

e-DASH: Modelling An Energy-Aware DASH Player

Benoy Varghese,^{*†} Guillaume Jourjon,[†] Kanchana Thilakarathne,^{*†} and Aruna Seneviratne^{*†}

^{*}University of New South Wales, Sydney, Australia

[†]Data61-CSIRO, Sydney, Australia

Email: firstname.lastname@data61.csiro.au

Abstract—Dynamic Adaptive Streaming over HTTP (DASH) is one of the most popular ways to stream videos at present. In this work, we propose a DASH player energy-aware plugin (eDASH) for mobile devices which help reduce the battery consumption of the device. The eDASH player utilises a novel bitrate and video brightness adaptation algorithm to determine the next chunk to download. This algorithm utilises an energy-aware QoE model which factors in power consumption of the device in conjunction with existing bitrate adaptation logic to determine the next chunk. We also propose a new DASH architecture which could be easily integrated with the existing one. Macro-benchmarking of energy consumption of a mobile device while streaming and playing back video is conducted to obtain energy profiles of various video qualities. This energy data is then used along with real world network traces to drive simulations to evaluate energy savings that could be achieved using eDASH. We observe that up to 45% energy savings could be achieved with minimal reduction in QoE. We also find that up to 80% data transfer savings could also be achieved with an eDASH client.

I. INTRODUCTION

Ever increasing capabilities of smart mobile devices, especially in the case of processing and display quality, have made smartphones an attractive option for users to consume video content. It is even possible to stream ultra-high-definition (UHD) videos on recent smartphones. As a result, mobile video is driving the exponential growth in mobile data traffic which has been predicted to exceed 80% by 2020 [1]. Moreover, UHD will account for 20.7% of total video on demand (VoD) traffic [1]. Mobile network operators and vendors are utilising every tool at hand to support this growing capability crunch both by increasing the capacity using current technologies and by upgrading their infrastructure with higher bandwidth technologies such as LTE.

On the other hand, delivering the best Quality of Experience (QoE) for multimedia traffic has long been one of the main objectives in both networking and multimedia research communities. As a result, we now have efficient architectures such as MPEG Dynamic Adaptive Streaming over HTTP (DASH) [2] or Apple HTTP Live Streaming (HLS) [3] capable of delivering, over HTTP, video streaming at a rate close to network capacity. As such, these mechanisms aim to maximise QoE under constraints from the network. However, these algorithms have been developed with a desktop client in mind and therefore do not integrate well with constraints of mobile devices such as energy constraints.

The energy problem is exacerbated by the fact that battery technology is not keeping up with the rapid advances in the capabilities of other device subsystems like displays and processors. This has prompted reduction in power consumption of mobile devices by other means such as energy efficient networking protocols [4] [5].

In this paper, we propose e-DASH, a pluggable component to the original adaptive video streaming DASH library that allows users to integrate energy constraints of the device in the optimisation of QoE while streaming videos. e-DASH architecture builds upon the experimental validation of the hypothesis that the higher the quality of the video, the larger the energy consumption of the device, not only for steaming the video but also for playback of the video. Moreover, we have also observed that display brightness significantly increases energy consumption. In light of this, we propose e-DASH that adaptively streams different quality video chunks and/or video chunks with different brightness levels that are pre-processed and stored in the media server according to the energy-related context parameters of the device, such as remaining battery capacity. The paper makes the following contributions:

- We experimentally profile the energy consumption of streaming video encoded at different brightness levels on modern smartphones.
- We propose a new Quality of Experience Model factoring in energy consumption for HTTP-based adaptive streaming application.
- We introduce a new architecture, e-DASH for HAS, which could be implemented seamlessly with existing architecture and with minor additions to DASH protocol.
- Using experimental data driven simulations with real network traces, we show that approximately up to 45% of device energy consumption for streaming videos can be saved with minimum reduction in overall QoE of the end user.

The remainder of this article is organised as follows. Section II motivates the need for the incorporation of energy constraints in bitrate adaption mechanisms. Section III presents the design principles and optimisation of our solution. In Section IV, we evaluate our proposal and show that we can save up to 45% of battery life in video streaming. Sections V and VI discuss open issues and related work.

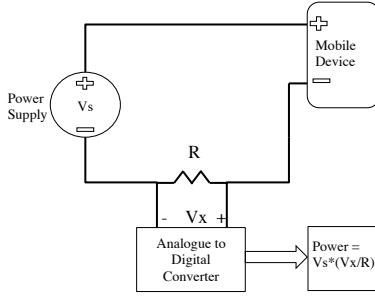


Fig. 1. Power measurement setup

II. BACKGROUND AND MOTIVATION

A. Adaptive Video Streaming

To achieve HTTP-based Adaptive Streaming (HAS), different versions of the original content that is to be delivered is created. Each version has a different bitrate and resolution. The encoding bitrates recommended by YouTube for video streaming are 8, 5 & 1 Mbps for video resolutions of 1080p, 720p and 360p respectively for frame rates upto 30 fps. Each version of the video is then split into smaller chunks (or segments), typically of 2-4 seconds in length. These chunks are then stored on a web server which can be accessed by clients on demand using simple HTTP GET requests from a video streaming client running one of the HAS algorithms. For each video, the relationships between the chunks and video information are stored in a manifest file. A user trying to stream a particular video is first presented with this manifest file. The video player used at the client-side uses information from the manifest file and the current network conditions to achieve adaptive streaming using MPEG-DASH standard is known as DASH player [2].

A DASH player utilises dynamic bitrate adaptation based on current network conditions to decide which chunk to download. Typically, bitrate adaptation algorithms are of two types: (i) *rate-based* and (ii) *buffer-based*. A *rate-based* bitrate adaptation logic works by measuring the network throughput and using that information to predict the future state of the network and decide which quality to pick next. A simple *buffer-based* system on the other hand, uses the current state of the playback buffer to decide on the quality of the chunk to download next. In the simplest form, the bitrate adaptation algorithm tries to download the highest possible quality chunk without congesting the network and to provide the best video quality possible to the displayed to the end user.

B. Quality of Experience

Traditionally the QoE of an end user in a telecommunication network was measured using a Mean Opinion Score (MOS) metric¹. For multimedia applications, MOS provides a numerical quantification of the user perceived quality of the media consumed. Users taking the test gives score value ranging from 1 to 5 for the least perceived quality to the highest

perceived quality respectively for the media they consume. The MOS score is the arithmetic mean of the scores from all the users who undertake the test. Psychological experiments are conducted in accordance with ITU specifications to obtain the subjective MOS for video quality [6].

The QoE for video streaming and playback is primarily dependent on the following factors [7]: (i) average video quality, (ii) number of re-buffering events, (iii) number of quality switches, and (iv) playback startup delay. Typically, DASH players aim to achieve a tradeoff between the number of re-buffering events to number of quality switches while maximizing the average video quality. A high average quality is considered to increase the QoE for the end user, while both re-buffering events and quality switching reduces it. For a given network condition, these are conflicting factors. For example, in a bandwidth constrained network, increasing average video quality while avoiding quality switching would result in more data being transferred which in turn causes re-buffering. Reducing the number of re-buffering events by switching qualities would reduce the average video quality as well as number of quality switches. Hence goal of a good DASH player is to find a tradeoff among these factors such that the QoE is maximised.

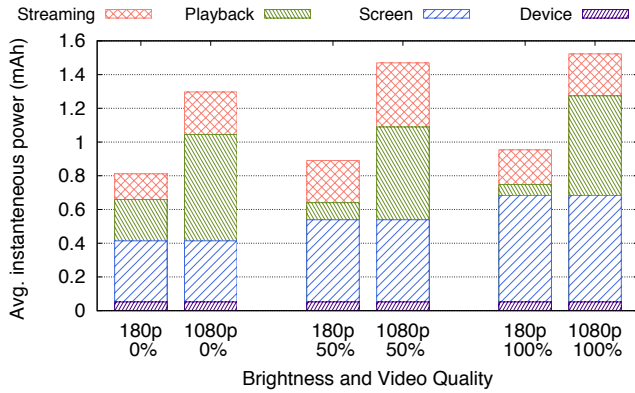
However, traditional QoE models do not consider the fact that different users have different device displays and the way content is displayed on their screen has a significant impact on the QoE [8]. Moreover, with advancements in display technology, we see that mobile devices these days are able to display high-definition videos with vibrant colours and high sharpness. One of the drawbacks of these high quality video displays is that the battery life of these devices is being adversely affected. This is because the battery technology has not advanced at the same pace as other mobile device component technologies. These high-resolution displays combined with the increasing popularity of mobile video streaming, has resulted in these devices running out of battery much before the end of the day. If the users do not have access to a charger they end up with a device they cannot use for several hours until they reach home. This unavailability of the device has never been captured in QoE modelling in the past. To this end, a more complete QoE model for video streaming and playback would capture display parameters as well as energy consumption parameters.

C. Quantifying Power Consumption of Video Streaming

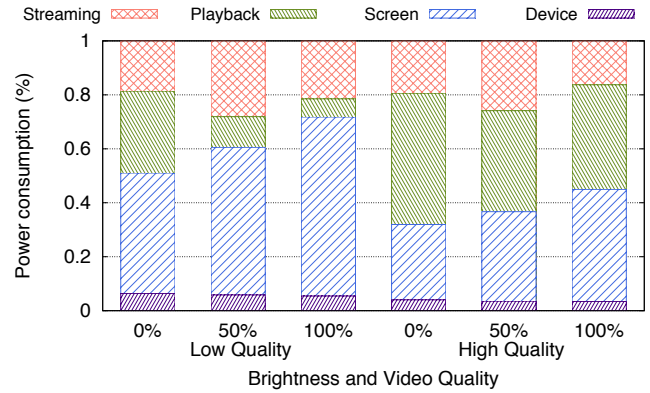
To understand the power consumption profile of a mobile device while streaming various video qualities, we performed a controlled set of experiments to quantify the percentage of power that is consumed by the video streaming and playback compared to the rest of the system.

Experimental setup: We used an Android smartphone (Samsung Galaxy S3-GT-I9300) that runs Android 4.1.2 and VLC Android version 2.0.3. Different qualities (1080p, 720p, 360p, 180p) of the same video are hosted in a local Apache web server that support DASH requests. The video used is “Big Buck Bunny” by Blender Foundation [9]. The length of the video is 9 mins and 56 seconds. The videos were encoded

¹<http://www.itu.int/rec/T-REC-P.800-199608-I/en>



(a) Instantaneous power consumption



(b) Percentage contribution of each subsystem

Fig. 2. Portion of contributions by different functionalities.

at the bitrates recommended by YouTube for their streaming service.

For power consumption measurements, smartphone was connected to an Agilent E3630A power supply to provide a constant 3.9V. We utilised a power supply unit instead of hijacking the battery to isolate the actual power consumption from the voltage drain characteristics of the battery pack and supply a constant voltage to the device. A known, low resistance $R = 0.5\Omega$ was used along with NI USB 6008 ADC to calculate the current drawn by the circuit by measuring the voltage V_x across R as shown in Fig. 1. The power consumed by the smartphone was given by $P = (V_s - V_x)I$, where V_s was the voltage provided by the power supply and I was the current through the resistor R , where $I = V_x/R$. The sampling rate used was 4 Hz. Once the power data was logged, we removed the first eight and last eight samples of the measurement to avoid the additional power consumption due to the touchscreen input.

Experimental procedure: Measurements were taken in four stages as follows:

- (i). The baseline power consumption of the device was measured when the device is in standby mode with screen, network connectivity and all background processes switched off for ten minutes.
- (ii). Then, the display was turned on with brightness at 0% with home screen displayed for ten minutes. The baseline power measurement was subtracted from this measurement to obtain the background power consumption of the operating system.
- (iii). Next, videos of various qualities were played back from the phone storage at various screen brightness levels to measure the playback contribution.
- (iv). Finally, the same videos were streamed via WiFi from the local server to measure the power consumption of the networking related components by subtracting the power consumption for all previous stages.

Results: Fig. 2 illustrates the average instantaneous power consumption for different functionalities with three screen

TABLE I
EXPECTED BATTERY LIFE.

Quality	Brightness	Battery life (hours)
Low Quality	100%	9.83
	50%	8.96
	0%	8.37
High Quality	100%	5.24
	50%	5.43
	0%	6.15

brightness levels. To highlight the amount of power that can be saved by reducing the quality if the video, results for the two extremes (highest quality and lowest quality) are shown. Even with the 100% screen brightness, approximately 40% of power can be saved by reducing the quality to the lowest level. Note that the significant portion of the power reduction is not due to the reduced amount of data streamed as expected, but during the playing back of the low quality video.

This fact is further depicted in Fig. 2b, where playback always contributes to more than 50% of power consumption irrespective of the brightness level. In contrast, for the low quality video, more than 50% of power is consumed for keeping the screen turned on. Surprisingly the amount of power consumed for streaming are comparable among all scenarios. One reason for that is the high throughput of the local WiFi connectivity. If there are network bottlenecks during streaming process, even greater power savings can be expected.

Majority of users measure power consumption of smartphones in terms of *time to recharge*. TABLE I interpolates the total battery life for different scenarios. It shows that battery life can be extended by 4 hours by reducing the quality level even if the user continuously stream videos throughout the day. Thus, these results experimentally validates our hypothesis of impact of video quality in power consumption of smartphones.

III. NEW HAS ARCHITECTURE FOR E-DASH

A. General Architecture

In traditional HAS architecture, as explained in Section II, original video is split into multiple quality levels before being

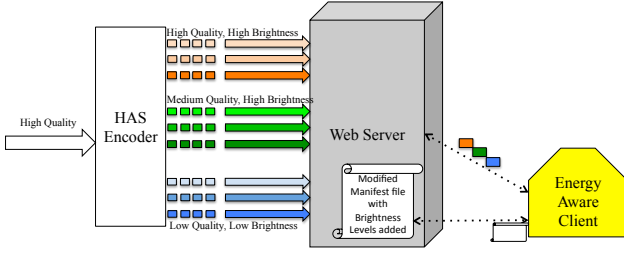


Fig. 3. Modified HAS Architecture.

served to the end user. The primary design goal was to achieve the ability to adapt to network conditions. This is possible because video chunks of lower qualities would have lower sizes and hence can be easily sent over a network constraint system. Since the end users energy constraint was never factored into the design, all recent studies to make systems energy efficient were done by making modifications to client side rather than rethinking the architecture design [10] [11].

Our modified HAS architecture factors-in the possibility of an energy constraint client. This is done by adding an extra layer of brightness information to each quality level. For example, for a video that is split into three quality levels, we define three additional brightness levels as well, resulting in a total of nine levels to choose from for each chunk. This new information is stored in the manifest file for the video. A client that wants to stream that particular video would make the bitrate adaptation decision based on not only the video quality, but also the brightness information. The modified HAS architecture is as shown in Fig. 3.

On the client side, our solution enhances any DASH video player with an energy aware plugin. This plugin takes as an input the result of the bitrate adaptation algorithm and the manifest file to choose the series of chunks that would optimise the Quality of Experience while minimising the energy consumption. This process is detailed in the remainder of this section.

B. Power optimization Models and solution

As explained above, our main goal is to minimise the power consumption while keeping the QoE for the end user maximum. We assume that once the DASH player has picked a video resolution as per its default bitrate adaptation algorithm, our energy plugin cannot pick a quality with an higher bitrate. This condition ensures that the network performance is not affected by our power optimization algorithm.

1) *Energy Model:* Without loss of generality, we define our optimisation problem as, the maximization of the QoE of end user while minimising the energy consumption.

Unlike traditional HAS systems, we define that a chunk is characterized by two properties: Bitrate, and Brightness. Hence, we assume that there exists a function, $f(\text{Bitrate}, \text{Brightness})$ which is used to define the Quality of Experience (qoe) of a given chunk. Also, we define a transition matrix T which maps the cost associated with transitioning from one type of chunk to another type of chunk.

TABLE II
TRANSITION MATRIX, T

	Hh	Hm	Hi	Mh	Mm	Ml	Lh	Lm	Li
Hh	0	0.1	0.2	0.4	0.5	0.6	0.8	0.9	1
Hm	0.1	0	0.1	0.5	0.4	0.5	0.9	0.8	0.9
Hi	0.2	0.1	0	0.6	0.5	0.4	1	0.9	0.8
Mh	0.4	0.5	0.6	0	0.1	0.2	0.4	0.5	0.6
Mm	0.5	0.4	0.5	0.1	0	0.1	0.5	0.4	0.5
Ml	0.6	0.5	0.4	0.2	0.1	0	0.6	0.5	0.4
Lh	0.8	0.9	1	0.4	0.5	0.6	0	0.1	0.2
Lm	0.9	0.8	0.9	0.5	0.4	0.5	0.1	0	0.1
Li	1	0.9	0.8	0.6	0.5	0.4	0.2	0.1	0

Hence, the decision for picking the next chunk is given by, *Maximize* QoE , such that total Energy consumption of the video ($\sum_n E$) is minimum.

Where,

$$QoE(k) = \frac{\mu}{n} \left(\left[\sum_{i=k-n}^{k-1} qoe(i) \right] + qoe(k) \right) - \alpha T_{(k-1),k} qoe(k) - \beta E_k qoe(k) \quad (1)$$

where, $(n-1)$ is the number of chunks already downloaded, $T_{(k-1),k}$ is a transition matrix of cost associated with transitioning from previous chunk $(k-1)$ to next chunk (k) , E_k is the energy cost associated with streaming/playing back a particular chunk, and $qoe(k)$ is the normalised Quality of Experience metric for a chunk of given quality. The qoe vector we use for our simulations is provided in Table III. The energy cost, E_k is based on measured power consumption values for both local playback and streaming video of various quality and brightness levels. This measurement was done experimentally, the results of which are summarised in Fig. 4.

In the following we will use the normalised transition matrix as shown in TABLE II. This transition matrix is created with the following logic: each step change in brightness corresponds to a penalty of 0.1 QoE, and each step change in quality corresponds to a penalty of 0.4. For example a transition from a chunk of High Quality with Medium Brightness (**Hm**) to another with Low Quality and Low Brightness (**Li**) would incur a transition cost factor of $0.4 + 0.1 = 0.5$.

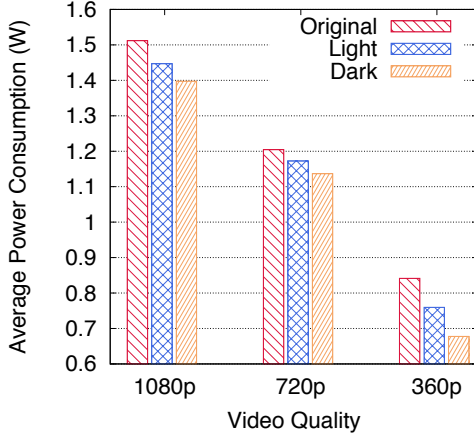
The problem can be later discretised by number of chunk as follows:

$$\text{Maximise } \sum_{i=1}^N QoE_i \quad s.t \quad (2)$$

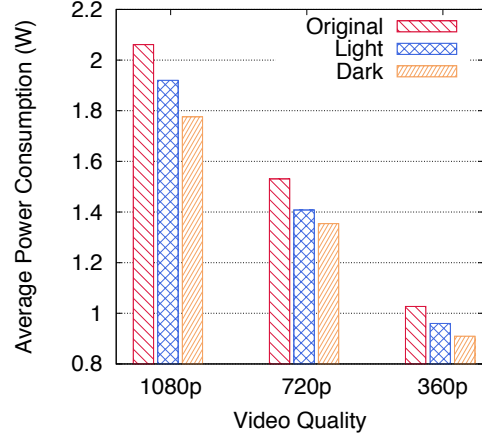
$$\sum_{i=1}^N E_i \leq E_{th} \quad (3)$$

Where, E_{th} is the energy threshold which is the maximum energy the client is allowed to utilise for a particular video.

2) *Optimization resolution:* In order to solve this maximization problem, we propose to reformulate it as a constraint flow optimization problem. In our scenario, we can represent this problem with Fig. 5. In this figure, each row represents a certain video characteristic in terms of bitrate and brightness



(a) Playback only



(b) Streaming with playback

Fig. 4. Power consumption comparisons for various luminance levels and video qualities for streaming video using VLC

TABLE III
QUALITY OF EXPERIENCE VALUES, $qoe(k)$

	Hh	Hm	Hi	Mh	Mm	Ml	Lh	Lm	Li
$qoe(k)$	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2

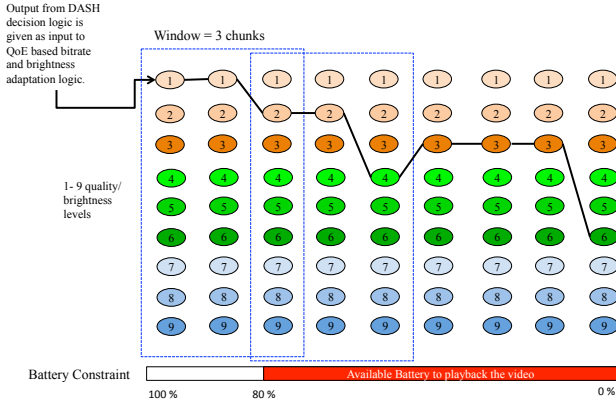


Fig. 5. Constraint Flow Optimization Problem.

(e.g. numbers 1 to 3 represent high bit rate and brightness varying from high to low) while each column is associated to a chunk number in the video. Therefore, we can solve our maximization problem by selected the path in the network shown in Fig. 5 that would both optimize the QoE while keeping the energy consumption under a predefined threshold.

In particular, we propose the following execution of our resolution algorithm. Let's define a horizon window of n chunks (e.g. $n = 3$ in Fig. 5). After either the download of n chunks or the detection of a network characteristic changes from the bit rate adaptation algorithm, which ever comes first, our algorithm will pick the n future chunk to be downloaded that would solve the optimization formulation. In addition, our algorithm is not allowed to pick chunk with a bit rate higher than the one given by the bit rate adaptation function.

Fig. 5 illustrates this algorithm as follows. At the beginning of the streaming, the original DASH algorithm estimate that the network can support the download of high quality as illustrated by the top-left input line in the figure. In our scenario, we have configured the algorithm with an horizon window of 3, therefore the algorithm would select the following 3 chunks, (High Quality-High brightness, High Quality-High brightness, High Quality-Medium brightness). At the end of this phase, network characteristics did not change and therefore our algorithm repeats itself taking the last selected chunk as origin and select the following chunks (High Quality-Medium brightness, High Quality-Medium brightness, Medium Quality-High brightness). Finally, this process will be repeated until the full video has been downloaded.

IV. RESULTS

To evaluate the energy aware QoE model, we simulated our energy aware DASH client (eDASH) and compared its performance to that of a normal DASH client. In particular, we define in the remainder of this section a normal DASH client as one that picks the best quality possible solely based on the varying network conditions at any given point in time.

A. Simulation setup

We use SimPy [12] to simulate the behaviour of the eDASH as well as the normal DASH clients. SimPy is a discrete-event simulation framework based on Python. The parameters used for simulation environment are shown in Table IV. The energy cost is based on power measurements of video playback and streaming for various quality and brightness values. To obtain these values, an experiment was done with a similar setup to that explained in Section II. The difference here is that we set the device brightness value to 100% for all the measurements, but we playback and stream videos encoded at different brightness levels. To encode videos with different brightness levels, we use *ffmpeg* and the encoding was done for 1080p, 720p, and 360p videos. The source video used was

TABLE IV
SIMULATION PARAMETERS.

Type	Value
Video Length	400 seconds
Chunk Length	4 seconds
No. of chunks	100
Bitrates	3000, 1000, 350 kbps
Window Sizes	3, 5
Battery threshold	0.7, 0.8, 0.9
Energy factor, β	0.5, 0.75, 1, 1.25, 1.5
Smoothness factor, α	1
Simulation Environment	SimPy

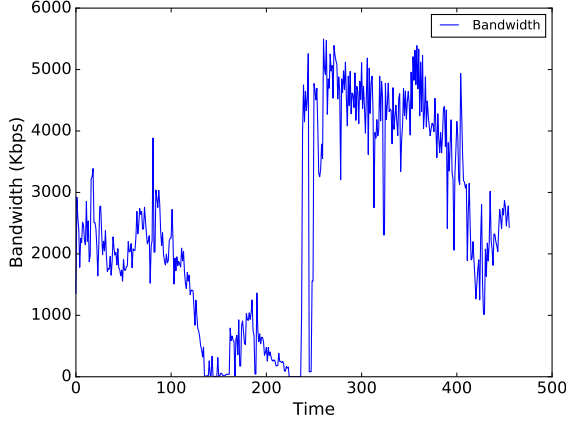


Fig. 6. Sample HSDPA network trace

an uncompressed raw version (*.yuv) of the Big Buck Bunny video by Blender Foundation [9]. Unless otherwise mentioned, we utilize these measured power values.

The simulation is driven using logs from TCP streaming sessions in a 3G/HSDPA network in Norway [13]. One of the sample network traces we used in our simulation is shown in Fig. 6. We assume both the DASH players have infinite playback buffer. This is done to avoid the buffer mechanics influencing the streaming mechanism. Simulations were run for multiple traces as well as at different portions of any given trace. The video chunks encodings used were 3000, 1000 & 350 kbps for High, Medium and Low qualities respectively. Each video chunk begins download as soon as the previous one has completed downloading. We also use chunks with three different luminance values for each bitrate. Hence we have a total of 9 brightness levels as shown in Fig. 7a.

For the normal DASH client, we use a throughput based bitrate adaptation logic where the client picks the best possible quality so as to maximise the network utilization. The bitrate adaptation logic for eDASH is a two-stage process. The first stage is similar to that of a normal DASH client which picks the best possible quality for the current network condition. This information is then given as input to the second stage which runs our eQoE algorithm which picks the chunk with the best possible quality and brightness that could minimize the energy consumption while maximising the QoE and keeping the overall power consumption for the video under the specified energy threshold (E_{th}). The energy threshold is defined

as the percentage of energy, normalised to 1, available to the eDASH to download a video had the video been streamed only using highest available bitrate. For example, for a video that consumes 100 units of power if only the highest possible bitrate was selected, an E_{th} of 0.8 means that the eDASH player has to finish downloading and playing back the entire video using 80 units of power. We use E_{th} values of 0.7, 0.8 and 0.9. for our simulations depending on whether we require aggressive, nominal and bare minimum energy savings respectively. Unless otherwise mentioned, we report all results corresponding to the balanced energy threshold.

The second stage returns the next n chunks depending of the window size, w specified. For our simulations, we use $n = w$. For example, a window size of 5 would mean the eDASH would return the next 5 possible chunks within the energy constraint E_{th} that maximizes the QoE. The window size is the horizon for which the eDASH assumes the network conditions will remain same. This is done to reduce the number of times the algorithm is run on the client side and hence reduce the load on the client. The operation of the window was explained earlier in Section III. In our simulations, we used two window sizes of 3 & 5 chunks.

B. Example scenario

In order to illustrate the behavioural differences between our proposal and a normal DASH player, we present in this section a typical run based on the network trace shown in Fig. 6. In particular, we are interested on how the chunks are selected by the eDASH for a single trace for downloading a single video file as well as their associated displaying and downloading energy fingerprint. In this specific scenario, eDASH was configured with a window size $w = 3$, and an energy threshold $E_{th} = 0.8$.

Fig. 7a shows the various Quality and Brightness levels picked (bitrate and brightness adaptation) by eDASH while Fig. 7b shows their corresponding chunk selection (bitrate adaptation) as well as displaying the corresponding network condition. We can see in Fig. 7b that our proposal respects the adaptive properties of DASH by not overloading the network and downloading unrealistic chunks. On the other hand, using the same network trace, Fig. 7c shows the chunk selection for a normal DASH client. As expected, we can observe that the normal DASH player is better suited to match network conditions, but this characteristic is having a significant impact on the energy consumption. Indeed, we present in Fig. 7d, the corresponding battery progress with time based on both the displaying and downloading energy cost. Overall, we can see that, in order to save energy, the eDASH picks lower video quality. Then it tries to maximise the QoE by maintaining a particular quality instead of following the network. This increases the smoothness of the video, as well as reduce the overall energy consumption and hence the end user QoE is maximised. Finally we see that our proposal was able to save nearly 30% of battery in this particular case.

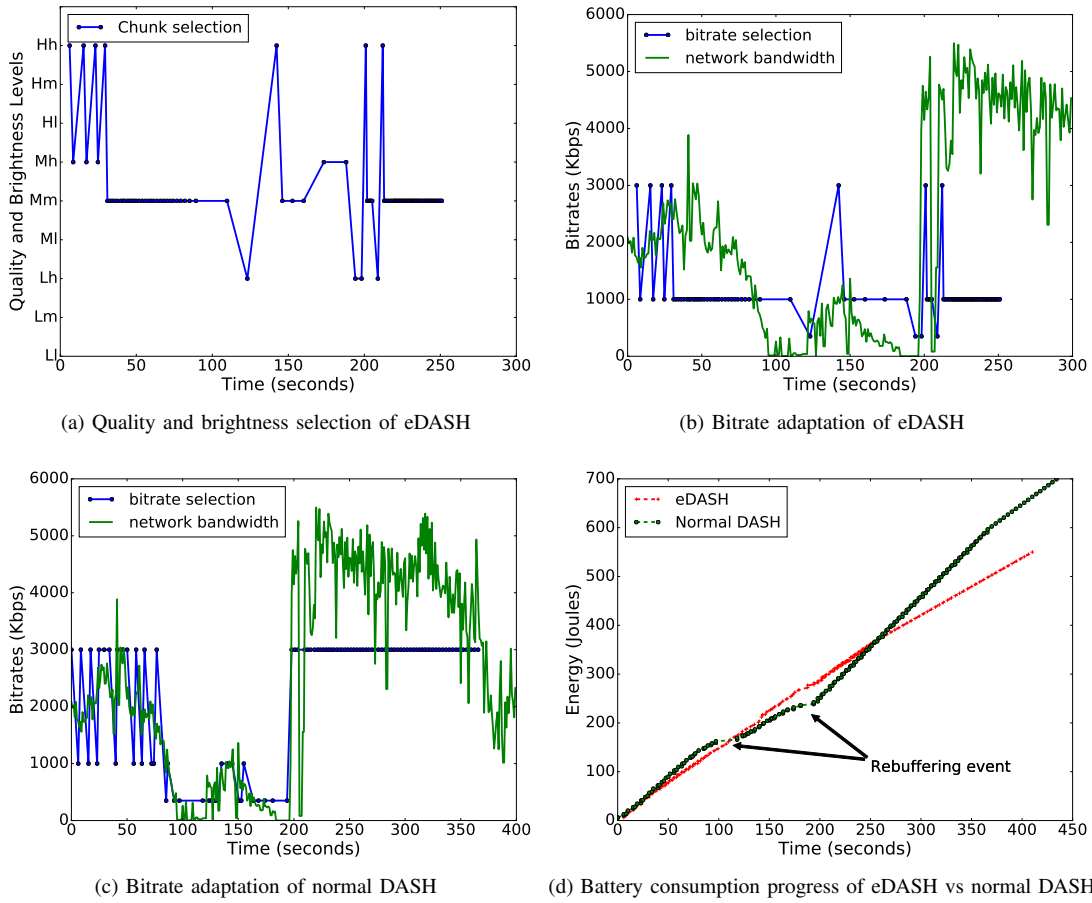


Fig. 7. Chunk selection comparison for eDASH vs Normal DASH for a sample network trace ($window = 3, E_{th} = 0.8, \beta = 1$)

C. Impact of energy parameters on QoE and power consumption

To investigate the impact of various energy parameters on the Quality of Experience of end user, simulations were run for varying β from (1) and Energy Threshold, E_{th} values. The simulations were run with multiple traces and varying starting points for each trace. Unless otherwise mentioned, the results shown are aggregate of these samples. We use a window sizes of 3 and 5, energy threshold values, E_{th} were varied between 0.7 to 0.9.

First, we study the influence of β on QoE by varying β from 0.5 to 1.5. As shown in Fig. 8, we can see that with increasing the significance of energy consumption of device (β), the normal DASH has its overall QoE reduced for the end user. With eDASH, even though the overall QoE is reduced, the user has relatively higher QoE as compared to normal DASH. This is because the user consumes less battery to play back the entire video as opposed to that for normal DASH which consumes more battery power as shown in Fig. 9.

To gain an insight into what contributes to this reduced battery consumption and to find out the amount of energy saving that could be achieved, we study the impact of battery constraint or energy threshold (E_{th}) on energy savings. Fig. 10 shows the energy savings that could be achieved for various

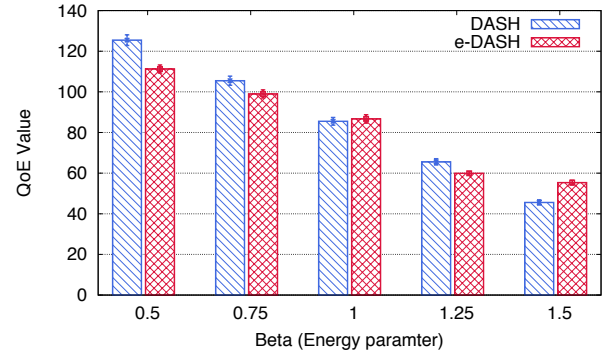


Fig. 8. Impact of variation of energy factor β on Quality of Experience ($window = 5, E_{th} = 0.8$)

energy thresholds. We also investigate the contribution of energy savings achieved from switching quality and that from brightness variations as well for a given set of parameters. We find that most power savings is obtained due to quality switches rather than brightness reduction. A combination of network condition and energy threshold parameter, E_{th} contributes to this behaviour. For example, for an E_{th} of 0.9, we see that 100% of the energy savings achieved is due to quality

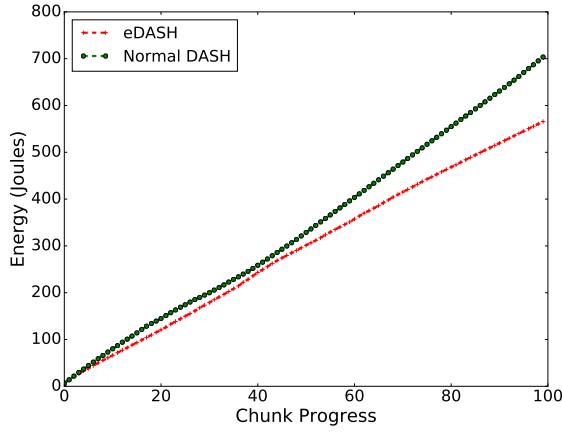


Fig. 9. Average battery consumption comparison for eDASH vs Normal DASH ($window = 5, E_{th} = 0.8$)

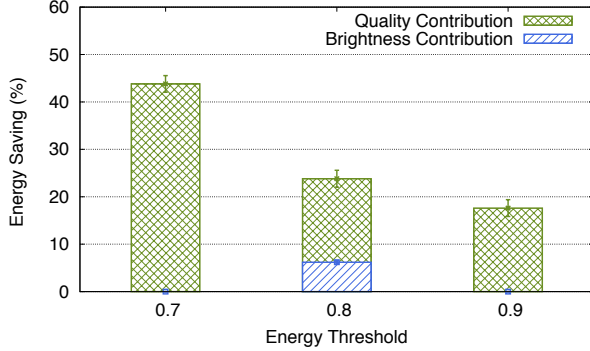


Fig. 10. Contribution of quality switching and brightness variations in observed energy savings for various energy thresholds ($window = 3, \beta = 1$)

variation. This is because, for a bandwidth constraint network, even at a high threshold (low battery constraint), since the network does not allow higher quality files to be downloaded, just selecting a medium or low quality chunk at its highest brightness value is sufficient to satisfy the battery constraint.

D. User benefits

The obvious benefit for end users is the increase in battery life due to energy savings that is achieved from downloading and playing back low quality chunks as well as those with lower brightness as shown in Fig. 9. The not so obvious user benefit is the reduction in amount of data that is downloaded as shown in Fig. 11. By reducing the quality of the chunks that is downloaded, the user downloads less data over their mobile network. For example, for an energy threshold of, $E_{th} = 0.7$, eDASH downloads 80% less data (Kb) than a normal DASH. This translates to less data used from their mobile data cap for those users with small download quotas. It also reflects as monetary savings in cost of data plans for mobile users since users need not pay exorbitant rates for bigger mobile data caps.

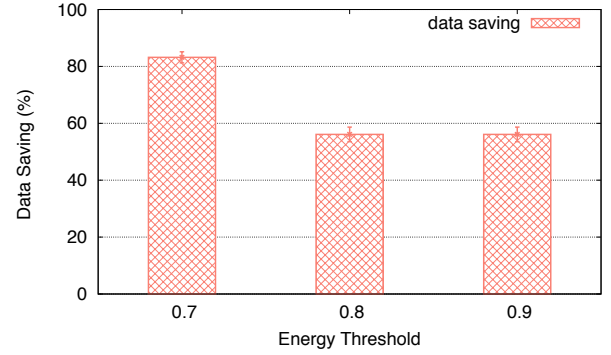


Fig. 11. Amount of data savings for various energy thresholds ($window = 3, \beta = 1$).

V. RELATED WORK

The focus of our work was manifold. First, we aimed to experimentally profile the energy consumption of video streaming in mobile devices, then to propose a new QoE model factoring in energy consumption of (mobile) client device for adaptive streaming application and finally to propose a new energy aware architecture which could be seamlessly integrated with existing one with minimal additions to DASH protocol. In the remainder of this section, we review the most relevant work for each aspect of our contribution.

There has been several complementary works that have looked at profiling energy consumption of smartphones [14] [15]. Carroll and Heiser [14] conducted a detailed component-level power consumption measurement and analysis for various applications including video streaming in smartphones. Their macro-benchmarking for video streaming observed that the display subsystem accounted between 38% and 68% of aggregate power consumption. Since then, to obtain a more detailed understanding of power consumption of smartphone display applications, Chen et al. [15] evaluated power consumption AMOLED displays for multimedia applications, including video streaming. They concluded that increasing video resolution had an higher impact on power consumption of AMOLED displays than increasing bitrate. They also concluded that the power consumption of these displays are highly content dependent. Several other studies were conducted for measuring, modelling and optimizing power consumptions of various state of the art display technologies [16] [17].

Park and Song [10] proposed an OLED power management scheme for video playback on Android smartphones. They showed that darkening the videos using alpha blending technique helped in saving 12%-36% power for video playback compared to standard playback. Trestian et al. [18] showed that energy savings can be achieved by reducing the quality level of the chunks without significantly reducing the user-perceived quality of video streaming. Similar display adaptation and colour transformation techniques have be proposed to achieve energy savings for video playback and display power consumption of smartphones [19] [20].

The impact of various streaming techniques, network connectivity options on energy consumption of video streaming application and how to save energy using smart bitrate adaptation techniques is another area of interest [11] [21].

Improving the the Quality of Experience of end users has traditionally been one of the main goals of adaptive streaming. Yin et al. [7] proposed a model predictive control (MPC) approach to maximize the users QoE while providing a good tradeoff among QoE parameters like rebuffering events, quality switches and startup delay. They also have a practical implementation of their MPC in standard DASH player.

Complementary to the above studies, we provide a renewed way of understanding QoE for video streaming applications by considering energy savings as an equally significant factor that affects the overall streaming experience for an end user.

VI. CONCLUSION

In this paper we proposed a new HTTP-based adaptive streaming architecture, which could be easily and seamlessly integrated with the existing one with minor additions to the DASH protocol. We also proposed a new Quality of Experience model for HAS application which factors in the energy consumption of client (mobile) device that forms part of an energy-aware DASH player known as eDASH. This new model was built upon some experimental profiling of the power consumption of a modern smartphone for video streaming application.

As a result, we have proposed the first HAS framework that takes into account the impact of energy consumption is the video quality selection. Based on real world traces and energy profile data obtained from our experiments, we later evaluated through simulations our newly introduced eDASH proposal. Overall, we observed upto 45% energy savings and upto 80% network utilisation saving while minimizing the reduction in QoE for the end user.

As future work, we intend to implement the eDASH QoE model as a plugin for existing DASH clients like VLC and evaluate its energy savings through controlled experiments as well as implementations in the wild using real users. Furthermore a MOS test could be designed which would provide a more realistic feedback on the potential for adoption of eDASH as a standard.

REFERENCES

- [1] Cisco, "Cisco vni forecast and methodology, 2015-2020," 2016. [Online]. Available: <http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html>
- [2] T. Stockhammer, "Dynamic adaptive streaming over http -: Standards and design principles," in *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, ser. MMSys '11. New York, NY, USA: ACM, 2011, pp. 133–144.
- [3] Apple, "Http live streaming overview." [Online]. Available: <https://developer.apple.com/library/content/documentation/NetworkingInternet/Conceptual/StreamingMediaGuide/Introduction/Introduction.html>
- [4] S. Deng and H. Balakrishnan, "Traffic-aware techniques to reduce 3g/lte wireless energy consumption," in *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '12. New York, NY, USA: ACM, 2012, pp. 181–192.
- [5] S. V. Rajaraman, M. Siekkinen, V. Virkki, and J. Torsner, "Bundling frames to save energy while streaming video from lte mobile device," in *Proceedings of the Eighth ACM International Workshop on Mobility in the Evolving Internet Architecture*, ser. MobiArch '13. New York, NY, USA: ACM, 2013, pp. 35–40.
- [6] ITU-T, "Subjective video quality assessment methods for multimedia applications, itu-t rec. p.910," 2008.
- [7] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over http," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, pp. 325–338, Aug. 2015.
- [8] A. Rehman, K. Zeng, and Z. Wang, "Display device-adapted video quality-of-experience assessment," pp. 939 406–939 406–11, 2015.
- [9] B. Foundation, "Big buck bunny," 2008. [Online]. Available: <https://peach.blender.org/download/>
- [10] M. Park and M. Song, "Saving power in video playback on oled displays by acceptable changes to perceived brightness," *Journal of Display Technology*, vol. 12, no. 5, pp. 483–490, May 2016.
- [11] M. Siekkinen, M. A. Hoque, J. K. Nurminen, and M. Aalto, "Streaming over 3g and lte: How to save smartphone energy in radio access network-friendly way," in *Proceedings of the 5th Workshop on Mobile Video*, ser. MoVid '13. New York, NY, USA: ACM, 2013, pp. 13–18.
- [12] N. Matloff, "Introduction to discrete-event simulation and the simpy language," 2008. [Online]. Available: <http://heather.cs.ucdavis.edu/~matloff/156/PLN/DESimIntro.pdf>
- [13] H. Riiser, P. Vigmostad, C. Griwodz, and P. Halvorsen, "Commute path bandwidth traces from 3g networks: Analysis and applications," in *Proceedings of the 4th ACM Multimedia Systems Conference*, ser. MMSys '13. New York, NY, USA: ACM, 2013, pp. 114–118.
- [14] A. Carroll and G. Heiser, "An analysis of power consumption in a smartphone," in *Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference*, ser. USENIXATC'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 21–21.
- [15] X. Chen, Y. Chen, Z. Ma, and F. C. A. Fernandes, "How is energy consumed in smartphone display applications?" in *Proceedings of the 14th Workshop on Mobile Computing Systems and Applications*, ser. HotMobile '13. New York, NY, USA: ACM, 2013, pp. 3:1–3:6.
- [16] F. Chesterman, G. Muliuk, B. Piepers, T. Kimpe, P. D. Visschere, and K. Neyts, "Power consumption and temperature distribution in wrgb active-matrix oled displays," *Journal of Display Technology*, vol. 12, no. 6, pp. 616–625, June 2016.
- [17] M. Dong and L. Zhong, "Power modeling and optimization for oled displays," *IEEE Transactions on Mobile Computing*, vol. 11, no. 9, pp. 1587–1599, Sept 2012.
- [18] R. Trestian, A. N. Moldovan, O. Ormond, and G. M. Muntean, "Energy consumption analysis of video streaming to android mobile devices," in *2012 IEEE Network Operations and Management Symposium*, April 2012, pp. 444–452.
- [19] Z. Yan and C. W. Chen, "Rnb: Rate and brightness adaptation for rate-distortion-energy tradeoff in http adaptive streaming over mobile devices," in *Proceedings of the 22Nd Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '16. New York, NY, USA: ACM, 2016, pp. 308–319.
- [20] M. Dong and L. Zhong, "Chameleon: A color-adaptive web browser for mobile oled displays," in *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '11. New York, NY, USA: ACM, 2011, pp. 85–98.
- [21] M. A. Hoque, M. Siekkinen, J. K. Nurminen, M. Aalto, and S. Tarkoma, "Mobile multimedia streaming techniques: Qoe and energy saving perspective," *Pervasive and Mobile Computing*, vol. 16, Part A, pp. 96 – 114, 2015.