Joint Media Streaming Optimization of Energy and Rebuffering Time in Cellular Networks

Zeqi Lai*, Yong Cui*, Yayun Bao†, Jiangchuan Liu‡, Yingchao Zhao§ and Xiao Ma*
*Department of Computer Science and Technology, Tsinghua University, Beijing, PR China
Email: uestclzq@gmail.com, cuiyong@tsinghua.edu.cn, thumax9@gmail.com
†State Key Laboratory of Networking and Switching Technology,
Beijing University of Posts and Telecommunications, Beijing, PR China
‡School of Computing Science, Simon Fraser University, Metro-Vancouver, Canada
§Department of Computer Science, City University of Hong Kong, Hong Kong, PR China
Email: jimmybao0730@gmail.com, jcliu@cs.sfu.ca, zhaoyingchao@gmail.com

***Abstract*—Streaming services are gaining popularity and have contributed a tremendous fraction of today's cellular network traffic. Both playback fluency and battery endurance are significant performance metrics for mobile streaming services. However, because of the unpredictable network condition and the loose coupling between upper layer streaming protocols and underlying network configurations, jointly optimizing rebuffering time and energy consumption for mobile streaming services remains a significant challenge. In this paper, we propose a novel framework that effectively addresses the above limitations and optimizes video transmission in cellular networks. We design two complementary algorithms, Rebuffering Time Minimization Algorithm (RTMA) and Energy Minimization Algorithm (EMA) in this framework, to achieve smoothed playback and energy-efficiency on demand over multi-user scenarios. Our algorithms integrate cross-layer parameters to schedule video delivery. Specifically, RTMA aims at achieving the minimum rebuffering time with limited energy and EMA tries to obtain the minimum energy consumption while meeting the rebuffering time constraint. Extensive simulation demonstrates that RTMA is able to reduce at least 68% rebuffering time and EMA can achieve more than 27% energy reduction compared with other state-of-the-art solutions.**

I. Introduction

Video streaming services, such as YouTube, Netflix, and Hulu are gaining immense popularity on mobile networks thanks to the rapid development of high speed 3G/LTE technologies. It has been estimated that more than half of the mobile data traffic is now contributed by streaming applications and it will grow to 69% by 2018 [3]. Significant efforts have focused on optimizing the playback in mobile environments [6], [10], [24], [27] to satisfy user experience. Besides the playback fluency, as the battery technologies have not been keeping up with the rapid advancement of mobile devices [20], energy consumption is also becoming an important concern when designing and implementing streaming protocols for mobile devices. Because of the sustained transferring property of streaming services, wireless interfaces are forced to be powered up and are responsible for a large amount of energy consumption when running video streaming applications in power-constraint mobile devices. It is crucial for both services providers and mobile network operators to optimize the user experience while achieving energy efficiency.

Maintaining the streaming fluency while optimizing the energy consumption in mobile environments is extremely challenging. First, the inherent variable bandwidth in mobile networks makes it difficult to guarantee smooth playback. The variability is caused by many unpredictable external factors such as signal strength, mobility and workload changes at the base station. The dynamic wireless environment will cause inevitable viewing interruption and rebuffering delay, i.e. the time interval separating the viewing time of a streaming event [32]. Previous works try to ensure smooth playback by limiting the waiting queue [17], satisfying either the video packet deadline [19] or the required data rate in each time slot [30]. However, only considering video tasks' deadline ignores the viewing interruption during the playback. The variable bandwidth and the resource competition in multi-user scenarios also make it difficult to satisfy the required data rate in every slot.

Second, the loose coupling between the streaming protocols and the underlying radio resource management also results in serious energy waste. In typical 3G/LTE networks, user equipments (UEs) follow the radio resource control (RRC) protocol for dynamic acquisition of radio resource. According to the RRC state machine, a UE will not switch to a low power state immediately after data transmission and will keep in the high-power state until an inactivity timeout. Such mechanism will lead to significant energy waste, namely tail energy, especially for typical streaming protocols implemented in real mobile video services. For example, YouTube, Dailymotion and Vimeo have implemented the ON-OFF protocol in their Android client player for pushing video chunks, through which a player uses a persistent TCP connection and simply stops reading from the TCP socket during an OFF period [14]. During the OFF period no data is received but the UE still keeps in high power state, resulting in significant energy waste.

Aiming at jointly optimizing the rebuffering time and the energy consumption for mobile streaming services, in this paper we propose a streaming content scheduling framework that can effectively address above challenges. To optimize the video delivery and manage the resource competition under a

0190-3918/15 $31.00 © 2015 IEEE
DOI 10.1109/ICPP.2015.49

dynamic wireless environment, our framework is designed as a gateway solution across multiple video flows of all connected users. The framework works in two complementary modes on demand: the rebuffering time minimization (RTM) mode for minimizing the rebuffering time while limiting the energy consumption in a desirable range, and the corresponding energy minimization (EM) mode that can optimize the energy consumption while limiting the rebuffering time in a tunable tolerance range. Furthermore, we design two online scheduling algorithms for each mode respectively. Through integrating cross layer parameters such as signal strength, RRC timer and video bit-rate, the proposed online algorithms are able to judiciously make data allocation decisions to jointly optimize the rebuffering time and the energy cost in various scenarios.

Our gateway-level solution has minimum dependence on specific cellular technologies, making it possible to be implemented in multiple 3G/LTE wireless access networks, such as UMTS, HSPA, LTE or LTE-A. In summary, our contributions in this paper can be concluded as:

- We propose a scheduling framework with two complementary modes to jointly optimize the rebuffering time and the energy consumption for mobile streaming services. Based on our framework, we integrate a set of cross layer parameters to model the rebuffering time and the energy consumption over cellular networks.
- We establish the hardness of the rebuffering time minimization problem and design an online algorithm RTMA in multi-users scenario to optimize the rebuffering time while keeping the energy consumption limited.
- We design a non-prediction online algorithm EMA based on Lyapnuov optimization to achieve the energy conservation while meeting the rebuffering time limitation. We also prove the bound of energy and rebuffering time theoretically.

The rest of the paper is organized as follows. Section II reviews existing proposals on performance optimization for mobile streaming applications. Section III presents our framework overview and modeling. In Section IV, we introduce the rebuffering time minimization problem and design an online algorithm. The energy minimization problem is raised and addressed in Section V. We evaluate the performance of our solution in Section VI, and finally conclude our work in Section VII.

II. Related Work

A large number of efforts have been focused on optimizing the video delivery in terms of user experience and energy consumption for mobile streaming services. In the following, we discuss the most important parts of them.

Several kinds of optimization techniques are proposed to deliver video for mobile users to guarantee user experience before. Ali *et al.* aim at characterizing the buffer size needed for a target probability of playback interruption over video session [6]. Xu *et al.* demonstrate the possibility of forecasting the short-term performance in cellular networks and uses it to improve the performance of mobile applications [24]. In [10], Chen *et al.* propose a scheduling framework to maintain high resource utilization and allocate stability of bit-rates to each user. Their framework is close to our work as a gateway solution in multiple cellular networks. Draxler *et al.* apply a cross-layer scheduling to improve the QoE of streaming users in [13]. They predict the channel capacity and schedule the data transmission to reduce playback interruptions and provide the best possible video quality to users. In [22], Dutta *et al.* adopt the Markov channel model to manage stalling for multiple streams over a time-varying bandwidth-constrained wireless channel. However, above works only focus on improving user experience but ignore the energy consumption of mobile devices.

There is a plethora of work, e.g., [8], [9], [17], [21], [23], [29], on saving energy consumption for communicating over 3G/LTE networks. RadioJockey [21] explores program execution in order to optimize the use of Fast Dormancy in 3G communication and reduce energy consumption. TOP [9] leverages the similar methodology but it needs active participation from programmers. Both of them are not targeting streaming service. Bartendr [8] reduces energy through prefetching content in condition of good signal strength. It needs the user to store historical signal record for prediction. SALSA [17], eTime [23] and PerES [29] have similar strategies on scheduling transmission to reduce energy consumption. They follow the principle that defers transmission until finding an appropriate time, depending on buffer queue length, or performance degradation for specific applications. SALSA, however, ignores the significant energy waste during tail time. Bartendr, eTime and PerES are all loosely coupled with the key metric for streaming services such as viewing latency or viewing interruption. They do not address the impact on viewing fluency when scheduling data to achieve energy efficiency. The authors of [31] design an energy-efficient resource allocation scheme over multi-users scenarios in LTE system but it ignores the significant tail energy. Hoque *et al.* make a detail analysis between burst size and power consumption of smart phones and design a cross-layer multimedia delivery system called EStreamer to save energy [16]. However this system ignores the impact of signal strength. Moreover, these previous works have not considered the resource competition among multiple users at the base station, which may cause serious unfairness.

In addition, many works measure and characterize 3G/LTE networks that touch on the subject of our work. Authors of [19] give a comprehensive analysis for the RRC state machine and the energy consumption in 3G networks. The LTE power characteristics are also studied in [11]. Shafiq *et al.* present a large scale measurement study on characterizing the impact of cellular network performance on mobile video user engagement [18]. These measurements offer important reference for parameter configuration in this paper.

Our work focuses on jointly optimizing the rebuffering time and the energy consumption for mobile streaming in cellular networks. We fill the gap between upper streaming protocol and underlying network through designing an integrated scheduling framework for video delivery, while considering

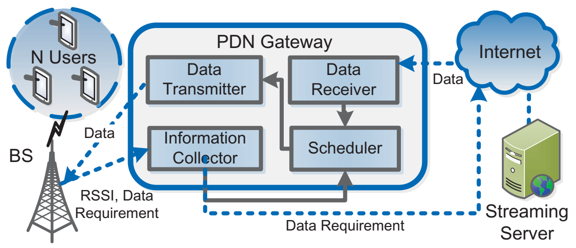


Fig. 1. Framework overview

the resource competition at the base station.

III. Framework and Modeling

In this section, we first present the overview of our framework. Based on the proposed framework, we then model the data transmission, the energy consumption and the rebuffering time of video streaming over cellular networks.

A. Framework Overview

Figure 1 shows the overview of our video delivery optimization framework. As a gateway solution, our framework works between the base station (BS) and Internet to manage the resources of each BS independently and schedule the video traffic. We design the framework at the packet data network (PDN) gateway (e.g. GGSN) because typical BSs have minimal computational resources since they need to be deployed at a large number of locations. Implementing our framework on BSs would require substantial increase in computational and memory requirements of BSs. Summarily, there are four primary components in our framework.

Data Receiver is designed as the queue to buffer downlink streaming data before forwarding them to users. Data Receiver leverages the resource slicing technique [26] to separate video flows among other downlink traffic. Only video traffic will be scheduled by our framework.

Information Collector is responsible for collecting the signal strength variation and the video data rate requirements from N users. These parameters are then sent to the Scheduler for resource allocation. Since most recent streaming protocols are based on HTTP or RTSP. The bit rate requirement can be obtained from DPI middleboxes that are part of existing cellular networks [2].

Scheduler works in two alternative modes, Rebuffering time minimization (RTM) mode and energy minimization (EM) mode for different scenarios. We design two algorithms in Section IV and Section V respectively for each mode. Both algorithms use required video data rate, real time signal strength, and BSs' available capacity to allocate resources and to achieve the jointly optimization.

Data Transmitter forwards the video content to each user according to the allocation made by Scheduler.

In the runtime, the operator first selects an appropriate mode depending on the concrete demand. RTM mode works for minimizing the rebuffering time while limiting the energy consumption in a certain range, while EM mode is able to optimize the energy consumption at the constraint of rebuffering time. Once the running mode is determined, the Information Collector extracts the required data rate and the signal strength information (e.g. RSSI) from the user request. Then these requests are forwarded to the streaming server and video contents are fetched and buffered at the Data Receiver. After that the Scheduler makes the data allocation decision to jointly optimize the rebuffering time and the energy and finally video contents are delivered to the users.

B. Transmission Model

In this subsection, we present the transmission model of streaming in cellular networks. For ease of exposition, we assume that time is slotted and each slot lasts for τ seconds. Further, we divide the downlink traffic from a streaming application into multiple data shards.

Definition 1. ***Data shard*** *of a streaming application is defined as the amount of data in byte transmitted in one slot for a single user.*

The received data shard can be used by streaming application only when it is fully accepted. At the beginning of slot n, the Scheduler allocates data shard $d_i(n)$ to user $i(i \in 1, 2, \cdots N)$. And $d_i(n)$ can be used only in the next slots. According to the transmission protocol of 3G/LTE [1], frame is the basic data unit during the transmission process, and in the physical layer, data are transmitted in frames with fixed length (denoted as δ) decided by the spreading factor. We define a decision parameter $\varphi_i(n)$ to indicate the data unit amount of user i transmitted in slot n, and we have $d_i(n) = \varphi_i(n)\delta$.

In 3G/LTE networks, both the throughput and the power of a mobile device are influenced by the channel quality, which can be characterized by Signal to Interference plus Noise Ratio (SINR) [12] or Received Signal Strength Indicator (RSSI) [8], [28]. As RSSI can be easily acquired on mobile devices, in this paper, we take RSSI value as a metric to indicate the channel quality. Since one slot can be very short, we assume that the signal strength of a user remains the same in a single slot. We make the following definitions in our paper.

Definition 2. ***Signal Strength*** *$sig_i(n)$ is defined as the signal strength value of user i in slot n.*

Previous works [8], [28], [29] have already demonstrated the relationship between the signal strength and the throughput. First, we define the function of throughput,

Definition 3. ***Throughput*** *$v(sig)$ is defined as the maximum data amount transmitted per second in byte under a certain signal strength.*

In real transmission, the data transmitted to a user in one slot is finite. The decision parameter $\varphi_i(n)$ should satisfy,

$$\varphi_i(n) \leq \lfloor \frac{\tau \cdot v(sig_i(n))}{\delta} \rfloor \tag{1}$$

Besides, due to the limited serving capability, it might be impossible for the BS to serve infinite users simultaneously.

We denote the serving capacity $S(n)$ as the maximum throughput of the BS in slot n. Thus the decision parameters among all users should satisfy:

$$\sum_{i=1}^{N} \varphi_i(n) \leq \lfloor \frac{\tau \cdot S(n)}{\delta} \rfloor \tag{2}$$

C. Energy Model

The total energy consumption of a device can be divided into the data transmission energy and the tail energy [19]. As we discussed above, the instant power of a device can be estimated via the real time signal strength [28]. Accordingly we give the definition of power consumption.

Definition 4. ***Power Consumption*** *$P(sig)$ is defined as the energy consumed per byte under a ceratin signal strength.*

When the Scheduler allocates $\varphi_i(n)$ data units to user i in slot n, the transmission energy $E_{i,trans}(n)$ can be computed as,

$$E_{i,trans}(n) = P(sig_i(n)) \times \varphi_i(n) \times \delta \tag{3}$$

If the Scheduler does not allocate data to a user in a slot, the mobile device will experience the tail process [19]. With regard to energy consumption, both 3G and LTE network interfaces have some similar features. Specifically, both of them have mechanisms to avoid the frequent switching between the idle state and the powered on state. When communicating, they operate in different modes, which correspond to different power states. In 3G networks, these modes, namely CELL_DCH (high power state), CELL_FACH (medium power state), and CELL_IDLE (low power state), map to different channel allocations [4]. Similarly, an LTE device can be either in the RRC_CONNECTED (high power state) or the RRC_IDLE (low power state) state [5]. The transition from an active to a more passive state is executed based on inactivity timers, and the timer value of 3G and LTE may extend to ten seconds. Consequently, mobile devices will not immediately switch to a low power state after data transmission until the regular timers expire. The energy consumption by keeping the radio on for the period specified by the timer is often called tail energy.

We define P_d and P_f as the instant power in CELL_DCH and CELL_FACH state, respectively in 3G networks. Let T_1 and T_2 denote the timers in state demotion of CELL_DCH $\rightarrow$ CELL_FACH and CELL_FACH $\rightarrow$ IDLE. If t is the time interval between two transmissions, the tail energy can be formulated as,

$$E_{tail}(t) = \begin{cases} P_d \cdot t, & 0 \leq t < T_1 \\ P_d \cdot T_1 + P_f \cdot (t - T_1), & T_1 \leq t < T_2 + T_1 \\ P_d \cdot T_1 + P_f \cdot T_2, & t \geq T_2 + T_1 \end{cases} \tag{4}$$

The tail energy in a slot is highly correlated with the resource allocation in previous slots. We denote the tail energy consumption of user i in slot n as $E_{i,tail}(n)$ which can be obtained by Eq. (4) if there is no transmission.

We define $E_i(n)$ as the total energy consumption of user i in slot n, depending on whether the resources are allocated to user i for receiving data, i.e.,

$$E_i(n) = \begin{cases} E_{i,trans}(n), & \varphi_i(n) \neq 0 \\ E_{i,tail}(n), & \varphi_i(n) = 0 \end{cases} \tag{5}$$

We define the energy average value as $\overline{PE}(\Gamma)$,

$$\overline{PE}(\Gamma) = \frac{1}{N\Gamma} \sum_{i=1}^{N} \sum_{n=0}^{\Gamma-1} E_i(n) \tag{6}$$

where Γ is the scheduling period.

D. Rebuffering Time

Due to the variance of transmission rate and the resource competition of multiple users, video display interruption may occur if the buffer in the client is lack of data. We consider the rebuffering time as the time period for resuming playback when the playback gets stuck. In other words, smooth playback demands the total rebuffering time to be small enough. In our model, we consider the video bit rate changes over time but remains same in a slot. Let $p_i(n)$ denote to be the requested data rate of the video for user i in slot n. The playback duration maintained by this data shard $d_i(n)$ can be formulated as $t_i(n) = d_i(n)/p_i(n)$.

Definition 5. ***Remaining Occupancy*** *$r_i(n)$ is defined as the playback duration that can be maintained by data in user i's buffer at the beginning of slot n.*

Note that the data in user i's buffer include both the received data in current slot and the remained data. And as a data shard can be used when fully accepted, define $r_i(0) = 0$ and $r_i(n)$ can be formulated as,

$$r_i(n) = \mathbf{max}\{r_i(n-1) - \tau, 0\} + t_i(n-1) \tag{7}$$

Viewing interruption will occur when the remaining occupancy is insufficient to support playback.

Definition 6. ***Rebuffering time*** *$c_i(n)$ is the duration that lacks of data for playback of user i in slot n.*

We employ the rebuffering time as a metric to characterize the playback experience. The smooth playback for better user experience demands $c_i(n)$ to be small enough. Rebuffering time in slot n is related with the remaining occupancy $r_i(n)$ in user's buffer:

$$c_i(n) = \begin{cases} \mathbf{max}\left\{\tau - r_i(n), 0\right\}, & m_i(n) < M_i \\ 0, & m_i(n) \geq M_i \end{cases} \tag{8}$$

where $m_i(n)$ is the elapsed playback time for user i and M_i is the total playback time needed by user i. We define the average rebuffering time as $\overline{PC}(\Gamma)$,

$$\overline{PC}(\Gamma) = \frac{1}{N\Gamma} \sum_{i=1}^{N} \sum_{n=0}^{\Gamma-1} c_i(n) \tag{9}$$

where Γ is the total number of time slot during scheduling.

Algorithm 1 RTMA

Inputs: User number N, Bandwidth $S(n)$, Slot length τ, Signal strength $\text{sig}_i(n)$, Required data rate $p_i(n)$

Outputs: Data Unit Allocation $\varphi_i(n), i \in [1, N]$

```
1:  $d_i(n) \leftarrow 0, \varphi_{sup}(i) \leftarrow 1 \; i \in [1, N]$;
2:  Sort $p_i(n)$ in ascending order;
3:  calculate $\varphi_{need}(i)$;
4:  while $S(n) > 0$ and $\varphi_{sup}(i) > 0, \exists \; i \in [1, N]$ do
5:      for $i = 1 \rightarrow N$ do
6:          if $\text{sig}_i(n) \geq \phi$ then
7:              calculate $\varphi_{sup}(i)$;
8:              if $\varphi_{sup}(i) > \varphi_{need}(i)$ then
9:                  $\varphi_i(n) = \varphi_i(n) + \varphi_{need}(i)$, $S(n) = S(n) - \delta\varphi_{need}(i)$;
10:             else
11:                 $\varphi_i(n) = \varphi_i(n) + \varphi_{sup}(i)$, $S(n) = S(n) - \delta\varphi_{sup}(i)$;
12:             end if
13:         end if
14:     end for
15: end while
16: return $\varphi_i(n), i \in [1, N]$;
```

IV. Rebuffering Time Minimization

Streaming services are delay-sensitive. To have a good viewing experience, the users may expect to obtain smooth playback even though sacrificing more energy consumption. However, the battery capacity is limited. We assume that there is an energy bound Φ, which is the expected maximum cost described as:

$$\overline{PE}(\Gamma) \leq \Phi \tag{10}$$

The first mode of our Scheduler, RTM, is designed to optimize the global rebuffering time while limiting the energy consumption in a certain range for multi-user scenarios. This problem, namely rebuffering time problem, can be formulated as:

$$\begin{array}{ll} \textbf{min} & \overline{PC}(\Gamma) \\ \text{s.t.} & \text{Constraints (1), (2) and (10)} \end{array} \tag{11}$$

This problem can be proved to be NP-hard by transforming the multi-choice Knapsack problem to it. We design an online heuristic algorithm, Rebuffering Time Minimization Algorithm (RTMA), to find an approximate solution for this problem. When getting the same amount of data, the rebuffering time is affected by the requested data rate $p_i(n)$. The basic idea of RTMA is to preferentially guarantee the data requirement of the users with the smallest required data rate while satisfying the energy limit.

Algorithm 1 shows how we allocate data to each user. The objective of RTMA is to decide how many data units should be allocated for each user in a slot. Given the same amount of data in a slot, the smaller the required data rate is, the longer time that the data will maintain a smooth playback. Hence in each slot, our algorithm first sorts the users based on the required data rate and initiates the allocation array to zero (steps 1-2). In step 3, we calculate the data requirement $\varphi_{need}(i)$, as the minimum for guaranteeing smooth playback, where $\varphi_{need}(i) = \lceil \tau p_i(n)/\delta \rceil$. After initiation, in order to improve the bandwidth utilization, we iteratively update $\varphi_i(n)$ until the available resource S is exhausted or the allocated size has reached its bound determined by its real time signal strength (steps 4-15). To meet the energy constraint in Eq. (10), we apply an approximate conversion to map the energy consumption to signal strength, according to Definition 4. Assume that a certain signal strength, say ϕ, satisfies the equation,

$$\Phi = \frac{1}{2}[P(\phi) \times v(\phi) \times \tau + \tau P_{tail}] \tag{12}$$

where Φ is the energy limitation estimated as the mean of the maximum transmission power and the tail energy in a slot. Given the energy constraint Φ, we can derive ϕ from Eq. (12). Thus we can consider ϕ as a "signal strength limitation" that if the signal is weaker than ϕ for user i, Scheduler will not allocate data to user i to satisfy Φ. This limitation is given in step 6 for each iteration. Note that the conversion there is stricter than Eq. (10) because we require that each user should satisfy the limitation.

If the signal strength is strong enough, we calculate $\varphi_{sup}(i)$, defined as the available data units that the BS can still support for allocation, before updating $\varphi_i(n)$ in step 7. The value of is $\varphi_{sup}(i)$ determined by the available throughput of user i and the capacity of the BS. In each iteration, $\varphi_{sup}(i)$ is updated:

$$\varphi_{sup}(i) = \lfloor \min\{v(sig_i(n)) - \delta\varphi_i(n), \tau S(n) - \delta\varphi_{sum}(n)\}/\delta \rfloor$$

where $\varphi_{sum}(n)$ is the total data units that has already been decided to allocate to users in slot n, and can be calculated as $\varphi_{sum}(n) = \sum_{i=1}^{N} \varphi_i(n)$. In order to maintain the smooth playback of the next slot and serve more users as possible, we only assign the needed data in a iteration. When allocating data in each iteration, if $\varphi_{sup}(i)$ is larger than $\varphi_{need}(i)$, RTMA appends $\varphi_{need}(i)$ to $\varphi_i(n)$ (step 9). Otherwise $\varphi_i(n)$ is updated by an increment of $\varphi_{sup}(i)$ (step 11). Finally the allocation result is obtained after iteratively updating $\varphi_i(n)$ in slot n.

The Algorithm RTMA is local optimal in one slot without the energy limitation Eq. (10). When the energy limitation exists, the algorithm will not be local optimal as RTMA applies a stricter limitation than Eq. (10). Some users could sacrifice their playback quality to satisfy the energy limitation of the whole system.

V. Energy Minimization

The battery life is also an important consideration for mobile users. To conserve the energy consumption, EM mode can be used to optimize the energy consumption and restrict the rebuffering time in a tolerable range. We have the following average rebuffering time constraint:

$$\overline{PC}(\Gamma) \leq \Omega \tag{13}$$

where Ω is the bound of rebuffering time to guarantee the viewing experience. The above optimization can be described as an energy minimization problem.

$$\begin{array}{ll} \textbf{min} & \overline{PE}(\Gamma) \\ \text{s.t.} & \text{Constraints (1), (2) and (13)} \end{array} \tag{14}$$

This problem can also be proved to be NP-hard by transforming the multi-choice Knapsack problem to it. We design an online algorithm that can be efficiently applied. It takes the user requirement on the viewing performance into account, and ensures stable system performance under dynamic user traffic and channel variance. Let Γ_i denote the total slot number of user i in a video session. Combined with Eq. (7) and Eq. (8), we can get the total rebuffering time as Γ_i minus the actual playback time,

$$PC_i(\Gamma_i) = \tau\Gamma_i - \sum_{n=0}^{\Gamma_i-1} t_i(n) \tag{15}$$

To deal with the rebuffering time limitation, we design a special queue mechanism to control the rebuffering time. Note that the receiving data of user i in slot n is $d_i(n)$, which can maintain the playback for $t_i(n) = d_i(n)/p_i(n)$ seconds and the slot length is τ. Then, we simply construct the rebuffering time recursion relation as follows,

$$PC_i(n+1) = PC_i(n) + \tau - t_i(n) \tag{16}$$

For all the scheduling period, $PC_i(n)$ is similar with the sum of the rebuffering time. We can get the Eq. (15) from Eq. (16) through accumulation. The queue length of rebuffering time $PC_i(n)$ during the playback can be positive or negative. A negative $PC_i(n)$ indicates that there still exists enough data in the i-th user's playback buffer. The remaining data in the buffer will contribute to the next slot's playback, while a positive $PC_i(n)$ indicates the accumulated rebuffering time. In order to control PC_i, we use the Lyapunov optimization framework and define the Lyapunov function as [33]:

$$L(n) = \frac{1}{2}\sum_{i=1}^{N}(PC_i(n))^2 \tag{17}$$

In order to keep the queue stable and push the Lyapunov function towards a lower congestion state, we introduce the Lyapunov drift $\Delta(n)$ from the perspective of mathematical expectation,

$$\begin{aligned} \Delta(n) &= \mathbb{E}\{L(n+1) - L(n)|PC(n)\} \\ &\leq B + \sum_{i=1}^{N}\mathbb{E}\{PC_i(n) \times (\tau - t_i(n))|PC(n)\} \end{aligned} \tag{18}$$

where $PC(n) = \sum_{i=1}^{N} PC_i(n)$ and $B = \frac{1}{2}\sum_{i=1}^{N}(\tau^2 + t_{max}^2)$, $t_{max} \geq t_i$, $\forall i \in N$, representing the maximum playback time that a data shard can support for any user i in a time slot.

Following the Lyapunov optimization approach [33], considering playback performance and energy consumption simultaneously, we construct the $drift - plus - penalty$ expression below,

$$\Delta(n) + V\mathbb{E}(E(n)|PC(n)) \tag{19}$$

where $E(n) = \sum_{i=1}^{N} E_i(n)$ and the conditional expectation $\mathbb{E}\{E(n)\}$ indicates the average energy consumption $\overline{PE}$. The parameter V indicates the weight of energy consumption. A larger V indicates more energy saving and vice versa. Therefore, we have transformed the original online optimization problem into the problem of minimizing the above expression.

Combined with Eq. (18), the $drift - plus - penalty$ expression of Eq. (19) is controlled within an upper bound as follows,

$$\begin{aligned} &\Delta(n) + V\mathbb{E}(E(n)|PC(n)) \\ &\leq B + V\mathbb{E}(E(n)|PC(n)) \\ &+ \sum_{i=1}^{N}\mathbb{E}(PC_i(n) \times (\tau - t_i(n))|PC(n)) \end{aligned} \tag{20}$$

Following the design principle of the Lyapunov framework, the underlying objective of our energy minimization problem is to minimize the upper bound of the $drift - plus - penalty$ expression. Minimizing the RHS of the equation above will guarantee the rebuffering time with the minimum energy consumption.Since B and τ are both constants, the energy optimization problem can be transformed into the following problem,

$$\begin{array}{ll} \textbf{min} & V \cdot E(n) + \sum_{i=1}^{N}(PC_i(n) \times (\tau - t_i(n))) \\ \text{s.t.} & \text{Constraints (1) and (2)} \end{array} \tag{21}$$

We continue to derive the objective and have,

$$\begin{aligned} &V \cdot E(n) + \sum_{i=1}^{N}(PC_i(n) \times (\tau - t_i(n))) \\ &= \sum_{i=1}^{N}[V \cdot E_i(n) + PC_i(n)(\tau - t_i(n))] \\ &= \sum_{i=1}^{N} f(i, \varphi_i(n)) \end{aligned} \tag{22}$$

where $t_i(n) = \delta\varphi_i(n)/p_i(n)$. $E_i(n)$ and $PC_i(n)$ are determined by the amount of decision parameter $\varphi_i(n)$ in slot n. Hence we simplify the equation into $\sum_{i=1}^{N} f(i, \varphi_i(n))$. Now the basic goal is to minimize the sum of $f(i, \varphi_i(n))$. From Eq. (22), if $PC_i(n)$ is negative, $(\tau - t_i(n))$ should be positive to minimize $\sum_{i=1}^{N} f(i, \varphi_i(n))$. In other words, if the buffer of user i is not empty ($PC_i(n) < 0$), to ensure the fairness among multiple users, we should not allocate too much data to user i.

To find the minimum solution of the above problem, we design an online algorithm Energy Minimization Algorithm (EMA) to solve the problem with the constraints of Eq. (1) and Eq. (2). To minimize Eq. (22), we build a dynamic programming equation of slot n as,

$$a[i][M] = \min\{a[i-1][M - \varphi_i(n)] + f(i, \varphi_i(n)), a[i][M]\}$$

where M is the number of total data units that the Scheduler decides to transmit to the previous $i-1$ users, and $\varphi_i(n)$ is the amount of data units that the Scheduler decides to transmit to user i in slot n.

Algorithm 2 EMA

Inputs: User number N, Bandwidth $S(n)$, Slot length τ, Signal strength $\mathrm{sig}_i(n)$, Required data rate $p_i(n)$, Lyapunov parameter V

Outputs: Data Unit Allocation $\varphi_i(n), i \in [1, N]$

```
1:  Compute $PC_i(n)$ using Eq. (12);
2:  $\varphi_i(n) \leftarrow 0, i = 1, 2, \ldots, N$;
3:  for $M = 0 \rightarrow \lfloor \tau v(\mathrm{sig}_1(n))/\delta \rfloor$ do
4:      $a[1][M] = \min\{f(1, M)\}$;
5:  end for
6:  $g(1, M) \leftarrow \arg\min_M a[1][M]$;
7:  for $i = 2 \rightarrow N$ do
8:      for $M = 0 \rightarrow \lfloor \tau \cdot S(n)/\delta \rfloor$ do
9:          for $\varphi_i(n) = 0 \rightarrow \lfloor \tau v(\mathrm{sig}_i(n))/\delta \rfloor$ do
10:             $a[i][M] = \min\{a[i-1][M - \varphi_i(n)] + f(i, \varphi_i(n))\}$;
11:         end for
12:         $g(i, M) \leftarrow \arg\min_{\varphi_i(n)} a[i][M]$;
13:     end for
14: end for
15: $D_N(n) \leftarrow \arg\min_M a[N][M]$; $\varphi_N(n) = g(N, D_N(n))$;
16: for $i = N - 1 \rightarrow 1$ do
17:     $D_i(n) = D_i(n+1) - \varphi_i(n+1)$; $\varphi_i(n) = g(i, D_i(n))$;
18: end for
19: return $\varphi_i(n), i \in [1, N]$;
```

Algorithm 2 shows the detail of EMA. In steps $3 - 5$, we get the border condition of the problem. The algorithm tries to find the minimum $\sum_{i=1}^{N} f(i, \varphi_i(n))$ (steps $7 - 14$). We use $g(i, M)$ to record the best $\varphi_i(n)$ when the total data units assigned to the previous $i - 1$ users are M. We finally obtain the data allocation through steps $16 - 18$.

We analyze the boundedness property of our design and prove that $\overline{PE}$ and $\overline{PC}$ are both controlled by two different upper bounds.

Theorem 1. *If* $\Gamma \rightarrow \infty$, $\overline{PE}_\infty$ *and* $\overline{PC}_\infty$ *hold,*

$$\begin{aligned} \overline{PE}_\infty &= \limsup_{\Gamma \rightarrow \infty} \frac{1}{\Gamma} \sum_{n=0}^{\Gamma-1} \sum_{i=1}^{N} E_i(n) \le E^* + \frac{B}{V} \\ \overline{PC}_\infty &= \limsup_{\Gamma \rightarrow \infty} \frac{1}{\Gamma} \sum_{n=0}^{\Gamma-1} \sum_{i=1}^{N} c_i(n) \le \frac{B + VE^*}{\varepsilon} \end{aligned} \tag{23}$$

where B and ε are both positive constants, and E^* is the theoretically optimal solution (minimum energy cost). Please refer to the proof in Appendix A.

VI. Performance Evaluation

We implement RTMA and EMA in around 1000 lines of C++ codes. In our experiment, the slot number is set to 10000 and each slot lasts $\tau = 1s$. The video length that users require is set as random value ranging from $250MB$ to $500MB$ with the variable required data rate from $300KB/s$ to $600KB/s$. The service capacity S at the BS is set to $20MB/s$ for all slots. Recall that the RRC models of 3G and LTE are similar and only different in certain parameters (e.g. timers and the power in each state) as described in Section III-A, it can be expected that we can obtain similar results in LTE networks. We set the radio parameters following the previous measurements in 3G networks [19].

According to Definition 3 and 4, the throughput v and the power cost per byte P are highly associated with the instant signal strength. Prior measurements [7], [11], [28] in cellular networks have shown that, both the power and the throughput can be numerically fitted as two functions related with signal strength. Therefore we exploit the results of [28] and use two functions to fit the throughput and the power.

$$\begin{aligned} v(\mathrm{sig}) &= 65.8 \times sig + 7567.0 (KBps) \\ P(\mathrm{sig}) &= -0.167 + \frac{1560}{v(\mathrm{sig})} (mJ/KB) \end{aligned} \tag{24}$$

The parameters of RRC state we set here are based on [29], where the power of CELL_DCH and CELL_FACH state is $732.83mW$, $388.88mW$ respectively in a 3G network. We set the timers T_1 and T_2 to $3.29s$ and $4.02s$. We set the signal strength variation following a sine function in the range of $-50dBm$ to $-110dBm$ with $30dBm$ white Gaussian noise intensity. Since the signal strength variation may be different from each other, we add different phase shifts for the N sine functions.

A. Evaluation of RTMA

We now verify the effectiveness of RTMA. For comparison, we implement a default streaming system as the baseline that delivers video contents to each user as much as possible to make full use of throughput and satisfy the required data rate. The value of average rebuffering time and energy consumption achieved in this method, which we denote as default rebuffering time and default energy, are used as baseline values. Note that the goal of RTMA is to optimize the rebuffering time while limiting the energy consumption within Φ. We set the energy constraint $\Phi = \alpha * E_{Default}$, where $E_{Default}$ is the default energy and α is a tunable constant coefficient. We set $\alpha = 1$ to compare the rebuffering time with or without RTMA under the same energy consumption. Furthermore, to evaluate how RTMA manages the resource competition at the BS in multi-user scenarios, we define a fairness metric for user i as $F_i = d_i/d_{need}(i)$, where d_i is the allocated data size for user i and $d_{need}(i)$ is the required data size in a slot. We define the fairness index as $(\sum_{i=1}^{N} F_i)^2/(N \sum_{i=1}^{N} {F_i}^2)$ based on the Jain Fairness Index [25]. A fairness index close to 1 indicates the fair allocation for all users.

We first compare the fairness index result with or without RTMA when user number is 40 and the average required data amount is $350MB$. As shown in Figure 2, the fairness index of RTMA is larger than 0.7 for more than 90% time slots for all users across the BS, while for default strategy, the fairness index is below 0.2 for about 50% slots. This is because there exists resource competition at the BS and the default strategy allocates data as much as possible, hence some users cannot obtain enough data. RTMA achieves much higher fairness compared to the default strategy since it iteratively allocates

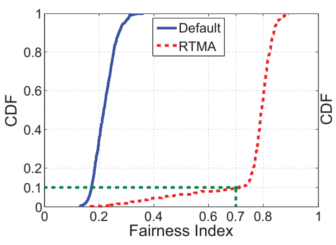


Fig. 2. Fairness

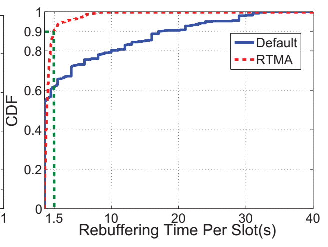
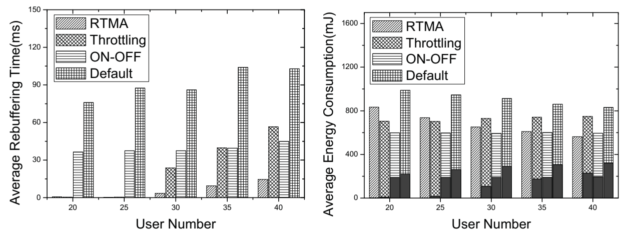
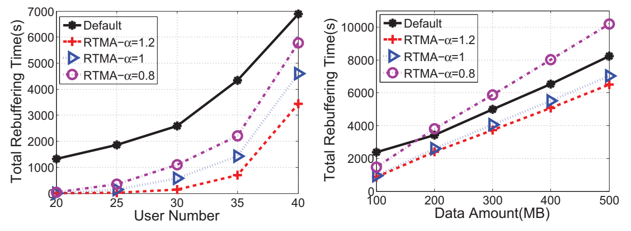


Fig. 3. Rebuffering time



(a) Rebuffering time (b) Energy consumption

Fig. 5. Performance comparison of RTMA



(a) Various user number (b) Various data amount

Fig. 4. Efficacy of RTMA

data for each user. Thus the prior users will not seize too much bandwidth and cause unfairness. To evaluate the reduction of rebuffering time, we also compare the rebuffering time per slot, c_i among all users. As depicted in Figure 3, RTMA has lower rebuffering time, and about 90% of the slots have less than $1.5s$ rebuffering time. We observe that about 57% users with the default strategy have a very low unsaturated time (close to zero) but more than 20% users have suffered rebuffering time more than $11s$. This imbalance of rebuffering time among multi-users is caused by the resource competition at the BS.

Further, we adjust α to evaluate how the energy constraint impacts on the rebuffering time. The computation runs independently when α is set to $0.8, 1, 1.2$ respectively in different scenarios with various user numbers or total required data amounts. The comparison results of rebuffering time are shown in Figure 4. As we adjust the energy constraint by tuning α, a loose energy constraint with $\alpha = 1.2$ provides more rebuffering time reduction. However, under a tighter constraint (e.g. $\alpha = 0.8$), RTMA can still achieve lower rebuffering time compared with the default strategy in certain cases. The results in Figure 4 indicate that we can adjust α to meet various energy demands.

We then evaluate the performance of RTMA with two other common online scheduling algorithms, i.e. "Throttling" [15] and "ON-OFF" [14]. We set the energy consumption limitation Φ of RTMA to the default energy consumption. We plot the average rebuffering time and energy consumption of each user per slot with different algorithms in Figure 5. Throttling delivers the video contents at a rate that is lower than the bulk transfer capacity but higher than the encoding rate, which ensures the continuous transmission of users to achieve small rebuffering time. As the user number raises, it is hard to meet all the rate requirement and hence the rebuffering time increases dramatically. ON-OFF is an algorithm that sets a low threshold of the buffer therefore it obtains a smaller rebuffering time than the default strategy. Since ON-OFF does not consider the resource competition among multiple users, the rebuffering time is much higher than that of RTMA. RTMA outperforms other algorithms with the increasingly resource competition, as shown in Figure 5(a). The black bar in Figure 5(b) indicates the tail energy of each strategy. When we set $\alpha = 1$, RTMA's energy consumption is smaller than the default strategy and slightly higher than "ON-OFF" which deliveries the data in burst to reduce the active time of wireless interface. When it comes to 40 users, RTMA can still maintain a smooth transmission of all users while reducing the tail energy.

In summary, we find that RTMA obtains the shortest rebuffering time because it considers the required data rate and makes the bandwidth full utilized while satisfying energy consumption constraint. RTMA reduces at least 68% rebuffering time compared with Throttling, ON-OFF and the default strategy.

B. Evaluation of EMA

EMA is designed to minimize energy consumption under the constraint of rebuffering time. Similar with the evaluation of RTMA, we use the default rebuffering time as the upper bound in Eq. (13), thus we set $\Omega = \beta * R_{Default}$, where $R_{Default}$ is the default rebuffering time and β is a constant coefficient. We plot the fairness index and the power consumption in each slot among all users with or without EMA in Figure 6 and Figure 7. The user number here is set to 40 and the average required data amount is $350MB$. As shown in Figure 6, EMA achieves higher fairness index because it designs a negative queue to ensure fairness. During the video session, EMA tries to allocate data under better signal strength to conserve energy. Consequently, as depicted in Figure 7, EMA conserves more energy than the default strategy and about 50% slots of EMA have the power consumption lower than $25J$.

To evaluate EMA in different scenarios, we vary the user number and the total data amount, and plot their energy consumption in Figure 8. Due to the increasing user number or the required data amount, the workload at the BS increases and raises intense competition. EMA ($\beta = 1$) can maintain the same rebuffering time as default strategy and achieve more than 48% energy reduction in various scenarios. We also

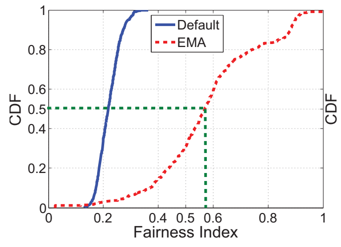


Fig. 6. Fairness

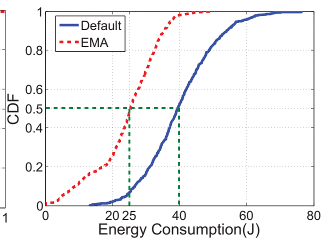
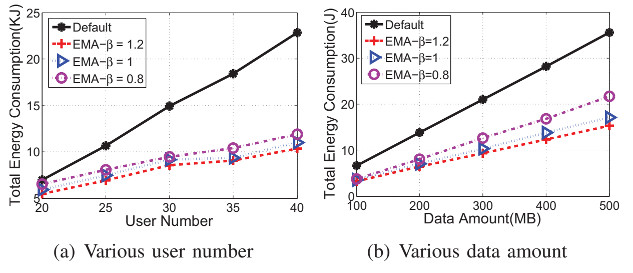
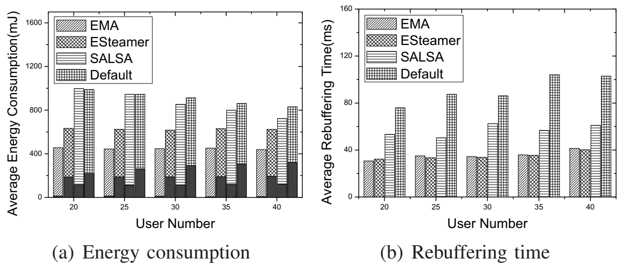


Fig. 7. Power per slot



(a) Various user number
(b) Various data amount

Fig. 8. Efficacy of EMA



(a) Energy consumption
(b) Rebuffering time

Fig. 9. Performance comparison of EMA

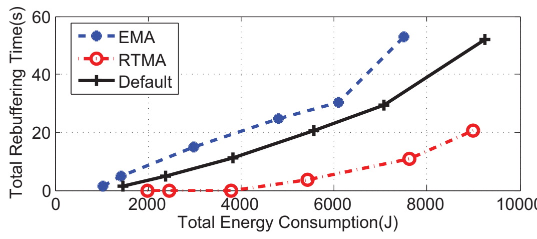


Fig. 10. RTMA and EMA comparison

observe that a tighter rebuffering time constraint ($\beta = 0.8$) can still reduce the energy consumption and conversely if we increase this upper bound ($\beta = 1.2$), EMA will achieve better energy efficiency. In other words, β can be tuned to satisfy various rebuffering time requirement.

In addition, we compare EMA with other two online energy efficient scheduling algorithms, i.e. "SALSA" [17] and "EStreamer" [16]. We set the rebuffering time bound Ω of EMA to the EStreamer's rebuffering time. We plot the energy consumption and the rebuffering time for each algorithm in Figure 9. SALSA intends to minimize the energy while keeping a finite waiting queue, but it ignores the significant tail energy, which weakens the effectiveness of conserving energy consumption. EStreamer is an algorithm that leverages the caching management to reduce the active time of wireless interface. Its energy consumption is higher than that of EMA because it raises significant tail energy in the idle period between the transmission bursts. EStreamer sets the burst size according to the buffer size, so its rebuffering time is smaller. Both SALSA and EStreamer do not take the impact of signal strength into consideration.

Clearly, EMA obtains less energy consumption because EMA jointly considers signal strength and tail energy to optimize the energy consumption while satisfying the rebuffering time constraint. In summary, EMA reduces at least 48% energy consumption compared with SALSA and the default strategy, and achieves more than 27% energy reduction compared with EStreamer.

C. Discussion of RTMA and EMA

The two algorithms, RTMA and EMA, are proposed to jointly optimize media streaming according to the concrete rebuffering time or the energy constraint. Essentially, the basic idea of both algorithms is to make trade-off between rebuffering time and energy. Through scheduling video delivery, they try to optimize one metric while meeting the constraint of the other one.

To better understand this trade-off, we plot the "rebuffering time"-"energy" panel of RTMA ($\alpha = 1$), EMA ($\beta = 1$) and default scheduler with different user numbers from 20 to 40 in Figure 10. It can be seen that, with the constraint of default energy, the "rebuffering time"-"energy" curve of RTMA performs a drift to the negative direction in rebuffering time axis for each point compared with baseline. With the same energy consumption, RTMA can always achieve lower rebuffering time than default strategy. Meanwhile, EMA performs a similar drift in the energy axis and optimize energy consumption with the limitation of default rebuffering time. The comparison results give the indication that our algorithms RTMA and EMA can achieve significant rebuffering time or energy reduction for various demands. Through our framework, it is flexible to select the appropriate mode and set the constraint to obtain rebuffering time or energy optimization.

VII. Conclusion

The playback fluency and battery endurance have become significant concerns for streaming services in cellular networks. In this paper, we investigate the joint media streaming optimization for energy consumption and rebuffering time. We first proposed an optimization framework to schedule video traffic in cellular networks. Further we designed two algorithms that leverage cross-layer features to jointly optimize energy consumption and rebuffering time on the proposed framework. Specifically, RTMA aims at achieving minimum rebuffering time with limited energy and EMA optimizes energy consumption while meeting the rebuffering time constraint. These two algorithms can work for the two complementary modes in our frameworks respectively. Extensive simulations demonstrate that the proposed algorithms achieve promising effectiveness on jointly optimizing energy and rebuffering time. The RTM algorithm reduces at least 68% rebuffering time and the EM algorithm achieves more than 27% energy reduction compared with other state-of-art strategies.

VIII. Acknowledgments

This work is supported by National Natural Science Foundation of China (no. 61120106008 and no. 61422206), National High Technology Development 863 Program of China (no. 2013AA010401). J. Liu's work is supported by an NSERC Discovery grant.

References

[1] The mobile broadband standard. https://http://www.3gpp.org/.

[2] Sandvine. http://www.sandvine.com.

[3] Cisco visual networking index: Global mobile data traffic forecast update, 2013–2018. 2014.

[4] 3GPP TS 25.331. Radio resource control (rrc) protocol specification. 1999.

[5] 3GPP TS 36.331 E-UTRA. Radio resource control (rrc) protocol specification. 2008.

[6] A.ParandehGheibi et al. Avoiding interruptionsła qoe reliability function for streaming media applications. *JSAC*, 2011.

[7] A.Rahmati et al. Context-for-wireless: context-sensitive energy-efficient wireless data transfer. In *MOBISYS*, 2007.

[8] A.Schulman et al. Bartendr: a practical approach to energy-aware cellular data scheduling. In *MOBICOM*, 2010.

[9] F.Qian et al. Top: Tail optimization protocol for cellular radio resource allocation. In *ICNP*, 2010.

[10] J.Chen et al. A scheduling framework for adaptive video delivery over cellular networks. In *MOBICOM*, 2013.

[11] J.Huang et al. A close examination of performance and power characteristics of 4g lte networks. In *MOBISYS*, 2012.

[12] L.Ekiz et al. Mimo performance evaluation of automotive qualified lte antennas. In *IEEE EuCAP*, 2013.

[13] M.Draxler et al. Cross-layer scheduling for multi-quality video streaming in cellular wireless networks. In *IEEE IWCMC*, 2013.

[14] M.Hoque et al. Dissecting mobile video services: An energy consumption perspective. In *IEEE WoWMoM*, 2013.

[15] M.Hoque et al. Using crowd-sourced viewing statistics to save energy in wireless video streaming. In *MOBICOM*, 2013.

[16] M.Hoque et al. Saving energy in mobile devices for on-demand multimedia streaming–a cross-layer approach. *ACM TOMCCAP*, 2014.

[17] M.Ra et al. Energy-delay tradeoffs in smartphone applications. In *MOBISYS*, 2010.

[18] M.Shafiq et al. Understanding the impact of network dynamics on mobile video user engagement. In *SIGMETRICS*, 2014.

[19] N.Balasubramanian et al. Energy consumption in mobile phones: a measurement study and implications for network applications. In *IMC*, 2009.

[20] N.Thiagarajan et al. Who killed my battery?: analyzing mobile browser energy consumption. In *WWW*, 2012.

[21] P.Athivarapu et al. Radiojockey: mining program execution to optimize cellular radio usage. In *MOBICOM*, 2012.

[22] P.Dutta et al. On managing quality of experience of multiple video streams in wireless networks. In *INFOCOM*, 2012.

[23] P.Shu et al. etime: energy-efficient transmission between cloud and mobile devices. In *INFOCOM*, 2013.

[24] Q.Xu et al. Proteus: network performance forecast for real-time, interactive mobile applications. In *MOBISYS*, 2013.

[25] R.Jain et al. A quantitative measure of fairness and discrimination for resource allocation in shared systems. Technical report, 1984.

[26] R.Kokku et al. Cellslice: Cellular wireless resource slicing for active ran sharing. In *IEEE COMSNETS*, 2013.

[27] S.Shen et al. An information-aware qoe-centric mobile video cache. In *MOBICOM*, 2013.

[28] S.Suneja et al. Envi: energy efficient video player for mobiles. In *ACM Workshop on Cellular networks: operations, challenges, and future design*, 2013.

[29] Y.Cui et al. Performance-aware energy optimization on mobile devices in cellular network. In *INFOCOM*, 2014.

[30] Y.Yu et al. Energy-efficient downlink resource allocation for mobile devices in wireless systems. In *GLOBECOM*, 2013.

[31] Y.Yu et al. Energy-adaptive downlink resource allocation mobile devices in wireless systems. *TMC*, 2014.

[32] Z.Wang et al. Studying streaming video quality: from an application point of view. In *MM*, 2003.

[33] M J.Neely. Stochastic network optimization with application to communication and queueing systems. *Synthesis Lectures on Communication Networks*, 2010.

Appendix

Appendix A: Proof of Theorem 1

Proof: Since the arrival process is strictly within the network capacity, there exists one stationary randomized control policy that can stabilize the queue, which satisfies the following properties:

$$\begin{aligned}\mathbb{E}\{E(n)\} &= E^* \\ \mathbb{E}\{\tau - t_i(n)\} &\le \varepsilon\end{aligned} \tag{25}$$

where we define E^* as the minimum achievable power expenditure using any control policy that achieves delay stability. Then we apply these two equations into Eq. (18):

$$\Delta(n) + V \cdot \mathbb{E}\{E(n)\} \le B + V \cdot E^* - \varepsilon PC(n) \tag{26}$$

where $PC(n) = \sum_{i=1}^{N} PC_i(n)$.

Taking an expectation for Eq. (26) and using the iterative expectation law:

$$\mathbb{E}\{L(n+1)-L(n)\}+V\cdot\mathbb{E}\{E(n)\} \le B+V\cdot E^*-\varepsilon\mathbb{E}\{PC(n)\}$$

Then, summing over all time slots $n \in \{0, 1, \cdots, \Gamma - 1\}$ and diving by Γ:

$$\frac{\mathbb{E}\{L(\Gamma) - L(0)\}}{\Gamma}+\frac{V}{\Gamma}\sum_{n=0}^{\Gamma-1}\mathbb{E}\{E(n)\} \le B+V\cdot E^*-\frac{\varepsilon}{\Gamma}\sum_{n=0}^{\Gamma-1}\mathbb{E}\{PC(n)\}$$

Since the Lyapunov function $L(n)$ is non-negative by definition and so is $E(n)$, we have:

$$\frac{1}{\Gamma}\sum_{n=0}^{\Gamma-1}\mathbb{E}\{E(n)\} \le \frac{B}{V} + E^* + \frac{\mathbb{E}\{L(\Gamma) - L(0)\}}{V \cdot \Gamma} \tag{27}$$

$$\frac{1}{\Gamma}\sum_{n=0}^{\Gamma-1}\mathbb{E}\{PC(n)\} \le \frac{B + V \cdot E^* + \mathbb{E}\{L(\Gamma) - L(0)\}/\Gamma}{\varepsilon} \tag{28}$$

When $\Gamma \to \infty$, we obtain equations Eq. (23). ■