Streaming Lower Quality Video Over LTE: How Much Energy Can You Save?

Azeem Aqil*, Ahmed O. F. Atya*, Srikanth V. Krishnamurthy*, George Papageorgiou*
*University of California, Riverside,
{*aaqil001, afath001, krish, gpagag*}@*cs.ucr.edu*

***Abstract*—Streaming video content over cellular connectivity impacts the battery consumption of a client (e.g., a smartphone). The problem is exacerbated when the channel quality is poor because of a large number of retransmissions; moreover, streaming high quality video in such cases can negatively impact user experience (e.g., due to stalling). In this paper, we develop an analytical framework which can provide the user with an estimate of "how much" energy she can save by choosing to view a lower quality stream of the video she wishes to view. The framework takes as input the network conditions (in terms of packet error rate or PER) and a coarse characterization of the video to be viewed (slow versus fast motion, resolution), and yields as output the energy savings with different resolutions of the video to be viewed. Thus empowered, the user can then make a quick, educated decision on the version of the video to view. We validate that our framework is extremely accurate in estimating the energy consumption via both simulations, and experiments on smartphones (within $\approx$ 5% of real measurements). We find that switching to a lower resolution video can potentially lead to $\approx$ 418 mW (23.2%) decrease in the consumed power for slow motion video, and $\approx$ 480 mW (26%) for fast motion video in bad channel conditions. This translates to an energy savings of 376.2 J and 432 J respectively, for video clips that are 15 minutes long.**

I. Introduction

Current reports indicate that video streaming to smartphones is experiencing an unprecedented growth [1]. The emergence of LTE (Long Term Evolution standard) [2], which offers significantly higher throughput compared to the previous generations of cellular networks, has fostered this growth. Streaming video over a cellular network however impacts the battery consumption of a client device. While User Equipment (UE) and specifically smartphones, have grown in complexity with better displays and faster CPUs, the battery technology has not been able to keep up. LTE, due to its ability to sustain significantly higher user throughput compared to 3G, exacerbates the battery problem during video downloads, since the higher downlink/uplink data rate translates to higher energy consumption [3]. It has been shown that the radio interface is a significant power consuming resource [3].

Today, adaptive bit rate streaming has become a common practice for streaming video [4]; by detecting the user's bandwidth, the quality (resolution/bit rate) of the video stream is adjusted so as to improve the user's quality of experience. However, to the best of our knowledge, changing the quality of the video to lower the energy consumption on the user's smartphone has not been previously studied. In particular, a user may choose a lower quality video stream, even when bandwidth/CPU resources are adequate, to reduce her battery drain. We seek to explore this dimension in this work.

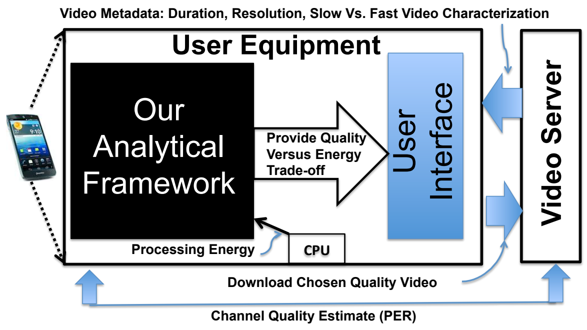
Video Metadata: Duration, Resolution, Slow Vs. Fast Video Characterization
User Equipment
Our Analytical Framework
Provide Quality Versus Energy Trade-off
User Interface
Video Server
Processing Energy
CPU
Download Chosen Quality Video
Channel Quality Estimate (PER)

Fig. 1: Applicability of our framework.

As one might expect, a user can decrease the energy consumed on her smartphone when downloading a video, by downloading a lower quality version of the same video. In some cases (poor channel conditions), downloading a lowered quality video could even improve user experience (prevent stalls); in fact, there has already recent work that advocate the use of lowered video quality (albeit in a wireline setting) to enhance user experience during downloads [5]. However, today there do not exist any tools that allow the user to get an estimate of how much energy she can save by choosing a lower quality video for downloads over cellular connectivity. Such an estimation is intricately hard because of the following reasons: **(i)** The estimation has to be made without downloading any of the versions of the video; in other words, it has to be based on a set of parameters that characterize the video to be downloaded; **(ii)** The savings from downloading a lower quality version would depend on how the UE state transitions (described later) [2] are affected by the arriving video traffic. This in turn would depend on the channel conditions perceived by the user at that time. These challenges essentially require that any framework must be holistic and tie in the interactions between the video flow characterization, the LTE scheduler and the energy transitions due to traffic arriving at the UE.

Goal and Vision: In this paper, we seek to develop an analytical framework which takes as inputs, factors that influence energy transitions at the UE (i.e., video characteristics, channel conditions) and yields as output an estimate of the energy savings possible with a lowered quality video download of a given stream. A pictorial representation of how our framework can be applied is shown in Fig. 1. Either the UE or the video server can perform a small set of calibration measurements to estimate the channel quality in terms of PER. Alternatively, a model that maps the signal strength to PER could be used to estimate the PER at the UE side. The video server would provide metadata [6] from the video clip chosen for download in the form of resolution, a coarse characterization of slow versus fast video (using tools such as AForge [7]) and the duration of the video. The UE will also locally estimate the energy required to process the received video frames for the different versions of the video. Our model yields as output an estimate of the energy consumed on the network interface with the different resolutions of the video, the user seeks to view. This combined with the the processing power provides the user with an estimate of the total energy with the different versions. She can then make an educated decision on the version of the video to download from the server.

Contributions: As our primary contribution we build a mathematical framework to capture the interactions between video traffic, the LTE scheduler, and the energy state machine at the UE. In addition to being useful for near real-time estimation of energy savings from choosing lowered quality videos for downloads, it provides a fundamental understanding of how and why different input factors influence energy consumption. In essence, the framework considers a general model of video traffic, and characterizes the arrival process of video packets at a client device (UE) after they traverse an LTE-based wireless link. The arrival process in turn calibrates the transitions between the different energy states in which the UE can reside, and the likelihood of being in each of those states. We validate our framework via extensive simulations and through experiments on a real smartphone in a variety of scenarios, thereby demonstrating its accuracy (the results are within ≈ 5 % of the real measured values) as well as generality.

Some interesting insights arising from our work are:

- Choosing a lower quality video stream incurs a small penalty in terms of the video PSNR (Peak Signal to Noise Ratio) but results in significant energy savings; specifically, a PSNR reduction of 10.1% can fetch energy savings of the order of 375 J for a video of duration 15 minutes. When a user views videos over extended periods, the energy savings can therefore be significant.
- While as expected, streaming higher resolution videos result in higher energy, the increase depends on whether it is fast or slow motion video. For example, in good channel conditions, moving to a lower resolution from a higher resolution results in a 480 mW (≈ 26 %) reduction in power (energy consumed per unit time) for fast motion video, but a 418 mW increase (≈ 23 %) for slow motion video.
- In poor channel conditions, moving to a lower resolution results almost in identical power savings for slow and fast motion video ($\approx$ a 19.5 % decrease in power). However, the savings in milliwatts is higher for fast motion video.
- For typical video transmissions, one observes that the time spent in some of the LTE energy states is insignificant regardless of the resolution.

Scope: Our framework primarily accounts for the energy consumed by the network interface on the UE. In addition, there is a processing energy consumed on a device for processing/playing back the video frames; this energy is device dependent and we use empirical results that are driven by experiments (this can be measured locally on any device).

For validation, we assume that videos are streamed using fixed bit rates. However, our framework can be applied to adaptive bit rate streaming as discussed in Section VI. We employ video resolution and PSNR as the metrics for quantifying video quality. It has been shown that the perceived video quality on mobile devices is affected by the size of the screen, user mobility, and ambient light [8]. Accounting for these factors is beyond the scope of this work and will be considered in the future. For analytical tractability, we also assume that the user is stationary during the course of video streaming.

While we validate our framework via simulations and experiments, we do not implement the complete system shown in Fig. 1. To implement such a system, we will need to make changes to the video server so as to deliver the appropriate metadata to the client UE, and also have a dynamic PER estimation tool for the LTE link; these are beyond the scope of this paper and will be considered in future work.

II. Relevant Background

In this section, we describe aspects of LTE that we seek to capture in our analytical framework.

The Radio Resource Control (RRC) State Machine: The LTE RRC state machine captures the different *energy* states that a UE can be in and has two primary states: **rrc_idle** and **rrc_connected**. The latter has three modes as shown on the right side of Fig. 2. If the UE is in **rrc_idle**, then any data exchange (even corrupted) triggers a transition to the **rrc_connected** state. The UE then enters the continuous reception mode and monitors the physical downlink control channel (PDCCH), on which control information is delivered from the base station (referred to as enB in LTE jargon [2]). At this time, the UE also starts its *continuous reception timer*, T_c. If no packets are received before the expiry of this timer i.e., in T_c, the UE enters the Short DRX mode. In this mode the UE alternates between ON and OFF periods (called DRX cycles) to save energy. If during any ON period, the UE receives data or has data to send, it returns to the continuous reception mode.

Upon entering the Short DRX mode, a different timer, T_s, is set. If there is no data transfer (received or sent) prior to the expiry of this timer, the UE enters the Long DRX mode. The Long DRX mode is similar to the Short DRX except that it has longer DRX cycles and a bigger timer value (T_l) associated with the time prior to exiting this state. Thus, $T_{tail} = T_c + T_s + T_l$ represents time for which no packets should be either received or sent in order to return to the **rrc_idle** state, and is referred to as the LTE tail period.

Packet transfers in LTE: The transitions between the states in the LTE RRC state machine are dictated primarily by packet receptions during video streaming. This in turn is handled at the MAC (and the PHY) layer of the LTE protocol stack. Our focus in this work is on the MAC layer, and we abstract the PHY in terms of packet error probabilities. Thus, we describe this layer in some detail below. A more detailed description of all the layers of LTE can be found in [2].

MAC and Physical Layers: The MAC layer implements a Hybrid Automatic Repeat reQuest (HARQ) protocol for reliable packet transfers. It also dynamically selects the Modulation and Coding Scheme (MCS) to be used at the PHY, based on the channel conditions. Transmissions in LTE are organized into frames that are 10 ms long. Each frame is divided into ten 1 ms subframes. The LTE transmission time interval (TTI) specifies the granularity at which packets are scheduled and this is done once every subframe (TTI = 1 ms).

The HARQ process: The HARQ process is different from a regular ARQ process in how it sends data and retransmissions. In LTE, *incremental redundancy* and *chase combining* are both supported. With these approaches retransmissions add required redundancies and are combined with prior transmissions to increase the likelihood of packet success.

LTE uses multiple HARQ processes simultaneously. The number of these processes, N, are chosen such that it is greater than the round trip time (RTT) of a single HARQ process (in time slots). The multiple processes transmit packets one after the other (round-robin), as a continuous stream without waiting for acknowledgments (ACKs) or negative acknowledgments (NACKs). Each process will have received an ACK or a NACK by the time it is its turn to transmit again. The number of HARQ processes for FDD (frequency division duplexed) LTE is 8 (the RTT with LTE is typically < 8 ms [2]). The number of transmission attempts that a single HARQ process makes before dropping the packet is generally 5, i.e., it makes 1 transmission and 4 retransmission attempts.

III. Our Analytical Framework

In this section, we build our mathematical framework for understanding the trade-offs between video quality and the battery consumption at the UE. As expected, the UE energy consumed will decrease if one were to lower the quality of the video that is downloaded. However, the savings will depend both on the type of video (resolution, slow versus fast video) as well as the channel conditions (poor versus good link quality).

To reiterate, we envision our framework to be used as depicted in Fig. 1. Prior to sending a video stream, the sender characterizes the video in terms of its type (slow or fast motion) and resolution. The sender and the receiver also perform a set of calibration measurements to estimate the link quality in terms of the PER. These parameters are then input to our framework, which then outputs the expected energy consumption at the UE for a download of a particular duration.

We first model the video input process. Subsequently we characterize how these video packets are processed by the LTE scheduler and thereby affect the transitions between the energy states at the UE. The notation used is summarized in Table I.

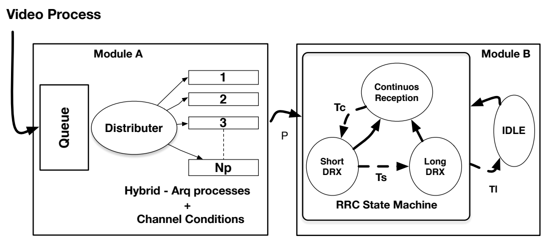

Fig. 2: A depiction of the system considered in our framework. Module A represents the LTE scheduler and Module B represents the UE RRC machine. The output of Module A, which is the probability of a packet being sent in a TTI, p, is the input for Module B.

Parameter	Description
π^{mmpp}	Steady sate vector of the 2-MMPP video process
R	Infinitesimal generator for the 2-MMPP video process
Λ	Rate Matrix of the 2-MMPP process
p	The probability of receiving something in a given TTI
N_p	The number of HARQ processes
r	Maximum number of transmission attempts of a single HARQ process
π^{harq}	The steady vector of a single HARQ process
π^{rrc}	The Steady state vector of RRC state machine Markov Chain

TABLE I: Key parameters in our mathematical framework

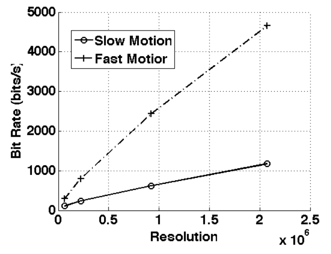

Fig. 3: Pixel Density (Resolution) VS Bit rate

A. Video Input

The input process characterizes the arrival of video packets into the LTE buffer. We assume that the video is composed of I, P and B frames [1]. A segment corresponding to an I frame is typically much larger than the MTU (maximum transmission unit) of the network and must be fragmented into multiple packets. The I frames are also less frequent than the much smaller P or B frames. The P and B frames are typically smaller than the MTU supported by the network. Since in terms of size, P and B frames are similar, we do not distinguish them in our model. In essence, we seek to capture two different phases of arrival which correspond to that of I frames and P/B frames, respectively. A natural choice for such a setup is the Markov modulated Poisson process (MMPP), which represents a doubly stochastic Poisson process [10]. The first state of the MMPP represents the arrival of I frames and the second, the arrival of P/B frames. The rate of transition from state 1 to 2 is r_1 and that from state 2 to 1 is r_2. When the process is in state 1, packets that are generated due to I frames arrive at a rate λ_1; in state 2, packets generated due to P/B frames arrive at a rate λ_2. The process can be represented by the infinitesimal generator R and the rate matrix Λ, given by:

$$R = \begin{bmatrix} -r_1 & r_1 \\ r_2 & -r_2 \end{bmatrix}, \qquad \Lambda = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \tag{1}$$

The steady state vector $\boldsymbol{\pi}^{mmpp}$ (which represents the probability of being in state $i, i \in \{1, 2\}$) is given by

$$\boldsymbol{\pi} = (\pi_1, \pi_2) = \frac{1}{r_1 + r_2}(r_2, r_1) \tag{2}$$

The expected arrival rate of the 2-MMPP process, λ_{avg}, is given by $\boldsymbol{\pi\lambda}$, where $\boldsymbol{\lambda}$ is the column vector with the diagonal elements in Λ, i.e., $\boldsymbol{\lambda} = \Lambda.e$, where, $e = (1, 1)^T$.

[1]For details on video representations please see [9].

Parameterizing video quality: In order to use the above representation we need to map the given video quality to the parameter λ_{avg} (this parameter influences the energy consumed as we will see later). Videos can be categorized as fast or slow motion videos [7]. Fast motion video is characterized by successive video frames that have little in common while slow motion video is characterized by successive frames that have a lot in common. Accordingly, fast-motion videos consume more bits in encoding than slow-motion videos. If the effective bit rate is known ($Bitrate$), $\lambda_{avg} \approx \frac{Bitrate}{E(Packet\ \ Size)}$.

If the video server can provide information with regards to the bit rates associated with different versions of the video clip as metadata, λ_{avg} can be readily computed. One can also empirically compute λ_{avg} from the resolution and the category of the video (slow or fast motion) if the bit rate is not readily available as follows.

The resolution of a video stream is essentially a function of the number of pixels per frame; the higher this number, the higher the resolution. To map the resolution to the bit rate, we perform measurements across 5 video streams for each type of video (fast or slow). We find that the bit rate is directly proportional to the resolution of the stream as shown in (shown in Fig. 3). The proportionality index depends only on whether the video is of fast or slow motion. With such a characterization (fast versus slow), we find that the prediction error is $< 5\%$. Thus, given a certain video type and its resolution, the server can determine the average arrival rate of packets to the MMPP process (using these offline measurements). Stated otherwise, the quality of the video in terms of its resolution can be used to characterize its arrival process to the LTE scheduler.

Fig. 3 demonstrates this relationship for each type of video. From the figure we can directly infer the decrease in λ_{avg} when the resolution is changed.

B. *Modeling LTE effects*

Next, we characterize the behavior of the RRC state machine at the UE given (i) the type of video being streamed and its resolution, and (ii) the state of the channel. The system has two distinct parts as shown in Fig. 2. The first part (Module A) reflects the process of transfer of the video traffic over LTE to a specific UE. The second part (Module B) characterizes the RRC state machine at the UE. The overall objective here is to determine the expected time spent in each of the RRC states. Module B essentially takes as input p, which represents the probability of receiving a packet (either a decodable packet or a corrupted packet) in a TTI. This probability essentially depends on the arrival process of the video flow, the functionality of the LTE scheduler (Module A) and the channel quality of the wireless link.

Assumptions: We make the following assumptions for analytical tractability. *First,* we assume that the UE is relatively stationary and thus, the channel conditions do not change (slow fading) for the duration of a transmission. However, we assume that they can vary between transmissions due to fading, and thus the likelihood of a packet succeeding in a transmission attempt is independent of what happens in other attempts. *Second,* we ignore synchronization issues. Note that we are only interested in the *average* energy due to the UE being in a state and not the transient energy behaviors while in a state (e.g., we only account for the average energy because of being in the Short DRX state).

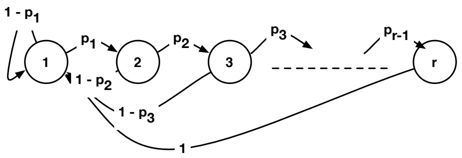

Fig. 4: Markov Chain representing a single process hybrid-arq. Each state $i, i \in \{1, 2...r\}$ represents the transmission attempt and r is the maximum number of transmissions allowed.

To begin with we assume that the UE in question is scheduled every TTI. We relax this later to account for the possibility that it gets scheduled once every N_p TTIs, on average.

Packet processing at the LTE transmitter: The packet processing at the LTE transmitter consists of two parts. First, we have an MMPP/G/1 queue to which the video packets arrive. The server of this queue essentially acts as a distributor and places the packets in one of N_p HARQ processes. Note that in order for the distributor to place a packet, at least one of the HARQ processes must be empty. Next, the HARQ process delivers the packet to the UE. First, we determine the service time distribution for our MMPP/G/1 queue and thereby compute its utilization. We later discuss how we map this to the probability of Module B receiving a packet in a TTI.

Characterizing the service time: The service time is influenced by the functions of the LTE HARQ processes. We denote it by S, which is essentially the time it takes for a packet to be assigned to a HARQ process by the distributor. We first describe how a single HARQ process functions and then discuss how we determine the distribution of S considering the N_p processes together.

Model of a HARQ process: We model a single HARQ process as a finite state discrete time Markov chain as shown in Fig. 4. The number of states, r, corresponds to the maximum number of transmission attempts allowed (after which the packet is dropped). The initial state (state 1) refers to a state wherein the HARQ process receives a new packet from the distributor. p_i is the probability of a packet being received by the UE in error, while in state i. A successful transmission while in state i, occurs with probability $1 - p_i$, and results in a transition to the initial state (state 1) and the process gets the next new packet. Because of additional FEC or change to a lower MCS (this is how HARQ functions), later re-transmissions have a greater chance of success, i.e., $p_i \geq p_{i+1}$ [2]. It is easy to see that the transition probability matrix for the

[2]In the NS3 LTE simulator we use [11], we see that all the p_is are nearly equal although this relation holds in general.

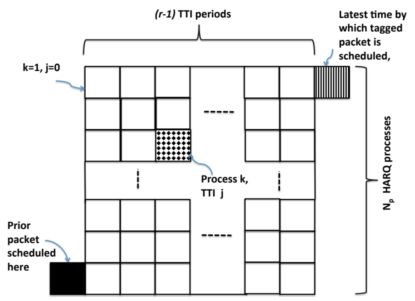
(r-1) TTI periods
Latest time by which tagged packet is scheduled,
k=1, j=0
Process k, TTI j
Prior packet scheduled here
N_p HARQ processes

Fig. 5: The state of the LTE HARQ processes

Markov chain, $\boldsymbol{P}$, is thus:

$$\boldsymbol{P} = \begin{pmatrix} 1-p_1 & p_1 & 0 & \cdots & 0 \\ 1-p_2 & 0 & p_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ \vdots & 0 & 0 & \cdots & p_{r-1} \\ 1 & 0 & 0 & \cdots & 0 \end{pmatrix} \tag{3}$$

The steady state probability vector, $\boldsymbol{\pi}^{harq} = (\pi_1, \pi_2, ..\pi_r)$, is then computed by solving the equations $\boldsymbol{\pi}^{harq}.\boldsymbol{P} = \boldsymbol{P}$ and $\sum_i \pi_i^{harq} = 1$. Specifically,

$$\pi_1^{harq} = \frac{1}{1 + \sum_{i=2}^{r} \prod_{j=1}^{i-1} p_i}, \qquad \pi_{k,\forall k>1}^{harq} = \pi_1^{harq} \prod_{i=1}^{k-1} p_k \tag{4}$$

Joint consideration of the HARQ processes: To derive the service time distribution, we need to characterize how the N_p HARQ processes function together as a whole. To recap, the N_p processes are served in a round robin fashion. In every TTI, there is only one HARQ process that is scheduled for transmission. Since $N_p >$ the RTT of in terms of TTI's, each transmitting process will receive an ACK or NACK by the time it is its turn to transmit again.

Deriving S: The distributor assigns the packet at the head of the queue to next available HARQ process. The time taken for this assignment, S, is essentially the time it takes for the distributor to find a *free* HARQ process.

Consider a packet, k, that is to be next assigned to one of the HARQ processes. For k to experience the "maximum assignment time", S^{max}, all the HARQ processes must be occupied for the maximum possible duration i.e., each process must perform the maximum number of (re)transmission attempts after k reaches the distributor. S_{max}, is then, the sum of (i) one TTI for the initial transmission of packet $k-1$ and (2) the $r-1$ transmission attempts made by each of the N_p processes subsequently (each of these processes must be occupied by a packet and must have at least performed one unsuccessful attempt already; else at least one would be empty and packet k can be assigned). It is easy to see that S^{max} is thus given by: $S^{max} = (r-1)\, N_p + 1$.

To aid the discussion on the calculation of S, we refer to Fig. 5. The rows in the figure correspond to the HARQ processes, while the columns correspond to time in terms of TTIs. Without loss of generality, assume that the prior packet that was assigned by the distributor, was assigned to the last HARQ process viz., process N_p; in the figure this preceding packet (relative to a tagged packet that we consider) is assigned to the black TTI (we refer to each block in the figure as a "slot" from here on). At this point, the tagged packet enters the distributor.

Let us assume that the tagged packet is assigned to a HARQ process, $jN_p + k$ slots later ($j \in \{0, r-1\}$, $k \in \{1, N_p\}$). This corresponds to the shaded slot in the figure. For this to happen, the following must hold true: **(i)** The processes from 1 to $k-1$, must be occupied for $j+1$ slots; **(ii)** The processes from $k+1$ to $N-p$ must be occupied for j slots; and, **(iii)** The process k must be occupied for j slots and must be free in the $(j+1)^{st}$ slot. The probability of the tagged packet assigned as above is given by Equation 5; in the following, we elaborate on how we arrive at this result. For simplicity, we simply refer to π_l^{harq} (recall Equation 4) as π_l.

Let us first consider the simple case wherein $j = 0$. If the process to which the packet is assigned (process k) is one of the first $(N_p - 1)$ processes (not the one to which the previous packet was assigned to), the conditions that must be satisfied are (i) the preceding $k-1$ processes must have been occupied and the k^{th} process is free. The likelihood of this is $(1-\pi_1)^{(k-1)}\pi_1$. If the process in question is the last (N_p^{th}) process, the requirement is that all the previous processes were occupied and the previous packet (transmitted in the black slot) was successful. The likelihood of this is $(1-\pi_1)^{(k-1)}.(1-p_1)$.

Next, let us consider the case where $j \neq 0$. To begin with let $k \neq N_p$. The probability of the first condition in the aforementioned list holding true for each of these processes, is essentially: $\pi_2.p_2.p_3 \dots p_{j+2+1} + \pi_3 p_3.p_4 \dots p_{j+3+1} + \dots\dots + \pi_{(r-1)-(j+1)}.p_{(r-1)-(j+1)} \dots p_{r-1}$. Using Equation 4 it can be shown that this long expression is simply $\sum_{l=j+2+1}^{r} \pi_l$. Together, for the $k-1$ processes (assuming that they are independent because of varying channel conditions between transmissions from these processes, due to fading), the probability of the first event is $(\sum_{l=j+2+1}^{r} \pi_l)^{(k-1)}$.

Similarly, $(\sum_{l=j+2}^{r} \pi_l)^{(N_p-k-1)}).\prod_{m=1}^{j} p_m$ gives us the probability of the second event. The last term corresponds to process N_p to which the packet preceding the tagged packet was assigned. For this packet, it is known that a transmission attempt was made in the black slot, and the last term accounts for this.

Finally, let us consider the the last required event. Since, process k is one of the first $(N_p - 1)$ processes, this event occurs with a probability $\sum_{l=j+2}^{r-1} \pi_l (1-p_l) + \pi_r$. This essentially corresponds to failed attempts in the first j slots followed by either a packet success or a packet drop for that process (k) in the j^{th} slot.

If $k = N_p$, things are slightly different since we know when the previous packet was scheduled. Thus, the likelihood of this process being free for the first time at the j^{th} TTI is simply $\prod_{l=1}^{j-1} p_l (1-p_j)$ for $0 < j \leq r-1$.

Determining the likelihood of a packet reception in a TTI: Having characterized the arrival and the service processes, we next determine the likelihood of a packet reception (either decodable or corrupted) in a TTI at the UE. A reception either transitions the UE to the active state or keeps it in one of the composite modes in that state.

$$P(S = jN_p + k) = \begin{cases} (\sum_{l=j+2+1}^{r} \pi_l)^{(k-1)}.(\sum_{l=j+2}^{r} \pi_l)^{(N_p-k-1)}.\prod_{m=1}^{j} p_m).(\sum_{l=j+2}^{r-1} \pi_l(1-p_l) + \pi_r) & j > 0, k \neq N_p \\ (\sum_{l=j+2+1}^{r} \pi_l)^{(k-1)}.\prod_{l=1}^{j-1} p_l(1-p_j) & j > 0,\ k = N_p \\ ((1-\pi_1)^{(k-1)}.\pi_1 & j = 0, k < N_p \\ (1-\pi_1)^{(k-1)}(1-p_1) & j = 0, k = N_p \end{cases} \quad (5)$$

The UE receives a packet in a TTI if the corresponding process has a packet to send. If not, there is no reception. Here, we make the following approximations. If the queue is non-empty it is unlikely that any of the N_p processes is empty and thus, the UE will receive a packet in each TTI. If the queue is empty and N of the N_p processes are occupied, the likelihood of the UE receiving a packet in a TTI is:

$$\beta = \sum_{N=1}^{N_p} \frac{N}{N_P} P(\text{No of Pkts in System} = N). \quad (6)$$

The probability of the MMPP/G/1 queue being non empty is simply given by:

$$\rho = \lambda_{avg} E(S) \quad (7)$$

Thus, the probability of the UE receiving a packet in a TTI is given by:

$$p = \rho + (1-\rho)\beta. \quad (8)$$

If ρ is high, the second term in Equation 8 tends to zero. On the other hand, if ρ is small, the value of N and thus, β is even smaller (meaning that if the queue is empty, the likelihood that some of the processes are occupied is very small). Thus, we ignore the second part and approximate $p \approx \rho$. We later validate that this approximation is reasonable via simulations (where we don't make this assumption).

C. Impact on the LTE RRC State Machine

Next, we seek to capture the impact of p on the LTE RRC finite state machine (FSM). We model the state machine as a finite-state discrete-time Markov chain parametrized by p, the probability of receiving a packet at a given TTI. The Markov chain is shown in Fig. 6. Initially the chain is in the idle state. A packet reception (with probability p), results in a transition to the continuous reception state (consisting of states 1 through T_c). A state transition from a higher RRC power state to lower state occurs if no packet is received (with probability $1-p$) for T_i, where T_i reflects the timer value associated with the currently occupied RRC state. Any packet reception triggers a timer reset and the machine transitions to the continuous reception state if in any other occupied state. The transition probability matrix for this FSM, $\boldsymbol{P}$, is given by

$$\boldsymbol{P} = \begin{pmatrix} 1-p & p & 0 & \cdots & \cdots & 0 \\ 0 & p & 1-p & 0 & \cdots & 0 \\ \vdots & p & 0 & 1-p & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & p & 0 & \cdots & \cdots & 1-p \\ 1 & p & 0 & \cdots & \cdots & 0 \end{pmatrix} \quad (9)$$

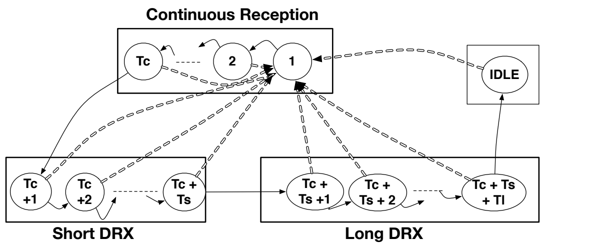

Fig. 6: Markov chain representing the LTE RRC state machine. State 0 corresponds to the Idle state. States 1 through T_c represent the continuous reception state. Dotted and solid arrows represent transitions with probability p and $1-p$, respectively.

The steady state probabilities $\boldsymbol{\pi}^{rrc}$ of being in each sub-state depicted in the figure is then computed using:

$$\boldsymbol{\pi}^{rrc}\boldsymbol{P} = \boldsymbol{P} \quad (10a)$$

$$\sum_i \pi_i^{rrc} = 1 \quad (10b)$$

To calculate the probability of each individual RRC state we can simply sum the probabilities of being in any of the component sub-states of that state (i.e., π_i^{rrc}'s that correspond to sub-states i within that state). From equations 10 one can compute the following:

$$\pi_{IDLE}^{rrc} = (1-p)^{T_{tail}} \quad (11a)$$

$$\pi_{RRC\_Connected}^{rrc} = 1 - \pi_{IDLE}^{rrc} \quad (11b)$$

$$\pi_{ContinousReception}^{rrc} = 1 - (1-p)^{T_c} \quad (11c)$$

$$\pi_{ShortDRX}^{rrc} = -(1-p)^{T_c}((1-p)^{T_s} - 1) \quad (11d)$$

$$\pi_{LongDRX}^{rrc} = (1-p)^{T_c+T_s} - (1-p)^{T_{tail}} \quad (11e)$$

where $T_{tail} = T_c + T_s + T_l$. Given the expected energy consumed in each TTI when being in each of the states of the RRC state machine, and the specifications of a video stream (resolution, slow versus fast) and its duration, one can now compute the expected energy consumed by network interface from the download.

Energy due to packet receptions: We wish to point out that while being in each of the RRC_CONNECTED states results in a baseline energy expenditure, there is additional energy consumed when packets are received [3]. This increase is directly proportional to the rate at which packets are received while in that state. This is especially important for the continuous state, since increased rate of packet receptions leads to this state almost inevitably. In such conditions, since ρ linearly increases with rate, the increase in energy is linearly proportional to ρ.

Resolution Types	EGA, CGA, HD, FHD
Streaming server	Darwin
Encoding Protocol	MPEG4
Wireless Device	Samsung Galaxy s5

TABLE II: Experimental Setup

Type	Protocol	Resolution	Fast Motion (Kb/s)	Slow Motion (Kb/s)
CGA	h.264	320x200	293	111
EGA	h.264	640x350	802	236
HD	h.264	1280x720	2433	619
FHD	h.264	1920x1080	4656	1174

TABLE III: Details of types of videos used

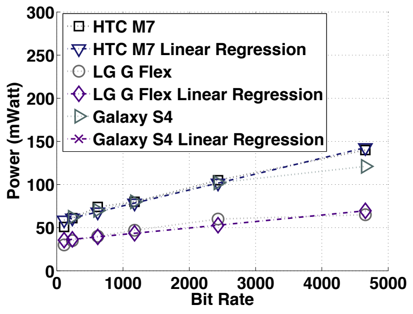

Fig. 7: Bit Rate Vs CPU power

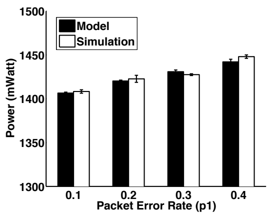

Fig. 8: Power consumption versus PER for slow motion, low resolution video

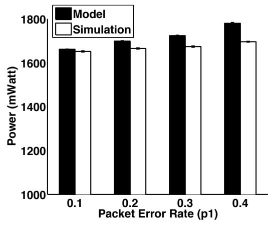

Fig. 9: Power consumption versus PER for slow motion, high Resolution video

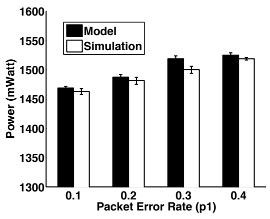

Fig. 10: Power consumption versus PER for fast motion, low resolution video

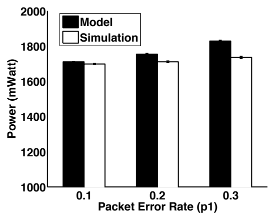

Fig. 11: Power consumption versus PER for fast motion, high resolution video

D. Impact of LTE Scheduling

Thus far, we assumed that the tagged UE is scheduled every TTI. However, depending on the number of users and the provider's policy, this may not be true. In between transmission attempts, there maybe TTIs where the UE is not scheduled, and this results in an additional service time component, viz., a waiting time wherein, no process is active and the distributor is simply in a wait mode. If the UE is scheduled every W slots (W is a random variable) and if we assume that this is independent of the service time rendered to a packet[3], the expected service time is scaled up by a factor $E(W)$ due to this scheduling policy. Thus, the probability that the queue is non-empty is now $\hat{\rho} = \rho E(W)$.

The likelihood of the UE receiving a packet in a TTI depends on whether or not the UE was scheduled in that TTI; the probability of the UE being scheduled in a TTI is $q = \frac{1}{E(W)}$ (we assume that in general $E(W)$ is small; if not, the UE is not scheduled for long durations and the queue may become unstable). If scheduled, a packet is received with probability $\hat{\rho}$. Thus, the probability of receiving a packet in any arbitrarily chosen TTI is $\hat{\rho}.q = \rho$. Thus, in essence, our prior analysis applies in this more general case as well. We have validated this via simulations.

E. Power consumption due to processing

The power consumed at the UE includes the power due to the processing of the received frames (decoding and playing the video) in addition to the power consumed on the (LTE) network interface. Our model only explicitly accounts for the latter. Higher resolution videos are typically larger in size and thus require higher processing/computing resources (and therefore power) at the UE.

The power consumed for processing is device dependent (i.e., it depends on the CPU and battery of the device). If the bit rate of the video stream is known (can be provided by the video server), the UE can locally estimate the processing power that will be consumed in addition to the LTE interface power discussed thus far. To show the viability of this approach, we experimentally profile the CPU power consumption for 3 different smartphones (Samsung Galaxy S5, HTC One M7 and LG G Flex) for downloading videos with different bit rates using the PowerTutor tool [12]. The results from our experiments are plotted in Fig. 7. We see that, for each phone, the power consumed due to CPU activities (processing/playback) increases (roughly) in a linear fashion with the bit rate; thus, the local device can easily estimate this power for different bit rate video clips once it does an initial calibration with a few video downloads.

Combining this CPU power with the estimated power consumed on the network interface (obtained using our model), we are able to get a estimate of the total power consumed by different versions of a video stream.

IV. Evaluations

In this section, we validate our analytical framework via simulations and experiments. We consider different types of video, and vary channel conditions to understand how video downloads affect the UE energy consumption. We first describe our simulation and experimental setups, discuss how we compute energy with our model, and later discuss our results.

[3]This is reasonable since the external load is independent of what the UE is attempting to download.

A. Configurations

Simulation settings: We use NS3 [13] in combination with the LTE module developed by the LENA Project [11]. We use the default channel model used in this project. Details can be found in [11]. Since we are only interested in the energy consumed at the UE, we set up a simple LTE topology with one enB serving a single UE. We log all the PHY packets that the radio processes (including packets that are in error). We vary the channel conditions. We use 8 different video clips each of which is about 60 seconds long. Videos are tagged and processed using the EvalVid [14] tool and then passed on to the enB to be transmitted to the UE. For each experiment, we perform 20 simulation runs.

Experimental Setup: Our experiments are on a Samsung Galaxy S5 LTE phone over the AT & T LTE Network (we did experiments with other models and T-Mobile and the results were similar). We connect our LTE phone to a Monsoon Power Monitor [15] and measure the power when different types of video (discussed next) are being downloaded from a Darwin server. We collect power traces for each resolution of fast and slow motion videos 10 times and estimate the average power. The details of our experimental set up are in Table II.

Video: We use four popularly used video resolutions listed in Table III. For each video, we translate the resolution into a bitrate, and estimate λ_{avg} as discussed in Section III.

Parameters for our analytical framework: To compute the power consumed, we use our analytical model in conjunction with the results in [3]. Specifically, from [3], we use the following: (a) In the idle state a constant power of 594 mW is consumed (594 mJ of energy is consumed in 1 second). In the continuous state, $power = \alpha\eta + \beta$. Where $\alpha = 51.97$ (power/Mbps), β is the baseline power in this state and is equal to 1288.04. η is the rate of reception in Mbps (and can be computed from ρ). For the Short DRX and Long DRX states, the power alternates between the power in the idle state and an active power; this active power is approximately 1680mW (note that this is larger than the continuous reception baseline power due to the power due to switching states). For the Short DRX state the switch from idle to active periods happens every 20 ms and for the long DRX state it happens every 40ms. We also use the results from [3] for the timer values that dictate the RRC state machine transitions. Specifically, $T_c = 100\ ms, T_s = 20\ ms, T_l = 11450\ ms$. We also set the value of all the p_is to be the same as p_1 as we observed in our simulations that these did not change by much from p_1.

Processing power at UE: The power consumed at the UE includes the power due to the processing of the received frames in addition to the power consumed on the network interface. Our model only explicitly accounts for the latter i.e., the power consumed due to the LTE interface. Higher resolution videos are typically larger in size and thus require higher processing/computing resources (and therefore power) at the UE. To estimate the energy consumption due to processing, instead of reinventing the wheel, we simply use the powerTutor tool [12]. This allows us to profile and estimate the power consumed by the CPU for the purposes and rendering the video. Combining this estimate with the estimated power consumed on the network interface (obtained with our model),

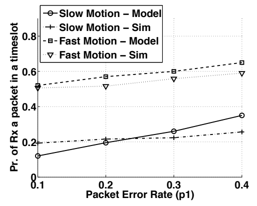

Fig. 12: The probability of receiving a packet in a TTI (decodable or corrupted) for low resolution videos

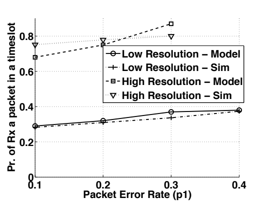

Fig. 13: The probability of receiving a packet in a TTI (decodable or corrupted) for high resolution videos

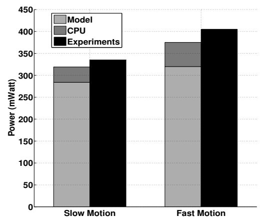

Fig. 14: Power reduction from lowering resolution from highest to lowest level under good channel conditions: Analytical and experimental results

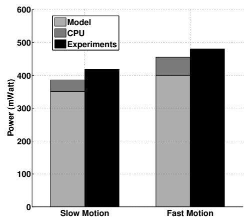

Fig. 15: Power reduction from lowering resolution from highest to lowest level in bad channel conditions: Analytical and experimental results

we are able to get a estimate of the total power consumed by different versions of a video stream.

B. Results and Inferences

Slow motion video: In Figs. 8 and 9, we present both the analytical and simulation results for slow motion videos of the lowest and highest resolutions. We see that the analytical results match well with those from simulations. We notice that at low packet error rates (PER), there is about a 280 mW (16.9 %) decrease in power when we switch from the high resolution video to low resolution video[4]. As the PER increases, the decrease is more pronounced (19.1 %, 340mW). This is because at high PERs, there are increased retransmissions, which essentially increase the probability of being in one of the active states. Furthermore, the increase in receptions trigger power increases even when in the continuous mode due to packets in error.

Fast motion video: Figs. 10 and 11 present analogous results with fast motion video. First, again, the analytical results match well with simulation results. We notice that with fast motion video, transitioning to a lower resolution results in about a 318mW (17.8 %) reduction in the consumed power in good channel conditions. In bad channel conditions we see a 384 mW (19.8 %) decrease in power consumption. This is because, with this type of video at high resolutions the packet arrival rate is as is high; the retransmissions further increase the energy consumed. We observe here that the UE is mostly in the continuous state regardless of whether the channel condition is good or bad. Note here that at $p_1 > 0.3$, the queue became unstable and we could not gather meaningful results with the simulations.

[4]To lower the video resolution, we switch from FHD to EGA.

	Decrease in PSNR	Bad Channel	Good Channel
Slow-Motion	10.11%	$376.2J$	$301.5J$
Fast-Motion	14.5%	$432J$	$364.5J$

TABLE IV: The energy saved as resolution is changed from high to low with the corresponding decrease in PSNR

Probability of reception in a TTI with slow and fast motion videos: In Figs. 12 and 13, we depict the probability of receiving a packet (decodable or corrupted) at the UE, in a TTI. As one might expect, the probability increases as the PER increases since there will be a higher number of retransmission attempts. Further, also as expected, this probability is higher for fast motion video, and for higher resolution videos, since the bit rates are higher. Most importantly, we see a close match between the simulation and analytical results; this shows that the assumptions made for analytical tractability do not influence the results by much.

Comparing analytical results with experimental results: In Fig. 14, we compare the results obtained using our framework, with that from real experiments on our Samsung Galaxy S5 LTE phone. First, we consider good channel conditions (4-5 bar coverage). Since we do not have access to the LTE PHY interface, we simply map the coverage level (5 bar) to a rough empirical characterization of the PER. Specifically, in this case, we set the PER to be 0.1 as a conservative estimate. The total power with our approach is the sum of the LTE interface power and the CPU processing power as discussed in Section III-E. We observe that the results derived from our approach are fairly good estimates of the total energy consumed in reality (within 5 % of the measured energy savings). We see that the power savings are significant with both fast and slow motion video when the user chooses a lower quality version. The savings are higher with fast motion video since, there is a bigger reduction in the video bit rate ($\approx$ 405 mW) as compared to slow motion video ($\approx$ 335 mW).

In Fig. 15, we plot the results with bad channel conditions (1 bar coverage). Here we empirically choose the highest PER that we could tolerate (in our simulations) without the queue becoming unstable (0.39) when we use our model. The total power saved is the sum of what is saved on the network interface and due to processing. We observe again that our results once again match very well with the results from the experiments (within 5% of the experimental results). We find that the savings increase to $\approx$ 480 mW and $\approx$ 418 mW respectively, for fast and slow motion video, potentially due to a decrease in the number of retransmissions.

Finally, in Table IV, we show the decrease in PSNR that comes with a corresponding decrease in energy for 15 minute long videos. Interestingly, a small decrease in PSNR (49.5 dB to 44.5 dB for slow motion and 48 dB to 41.5 dB for fast motion, which corresponds to 10.11 % and 14.4 % in the two cases, respectively) results in considerable energy savings (e.g., 376 J for slow motion video and 432 J for fast motion video in bad channel conditions). The reason for only a slight reduction is that the PSNR decreases only due to a reduction in resolution; due to TCP there are no losses that degrade quality on the channel itself. Thus, by lowering the video quality slightly, the user can conceivably gain significant energy savings.

Impact of LTE scheduling: In Fig. 16, we show the impact of scheduling the UE once every W TTI on average. W is chosen as per a uniform distribution between 1 and twice the average value. We observe that the power consumed does not change by much with varying W. This validates our analysis in Section III-D.

Times spent in each of the LTE states: In Fig. 17 we show the probability of being in each LTE energy state. We see that if λ_{avg} increases beyond an extremely low value, the UE is almost never in the IDLE state. As one might expect, as λ_{avg} increases the likelihood of being in the continuous mode increases, while the likelihood of being in the Long DRX state decreases. To begin, the likelihood of being in the short DRX state increases; here the time is shared between this state and the continuous state. But after a certain point, the probability of being in this state decreases and the UE is almost always in the continuous state.

Power consumption decreases with resolution: For completeness, we present a plot wherein we show the variations in the consumed power with fast and slow motion video as we vary the quality in terms of resolution. We assume $p_1 = 0.1$. The difference in power consumption between FHD and CGA are as reported earlier. Interim resolutions offer different trade-offs between power and video resolution.

V. Related Work

Almost all of the power models for LTE are empirically derived. In [3], the authors empirically derive a power model based on an experimental study of LTE performance. However, the work is mainly intended for developers and does not provide an understanding of how the energy consumption varies with different types of downloads, or in varying channel conditions. In [16], the authors experimentally characterize how variations in signal strength can affect UE power consumption. They develop an approach to schedule communications during periods of strong signal strength. However, they do not account for traffic characteristics (video) on RRC state transitions. The authors in [17] study the impact of signal strength on battery drain and evaluate various energy saving schemes for 3G networks; however, the impact of traffic patterns is not considered.

The authors in [18] present a model of the LTE radio interface. In [19], the authors optimize the LTE timers to save energy. Zhou et al., [20] investigate the trade off between saving power and wake up delay. In [21] the authors investigate the effects of varying the LTE parameters on user experience. However, none of these efforts are focused on video, nor do they capture the interactions between the video characteristics and the LTE energy states. In [22], the authors model the LTE HARQ process as a Markov chain to evaluate the energy implications of the HARQ process but the model does not capture the characteristics of video traffic or how the LTE scheduler influences transmissions.

Some of the work that looks specifically at energy consumption due to video transfers over LTE (e.g., [23]), target the optimization of the LTE timer values to reduce energy; they cannot be directly applied to determine energy savings due to lowered video quality. In [24], the authors study how

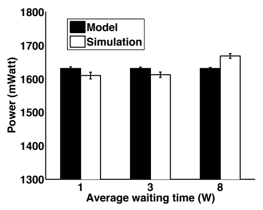

Fig. 16: Power consumption with different expected waiting times ($E(W)$).

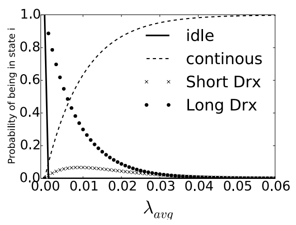

Fig. 17: The change in state occupancy with λ_{avg}

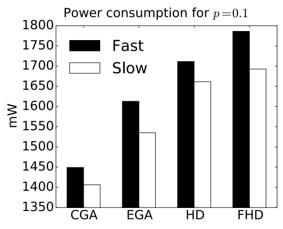

Fig. 18: Comparison in power consumption with different resolution types.

YouTube traffic affects energy. He et. al. present a model for saving energy for videos streamed over a wireless link in [25]; however, their work focuses on video encoding and does not address radio power. Lu et.al [26] try to minimize transmission power based on the state of the wireless channel. Mohapatra et. al. [27] identified several architectural techniques which can be coupled with OS level approaches to save CPU and memory energy while streaming video. In [28] the authors model the interdependence between different video packets to determine the optimal retransmission scheme for HARQ processes but do not consider the energy implications.

There is other work, where video delivery is optimized for quality (e.g., [29] and citations therein). These efforts do not explicitly consider energy due to video streaming.

VI. Discussion

Adaptive bit rate video: Current adaptive bit rate streaming technologies such as Adobe and Silverlight (used by Youtube and Netflix, respectively) are relatively simple. They work as follows. The server stores different bit rate versions of the same video. When the client requests a video, the server sends the client a list of different bit rate videos it has in its storage. The client then makes decisions on what bit rate video it wants to download on a per chunk basis, where a chunk is simply a small portion of the video [30]–[32]. Chunk sizes (in terms of viewing time) can range anywhere from 2 seconds to 10 seconds. The default bit rate is applied to the first chunk and usually corresponds to the maximum available bandwidth. Clients select the highest bit rate that is supported by their bandwidth [33]. This bandwidth is estimated by looking at the time it takes to download a chunk and the size of that chunk [33]. Current adaptive bit rate clients try to maximize bit rate given this single constraint on bandwidth. We argue that energy is a concern for clients and our model allows this factor to be taken into perspective.

Our experiments show that transferring a high-resolution video over a channel with high PER results in very high energy consumption. Bandwidth aware clients could still request this video because it can still be transmitted over the channel (bandwidth suffices). However, the user may not want to spend that much energy. Further, while PER affects the bandwidth of the link, other factors such as load or how busy the enB is at that time, could be arguably more important in determining the bandwidth available for a session as shown in [34].

Our model applies to adaptive bit rate videos; the discussion was focused on fixed bit rate videos for the purposes of clarity. Some of the experiments compute energy savings assuming a fixed bit rate video, again for the purposes of making things clear. We point out that our results, in particular Figs. 8 to 11, demonstrate the effectiveness of our model in predicting power consumed at short time scales. The same experiments also demonstrate the accuracy of our model across different bit rates. Recall that two primary inputs to our model are PER and video bit rate. Power estimates can be made as often as necessary whenever one of the inputs changes.

An energy aware adaptive bit-rate scheme would make similar chunk wise decisions as its non-energy aware counterpart. The client now has an added constraint, namely power. The UE can estimate the power consumption with any bit rate video given the state of the channel using our model. The client would now estimate (a) the highest bit rate video given its bandwidth constraints (say $Bitrate_B$ and (b) the highest bit rate video given an energy constraint ($Bitrate_E$). It would choose the bit rate that satisfies both these constraints i.e., min($Bitrate_B$, $Bitrate_E$).

Buffered Videos: Our model does not make assumptions about buffering. In fact, the streaming client used in our experiments does an initial buffering (approximately for 2 seconds).

Studies have shown that major video streaming services (e.g., Youtube and Netflix) do not buffer too much. The buffering is often limited to a couple of seconds [30]. Since Clients often terminate videos prematurely, the service providers are concerned with wasted bandwidth and for this reason never buffer more than what is necessary. Netflix clients sometimes buffer up to 2 minutes because Netflix hosts videos that are often hours long and some extra buffering is supported in case clients want to skip ahead.

Our model applies in cases where there is no buffering, any level of buffering or in the extreme case where a client buffers the entire video prior to playback. The video in question will still have to be transferred over an error prone wireless channel (which will result in queuing and retransmissions). If a client is allowed to buffer the entire video clip, all packets from the clip are inserted into the server buffer i.e., that buffer does not become empty until the entire video is transferred). This simply translates to very high values for λ_1 and λ_2. Our model can be still applied and the power consumption for different qualities can be calculated accordingly.

Motion in a video: Our framework accounts the type of dynamics in a video stream (slow or fast motion). For ease of discussion, we have assumed that the video is homogeneous in this regard. However, a video may transition between periods of fast motion to slow motion and vice versa. Tools such as AForge [7] can estimate the expected durations of fast and slow motion in such cases; with these estimates (done over chunks of buffered video at the server side), the expected energy savings can be easily computed with our framework.

Mobility If a UE is moving about a lot, this will cause the state of the channel to change (PER will change). In such cases, our model can simply recalculate power consumption with the new PER (can be potentially measured as videos are downloaded). However when a UE moves out of the coverage area of an enB an LTE handover will be necessary. Our model does not capture this characteristic.

Is a simple model enough?: One can conceive a simple model which assumes that the entire video is downloaded continuously and then displayed to the user. In such a case, there will be no RRC state transitions (as discussed later, it will be in a continuous reception state). Now, one might argue that one can calculate the time for which the interface is active provides an indicator of the energy spent. One could proportionally increase this time based on the perceived packet error rate. In practice however, two important characteristics are worth pointing out. First, as we've pointed out already, buffering is purposefully limited by popular video services Thus, it is quite possible that as network conditions change, the server side buffer becomes temporarily empty (and thus, the video is not continuously downloaded) which could trigger RRC state transitions. Our model is generic enough to take care of such cases; it is also applicable to the case where the server has a steady stream of video packets for a client for continuous downloads.

Second, we also point out here that the video stream is not necessarily transferred at the bit rate specified at the server. If that was the case, one could simply use a linear model to determine how the energy would vary with bit rate. The video traffic is shaped not only because the server may have varying loads (and thus transfers video at the specified bit rate on average, but in a bursty manner), but also because of retransmissions on the wireless channel due to varying channel characteristics. Thus, it is inevitable that the packets will experience some form of queuing at the LTE server

VII. Conclusions

In this paper, we build an analytical framework to capture the energy consumption due to video downloads over LTE. Our framework takes as input, the type of video (fast vs slow motion and resolution), the link qualities experienced (in terms of PER) and the power consumed in each LTE state. It provides a quick and effective means of determining the power consumption with various types of videos under different network conditions. We validate our framework via extensive simulations and real experiments.

Acknowledgements: This work was supported by the NSF NeTS grant 1320148. We thank our shepherd Samir Das for his constructive comments which helped us significantly improve the paper.

References

[1] "Cisco Visual Networking Index: Forecast and Methodology," http://bit.ly/LVhmuL.

[2] S. Sesia, I. Toufik, and M. Baker, *LTE - The UMTS Long Term Evolution: From Theory to Practice*, 2nd ed. Wiley, Sep. 2011.

[3] F. Q. J. Huang, A. Gerber, S. S. Z. M. Mao, and O. Spatscheck, "A close examination of performance and power characteristics of 4G LTE networks," ACM MobiSys 2012.

[4] B. Vandalore, W. chi Feng, R. Jain, and S. Fahmy, "A survey of application layer techniques for adaptive streaming of multimedia," *Real-Time Imaging*, vol. 7, no. 3, pp. 221 – 235, 2001.

[5] A. Anand, A. Balachandran, A. Akella, V. Sekar, and S. Seshan, "Enhancing video accessibility and availability using information-bound references," in *ACM CoNEXT*, 2013.

[6] S. Yi, S. K. Fayazbakhsh, Y. Guo, V. Sekar, Y. Jin, D. Kaafar, and S. Uhlig, "Trace-Driven Analysis of ICN Caching Algorithms on Video-on-Demand Workloads," in *ACM CoNEXT*, December 2014.

[7] "AForge.NET," http://www.aforgenet.com/framework/features/motion_detection_2.0.html.

[8] J. Xue and C. W. Chen, "Mobile video perception: New insights and adaptation strategies," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 8, no. 3, pp. 390–401, 2014.

[9] A. C. Bovik, *The Essential Guide to Video Processing*. Academic Press, 2009.

[10] O. Ibe, "Markov Process for Stochastic Modelling," Academic Press, 2008.

[11] "LENA," http://bit.ly/UEleow.

[12] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang, "Accurate online power estimation and automatic battery behavior based power model generation for smartphones," in *CODES/ISSS*, 2010.

[13] "NS3," http://www.nsnam.org/.

[14] "EvalVid," http://bit.ly/UOPQ6v.

[15] "Monsoon Power Monitor," http://bit.ly/1lh8JX1.

[16] A. Schulman, V. Navda, R. Ramjee, N. Spring, P. Deshpande, C. Grunewald, V. N. Padmanabhan, and K. Jain, "Bartendr: A practical approach to energy-aware cellular data scheduling," in *ACM Mobicom*, 2010.

[17] N. Ding, D. Wagner, X. Chen, A. Pathak, Y. C. Hu, and A. Rice, "Characterizing and modeling the impact of wireless signal strength on smartphone battery drain," in *ACM SIGMETRICS*, 2013.

[18] A. Jensen, M. Lauridsen, P. Mogensen, T. Srensen, and P. Jensen, "LTE UE power consumption model: For system level energy and performance optimization," in *IEEE VTC*, 2012.

[19] C. Bontu and E. Illidge, "DRX mechanism for power saving in LTE," *IEEE Comm. Magazine*, 2009.

[20] L. Zhou, H. Xu, H. Tian, Y. Gao, L. Du, and L. Chen, "Performance analysis of power saving mechanism with adjustable DRX cycles in 3GPP LTE," in *IEEE VTC 2008 (Fall)*, 2008.

[21] T. Kolding, J. Wigard, and L. Dalsgaard, "Balancing power saving and single user experience with discontinuous reception in LTE," in *IEEE ISWCS*, 2008.

[22] J. Dohl and G. Fettweis, "Energy aware evaluation of lte hybrid-arq and modulation/coding schemes," in *Communications (ICC), 2011 IEEE International Conference on*, 2011, pp. 1–5.

[23] 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*, 2013.

[24] Y. Xiao, R. Kalyanaraman, and A. Yla-Jaaski, "Energy consumption of mobile YouTube: Quantitative measurement and analysis," in *NGMAST*, 2008.

[25] Z. He, Y. Liang, L. Chen, I. Ahmad, and D. Wu, “Power-rate-distortion analysis for wireless video communication under energy constraints,” *IEEE Transactions on Circuits and Systems for Video Technology*, 2005.

[26] X. Lu, Y. Wang, and E. Erkip, “Power efficient h.263 video transmission over wireless channels,” in *International Conference on Image Processing*, 2002.

[27] S. Mohapatra, R. Cornea, N. Dutt, A. Nicolau, and N. Venkatasubramanian, “Integrated power management for video streaming to mobile handheld devices,” in *ACM Multimedia*, 2003.

[28] L. Badia and A. Guglielmi, “A markov analysis of automatic repeat request for video traffic transmission,” in *A World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2014 IEEE 15th International Symposium on*, 2014.

[29] A. Pande, V. Ramamurthi, and P. Mohapatra, “Quality-oriented video delivery over lte using adaptive modulation and coding,” in *IEEE GLOBECOM*, 2011.

[30] C. Müller, S. Lederer, and C. Timmerer, “An evaluation of dynamic adaptive streaming over http in vehicular environments,” in *Proceedings of the 4th Workshop on Mobile Video*, ser. MoVid ’12. ACM, 2012.

[31] S. Akhshabi, A. C. Begen, and C. Dovrolis, “An experimental evaluation of rate-adaptation algorithms in adaptive streaming over http,” in *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, ser. MMSys ’11. ACM, 2011.

[32] 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. ACM, 2011.

[33] J. Jiang, V. Sekar, and H. Zhang, “Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive,” in *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT ’12. ACM, 2012.

[34] A. Chakraborty, V. Navda, V. N. Padmanabhan, and R. Ramjee, “Coordinating cellular background transfers using loadsense,” in *Proceedings of the 19th Annual International Conference on Mobile Computing & Networking*, ser. MobiCom ’13. ACM, 2013.