Flash) Throttling Factor=1.25	-	(App) Throttling Factor=1.25	(App) ON-OFF		(App) Fast Caching(288p), ON-OFF(>288p)		(App) Throttling Factor=1.25
Quality	SD(288p)	-	LD(240), SD(288,480p)	SD(270p)		SD(288,480p),HD(720p)		SD(288p)
Container	mp4	-	mp4	mp4		mp4		mp4

A. Bitrate Streaming

Bitrate technique is used to deliver content at the encoding rate of the stream. A streaming session begins with Fast Start and then the player receives content at the encoding rate from the server. This rate control is applied by a streaming client. YouTube players in N9, Nokia 701 and Galaxy S3 and the Dailymotion player in N9 use Bitrate streaming technique. In these cases, player's playback buffer and TCP receive buffer are filled during Fast Start. During playback the player decodes content at the encoding rate and the corresponding space is freed from the playback buffer. As a result the player draws data from TCP receive buffer at the encoding rate and TCP flow control mandates the sending TCP to follow this rate. The variation in TCP receive window size, in Figure 4, confirms that the downloading rate is controlled by the client.

B. Throttling

Throttling refers to the technique which delivers content to a client at a limited rate but which is higher rate than the encoding rate. The sending rate is controlled by the server. For example sending a 300 kbps stream at the rate of 600 kbps. Table I indicates that iPhone 4S, Nexus S and Galaxy S3 receive content at throttled rates from YouTube. Dailymotion also throttles data rate when sending content to Nokia N9, iPhone 4S and Lumia 800.

The rate at which the server sends the stream data depends on the case. Typically the client can tell the throttle factor (the ratio of throughput and average encoding rate) to the server in the URL (e.g. for YouTube algorithm=throttle-factor and factor = 1.25) or a service specific default throttle factor is used. YouTube and Dailymotion players commonly request 1.25 as the throttling rate while the iPhone player receives at a factor of 2.0 from the YouTube server.

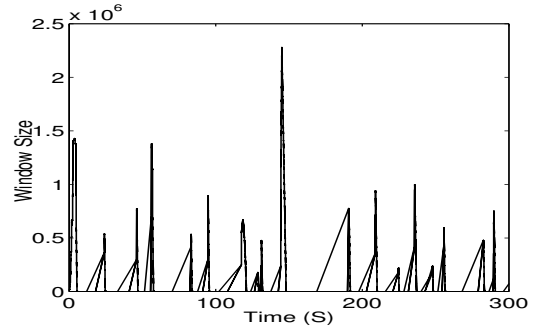


Fig. 4. TCP rwnd size (Bytes) advertisements during YouTube video streaming in Nokia N9.

1) *Single TCP connection*: In most cases we observed that a single TCP connection was used to deliver the throttled content. While throttling, server sends data to the client in small chunks. Figure 5 shows a CDF plot of the chunk sizes computed so that packets with interval shorter than 50 ms were grouped together in a chunk. We observed, like the authors in [14] and [7], that YouTube servers sends 64 KB chunks periodically to a Flash player in mobile browser. The chunk size for iPhone is 192 KB when streaming video of 720p quality. Furthermore, several chunk sizes seem to be used in the case of Galaxy S3. These chunks are separated from each other by few hundred milliseconds to 1.2 seconds which is shown in Figure 6. The interval decreases as the encoding rate of the video increases. This behavior is irrespective of using either Wi-Fi or 3G. However, this kind of burstiness was absent in Dailymotion and Vimeo traffic.

2) *Multiple TCP connection*: The YouTube app in iPhone uses multiple TCP connections when streaming HD quality videos. The player maintains a fixed playback buffer of 25

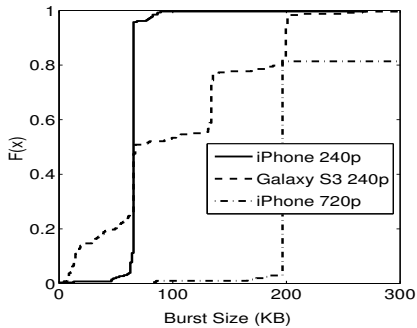


Fig. 5. CDF of the chunk sizes received by the clients using bitrate throttling from YouTube.

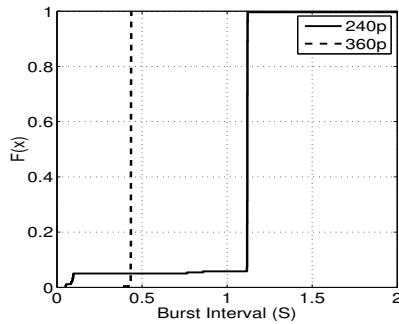


Fig. 6. CDF of the time intervals between two consecutive chunks in YouTube streams.

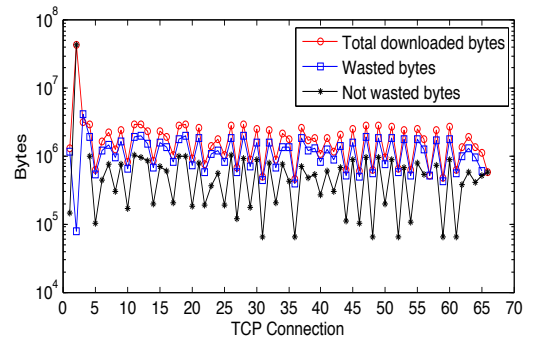


Fig. 7. iPhone 4S downloads the double amount of data from YouTube servers than the actual size of the content.

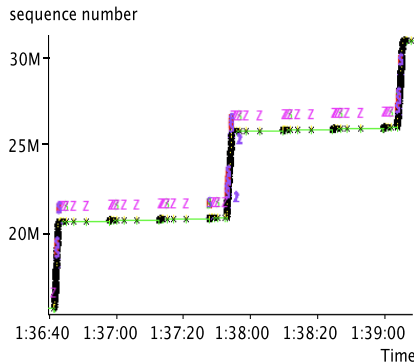


Fig. 8. Traffic trace for an ON-OFF streaming session to Nexus S.

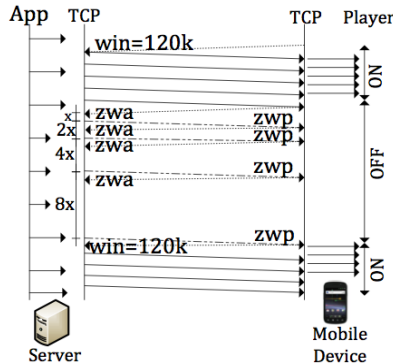


Fig. 9. TCP flow control messages (zwa and zwp) during an OFF period.

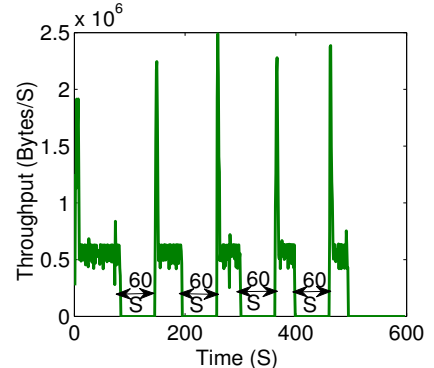


Fig. 10. In Galaxy S3, the YouTube app receives a video in multiple TCP connections.

MB and closes the existing TCP connection with the server whenever this buffer is filled. The player creates TCP connection with the server when some buffer space is freed. The time interval between two consecutive connections range between few hundred milliseconds to maximum three seconds.

We noticed, as did Finamore et al. [8], that the player receives more data than the actual video size (see Figure 7). By analysing traffic traces and video frames we identified that YouTube server sends media content from the beginning of a key frame for any partial content request. The player does not track the end position of the current key frame or the beginning of the next key frame beforehand and it may close the connection in the middle of a key frame transmission. Subsequently, the player makes every new request each time from the beginning of a key frame which was received partially for the previous request and all the data of that partially received key frame is wasted. Figure 7 shows that the consequence of this mismatch is significant data waste even though a user watches the complete video.

C. ON-OFF

ON-OFF technique is based on the playback buffer status of a player. When the player has enough content to play, it informs the server to pause the transmission. The server resumes data transmission only when the buffer falls below a threshold at the client.

1) *Single Persistent TCP Connection*: ON-OFF streaming generates a traffic pattern consisting of large bursts separated

by correspondingly long idle intervals. One traffic trace is shown in Figure 8. The bursts are easy to identify. Client player causes this pattern by reading TCP socket periodically. In between these reading events, the filled up TCP receive buffer and flow control make sure that the TCP sender at the server end must pause the transmission. The behavior is illustrated more clearly in Figure 9. In between bursts, because the TCP sender at the server side has data from the server application buffered to be sent, it sends zero size packets (zwp) in order to check whether the TCP receiver's buffer status has changed. The receiver replies with an ACK with zero window size (zwa). When the player application realizes that the amount of buffered stream drops below a certain threshold, it reads a burst of data from TCP socket again, which allows TCP to reset the receive window to a higher value and the sender to resume data transmission.

2) *Non-persistent TCP connections (ON-OFF-M)*: The YouTube app and HTML5 player in Galaxy S3 use multiple sequential non-persistent TCP connections to stream a video. We term this technique ON-OFF-M to distinguish from the use of a single persistent connection. The player maintains dynamic low and high level buffer thresholds. When the playback buffer touches the low level threshold the OFF period ends and the ON period begins by sending a new request using HTTP Range: bytes=X-Content-Length header. The player ceases downloading when the buffer reaches the high level threshold and the ON period ends. In this way, the player initiates TCP connections after every 60 second interval as shown in Figure 10.

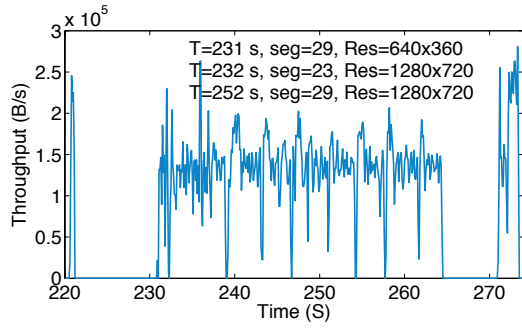


Fig. 11. iPhone 4S, using HLS, downloads the chunks of high quality and discards the downloaded lower quality content from the playback buffer whenever the player switches the video quality.

D. DASH

Using any of the streaming techniques, discussed so far, a client player can play a video of a single quality during a streaming session. It is possible to change the quality only by interrupting the playback. DASH, on the other hand, allows the player to switch the stream quality on the fly in order to adapt to bandwidth fluctuations. The Vimeo player in iPhone 4S uses Apple’s version of DASH called HTTP Live Streaming (HLS). The player receives content in chunks and it requests each chunk separately specifying the quality. We identified that a chunk contains 10 seconds of playback content.

From traffic traces we identified that the Vimeo player in iPhone always keeps 60 seconds playback content in the buffer when streaming via W-Fi. Whenever the player switches to a higher quality, this buffered content is wasted by the player for the sake of providing faster response to quality change. One such scenario is presented in Figure 11 where the player switches from 360p to 720p at 232 s and downloads from 23rd to 29th segments of 720p quality. In the case of 3G, the player wastes 20 seconds equivalent content. This observation can change with bandwidth variation. Although both ON-OFF-M and DASH exhibit bursty traffic behavior, in ON-OFF-M mechanism a streaming client receives content at server throttled rate, while iPhone 4S receives each burst using maximum bandwidth when applying DASH.

E. Fast Caching

Fast Caching refers to downloading the whole video using the maximum bandwidth. The client player decodes content at the encoding rate and at the same time maintains a large growing buffer. We found that Lumia 800 downloads YouTube and Vimeo videos and Nokia 701 Vimeo videos in this way. In the case of YouTube, the player uses `ratebypass=yes` parameter in the HTTP request to deactivate any rate control at the server side. For example, the YouTube player of Lumia 800 downloaded a 10-minute long 270p video within 120 seconds via 3G in our experiments.

F. Summary

Table I summarizes our findings on the usage of different techniques in different mobile platforms using the three video services. In our repeated experiments, we did not find any exceptions which confirms that these techniques are the results

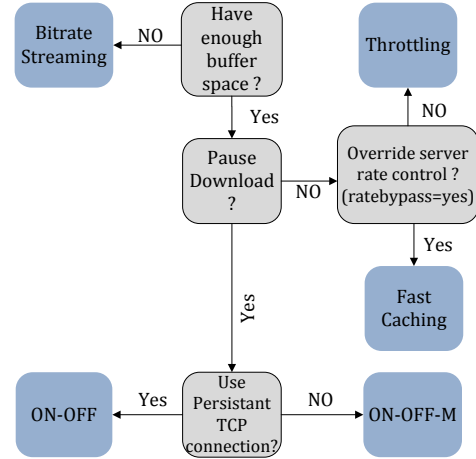


Fig. 12. The choice of a streaming technique by the client player.

of the described behavior of the video streaming services. Figure 12 illustrates how the client app behaviour leads to the choice of particular streaming technique. We sum up our main observations below:

- There is no systematic use of a given technique by a given streaming service. Neither is a particular technique tied to specific mobile platforms. Instead, the technique used depends on their combination plus the stream quality and the player type. It is notable that the used wireless interface does not influence streaming technique selection.
- Streaming servers use either throttling or Fast Caching to deliver video to mobile devices. The choice between these two is influenced by client player’s request. Some native mobile apps continuously pause and resume downloading leading to ON-OFF traffic pattern. ON-OFF-M uses non-persistent connections and each new byte range request begins with Fast Start followed by throttling phase if the requested range exceeds Fast Start phase. Bitrate streaming is the result of small playback buffer at the client buffer.
- The amount of data wasted by the YouTube player in iPhone is significant even when a user watches the complete video. This data waste could be a problem for users with quota based mobile broadband subscriptions. However, this problem could be solved with a smarter player implementation.
- ON-OFF mechanism with persistent TCP connection forces the sending TCP to pause sending data which keeps piling up in the send buffer. Many such clients simultaneously can generate significant memory pressure for a streaming server making it undesirable from service provider’s perspective.

V. STREAMING SERVICES AND POWER CONSUMPTION

We also measured the current consumed by the smartphones during the streaming sessions. The measured value was for the entire device but we separated the total current drawn into the average video playback and wireless interface current consumption. The playback current consumption includes decoding and display current. We can identify this current

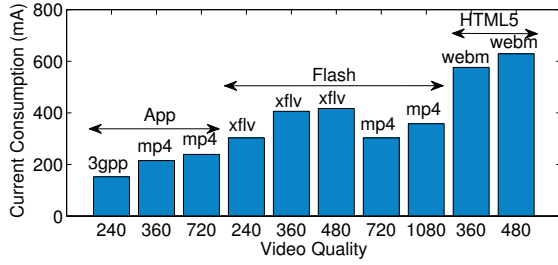


Fig. 13. Avg. playback current draw when streaming 240 – 1080p YouTube videos to the app and browser in Galaxy S3.

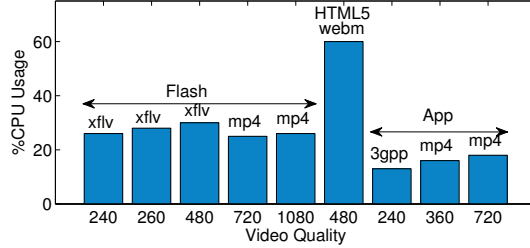


Fig. 14. Amount of CPU used by different video players in Galaxy S3 while playing different quality videos of different containers.

draw at the end of the power trace of each streaming session when the content has been fully delivered but playback still continues since some of content is always buffered at the end regardless of streaming technique used. During this time the Wi-Fi or 3G interface is in PSM sleep or CELL_PCH/IDLE mode. We computed the average wireless communication current, which we refer to as streaming current, by subtracting the average playback current from the average total current. The results presented in this section are the average of repeated measurements.

A. Impact of Video Quality, Player and Container

1) *Video Quality*: In Figure 13, we can see that playback current draw of Galaxy S3 increases as the quality of YouTube video increases as long as the same container is used. We also observed similar pattern for watching Dailymotion videos in iPhone 4S and Galaxy S3. It is logical that high quality videos have more information to present than low quality videos and, therefore, more current is drawn. However, in some cases even doubling the resolution adds a relatively small increment to the average playback current.

2) *Video Player*: In order to play YouTube LD, SD and HD videos, the browser loads a Flash player. Flash has support for different kind of codecs and containers, such as x-flv, mp4 and H.264. The browser loads HTML5 player to play webm videos. Figure 13 compares the energy consumption when using different players for streaming. It is noticeable that the native YouTube application consumes the least amount of energy. In contrast, browser-based players can draw even more than the double current compared to the app when playing the same video. We discovered that during playback the Flash player does not leverage any native system support to decode the video but consumes a significant amount of more CPU than the native application (see Figure 14). Although the HTML5

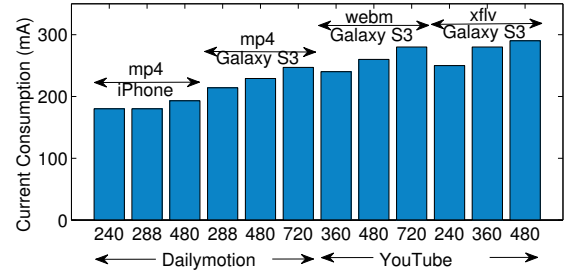


Fig. 15. Avg. playback current consumption while playing different quality videos of different containers.

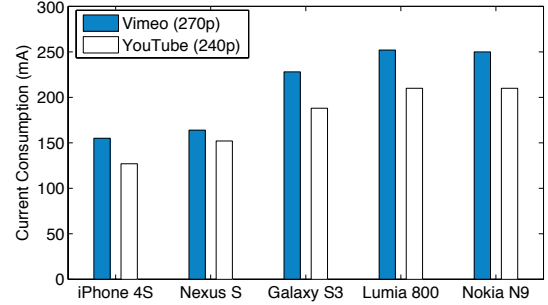


Fig. 16. Avg. playback current draw in different smartphones during the playback of a Vimeo and a YouTube video.

player takes native system support, it consumes 60% of CPU even during the playback of a 480p video. It seems that HTML5 player is required to go through further optimization to be used in mobile platforms.

3) *Video Container*: We have already shown how the videos of different quality and different players affect the energy consumption of smartphones. In Figure 13, we can see that playback of a 240p 3gpp video requires less energy than that of an x-flv video of the same quality. It is also illustrated that the same 240p x-flv requires more current than a 720p mp4 video. Although from Figure 13 we can infer that 3gpp is the least and webm is the most energy consuming containers, it is difficult to isolate the effect of the corresponding video containers since some videos can be played only using browsers. Besides, the energy consumption of the browser-based players are very high. Therefore, we downloaded some YouTube videos of x-flv and webm formats and then measured energy consumption during playback. The results are shown in Figure 15. This figure also illustrates that playback energy consumption does not change significantly when the quality of video changes with the same container category.

B. Display Variation

Our overall observation is that the average playback current consumption of the same video among multiple smartphones varies significantly, as expected. This variation could be caused by the display resolution, display type or display size. The resolution is unlikely to be the main cause because iPhone 4S has a higher resolution display than any other phones except Galaxy S3 and still iPhone is among the least playback current consuming devices. If we consider the display types, from

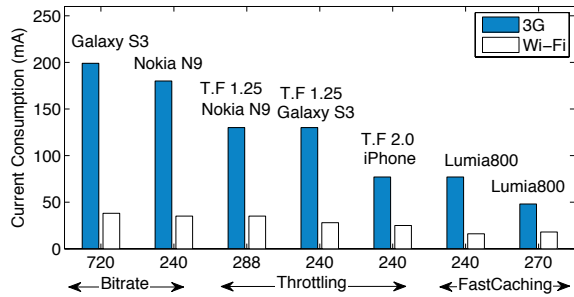


Fig. 17. Avg. streaming current consumption while playing different quality videos using different techniques in four different smartphones. 288p and 270p videos are from Dailymotion and Vimeo respectively. The rest of the videos are from YouTube.

Figure 16 we notice that iPhone with an IPS LCD display consumes least current and Nexus S which has a Super AMOLED display consumes a little bit more. Compared to these two, Galaxy S3 consumes more current with a Super AMOLED Plus. The reason can be the larger display. Contrastingly, Nokia N9 and Lumia 800 have smaller displays than Galaxy S3 but their current consumption is higher. Although the kind of content displayed on the screen has an impact on current consumption, from the above observations we can say that AMOLED display consumes more energy than the other kinds when viewing videos.

C. Impact of Streaming Techniques

In this section we discuss the effect of different streaming techniques on the energy consumption in smartphones. Since all the techniques are not available in a single platform, it is difficult to compare the energy efficiency of the techniques we identified. Therefore, we compare only the current consumed by the wireless interfaces of the smartphones and exclude the playback current in order to provide a comparison ground. We compare them in Figure 17 and 18.

1) *Bitrate Streaming*: In this case, the content is delivered continuously throughout the entire streaming session and the wireless interface is active all the time. For example, downloading a 6 minute video would require approximately six minutes. As a consequence, the average streaming current drawn by Galaxy S3 and Nokia N9 is very high for the YouTube videos. Figure 17 also shows that Galaxy S3 consumes around 30 mA for Wi-Fi and 200 mA for 3G (HD video using browser). The high current consumption of 3G is expected since the interface is constantly in the highest power consuming CELL_DCH state. However, power consumption over Wi-Fi is low with respect to the usage of the interface. It could be that the smartphones use some physical layer mechanism, such as dynamic modulation scaling, where power consumption of the interface is dynamically controlled according to the bit rate [5].

2) *Throttling*: When throttling is used by the server, the length of the video delivery phase depends on the throttle factor, which in turn determines how long the 3G or Wi-Fi radio will be powered on. Energy consumption for three throttled session is presented in Figure 17. The first one is of Nokia N9 for streaming from Dailymotion. The second

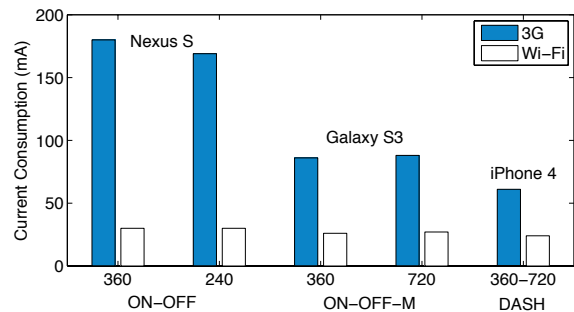


Fig. 18. Avg. streaming current consumption using ON-OFF, ON-OFF-M and DASH. Nexus S and Galaxy stream from YouTube and iPhone streams from Vimeo using the native applications. In each connection, Galaxy S3 receives content at a throttling factor of 2.0 after Fast Start.

session is of Galaxy S3, where the browser is used to stream LD videos from YouTube. Both exhibit comparatively high current consumption over 3G, but Galaxy S3 draws clearly less current than N9 when streaming over Wi-Fi. The reason is that unlike Dailymotion and Vimeo the YouTube server sends traffic in small chunks, as we explained in Section IV-B, and Wi-Fi interface manages to transition to sleep state in between the chunks due to the short timeout (200 ms). On the contrary, the 3G interface cannot leverage these small intervals because of the much longer inactivity timer values. In these two cases, throttle factor is 1.25 and we observe that current consumption is reduced by 50 mA over Bitrate streaming via 3G. The third case is shown for iPhone 4S which consumes even less current for both Wi-Fi and 3G. The obvious reason is the faster download at twice the encoding rate. Therefore, throttling delivers energy savings over Bitrate streaming as interface usage time is reduced to half of the total video duration.

3) *ON-OFF*: Figure 18 shows that Nexus S consumes very little current for streaming via Wi-Fi. In contrast, it uses a lot over 3G and almost is in the level of Bitrate streaming as shown in Figure 17. The reason is the use of a single persistent TCP connection which sends the TCP zero win advertisements and probes constantly and keeps the 3G interface in CELL_DCH state all the time. From traffic traces we computed that the maximum interval between the packets can be 5 seconds, while the 3G's T1 timer is set to 8 s. Wi-Fi can sleep most of the intervals in between these control packets. The same figure also includes a case where Galaxy S3 applies the ON-OFF-M technique using multiple TCP connections in between which there are no packets exchanged. Even though in both cases energy is saved over Wi-Fi compared to encoding rate streaming, Galaxy S3 saves more than 50% in average current when streaming over 3G compared to Nexus S and other Bitrate streaming sessions.

From Figure 18, we also observe that Galaxy S3 consumes almost the same amount of current while streaming a 720p and a 360p video via 3G. Considering the bit rate of the streams, this result also reflects the fact that energy consumption in CELL_DCH state remains stable regardless of the data rate. However, if we compare iPhone with throttling (Figure 17) and Galaxy S3 with ON-OFF-M (Figure 18), it can be seen that ON-OFF-M does not outperform throttling in energy savings

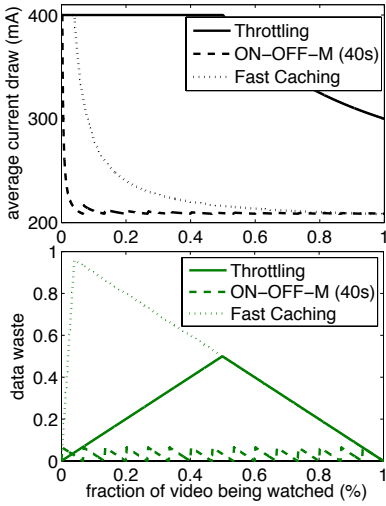


Fig. 20. ON-OFF-M provides the most attractive energy and data waste tradeoff for Wi-Fi access

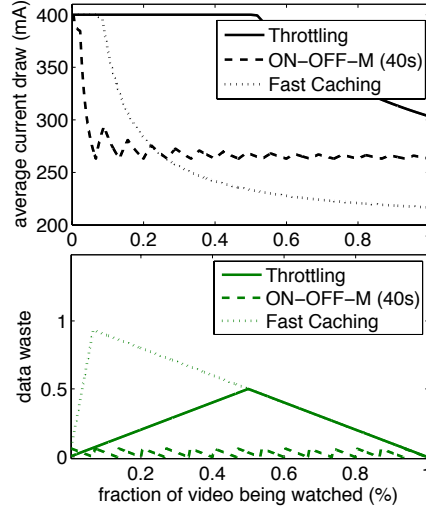


Fig. 21. Fast Caching is energy efficient over 3G access but potential data waste is high.

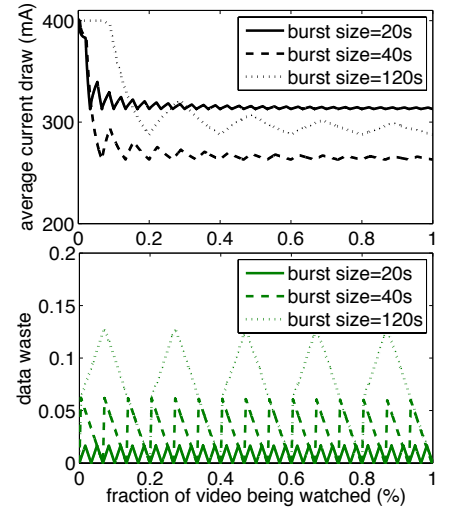


Fig. 22. Increasing burst size beyond Fast Start threshold increases average current.

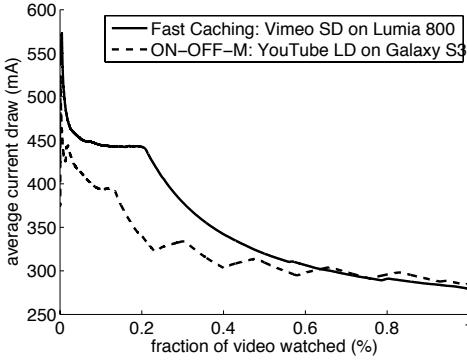


Fig. 19. Average draw of current as a function of viewing time for 3G access.

as the player receives data in each connection at the same throttled rate. Therefore, from the energy savings perspective both are of the same potential.

4) *DASH*: Only the Vimeo player in iPhone 4S uses HTTP rate adaptive streaming. The player receives video content as chunks of roughly 10 seconds which enables the wireless interfaces to consume less current. The Wi-Fi interface can sleep in between the chunks. In this case, iPhone consumes even less current than Galaxy S3 over 3G. The reason is that iPhone triggers Fast Dormancy after receiving a chunk which cuts the tail energy by half.

5) *Fast Caching*: Fast caching is used to download content at the client with as high throughput as possible. As a result the wireless interface is maximally utilized for as little time as possible. Figure 17 shows the current draw for two such example sessions in Lumia 800 for YouTube and Vimeo. We notice that Lumia 800 consumes less current than the other smartphones presented in Figure 17 and 18.

D. Energy vs. Possible Data Waste

Throttling, ON-OFF-M, and Fast Caching seem to consume the least energy compared to the other techniques. However,

if the user does not watch the whole video, the data is wasted and the energy is also wasted to retrieve that unwanted content. Furthermore, using the cellular access to download unnecessarily content is problematic for users having small quota in their data plan and for network resources.

For example, in [8] Finamore et al. analyzed YouTube traffic to desktop computers and iOS devices accessed via Wi-Fi and discovered that 60% of videos were watched for less than 20% of their duration. Therefore the impact of interrupted streaming sessions can be significant.

In Figure 19, we plot the average current draw for example cases of the aforementioned three techniques as a function of percentage of watched video computed out of the complete power traces. We see that interrupting the video watching early on would cause a hefty penalty in terms of wasted energy in both cases but the penalty gets smaller faster with the ON-OFF-M streaming making it a more attractive technique since it is common to not watch the video completely. To better understand the trade-offs, we derived simple models of the power consumption vs. data and energy waste. We assume constant power draw during download of data depends on inactivity timers, as explained in Section II. Based on observed power traces, we can assume constant power draw for 3G's CELL_DCH (during tail, no data transfer), CELL_FACH, and CELL_PCH states, and for Wi-Fi's idle and sleep states. The power values are extracted from the Galaxy S3 measurements (results would somewhat vary between devices). Wi-Fi is assumed to transition from idle to sleep mode after no packets have been observed during 200 ms. As for 3G, we use typical inactivity timer values ($T_1=8$ s, $T_2=3$ s, $T_3=29$ min) and assume no delay in state transitions.

Figures 20 and 21 plot the data waste and average current draw as a function of viewing time for three different strategies. Clearly, ON-OFF-M streaming is the most attractive for Wi-Fi from both energy and data waste point of view. However, given the much larger tail energy with 3G access, Fast Caching appears beneficial from the energy consumption perspective if the user watches the complete video. Intuitively,

using a larger burst size with ON-OFF-M technique would be useful. Figure 22 shows that it is not the case when server uses throttling. The reason is that if a burst is larger than what is delivered during the Fast Start phase (in this case 40 s worth of content), the rest is streamed at a server throttled rate which is less energy efficient, as we explained in Section V-C3. However, this limitation does not exist when server uses Fast Caching. In this case, higher burst size can be chosen and energy consumption can be reduced further with the increasing risk of more data waste.

E. Impact of Network Conditions

We obtained our results in “ideal” conditions in an isolated RF room. This approach was necessary because otherwise it would have been difficult to obtain comparable results. However, we also did a second set of measurement for most of the services and devices where we limited the available bandwidth close to the video encoding rate using a software based rate limiter. In this way we emulated a case of a loaded cell in 3G network. The main observation is that throttling and ON-OFF techniques are visible in the traffic pattern as long as available bandwidth is at least twice the encoding rate. After that if the bandwidth reduces further, different techniques start to resemble encoding rate streaming which is expected since there is no longer leftover bandwidth to leverage. Signal strength can also vary depending on the user’s location within a cell. We estimate that in such a case all the techniques would be penalized by having to use more transmit power and to amplify more the received signal. Such situation would hurt more the techniques that need to keep radio in rx/tx mode longer time meaning that the difference in average current consumption between the most and least energy efficient techniques would increase.

VI. LESSONS LEARNED AND FUTURE WORK

A. Streaming Techniques

The main lesson concerning the different streaming techniques is that none of the identified ones is able to provide a minimal energy consumption in all situations. In case of Wi-Fi access, ON-OFF provides the best trade-off between energy consumption and data waste given that users often interrupt streaming sessions. For 3G, Fast Caching is a competitive technique from the energy consumption perspective but not from the data waste perspective. An effective ON-OFF streaming technique should use non-persistent TCP connections and a burst size which does not exceed the amount of content delivered by the server during the Fast Start period in order to avoid server throttling phase. While server throttling could be a useful mechanism for the service provider to manage bandwidth demand and provisioning, it is undesirable from the clients perspective due to the increasing effect on the energy consumption. Bitrate streaming causes clearly the largest amount of energy consumption and should be considered as an unintended result of the client buffer being too small. DASH is similar to ON-OFF-M but higher chunk size would reduce energy consumption more.

LTE power consumption closely resembles 3G when discontinuous reception (DRX) is not supported. Then it exhibits

a long tail energy (e.g. 10s). If DRX is supported, the tail energy is reduced and the power characteristics begin to resemble more Wi-Fi. Consequently, ON-OFF-M would be most attractive option for DRX supported LTE and Fast Caching for certain situations where DRX is not supported.

An energy optimal streaming technique would adapt the download strategy to access network type and user behaviour, none of which the identified techniques do. Furthermore, a technique that also balances the data waste and energy consumption would consider user preferences. GreenTube takes the first step towards this direction [15] but, unfortunately it does not take the harmful impact of server throttling into account, and only considers local user viewing history in predicting the interrupt time which is also likely to depend on the content being viewed.

B. Player and Video Characteristics

We learned that using native apps is highly recommended because their implementations exhibit a much more optimized use of computational resources. Flash seems to be giving way to HTML5 which is likely to be an important technology in the future. Hence, optimizing the HTML5-based player implementations would be important future work.

According to our measurements, the video container/codec can have a significant impact on the energy consumption (3gpp seems more efficient than mp4), while video quality has a small impact. As a consequence, it makes little sense to try to reduce energy consumption by trading off video quality but the focus should rather be on choosing optimal container/codec. Even the impact on transmission energy is small in our measurements if the two video streaming sessions of different qualities use the same streaming technique. Having said that, an interesting question for future work would be to study how the user perceives different quality video streams on mobile devices. It is a waste of bandwidth to stream “too high” quality video when user detects no difference to a lower quality stream. Many factors play a role here, such as subjective opinion, video content, and display technology.

VII. RELATED WORK

The diverse nature of existing popular mobile streaming services and the resulting energy consumption characteristics have so far not been completely uncovered. Many papers have studied the energy efficiency of multimedia streaming over Wi-Fi and developed custom protocols or scheduling mechanisms to optimize the behavior. Examples of such work range from proxy based traffic shaping and scheduling to traffic prediction and adaptive buffer management [5]. However, streaming over 3G and the specific nature of the streaming services and client apps provide new challenges that these solutions cannot overcome. Balasubramanian et al. [16] studied 3G power characteristics in general and quantified the so called tail energy concept.

The most popular streaming services, especially YouTube, have been subject to numerous measurement studies in recent few years. Xiao et al. [17] measured the energy consumption of different Symbian based Nokia devices while using a YouTube application over both Wi-Fi and 3G access. A similar study was done by Trestian et al. [18] for Android platform. They

investigated energy consumption while streaming over Wi-Fi at different network conditions and studied the effect of video quality on energy consumption. However, these studies did not consider the details of traffic patterns and their impact on the energy consumption.

In a measurement study, Rao et al. [7] studied YouTube and Netflix traffic to different smartphones (iOS and Android) and web browsers accessed via Wi-Fi interface. They found three different traffic patterns of YouTube. In a similar passive measurement study, Finamore et al. [8] also analyzed YouTube traffic to PCs and iOS devices accessed via Wi-Fi and demonstrated that iPhone and iPad employ chunk based streaming. Qian et al. [19] explored RRC state machine settings in terms of inactivity timers using real network traces from different operators and proposed a traffic shaping solution for YouTube which closely resembles the ON-OFF streaming technique.

Liu et al. [20] studied power consumption of different streaming services. However, the scope of their study is considerably different from ours. They limit their study to streaming over Wi-Fi and performed experiments with only iPod, while we explored all the major mobile platforms and contrasted Wi-Fi with 3G. Our methodology is centered around fine grained power measurements with external instruments in controlled environment.

In contrast to these studies, we investigated which characteristics influence the choice of the streaming technique and quantified their impact on the energy consumption on different smartphones using both Wi-Fi and 3G. We also studied the effect of video qualities, video containers, players, and display types on playback energy consumption.

VIII. CONCLUSIONS

We analyzed the performance, and especially the energy consumption, of mobile video streaming. Based on measurements with six smartphones and three popular services we identified five different streaming techniques. The used technique depends on the service, client device, player type, and video quality. In general, we can say that in most cases the video streaming energy consumption is far from optimal. Specifically, our results demonstrate that none of the used stream delivery techniques is optimal but there are clear winners and losers. We pointed out several concrete suggestions on how the energy consumption, and also potential data waste, could be easily optimized using current techniques, also considering the player, codec, and video quality. Finally, our study paves the way for designing an optimized video streaming service.

ACKNOWLEDGEMENT

This work was supported by the Academy of Finland: grant number 253860 and The Future Internet Graduate School (FIGS).

REFERENCES

[1] YouTube Statistics. <http://www.youtube.com/yt/press/statistics.html>. July 2012.
 [2] Cisco visual networking index: Global mobile data traffic forecast update, 20112016, February 2012.
 [3] J.D. Power and Associates. 2012 u.s. wireless smartphone and traditional mobile phone satisfaction studies - volume 1, March 2012.

[4] Lei Guo, Enhua Tan, Songqing Chen, Zhen Xiao, Oliver Spatscheck, and Xiaodong Zhang. Delving into internet streaming media delivery: a quality and resource utilization perspective. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement, IMC '06*, pages 217–230, New York, NY, USA, 2006. ACM.
 [5] Mohammad A. Hoque, Matti Siekkinen, and Jukka K. Nurminen. Energy efficient multimedia streaming to mobile devices – a survey. *To Appear in Communications Surveys Tutorials, IEEE*, PP(99):1–19, 2012.
 [6] Thomas Stockhammer. Dynamic adaptive streaming over http –: standards and design principles. In *Proceedings of the second annual ACM conference on Multimedia systems, MMSys '11*, pages 133–144, New York, NY, USA, 2011. ACM.
 [7] Ashwin Rao, Arnaud Legout, Yeon-sup Lim, Don Towsley, Chadi Barakat, and Walid Dabbous. Network characteristics of video streaming traffic. In *Proceedings of the Seventh Conference on emerging Networking EXperiments and Technologies, CoNEXT '11*, pages 25:1–25:12, New York, NY, USA, 2011. ACM.
 [8] Alessandro Finamore, Marco Mellia, Maurizio M. Munafò, Ruben Torres, and Sanjay G. Rao. Youtube everywhere: impact of device and infrastructure synergies on user experience. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference, IMC '11*, pages 345–360, New York, NY, USA, 2011. ACM.
 [9] Jeffrey Erman, Alexandre Gerber, K. K. Ramadrishnan, Subhabrata Sen, and Oliver Spatscheck. Over the top video: the gorilla in cellular networks. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference, IMC '11*, pages 127–136, New York, NY, USA, 2011. ACM.
 [10] Fast Dormancy. Fast dormancy best practices. GSM association, network efficiency task force, 2010.
 [11] Matti Siekkinen, Mohammad A. Hoque, Jukka K. Nurminen, and Mika Aalto. Streaming over 3G and LTE: How to save smartphone energy in radio access network-friendly way. In *5th ACM Workshop on Mobile Video, MoVid'13*, pages 1–6. ACM, 2013.
 [12] AirPcap Nx. www.cacotech.com/documents/airpcapJan 2012.
 [13] Monsoon: www.msoon.com.
 [14] Shane Alcock and Richard Nelson. Application flow control in youtube video streams. *SIGCOMM Comput. Commun. Rev.*, 41(2):24–30, April 2011.
 [15] Xin Li, Mian Dong, Zhan Ma, and Felix Fernandes. GreenTube: Power optimization for mobile video streaming via dynamic cache management. In *Proceedings of the ACM Multimedia, ACM MM'12*, New York, NY, USA, 2012. ACM.
 [16] Niranjan Balasubramanian, Aruna Balasubramanian, and Arun Venkataramani. Energy consumption in mobile phones: a measurement study and implications for network applications. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference, IMC '09*, pages 280–293, New York, NY, USA, 2009. ACM.
 [17] Yu Xiao, Ramya Sri Kalyanaraman, and Antti Yla-Jaaski. Energy Consumption of Mobile YouTube: Quantitative Measurement and Analysis. In *Proceedings of the 2008 The Second International Conference on Next Generation Mobile Applications, Services, and Technologies*, pages 61–69, 2008.
 [18] Ramona Trestian, Arghir-Nicolae Moldovan, Olga Ormond, and Gabriel-Miro Muntean. Energy consumption analysis of video streaming to android mobile devices. In *Proceedings of the Network Operations and Management Symposium (NOMS), 2012 IEEE*, pages 444–452. IEEE, 2012.
 [19] Feng Qian, Zhaoguang Wang, Alexandre Gerber, Zhuoqing Morley Mao, Subhabrata Sen, and Oliver Spatscheck. Characterizing radio resource allocation for 3g networks. In *Proceedings of IMC 2010*, pages 137–150, New York, NY, USA, 2010. ACM.
 [20] Yao Liu, Lei Guo, Fei Li, and Songqing Chen. An empirical evaluation of battery power consumption for streaming data transmission to mobile devices. In *Proceedings of the 19th ACM international conference on Multimedia, MM '11*, pages 473–482, New York, NY, USA, 2011. ACM.