Multi-User Competitive Energy-Aware and QoE-Aware Video Streaming on Mobile Devices

Kristina Wheatman
Pennsylvania State University
kms674@psu.edu

Fidan Mehmeti
Pennsylvania State University
fzm82@psu.edu

Mark Mahon
Pennsylvania State University
mpm114@psu.edu

Thomas La Porta
Pennsylvania State University
tfl12@psu.edu

Guohong Cao
Pennsylvania State University
gxc27@psu.edu

ABSTRACT

Video streaming on mobile devices necessitates a balance between a users' Quality of Experience (QoE) and energy consumption. Given large data sizes from video, extensive amounts of battery power are required for downloading, processing, and playing each video segment. However, video quality may suffer drastically in an effort to pursue energy efficiency. In balancing these two objectives, our research incorporates advanced energy models, processor clock rates, buffer management, and network quality aware downloading. Furthermore, multi-user systems present unique challenges in optimization given competition for network resources. Our algorithm accounts for these fluctuations and concurrent demands across both many users and many cells. Present application in LTE networks and foundations for future implementation in 5G systems are provided.

CCS CONCEPTS

• **Networks** → **Network algorithms**; *Network simulations.*

KEYWORDS

Energy Efficiency; QoE; Cellular Networks; CPU Frequency Scaling; Video Streaming.

ACM Reference Format:
Kristina Wheatman, Fidan Mehmeti, Mark Mahon, Thomas La Porta, and Guohong Cao. 2020. Multi-User Competitive Energy-Aware and QoE-Aware Video Streaming on Mobile Devices. In *16th ACM Symposium on QoS and Security for Wireless and Mobile Networks (Q2SWinet'20), November 16–20, 2020, Alicante, Spain.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3416013.3426455

1 INTRODUCTION

Mobile video streaming through applications such as Youtube, Netflix, Hulu, and Amazon Prime has increased greatly over the past decade, while video streaming as a whole is expected to generate 82% of total cellular traffic by 2022 [1]. Given the vast data size and processing demands of video compared to static information in e-mails and websites, the demands of streaming cause significant battery drainage in mobile devices. Deciding what, when, and how much video data to download to mobile devices has been a topic of many research studies [4][11][12][27], resulting in myriad protocols across different streaming apps.

Our research produces an aggregate model to balance the demands of QoE and energy efficiency. We include elements previously studied separately or in smaller groups:

(1) Adaptive bitrate algorithm (ABR) to adjusting video resolution based on current network conditions.
(2) Comprehensive energy models including promotion energy and the "long tail" in data downloading.
(3) Impact of CPU scaling on energy consumption.
(4) Energy conservation to ensure full video playout.
(5) Buffer status awareness to prevent rebuffering in the event of poor channel conditions.
(6) Multi-user competition for network resources.
(7) Maximize each user's quality of experience (QoE).
(8) Balance between energy conservation and QoE.

Studies [4][12] incorporate (1); researchers in [25][27] consider (2) and (3). Furthermore, the analysis in [10] heavily relies on (5) for their implementation of (1). An additional paper [11] provide an appropriate actualization of (7). Finally, a recent study [5] provides a comprehensive analysis of (1), (4) and (8). However, none of these findings incorporate all eight of these items.

Our algorithms do not rely on any changes to network resource allocation or scheduling algorithms and are executed only on the mobile devices. Our optimization builds off of the foundations of [5] and [25]; however, [5] lacks incorporation of CPU scaling in their model while [25] isolates energy effects while ignoring QoE. Neither provide an energy conservation approach seeking to ensure full video playout before battery depletion. We add the additional effects of multi-user network resource competition to simultaneously demonstrate the effectiveness of our video data downloading protocols across different users.

Our results show the effectiveness of our algorithm in managing the tradeoffs between user QoE and energy constraints over realistic wireless networks in which users have varying channel quality and experience competition for network resources.

The remainder of this paper is organized as follows. In Section 2 we introduce the video streaming model and the energy model. The optimization formulation is presented in Section 3. This is followed

Q2SWinet'20, November 16–20, 2020, Alicante, Spain

ACM ISBN 978-1-4503-8120-8/20/11…$15.00
https://doi.org/10.1145/3416013.3426455

by the the heuristic algorithm in Section 4, and some set of results for the single-user case. The case with multiple users is presented in Section 6. Some related works are given in Section 7. Finally, Section 8 concludes the work.

2 SYSTEM MODEL

2.1 Video Streaming

We consider a system in which users stream videos stored on a remote server over shared wireless links. The users buffer a portion of the video before starting playout. The client may provide a very large buffer that can store the entire video, in which case the video may be downloaded as fast as the network and server can support while the user is playing the video out on their device. One drawback with this approach is that a user may not watch the entire video in which case the resources used to download the video, including device energy and network capacity, are wasted. Therefore clients may only wish to buffer a portion of the video for playout, and fetch further segments to buffer as the playout is progressing. These clients will use a typical high- and low-buffer thresholds to signal the start and stop of downloading.

The video server has a file reserve with a set of resolutions for each video segment, typically ranging between 144p and 1080p or above for mobile devices. The resolution of the video downloaded to the users is governed by what the algorithm requests and what the network can deliver. A user will request a resolution based on the size of their screen and the energy they wish to expend on downloading and playing the video. This requested resolution may be adjusted by the client depending on the status of the playout buffer and the available transfer rates from the network.

Our work seeks to demonstrate the ideal policy behavior for when and at what resolution video segment downloads ought to occur to maximize a user's quality of experience over time within energy constraints on the mobile device.

2.2 Energy Model

Mobile device energy consumption during video streaming includes (i) energy consumed during data reception across the mobile network interface and (ii) energy consumed from CPU processing such as decoding. On smartphones, power consumption is high during data downloading but much lower when the phone is no longer connected to the network and simply playing out buffered video data. In 4G LTE networks, the mobile network power consumption during video streaming exists in one of five states: promotion, data download, tail, idle, and rebuffering. In the absence of data downloading, the phone stays in idle state and consumes very little power. Once an application on the client requests a data download, the phone goes through a higher-energy promotion state, communicating with the mobile network base station to obtain the data transmission channel.

After promotion, data reception can occur at a high, variable speed. Upon full receipt of the requested data package, the phone switches to the tail state while holding the data transmission channel. This tail state is extremely useful for continuity of video streaming since the phone can immediately proceed with subsequent data downloads without the need for additional promotion energy. Once the phone has left the tail state, it moves into the lower-energy idle state and will require promotion energy to begin downloading again; hence, the tail energy spent on maintaining network connection is often wasted. This tail energy is significant as it consumes as much as 60% of the amount of energy for data downloads [3][19]. CPU energy consumption is impacted by its working frequency [15] which can be adjusted on demand, referred to as CPU scaling. Often integrated in the smartphone community as Dynamic Voltage and Frequency Scaling (DVFS), CPU scaling occurs at discrete frequencies set by the smartphone manufacture. The CPU frequency impacts the rate at which data can be received by a device because of TCP processing and the resolution at which videos can be played out. To illustrate, the Samsung Galaxy S5 equipped with a Qualcomm Snapdragon 801 can work at 15 different frequencies; the Samsung Galaxy S6 and LG Nexus 5x[1] both include Cortex-A53 and can work on 9 different frequencies from 384 MHz to 1.44 GHz.

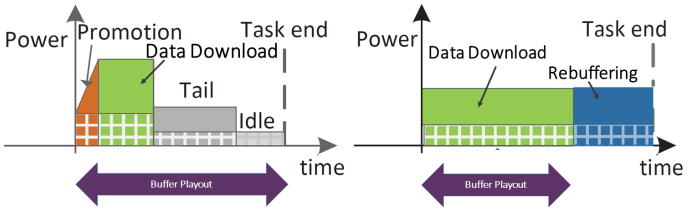

Figure 1: Energy model examples for video streaming, including promotion, data download, tail, and idle portions, with rebuffering if necessary.

Fig. 1 gives a simplistic representation of these varying energy states. If a subsequent data download begins before the end of the tail waiting period, no promotion energy is needed; however, once the phone has transferred into the idle state, promotion energy is needed.

3 PROBLEM FORMULATION

3.1 Optimization

Our optimization objective is a composite function of both energy costs and the utility obtained from a user's quality of experience. The orchestrator sets the weighted priority between these elements at the beginning of the algorithm. Energy consumption becomes a function of the chosen video resolution v_{uij} downloaded from the server, as well as the CPU clock rate f_{uijk} during download and playout. The quality is influenced primarily by the resolution at which a video is played out (which may be equal to or less than the downloaded resolution, depending on battery constraints), combined with penalties for any decreases in resolution as well as any rebuffering events.

To explore these effects, we divide a specific video into N chunks of equivalent playout duration to be downloaded from the server. The quantifiable battery drainage and QoE values for each are then scaled and summed up over all segments for a given user and then across all UEs. Below we first describe each component of our optimization, i.e., energy and QoE, and then present the formal optimization problem.

[1]Cortex-A57 on LG Nexus 5x is typically turned off during video streaming.

3.1.1 Energy. Each video is divided into discrete segment tasks T_{uijk}, given users $u = 1, 2, \ldots, U$, video segments $i = 1, 2, \ldots, N$, bitrate versions $j = 1, 2, \ldots, V$, and CPU clock rate frequencies $k = 1, 2, \ldots, F$[2]. A task (T_{uijk}) is defined as the period of time beginning with the the power promotion and download of video data for segment i, continuing until the subsequent segment process begins. The energy consumption equation $E_u\left(T_{uijk}\right)$ for task T_{uijk} is a summation of all energy consuming functions including promotion time, download time, tail time, idle time and rebuffering. The time duration and power for promotion $\left(t_{ui}^{pro},\ P_{pro}\right)$, data download $\left(t_{ui}^{down},\ P_{down}\right)$, tail period $\left(t_{ui}^{tail},\ P_{tail}\right)$, idle duration $\left(t_{ui}^{idle},\ P_{idle}\right)$, and rebuffering $\left(t_{ui}^{rebuff},\ P_{rebuff}\right)$ are taken from [5][25]. Equations and values are given in Tables 1 and 3 respectively. [3]

3.1.2 Quality of Experience. QoE is quantified using a 5-level rating scale, and model values are taken from experimental data [5]. We use the QoE formulation as follows[4]:

$$Q_u\left(T_{uijk}\right) = \Omega_{uj} - I_{uijk}^{tran}, \tag{1}$$

where Ω_{uj} is the *MOS*5 mean opinion score at designated bitrate j for user u, and I_{uijk}^{tran} is the *MOS*5 impact from data transmission preventing the user from experiencing the full QoE value for a given bitrate j. Specifically, the transmission impact I_{uijk}^{tran} is a function of the frequency of rebuffering and bitrate decrease, and their respective impacts. Prior research [26] has determined that a 3.0 Mbps decrease in resolution has the same impact on a viewer's quality as 1 s of rebuffering. Hence, we set $I_{uijk}^{rebuff} = I_{uij}^{bitrate}$ and scale the bitrate "frequency" changes $f_{uij}^{bitrate}$ by 3 Mbps. Formulations may be found in Table 4 and are taken from [5].

3.1.3 Objective Function. Our objective is to optimize the tradeoff between energy and quality of experience across multiple users in an LTE mobile network. We input $\gamma_u \in [0, 1]$ as the desired proportional weighting between an individual's energy consumption versus QoE, and the user's battery state ($Battery_{u0}$). Each user is assumed to have been given a received throughput rate $\bar{r}_{ui}$. The algorithm selects the resolutions of the downloaded segments, and requested network rate, and the CPU frequency of the mobile device.

Variables, equations, and definitions are found in Tables 1, 2, and 4. The objective function is

$$\min_{\eta} \sum_{u=1}^{U}\sum_{i=1}^{N}\sum_{j=1}^{V}\sum_{k=1}^{F} \eta_{uijk}\left(\gamma_u \frac{E_u(T_{uijk})}{E_u^{min}} - (1-\gamma_u)\frac{Q_u(T_{uijk})}{Q_u^{\max}}\right)$$

subject to

$$\begin{aligned} & f_{uijk} \geq f_{u\{i-g\}jk}^{\min}(v_{u\{i-g\}j}), && \forall\ u, i, j, k, \\ & 0 \leq \rho_{ui} \leq \beta_{\max}, && \forall\ u, i, \end{aligned} \tag{2}$$

where unique CPU clock rate frequencies $f_{uijk} \in \left[f_{uij1}, \ldots, f_{uijF}\right], \forall\ u, i, j, k$ and resolution bitrates $v_{uij} \in [v_{ui1}, \ldots, v_{uiV}]\,,\ \forall\ u, i, j$ are selected. η_{uijk} denotes the user's binary indicator variable of the selected resolution and clock rate for segment i, such that $\eta_{uijk} = 1$ for the chosen combination and $\eta_{uijk} = 0$ otherwise. For normalization purposes, we set the energy-scaling to be the energy consumed when downloading the lowest resolution ($j = 1$) at the lowest clock rate ($k = 1$), meaning $E_u^{min} = E_u\left(T_{ui(j=1)(k=1)}\right)$. Similarly, the QoE-scaling is defined as the mean opinion score for the highest available resolution ($j = J$), such that $Q_u^{\max} = \Omega_{u(j=J)}$.

Constraints: The first constraint requires the chosen CPU frequency f_{uijk} for downloading segment i to be greater than or equal to the minimum clock rate $f_{u\{i-g\}jk}^{min}$ required to playout segments in set $\{i-g\}$ at resolution $v_{u\{i-g\}j}$; see Table 5 for explicit values. The set $\{g\} = \left\{\lceil Z_{ui}\rceil, \ldots, \lceil Z_{ui}\rceil + 1 - \left\lceil \frac{t_{ui}^{pro}+t_{ui}^{tran}}{L}\right\rceil\right\}$ constitutes[5] the range of all previously downloaded segments to be played out during the data download of current segment i, indicated by set $\{i-g\}$. The second given constraint corresponds to the limitations of the buffer status, not to exceed the maximum resources of the phone.

Outputs: Solving the optimization given in Eq. (2) gives a chosen CPU frequency f_{uijk}^{*} and bitrate v_{uij}^{*} for each task i for every user u. These are indicated by the specific non-zero binary indicator variable η_{uijk}^{*}.

3.1.4 Complexity. Observe that Eq. (2) is a Mixed-Integer Nonlinear Programming (MINLP) problem; as such, the optimization has shown to be NP-hard [23]. Furthermore, perfect optimization requires omniscient knowledge of all past, present, and future network conditions for all UE devices as well as the starting time and duration for each individual's smartphone video streaming usage. Our heuristic seeks to approach the ideal in providing consideration of both network fluctuations and variable multi-user behavior.

4 HEURISTIC

Our heuristic relies on knowledge available to the mobile device at the start of each segment download, including its current battery state and network state.

We outline our heuristic in Algorithm 1, where individual users optimize each individual segment download task. The algorithm assures that a user has enough remaining energy to play out all segments downloaded. It adjusts its requested network rate and target resolution for each segment to maintain buffer occupancy sufficiently to prevent rebuffering events. Once it receives a rate allocation from the network, it may adjust its requested download resolution and sets is CPU frequency to be the lowest required, thus saving energy.

Specifically, for each segment i of user u, the objective function is expressed as

$$\min_{\eta,\, r_{ui}^{req.}} \sum_{j=1}^{\tilde{V}_{ui}}\sum_{k=1}^{\tilde{F}_{ui}} \eta_{uijk}\left(\gamma_u \frac{E_u(T_{uijk})}{E_u^{min}} - (1-\gamma_u)\frac{Q_u(T_{uijk})}{Q_u^{\max}}\right)$$

[2] Assuming all phones operate at the same discrete CPU frequencies.
[3] Notation: $(x)_+ = \max(x, 0)$.
[4] The impact of vibration, I_{ui}^{vib}, has been removed from the QoE equation to restrict the scope of evaluation.
[5] Notation: $\lceil x\rceil =$ unique integer satisfying $\lceil x\rceil - 1 < x \leq \lceil x\rceil$.

Table 1: Objective Function Equations

VARIABLE	Equation	Definition
$E_u(T_{uijk}) =$	$t_{ui}^{pro} \cdot P_{pro}\left(f_{uijk}\right) + t_{ui}^{down} \cdot P_{down}\left(f_{uijk}, v_{uij}, c_{ui}\right) + t_{ui}^{tail} \cdot P_{tail}\left(f_{u\{i-g\}jk}^{min}\right) + t_{ui}^{idle} \cdot P_{idle}\left(f_{u\{i-g\}jk}^{min}\right) + t_{ui}^{rebuff} \cdot P_{rebuff}\left(f_{uijk}, v_{uij}, c_{ui}\right)$	Energy to download video chunk i
$t_{ui}^{pro} =$	$\begin{cases}\tau_{pro}, & t_{u(i-1)}^{idle} > 0 \\ 0, & t_{u(i-1)}^{idle} = 0\end{cases}$	Length of promotion, if necessary
$t_{ui}^{down} =$	$\min\left\{\frac{d_{uij}}{\min(W_{ui}(f_{uijk}), \bar{r}_{ui})}, (\rho_{ui} - t_{ui}^{pro})_+\right\}$	Download period during playout
$t_{ui}^{tail} =$	$\min\left\{\left(\rho_{ui} - \beta_h + L - t_{ui}^{pro} - \frac{d_{uij}}{\min(W_{ui}(f_{uijk}), \bar{r}_{ui})}\right)_+, \tau_{tail}\right\}$	Length of tail period
$t_{ui}^{idle} =$	$\left(\rho_{ui} - \beta_h + L - t_{ui}^{pro} - \frac{d_{uij}}{\min(W_{ui}(f_{uijk}), \bar{r}_{ui})}\right)_+$	Length of idle period
$t_{ui}^{rebuff} =$	$\left(t_{ui}^{pro} + \frac{d_{uij}}{\min(W_{ui}(f_{uijk}), \bar{r}_{ui})} - \rho_{ui}\right)_+$	Length of rebuffering time
$P_{pro}\left(f_{uijk}\right) =$	$p_{11} + p_{12} \cdot f_{uijk}$	Power required for promotion
$P_{down}\left(f_{uijk}, v_{uij}, s_{ui}\right) =$	$p_{21} + p_{22} \cdot f_{uijk} + p_{23} \cdot f_{uijk}^2 + p_{24} \cdot v_{uij} + p_{25} \cdot v_{uij}^2 + p_{26} \cdot s_{ui} + p_{27} \cdot s_{ui}^2$	Power for download with playout
$P_{tail}\left(f_{u\{i-g\}jk}^{min}\right) =$	$p_{31} + p_{32} \cdot f_{u\{i-g\}jk}^{min}$	Power during tail period
$P_{idle}\left(f_{u\{i-g\}jk}^{min}\right) =$	$p_{41} + p_{42} \cdot f_{u\{i-g\}jk}^{min}$	Power during idle period
$P_{rebuff}(f_{uijk}, v_{uij}, s_{ui}) =$	$p_{51} + p_{54} \cdot v_{uij} + p_{55} v_{uij}^2 + p_{56} \cdot s_{ui} + p_{57} \cdot s_{ui}^2$	Length of rebuffering time
$Q_u(T_{uijk}) =$	$\Omega_{uj} - I_{uijk}^{tran}$	QoE for user u for task i

subject to

$$\begin{aligned} f_{uijk} &\geq f_{u\{i-g\}jk}^{\min}(v_{u\{i-g\}j}), && \forall\, u, i, j, k, \\ 0 < \beta_\ell \leq \rho_{ui} &\leq \beta_h < \beta_{\max}, && \forall\, u, i, \\ E_u^{min} &\leq E_u\left(T_{uijk}\right) \leq \varepsilon_{ui}^{max}, && \forall\, u, i. \end{aligned} \tag{3}$$

The second constraint now sets a lower β_ℓ and upper β_h threshold on the buffer state desired by the user during playout. To account for the lower bound, we use a modified version of the current buffer state ρ_{ui}, shown in Eq. (4), in solving each individual segment optimization

$$\tilde{\rho}_{ui} = (\rho_{ui} - \beta_\ell)_+ \tag{4}$$

The third constraint ensures that as many video segments as possible may be played out before the battery depletes, even if this necessitates downloading at a lower resolution, where a maximum energy allotment is given as

$$\varepsilon_{ui}^{max} = \max\left[\sigma \cdot \left(\frac{Battery_{ui} - Threshold_u}{N - Z_{ui} + 1}\right),\ E_u^{min}\right]. \tag{5}$$

Analogous to Eq. (2), solving the heuristic gives a chosen CPU frequency f_{uijk}^*, bitrate v_{uij}^*, and requested network throughput rate $r_{ui}^{requested}$. Once the network responds with the allocated resource blocks ($s_{ui} \cdot M$), the user can then choose to adjustment the chosen CPU frequency f_{uijk}^* and bitrate v_{uij}^* values if needed. The resultant received network throughput rate $\bar{r}_{ui}$ and next buffer status $\rho_{u(i+1)}$ are then calculated from these values.

Algorithm 1 Outline: At the start of download we know the threshold below which a user is not willing to allow their battery to drain, and the number of segments a user wishes to download before starting playout. Below we review the processing for each segment; we note that the processing for the initial segment is slightly different as is shown.

In each iteration we first ensure the battery state ($Battery_{ui}$) is above the set threshold (line 2). We then solve the optimization in Eq. (3) as per line 12. The algorithm produces the associated network throughput rate requested $r_{ui}^{requested}$ and the associated resolution and clock rate, j_{ui}^* and k_{ui}^*, respectively, for segment download i. We limit the increase in resolution to one level per segment to make video playout more smooth.

If rebuffering was necessary to download the prior segment ($t_{u(i-1)}^{rebuff} > 0$), the resolution chosen must be less than or equal to that of the prior downloaded segment (line 10). The optimal resolution v_{uij}^*, CPU frequency f_{ui}^*, and requested network throughput rate $r_{ui}^{requested}$ emerge from Eq. (3) with modified buffer status $\tilde{\rho}_{ui}$ from Eq. (4), shown in line 12.

After receiving the requested throughput rate $r_{ui}^{requested}$ and per-resource-block-rate $b_{ui}(t)$, the network base station then allocates a certain number of resource blocks to u, thereby giving the recipient a throughput rate of $r_{ui}(t)$, producing an estimated average throughput rate $\bar{r}_{ui}$ throughout the duration of downloading segment i (line 14). Line 15 explores two igniting conditions: a chosen resolution j_{ui}^* more than two indices lower than the resolution selected in the prior iteration, OR the rate received is less than the rate requested. The chosen resolution index j_{ui}^* is reset as the maximum resolution permitted that does not require rebuffering

Table 2: Variables

VARIABLE	Definition
γ_u	Weighting of Energy vs. QoE
L	Standardized length of each video chunk, $O(10\ s)$
v_{uij}	Bitrate j resolution for segment i of user u
f_{uijk}	Discrete CPU frequency
V	Total number of available resolutions on server
F	Total number of discrete frequencies[6]
η_{uijk}	Binary indicator
d_{uij}	Data to receive for task i at bitrate j
$W_{ui}(f_{uijk})$	Maximum TCP Throughput permitted at f_{uijk}
Z_{ui}	Number of video segments (tasks) in buffer at the start of downloading segment i
ρ_{ui}	Full playout time of data in buffer at the start of downloading segment i
β_ℓ	Lower bound for buffer
β_h	Upper bound for buffer
β_{max}	Maximum possible buffer status
$s_{ui}(t)$	Signal strength at time t when downloading segment i for user u
$\bar{s}_{ui}$	Average signal strength downloading segment i for user u
$r_{ui}^{requested}$	Requested throughput rate from network for task i
$r_{ui}(t)$	Received network throughput downloading T_{uijk} at time t
$\bar{r}_{ui}$	Estimated average network throughput received when downloading (T_{uijk})
$c_u(t)$	Percentage of resource blocks allocated to user u during task i at time t
M	Number of resource blocks
$b_u(t)$	Time-varying per-resource-block-rate, function of signal-to-interference-plus-noise
$\Psi(t)$	Number of users requesting network resources at time t
$Battery_{ui}$	Battery percentage for user u at start of segment i download
σ	Energy to battery scaling parameter
ε_{ui}^{max}	Maximum energy allocated to user u to download segment i
k_i^{max}	Maximum clock rate f_{uijk} from prior download $(i-1)$ to support download and full video playout given battery state
j_i^{max}	Index of maximum resolution v_{uij} supported by CPU clock rate index k_i^{max} (see Table 5)
Ω_{uj}	Mean opinion score at designated bitrate j for user u
I_{uijk}^{tran}	Transmission impact
I_{uijk}^{rebuff}	Rebuffering impact
f_{uijk}^{rebuff}	Frequency of rebuffering
$I_{uij}^{bitrate}$	Bitrate impact
$f_{uij}^{bitrate}$	Frequency of bitrate changes

$t_{ui}^{pro} + d_{uij}/\bar{r}_{ui} \leq \tilde{\rho}_{ui}$, else the lowest resolution is selected $j_{ui}^* = 1$. In the case that the chosen CPU clock rate index k_{ui}^* does not support the full received throughput rate (line 17), a higher frequency is selected in lieu.

Finally, data from segment i at bitrate v_{uij}^* corresponding to resolution index j_{ui}^* is requested from the server (line 20). This video data is then received at rate min $\left\{W_{ui}(f_{uijk^*}), \tilde{r}_{ui}\right\}$, designating the minimum of either the TCP throughput rate supported at clock rate f_{uijk}^* or the allocated throughput rate received from the mobile network $\bar{r}_{ui}$.

Once all video segments have been downloaded, or the battery has reached its given download threshold, remaining buffer segments are played out at the minimum clock rate required to sustain the resolution for each downloaded chunk until completion or battery depletion.

5 RESULTS

Data values are taken from [5] and [25] using LG Nexus 5x smartphones. Furthermore, the minimum CPU frequency scalings required to playout a given resolution for LG Nexus 5x are shown in Table 5. Fig. 2 depicts the successful simulation of the heuristic. The top portion of the figure shows when segments are downloaded over time, the time and resolution at which each segments is played out, and the corresponding CPU clock rate. The bottom portion of the graph shows the requested and received network rate, the evolution of the buffer at the client, and the residual energy at the client.

As can be seen, the client downloads segments as fast as possible until the buffer is full. The resolution of the segments increase from segment one to two as the device senses it has enough energy to download the video as the 144P resolution. It then requests further segments as the buffer occupancy decreases. The CPU clock rate is high when the videos are being downloaded, but lowers to the required rate to playout the videos when no download is ongoing. Validation plots for heuristic algorithm, including the impact of incorporating tail energy, buffer status, battery state, and CPU clock rate frequency are shown below. The following algorithm elements are given and their impact described below.

Buffer Status Impact: Eq. (4) from line 13 of Algorithm 1 allows the heuristic to detect when the buffer occupancy is approaching its lower threshold so it can request more segments to refill the buffer before playout is interrupted. Furthermore, lines 3 and 4 of Table 1 provide the time duration of the tail and idle periods chosen for each segment download. The upper buffer bound β_h drives/motivates the waiting period between downloads once the buffer has filled up to its ideal capacity. As seen in Fig. 2, the network provides very low network rate for the download of segments 4 and 5, so the client chooses not to increase the segment resolution to ensure segments can be downloaded fast enough to avoid rebuffering.

Battery Status Impact: Energy conscious behavior emerges from limiting the maximum allowable energy expenditure, line 7, as well as the upper thresholds on clock rate and resolution, lines 8 and 12. Fig. 2 illustrates behavior when energy conservation is considered, while Fig. 3 illustrates when it is not. As a result, when energy conservation is not considered, the user cannot playout all of the segments it has downloaded.

Algorithm 1: Energy-Aware and QoE-Aware Video Streaming on Mobile Devices (EQMS)

Data: Determine initial UE battery state ($Battery_{u0}$), battery threshold ($Threshold$), initial segments to download before playout ($Initial\_Segments$), and the desired energy versus QoE weighting (γ_u).

1 **for** $i = 1, \ldots, N,$ **do**
2 **if** $Battery_{ui} > Threshold$ **then**
3 **if** $i \leq Initial\_Segments$ **then**
4 Set $\tilde{V}_{ui} = \begin{cases} V & i = 1 \\ j^*_{u(i-1)} + 1 & i > 1 \end{cases}$
5 Solve Eq. (3) for j^*_{ui},f^*_{ui}, and $r^{requested}_{ui}$, replacing ρ_{ui} with δ.
6 **else**
7 Find ε^{max}_{ui} using Eq. (5); Set
8 $\tilde{V}_{ui} = \min\left[\left(j^*_{u(i-1)} + 1\right), j^{max}_{ui}\right]$, $\tilde{F}_{ui} = k^{max}_{ui}$;
9 **if** $t^{rebuff}_{u(i-1)} > 0$ **then**
10 Set $\tilde{V} = j^*_{u(i-1)}$;
11 **end**
12 Solve Eq. (3) with adjusted values $\tilde{\rho}_{ui}$ from Eq. (4) for j^*_{ui},f^*_{ui}, and $r^{requested}_{ui}$;
13 Solve for $k^{max}_{u(i+1)}$ and $j^{max}_{u(i+1)}$; Request rate $r^{requested}_{ui}$ given per-resource- block-rate $b_{ui}(t)$ from base station;
14 Receive throughput rate $\bar{r}_{ui}$ from network.
15 **if** $j^*_{ui} < j^*_{u(i-1)} - 2$ **OR** $r^{requested}_{ui} > \bar{r}_{ui}$ **then**
16 $j^*_{ui} = \max\left\{j : j \in 1, \ldots, j^{max}_{ui}\right\}$ $s.t.\ t^{pro}_{ui} + d_{uij}/\bar{r}_{ui} \leq \tilde{\rho}_{ui}$, else $j^*_{ui} = 1$;
17 Increase k^*_{ui} such that $W_{ui}(f^*_{uijk}) \geq \bar{r}_{ui}$;
18 **end**
19 **end**
20 Request bitrate v^*_{uij} for segment i from server;
21 Receive data at rate $\min\left\{W_{ui}(f_{uijk^*}), \tilde{r}_{ui}\right\}$;
22 Playout first segment(s) in the buffer with CPU clock rate set to $\max\left(f^*_{uijk}, f^{min}_{u\{i-g\}jk}\right)$ during promotion, download and rebuffering;
23 Set to $f^{min}_{u\{i-g\}jk}$ during tail and idle phases;
24 Update $\rho_{u(i+1)}$ and $(Battery_{u(i+1)})$;
25 **end**
26 **end**
27 **while** $Battery_u(t) > 0$ **do**
28 Playout remaining video chunks in buffer with CPU set to f^{min}_{uijk} for each segment.
29 **end**

Table 3: Constants

VARIABLE	Value	Definition
$\tau_{tail} =$	10.35 s	Standard tail time
$\tau_{pro} =$	0.91 s	Promotion energy time
$I^{rebuff}_{uijk} =$	0.742 $MOS5$	Rebuffering impact
$I^{bitrate}_{uij} =$	0.742 $MOS5$	Bitrate impact
$L =$	20 s	Length of video segments
$\beta_\ell =$	10 s	Lower buffer threshold
$\beta_h =$	80 s	Upper buffer threshold
$\delta =$	2 s	Adjusted download time
$Threshold_u =$	10%	Minimum download battery
$Initial\_Segments =$	2 $segments$	Pre-playout initial buffering
$\gamma_u =$	0.35	Energy vs. QoE scaler
$\sigma =$	$2.5 \cdot 10^{-6}\ \frac{Watts}{\%\ Battery}$	Energy-battery scaling

Table 4: Formulas

VARIABLE	Equation
$Z_{ui} =$	$\lceil \rho_{ui}/L \rceil$
$\rho_{ui} =$	$\sum_{z=1}^{Z_{ui}} \sum_{j=1}^{V} \eta_{u(i-1-Z+z)jk} \left(\frac{d_{u(i-1-Z+z)j}}{v_{u(i-1-Z+z)j}}\right)$
$r_{ui}(t) =$	$b_u(t) \cdot c_u(t) \cdot M$
$I^{tran}_{uijk} =$	$f^{rebuff}_{uijk} \cdot I^{rebuff}_{uijk} + f^{bitrate}_{uij} \cdot I^{bitrate}_{uij}$
$f^{rebuff}_{uijk} =$	$\frac{\left(t^{pro}_{ui} + \frac{d_{uij}}{\min\left(W_{ui}(f_{uijk}), \bar{r}_{ui}\right)} - \rho_{ui}\right)_+}{\rho_{ui}}$
$f^{bitrate}_{uij} =$	$\frac{\left(v_{u(i-1)j} - v_{uij}\right)_+}{3.0\ Mbps}$

Table 5: Video Streaming Resolution Metrics for LG Nexus 5x [5][25]

RESOLUTION	Bitrate	Min CPU Frequency
144p	0.10 $Mbps$	384 MHz
240p	0.375 $Mbps$	450 MHz
360p	0.75 $Mbps$	450 MHz
480p	1.50 $Mbps$	600 MHz
720p	3.00 $Mbps$	652 MHz
1080p	5.80 $Mbps$	883 MHz

Advanced Energy Model: Solving Eq. (3) in lines 5 and 12 Algorithm 1 allows us to incorporate the advanced energy model described in Section 2.2. Energy of promotion, download, tail, idle, and rebuffering in the model drives the user to download segments to fill up the buffer back to back to prevent wasted tail or promotion energy. Fig. 2 effectively chooses to download segments 1 through 8 at a chosen rate and resolution to avoid tail energy altogether, while segments 9 and 10 place the downloads close enough together to avoid unnecessary promotion energy.

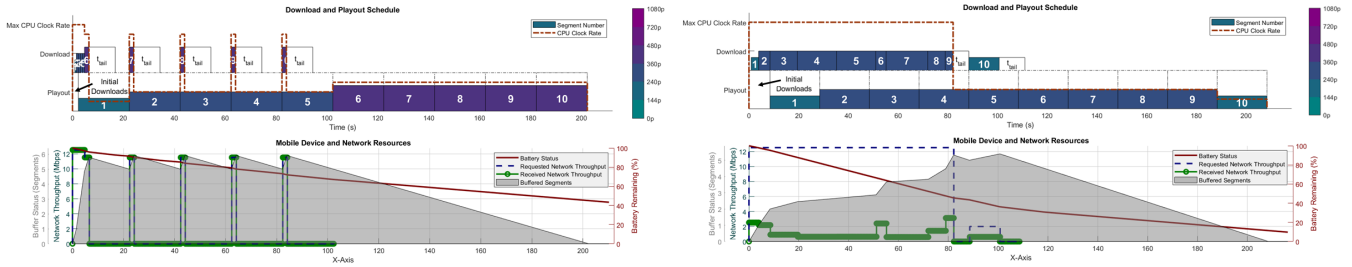

Figure 2: (Left) Ideal scenario implementing heuristic from Algorithm 2. Full requested network throughput rate is received. (Right) Prevention of buffer depletion by implementing heuristic from Algorithm 2. Only fraction of requested throughput rate is received.

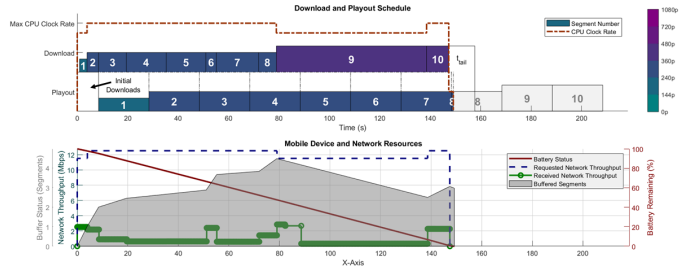

Figure 3: Lack of energy conservation by incomplete implementation of heuristic from Algorithm 2 (without lines 6, 7, 11). Only a fraction of requested throughput rate is received. The downloading schedule is not conservative enough without the full heuristic.

Algorithm 2: Network Resource Allocation with GBR and MBR Constraints

1 **for** $t = t_{start}, \ldots, t_{final}$ **do**
2 Find the number of users $\Psi(t)$ requesting network resources at time t.
3 **if** $\sum_{u=1}^{U} \frac{r_{ui}^{requested}(t)}{b_u(t)} \leq M$ **then**
4 Set $c_u(t) = r_{ui}^{requested}(t)/(b_u(t) \cdot M)$;
5 **else**
6 **for** $u = 1, \ldots, U$ **do**
7 $c_u(t) = \min\left[\Psi(t)^{-1}, \frac{r_{ui}^{requested}(t)}{b_u(t) \cdot M}, \frac{\Xi^{max}}{b_u(t) \cdot M}\right]$;
8 $c_u(t) = \max\left[c_u(t), \frac{\Xi^{min}}{b_u(t) \cdot M}\right]$;
9 **end**
10 **if** $\sum_{u=1}^{U} c_u(t) < 1$ **then**
11 Divide remaining resource blocks equally among all under-receiving users satisfying $r_{ui}^{requested}(t) > b_u(t) \cdot c_u(t) \cdot M$.
12 **end**
13 **end**
14 **end**

6 MULTI-USER RESULTS

6.1 Network Resource Allocation Model

6.1.1 Network Conditions. To simplify our network resource allocation model, we use "per-resource-block-rates" b_u [18] for individual users given our chosen 20 MHz bandwidth (100 resource blocks in LTE). This "per-resource-block-rate" incorporates signal to noise ratios (including interference and distortion impacts), coding rates and the corresponding efficiency. All bandwidth frequencies are assumed to have the same signal quality in all following examples and simulations.

An individual's received throughput rate is given by:

$$r_{ui}(t) = b_u(t) \cdot c_u(t) \cdot M. \tag{6}$$

Intuitively, constraints on the resource block allocation emerge as $\sum_{u=1}^{U} c_u(t) \leq 1,\ c_u(t) \in [0, 1],\ \forall\ u, t$, to prevent resource over-allocation.

6.1.2 Resource Allocation. We incorporate a basic network resource allocation scheme in our simulations, as outlined in Algorithm 2. Specifically, at each time increment, our network seeks first to provide every user with the throughput rate they request given their per-channel-rates (lines 3, 4). If allocating all users their requested rates would exceed the number of available resource blocks, the algorithm kicks in (line 6). All requesting users are given equivalent resources, not to exceed the throughput rate requested given each UE's per-channel-rate (line 7). These rates are further bounded as we require the network to provide a minimum guaranteed bitrate (GBR, denoted by $\Xi^{min} = 1.0\ Mbps$), not to exceed the maximum bitrate (MBR, $\Xi^{max} = 12.0\ Mbps$) range [22] (line 8), for each user given our QCI value of 4 for buffered video [6]. Remaining resource block resources are equally divided among users receiving a lower throughput rate than initially requested (line 11). Algorithm 2 outlines this network resource allocation with GBR and MBR constraints [17].

6.2 System Parameters

We consider a range of per-resource-block conditions ranging from 45 $kbps/block$ to 800 $kbps/block$. We choose the standard 15 CQI index levels for LTE and arbitrarily set a CQI of 1 as a per channel rate of 45 $kbps/block$ and a CQI of 15 as an 800 $kbps/block$ for simplicity [18]. While these values would be dependent on the proprietary mapping of any specific service provider, they are merely

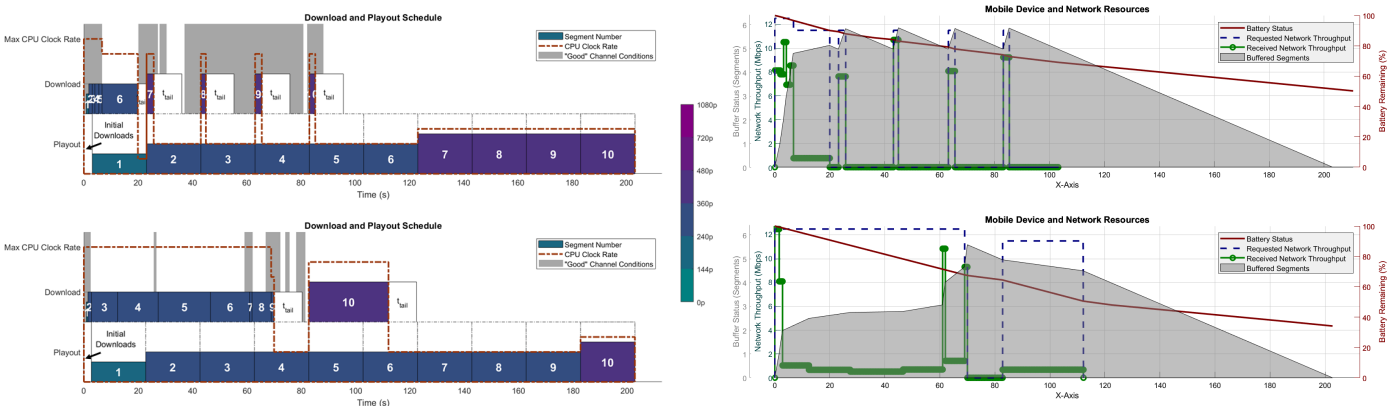

Figure 4: Video streaming behavior throughout playout for (Top) User 1 with primarily "good" channel conditions and (Bottom) User 2 with primarily "poor" channel conditions, given competition between 20 users.

representative of a range of conditions users might experience in LTE.

We define “good” channel conditions as CQIs ranging uniformly between 8 through 15 (explicitly, $250 - 800\ kbps/block$), with our simulation average of 500 $kbps/block$. “Poor” channel quality encompasses CQIs between 1 through 8 ($45 - 250\ kbps/block$), averaging around 121 $kbps/block$.

Block-rate conditions are set to fluctuate on the order of 100 ms. The exchange between good and bad states is modeled using a two-state Markov chain. Primarily “good” users have a probability of remaining in the good state as $q_{good} = 140/141$, with an expected residency time of 14 s, and a probability of remaining in a bad state as $q_{bad} = 60/61$, with an expected residency time of 6 s. Similarly, “bad” users have a probability of remaining in the good state as $q_{good} = 60/61$, with an expected residency time of 6 s, and a probability of remaining in a bad state as $q_{bad} = 140/141$, with an expected residency time of 14 s. "Moderate" users fluctuate between "good" and "bad" channel conditions, each with an expected residency time of 10 s where $q_{good} = q_{bad} = 100/101$. Reported RSRP values range from -140 dBm to -44 dBm [2]. While not directly correlated, we set the signal strength for users with CQI of 1 to $RSRP = -120\ dBm$ and users with CQI of 15 to $RSRP = -50\ dBm$ for simplicity. The remaining RSRP values are obtained by interpolating between these two anchors [13].

6.3 Results

This section illustrates multi-user energy-aware and QoE-aware video streaming protocols on mobile devices given the network resource allocation presented in Section 6.1. Our simulation has been scaled to twenty users sharing a wireless interface. Eight users have primarily “good” channel conditions, while another eight users are given mostly “poor” channel conditions throughout. The remaining four users have "moderate" channel conditions.

As shown in the left of Fig. 4, User 1 with primarily "good" channel conditions downloads the first 5 video segment chunks relatively quickly before the middle of the first video segment has played out, with segment 6 being downloaded at a slower rate due to decreased channel conditions. User 2 with mostly poor channel conditions instead follows a slower download pattern and does not finish downloading segment 6 until almost the end of segment 3 playout. User 1 only proceeds to the tail state after segment 6 since the upper buffer threshold has been reached; given a relatively full buffer and longer periods of better channel conditions, segments 7 through 10 are downloaded at an even higher resolution to further improve the QoE for User 1. The tail energy is effectively utilized when downloading segment 7 for User 1 since no promotion energy is needed. However, given a relatively full buffer for User 2 after downloading segment 9, the choice to download segment 10 at a higher resolution comes at a penalty of an extremely long download period of time lasting throughout two full playout segments. Nevertheless, both Users 1 and 2 are able to complete full playout of the video before battery depletion, even without reducing video quality over time. No rebuffering events occur due to the effective heuristic implementation and energy conservation measures.

As shown in the right of Fig. 4, neither User 1 nor User 2 receive the network throughput rate they request. This creates competition for network resource blocks. As resource blocks are equally divided among users, those with higher per-resource-block rates ultimately receive higher network throughput for the same number of allocated resource blocks.

Both users are energy-conscious in allowing for full video download and playout before battery depletion. User 1 has a larger energy reserve at the end of playout than User 2, yet both users have achieved their current viewing goals to completion.

7 RELATED WORK

While adaptive bitrate (ABR) streaming protocols have been widely explored, studies have limited their scope to a few objectives. One buffer-based approach hones in on the negative impact of rebuffering on user experience as a motivation to aggressively stabilize and tend towards a full buffer [10]. Our study incorporates an aggressive buffer-filling prioritization scheme without fully sacrificing energy and QoE impacts.

Another researcher provides a detailed exploration of mobile device energy consumption from both the network interface and CPU [24], yet does not consider the influence on users' quality of experience. Our results provide insight to the special of case of video streaming by *minimizing* this energy consumption. Prior studies [3][7][21] have investigated the "long tail problem," with cellular systems, but the literature is lacking research on the effects of CPU frequency scaling on energy consumption. Researchers [8][9][25] have begun to quantify the significant impacts on CPU scaling and energy conservation; nevertheless, none have explored CPU scaling energy consumption in conjunction with mobile users' Quality of Experience.

One study solely regarding mobile video streaming QoE [14] considers the influence of content, picture quality, sound quality, interest matching, fluidness and loading speed on an individual's quality rating. Our paper restricts the scope to the impact of mobile network conditions on QoE to allow for clearer conclusions.

While the balance between performance and energy for local computation tasks has been explored [16][20][27], our research is unique in considering the impact of CPU frequency on TCP throughput while balancing energy conservation with user's quality of experience.

Some energy models for video streaming may only include the data download energy and the idle state buffer playout energy [9]. Our model incorporates more detailed effects, including the promotion energy sometimes needed before a data download as well as the period of tail energy after the segment data download when the phone is holding the data transmission channel.

8 CONCLUSION

Our work provides an effective heuristic for ideal policy behavior, when and at what resolution, for video segment downloads ought to occur to maximize a user's quality of experience over time within energy constraints on the mobile device. Our research contributes insight into multi-user mobile video streaming given CPU scaling considerations, energy conservation, and QoE weighting. Implementation of said heuristic in conjunction with a given network resource allocation model produces superior results to both default isolated segment optimization settings as well as existing heuristic models [5][25]. Present application in LTE networks is suggested and future implementation in 5G systems are to be explored.

ACKNOWLEDGEMENT

Research was sponsored by the National Science Foundation (NSF) under grant number CNS-1815465.

REFERENCES

[1] 2020. Cisco Annual Internet Report (2018–2023) White Paper. http://goo.gl/DXWFyr

[2] 3GPP. 2010. *LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Requirements for support of radio resource management.* Technical Specification (TS) 36.133. 3rd Generation Partnership Project (3GPP). https://www.3gpp.org/dynareport/36133.htm Version 8.9.0.

[3] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. 2009. Energy Consumption in Mobile Phones: A Measurement Study and Implications for Network Applications. In *Proc. of 9th Conf. on Internet Measurement (IMC).* ACM SIGCOMM, New York, NY. https://doi.org/10.1145/1644893.1644927

[4] A. Bentaleb, B. Taani, A.C. Begen, C. Timmerer, and R. Zimmermann. 2018. A survey on bitrate adaptation schemes for streaming media over HTTP. *IEEE Communications Surveys & Tutorials* 21, 1 (2018), 562–585.

[5] X. Chen, T. Tan, and G. Cao. 2019. Energy-Aware and Context-Aware Video Streaming on Smartphones. In *2019 IEEE 39th Intl. Conf. on Distributed Computing Systems (ICDCS).* 861–870.

[6] C. Cox. 2012. *An introduction to LTE: LTE, LTE-advanced, SAE and 4G mobile communications.* John Wiley & Sons.

[7] Y. Cui, S. Xiao, X. Wang, M. Li, H. Wang, and Z. Lai. 2014. Performance-aware energy optimization on mobile devices in cellular network. In *Proc. of IEEE Conf. on Computer Communications (INFOCOM).* 1123–1131.

[8] M. Dasari, S. Vargas, A. Bhattacharya, A. Balasubramanian, S.R. Das, and M. Ferdman. 2018. Impact of Device Performance on Mobile Internet QoE. In *Proc. of 2018 Internet Measurement Conf. (IMC '18).* ACM, New York, NY, USA, 1–7. https://doi.org/10.1145/3278532.3278533

[9] W. Hu and G. Cao. 2015. Energy-aware video streaming on smartphones. In *Proc. of IEEE Conf. on Computer Communications (INFOCOM).*

[10] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson. 2014. A Buffer-Based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service. In *Proc. of the 2014 ACM Conf. on SIGCOMM.* ACM, New York, NY, USA, 187–198. https://doi.org/10.1145/2619239.2626296

[11] I. Irondi, Q. Wang, C. Grecos, J.M.A. Calero, and P. Casaseca-De-La-Higuera. 2019. Efficient QoE-Aware Scheme for Video Quality Switching Operations in Dynamic Adaptive Streaming. *ACM Trans. Multimedia Comput. Commun. Appl.* 15, 1, Article 17 (Feb. 2019), 23 pages. https://doi.org/10.1145/3269494

[12] S. Jabbar, D. Kadhim, and Y. Li. 2018. Improving Video Quality in DASH Systems by Proposing Adaptive Bitrate Scheme based on Variable Segment Size Approach. *Intl. Journal of Computer Applications* 180 (02 2018), 13–18. https://doi.org/10.5120/ijca2018916416

[13] M. T. Kawser, B. Hamid, N. Hasan, M. S. Alam, and M. M. Rahman. 2012. Downlink SNR to CQI mapping for different multiple antenna techniques in LTE. *Int. J. Inf. Electron. Eng.* 2, 5 (2012), 757–760.

[14] I. Ketykó, K. De Moor, T. De Pessemier, A.J. Verdejo, K. Vanhecke, W. Joseph, L. Martens, and L. De Marez. 2010. QoE measurement of mobile YouTube video streaming. In *Proc. of the 3rd Workshop on Mobile Video Delivery.* 27–32.

[15] J. Kwak, O. Choi, S. Chong, and P. Mohapatra. 2014. Dynamic speed scaling for energy minimization in delay-tolerant smartphone applications. *Proc. of IEEE Conf. on Computer Communications (INFOCOM)*, 2292–2300. https://doi.org/10.1109/INFOCOM.2014.6848173

[16] K. Kwon, S. Chae, and K. Woo. 2013. An application-level energy-efficient scheduling for dynamic voltage and frequency scaling. In *2013 IEEE Intl. Conf. on Consumer Electronics (ICCE).* 3–6.

[17] M. Mamman, Z. M. Hanapi, A. Abdullah, and A. Muhammed. 2019. Quality of Service Class Identifier (QCI) radio resource allocation algorithm for LTE downlink. *PLOS one* 14, 1 (2019).

[18] F. Mehmeti and C. Rosenberg. 2019. How Expensive is Consistency? Performance Analysis of Consistent Rate Provisioning to Mobile Users in Cellular Networks. *IEEE Trans. on Mobile Computing* 18, 5 (2019).

[19] R. Mittal, A. Kansal, and R. Chandra. 2012. Empowering Developers to Estimate App Energy Consumption. In *Proc. of the 18th Annual Intl. Conf. on Mobile Computing and Networking (Mobicom '12).* ACM, New York, NY, USA, 317–328. https://doi.org/10.1145/2348543.2348583

[20] P. Pillai and K.G. Shin. 2001. Real-Time Dynamic Voltage Scaling for Low-Power Embedded Operating Systems. In *Proc. of the Eighteenth ACM Symposium on Operating Systems Principles (SOSP '01).* ACM, New York, NY, USA, 89–102. https://doi.org/10.1145/502034.502044

[21] F. Qian, Z. Wang, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. 2010. TOP: Tail Optimization Protocol For Cellular Radio Resource Allocation. In *The 18th IEEE Intl. Conf. on Network Protocols.* 285–294.

[22] B. Rebekka, S. Sudheep, and B. Malarkodi. 2015. An Optimal and Priority Based Rate Guaranteed Radio Resource Allocation Scheme for LTE Downlink. *Wirel. Pers. Commun.* 83, 3 (Aug. 2015), 1643–1661. https://doi.org/10.1007/s11277-015-2468-1

[23] N. V. Sahinidis. 2019. *Optimization and Engineering.* Springer. 301–306 pages. https://doi.org/10.1007/s11081-019-09438-1

[24] M. Yan, C. A. Chan, A. F. Gygax, J. Yan, L. Campbell, A. Nirmalathas, and C. Leckie. 2019. Modeling the Total Energy Consumption of Mobile Network Services and Applications. *Energies* 12, 1 (2019), 184.

[25] Y. Yang, W. Hu, X. Chen, and G. Cao. 2019. Energy-Aware CPU Frequency Scaling for Mobile Video Streaming. *IEEE Trans. on Mobile Computing* 18, 11 (2019), 2536–2548.

[26] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli. 2015. A Control-Theoretic Approach for Dynamic Adaptive Video Streaming over HTTP. In *Proc. of 2015 ACM Conf. on Special Interest Group on Data Communication (SIGCOMM).* ACM, New York, NY, USA, 325–338. https://doi.org/10.1145/2785956.2787486

[27] J. Zhuo and C. Chakrabarti. 2008. Energy-Efficient Dynamic Task Scheduling Algorithms for DVS Systems. *ACM Trans. Embed. Comput. Syst.* 7, 2, Article 17 (Jan. 2008), 25 pages. https://doi.org/10.1145/1331331.1331341