This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2016.2604565, IEEE Transactions on Multimedia

Resource-efficient Mobile Multimedia Streaming with Adaptive Network Selection

Joohyun Lee[†], Kyunghan Lee[*‡], Choongwoo Han[♭], Taehoon Kim[‡] and Song Chong[♯]

***Abstract*—From the advancements of mobile display and network infrastructure, mobile users can enjoy high quality mobile video streaming anywhere anytime. However, most mobile users are still reluctant to use high quality video streaming when they are mobile due to costly cellular data and high energy consumption. In this work, we develop scheduling algorithms for resource-efficient mobile video streaming, which minimize the weighted sum objective of cellular cost and energy consumption. We first model the scheduling problem as a Markov decision process and propose an optimal scheduling algorithm based on dynamic programming. Then, we derive a heuristic algorithm that approximates the optimal algorithm. To evaluate the performance of proposed algorithms, we run simulation over YouTube video traces with audience retention graphs and mobility/connectivity traces in public transportation (e.g., commuting). Through extensive simulations, we show that our proposed scheduling algorithm has negligible performance loss compared to the optimal scheduling algorithm, where it saves 59% of cellular cost and 41% of energy compared to the YouTube default scheduler. We also implement our scheduling algorithm on an Android platform, and experimentally evaluate the performance compared to existing streaming policies.**

I. Introduction

Mobile video is expected to dominate the Internet traffic in near future according to a traffic forecast report [1]. This is plausible given that the screen size and resolution of mobile devices are getting larger and higher, and wireless networks are getting faster by the adoption of new technology such as LTE-Advanced networks. The report forecasts that the portion of video traffic will reach 66 percent of the total mobile data traffic, where the per-smartphone video traffic is projected to be 1.7GB/month in year 2016.

This trend of demanding more video traffic while being mobile is increasingly burdening users for significant inflation in the cellular cost as well as energy consumption. For instance, the streaming of 720p or 1080p YouTube video for an hour eats up 1.8GB or 4GB. Given that 15GB monthly data plan costs $100 at AT&T as of October 2015 [2], the streaming cost over LTE networks is never negligible. Note that unlimited data plan is provided only to a limited number of countries, but with significant throttling of data rates after a certain data quota. This is a major hurdle for users who want to watch high quality videos even outside WiFi coverage. The energy consumption for streaming videos is also of a critical issue. This is because streaming videos uses the most energy-hogging chipsets of mobile devices such as LTE, LCD, CPU, and GPU all at the same time [3]. According to [4], streaming applications (e.g., YouTube and Vimeo) drain the battery of Samsung Galaxy Note (2500mAh) and Motorola RAZR MAXX (3300mAh) within 4 or 5 hours. Another report [5] also revealed that when streaming video, the battery lifetimes are only 5 and 7 hours in Galaxy S4 (2600mAh) and S5 (2800mAh) smartphones, respectively.

We tackle these problems in mobile streaming by exploiting the following two characteristics: (1) *link rate fluctuation from multi-modal networking interfaces* (e.g., LTE, WiFi) and (2) *user churning behaviors* while streaming videos. The former is observed as frequent switching between WiFi and LTE interfaces especially when users roam around an indoor complex or pass by subway platforms providing WiFi coverage. The latter describes a typical user behavior of stopping watching a video in the middle of its streaming due to various reasons (e.g., loss of interest). The measurement study in [6] revealed that 60% of the YouTube videos are watched for less than 20% of their duration. These characteristics in mobile streaming are important as they provide huge opportunities in saving data cost and energy. For instance, if a device connected to a cellular network is expected to get a faster WiFi connection in near future, deferring the buffering of streaming saves both energy and cellular cost. In the same manner, if the video being watched is soon to be stopped, minimizing the amount of playback buffer saves both cellular cost and energy.

In this work, we propose a new scheduler for non-live streams that is far more resource-efficient than existing streaming algorithms being used in YouTube or Vimeo. Our scheduler utilizes a known statistics, *retention graph* to extract user churning behavior. The retention graph that shows the variation of views for each moment of a video is readily available in YouTube for the owner [7]. Link rate variation can be extracted from the history of the user or crowdsourced data rates over the user's mobility path.

Based on the user churning and link rate variation model, we formulate our scheduling problem as a Markov decision process, where the objective is to minimize the weighted sum of average cellular cost and energy consumption in streaming. We first derive an optimal scheduling algorithm using dynamic programming that is unusable in practice due to the *curse*

[†]J. Lee is with the Department of Electrical and Computer Engineering, the Ohio State University, e-mail: lee.7119@osu.edu. He conducted this research while he was at KAIST.

[*‡]K. Lee, the corresponding author, and T. Kim are with the School of Electrical and Computer Engineering, UNIST (Ulsan National Institute of Science and Technology), South Korea, e-mails: {khlee, carpedm20}@unist.ac.kr.

[♭]C. Han is with the School of Computing, KAIST (Korea Advanced Institute of Science and Technology), e-mail:cwhan.tunz@kaist.ac.kr.

[♯]S. Chong is with the School of Electrical Engineering, KAIST (Korea Advanced Institute of Science and Technology), e-mail: songchong@kaist.edu.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2016.2604565, IEEE Transactions on Multimedia

of dimensionality. We then develop an approximate algorithm that demands significantly lower information overheads and computational complexity. Through trace-driven simulation over real mobility and video streaming traces, we show that our proposed scheduler saves 59% of cellular cost and 41% of energy consumption compared to the current YouTube scheduler. We also implement our scheduler on an Android platform, and show that a similar level of saving as in trace-driven simulations is achievable in practice.

II. Related work

There are several categories of previous studies related to our paper. Many researchers studied mobile streaming to improve QoS (Quality of Service) without considering resource consumption. Examples of QoS metrics are the video stalling probability and video quality (e.g., PSNR (Peak Signal to Noise Ratio)). Recently, cost minimization (mainly energy) in mobile streaming has received a lot of attention in many papers. To improve cost efficiency, these works used additional information such as channel variations or user viewing statistics. Another line of research in delayed offloading exploited mobility and additional delay to alleviate cellular network congestion opportunistically. We summarize each category in the following paragraphs.

In a decade ago, early studies on multimedia streaming [8], [9] mainly focused on providing a better video quality guarantee and gave little attention to the energy efficiency because non-mobile streaming devices are typically not battery constrained. Chou et al. [8] developed nearly optimal streaming over a lossy packet network. In [9], Jurca et al. considered multipath networks and solved the problem of joint path selection and source rate allocation in order to maximize the video streaming quality. The proposed scheduling algorithms are shown very effective in quality control over lossy channels, but they may suffer from achieving high resource efficiency because it was not considered in the framework. Considering that mobile streaming users are sensitive to energy consumption as well as data cost, our work is filling the gap between [8], [9] and a more realistic user satisfaction.

As mobile streaming gets prevalent and energy/cost become scarce resources, scheduling and controlling mobile video streaming traffic has attracted much attention recently [10]–[23]. The common goal in recent studies is to obtain the best trade-off between resource consumption (e.g., energy, cellular cost) and user experience (e.g., video quality, rebuffer rates). Go et al. [10] developed an energy-efficient HTTP adaptive streaming algorithm that provides seamless high-quality video streaming services with networking cost constraints over heterogeneous wireless networks. Similarly, Guruprasad et al. [11] developed an adaptive steaming techniques including stopping video download at times, depending on mobile device buffer levels and the channel conditions experienced, to maximize battery life. Almowuena et al. [12] efficiently utilized both unicast and multicast in cellular networks to minimize the energy consumption of mobile devices and minimize the cellular load. Hoque et al. [13] studied the impact of traffic burst size on power consumption of smartphones in TCP, and showed that the power consumption of a smartphone decreases when the receiving burst size increases. Using this knowledge, they developed EStreamer that minimizes the wireless communication energy spent by mobile devices for TCP-based multimedia streaming. To improve the energy-efficiency of cellular communication in general mobile data, Bartendr [24] exploited network traces to schedule traffic over better channel conditions.

In [14], Hoque et al. additionally exploited crowd-sourced video viewing statistics about the audience retention graph to develop an energy-efficient scheduling algorithm, eScheduler. In [15], the same authors also proposed two versions of eSchedule, optimal one based on dynamic programming and another heuristic with less computational complexity. They also implemented an Android prototype of the scheduler and showed that the energy saving is up to 80% compared to the default Android YouTube application. GreenTube [16] also utilizes user-dependent (but not video-dependent) viewing history to calculate the expected remaining viewing time. Then, they developed a heuristic algorithm to dynamically manage the downloading cache based on the expected viewing time and network condition, for energy and data waste reduction.

Huang et al. [17] proposed "buffer-based" rate selection algorithms that adaptively control the delivered video quality based on the amount of the playback buffer. Their algorithms reduce rebuffer rates (i.e., the probability that the playback buffer becomes empty) by 10-20% compared to Netflix while delivering a similar video rate. Bokani et al. [19] controlled the quality level of the mobile HTTP-based adaptive streaming to optimize three dimensions of streaming performance (i.e., picture quality, deadline miss, and the frequency of quality change) based on a Markov decision process (MDP). They applied three simplification approaches to reduce the overhead of MDP, where the best performance is achieved with x-MDP that recomputes the optimal strategy for every x meters of the travel using offline statistics for each x-meter of the road MDP. Although these papers can improve QoS by chunk scheduling, resource consumption for chuck downloading were not considered and thus they have no guarantee in restricting the costs of energy and cellular data.

Chen et al. [20] proposed to adaptively change the video quality to use cellular data more efficiently. They developed an online video quality adaptation system, QAVA, that dynamically controls the quality (e.g., resolution) of a streaming video that maximizes user experience under a given data quota (i.e., data plan) of the user. An optimal policy is derived from standard backward induction techniques for finite-horizon Markov decision processes. Chen et al. [21] also designed AVIS that schedules HTTP-based adaptive video flow on cellular base stations. It optimally computes bit-rate allocation of each user for the optimal resource allocation in consideration of user quality of experience (QoE).

For general mobile data traffic including video traffic, several papers [25]–[28] have studied mobile data offloading through WiFi networks to reduce the amount of cellular data (i.e., cellular costs), where they exploit mobility and additional delay to offload more data to WiFi networks opportunistically.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2016.2604565, IEEE Transactions on Multimedia

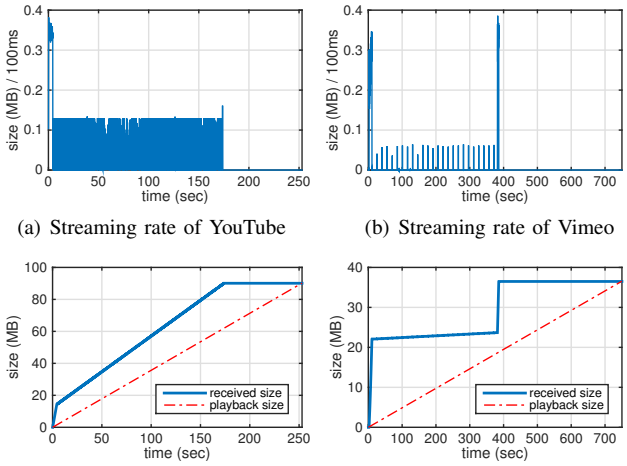
However, our proposed technique in this work treats the deadline for downloading the latter part of a video as dynamic as the video gets played more. This brings a difference in evaluating the offloading opportunity and we find that our technique creates substantially more opportunities. This become possible because of the playback buffer that conceals some amount of delay before it is being played. In other words, video chunks only require to be downloaded before the time they are scheduled to be viewed. For example, in a 10-minutes long video, the last part of the video can tolerate 10 minutes of delay. Thus, if WiFi connection availability is predicted within in the time that a piece of video needs to be played, a smart scheduler can exploit the opportunity to offload mobile video traffic to WiFi networks.

III. Measurement Study

A. Existing streaming techniques: YouTube and Vimeo

In this subsection, we examine existing streaming techniques in two popular streaming services, *YouTube* and *Vimeo* player apps, running on an Android smartphone, Samsung Galaxy Note 2 under a LTE network in March 2014.[1] We capture download packets using an Android version of *tcpdump* [29], while we measure energy consumption using the Monsoon power monitor [30] simultaneously. In YouTube, we stream a 253 seconds long video whose size is 90MB, and in Vimeo, we stream a 750-seconds long video whose size is 37MB. Since their encoding algorithms are different[2], we select different videos that best show the characteristics of their streaming algorithms. We depict streaming rates (download amount per 100ms) and received sizes (i.e., accumulated downloaded amount) of YouTube and Vimeo in Fig. 1. In Table I, we summarize the power consumption of YouTube and Vimeo streaming as well as the power consumption of fast caching when the video is downloaded at the maximum achievable capacity. We calculate the power consumption only for video streaming (i.e., downloading) by taking off the power consumption for the same downloaded video being played without any communication, which is about 1064mW (mJ/s).

Figs. 1(a) and 1(c) show that the YouTube streaming application prefetches the front part of the video (about 14MB) from the streaming server at the maximum achievable data rate of a TCP session. Followed by the fast buffering, the sending rate is throttled down to the average playback rate of the video multiplied by a factor of 1.25, which is also verified by the measurement in [14]. This results in a small playback buffer as shown in Fig. 1(c), but the wireless interface is continuously used until 175 seconds and consumes larger energy compared to the case when the video is downloaded at the maximum achievable data rate from the beginning to the end. This is often called *fast caching* in literature. This is because the network energy consumption is not reduced significantly even if the download rate is throttled, and the cellular interface consumes tail energy for about 10 seconds



(a) Streaming rate of YouTube (b) Streaming rate of Vimeo

(c) Received size of YouTube download (d) Received size of Vimeo download

Fig. 1. Download packet traces of existing streaming techniques: YouTube (throttling with a factor of 1.25) and Vimeo (on-off streaming with a chunk size of 22MB)

(which will be clarified in the next subsection), even if the data transfer is paused. In Table I, we find that for the tested video in Fig. 1(c), throttling the download rate at 1.25 times the playback rate consumes 3.7 times more communication energy than fast caching.

In the case of Vimeo streaming application (in Figs. 1(b) and 1(d)), the video is downloaded as *on-off streaming*, where each part of the video (e.g., 22MB) is downloaded in an ON period at the maximum achievable data rate, and the video downloading stops in an OFF period. An OFF period lasts until the playback buffer becomes smaller than a threshold. As the video is downloaded through multiple ON periods, the buffer size is always smaller than the size of each part, which is 22MB in the tested video. However, even in OFF period, wireless connection stays alive to keep the TCP session and also goes through tail energy consumption. Thus, on-off streaming obviously consumes larger energy than fast caching. We find that the on-off streaming of Vimeo consumes 8 times more communication energy than fast caching, as summarized in Table I. For other existing streaming techniques, we refer readers to [14], [31].

In short, existing streaming techniques of YouTube and Vimeo try to keep buffer sizes small by throttling the download rate or by on-off streaming to mitigate data wastes from user churning, but this sacrifices large communication energy.

B. Wireless communication energy

We measure cellular communication energy of a smartphone by letting the smartphone download a file from our server via a TCP socket using a Monsoon power monitor [30]. We refer readers to the experimental setting in [32], as we repeated the measurement methodology explained in the work. As shown in Fig. 2, a smartphone's cellular module is in either *idle*, *transfer*[3], or *tail*[4] mode, where the power consumption (in

[1]The installed Android version is Jelly Bean 4.3.

[2]Even if a user uploads the same video, the playback rate of the uploaded videos can be different across streaming services, mostly due to different quality adaptation algorithms.

[3]This mode is called DCH (Dedicated CHannel) mode in 3G networks.

[4]This mode is called FACH (Forward Access CHannel) mode in 3G networks.

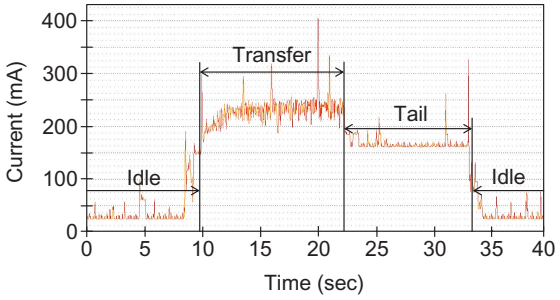
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2016.2604565, IEEE Transactions on Multimedia

TABLE I
MEASURED POWER CONSUMPTION OF YOUTUBE AND VIMEO VIDEOS.

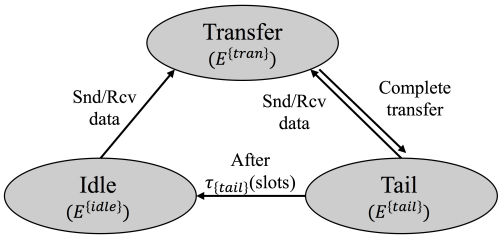
Video	YouTube (90MB, 253 secs long)	
Streaming	YouTube (throttling)	Fast caching†
Avg. power♯	678mW (3.7x)	182mW (1x)

Video	Vimeo (37MB, 750 secs long)	
Streaming	Vimeo (on-off)	Fast caching†
Avg. power♯	609mW (8x)	76mW (1x)

† Downloading the whole video at the maximum achievable capacity at the beginning.
♯ Excluding power consumption in playing the video (e.g., screen and CPU), which is about 1258mW for the YouTube video and 911mW for the Vimeo video.



(a) Consumed power in states (cellular networks).



(b) State diagram.

Fig. 2. State transitions in wireless modules.

TABLE II
LTE/WIFI ENERGY PROFILES.

	Device Name	E^{tran} (J/s)	E^{tail} (J/s)	E^{idle} (J/s)	τ^{tail} (sec)
LTE-I	Galaxy Note 2	1.35	0.67	0.02	9.74
LTE-II	Galaxy Note 2	1.38	0.65	0.02	10.01
WiFi	Galaxy Note 2	0.897	-	-	-

Joule/sec) in each mode is denoted by E^{idle}, E^{tran}, and E^{tail}. Each value is given in Table II for two different cellular networks and a WiFi network. Starting from *idle* mode, the mode is changed to *transfer* when it has data to transmit to or to receive from a base station. After completing the transfer, the mode is changed from *transfer* to *tail*, and the cellular module stays in the *tail* mode at least for a predefined tail duration, which is denoted by τ^{tail}, before going back to *idle* mode. We use the LTE-I network throughout this paper, as LTE-I and LTE-II networks show very similar energy profiles. We also measure power consumption in downloading data traffic through WiFi APs, which is also summarized in Table II. Note that the tail duration in WiFi is negligible (about 0.2 second as shown in [13]) compared to cellular networks.

TABLE III
SUMMARY OF MAJOR NOTATION

Variable	Definition
L	video length
r_p	CBR playback rate (in Mbps)
t_0	start-up delay
$B_{\min}$	minimum buffer size
$\mathcal{A}$	$=\{1,2\}$, set of access networks (1: LTE, 2:WiFi)
$r_a(t)$	data rate of network $a \in \mathcal{A}$
$b(t)$	received video size at time t
$\frac{b(t)}{r_p}$	received duration at time t
λ_{off}	WiFi AP arrival rate
λ_{on}	WiFi AP departure rate
T_{IAP}	random variable of inter-AP time
$\omega(t)$	$\in\{0,1\}$, WiFi connection indication at time t
T_c	$\leq L$, random variable of user churning time
$p_c(t)$	$=\mathbb{P}[T_c=t\|T_c\geq t]$, churning rate at time t
$\tau(t)$	remaining tail time at time t
s	system state, a tuple of t^s, $\frac{b^s(t^s)}{r_p}$, $\tau^s(t^s)$, $w^s(t^s)$
s_t	system state at time t
s_∞	terminating state
u	$\in\{0\}\cup\mathcal{A}$, control (i.e., network selection)
$\mu \in U$	scheduling policy, a set of controls $\mu(s)$ in each s
$U(s)$	set of feasible controls in state s
$p_{i,j}(u)$	state transition prob. from i to j under a control u
$g_{i,j}(u)$	instantaneous cost in transition from i to j under u
γ	trade-off parameter (cellular data over energy)
$E^{\text{tran}}_{\text{LTE}}$	unit LTE transfer energy
$E^{\text{tran}}_{\text{WiFi}}$	unit WiFi transfer energy
$E^{\text{tail}}_{\text{LTE}}$	unit LTE tail energy
τ^{tail}	LTE tail duration (in slots)

IV. FINITE-HORIZON COST MINIMIZATION PROBLEM

In this section, we describe our system model and formulate our problem as a finite horizon cost minimization problem given the audience retention probability and WiFi contact statistics (i.e., inter-AP and AP connection time[5] distributions). The audience retention probability captures the statistical departure probability (i.e., the probability of quitting the video watch) of the viewers for each moment of a video. By the definition, it is possible for us to estimate the probabilistic user churning behaviors from this graph. As a representative example, the YouTube API for retention data returns 100 data points, i.e., the granularity is 1% [7]. We first derive an optimal online algorithm based on dynamic programming. Then, we develop a simple heuristic algorithm, *Maximum Buffer Control (MBC)*, which is a threshold-based policy that mimics the optimal algorithm.

A. System Model

1) Network and Traffic (video) model: We summarize major notations in Table III. We consider a scenario where a user requests a single streaming video of length L (in timeslots) which is encoded in constant bit rate (CBR) with the playback rate r_p (in Mbps). We assume a time-slotted model and index a time slot by $t \in \{1,...,T\}$, where $T = L + t_0$ and t_0 is a start-up delay. The length of a time slot is one second. We assume that users are always under the coverage of a cellular base station, but not necessarily of a WiFi access

[5]AP connection time is the net amount of continuous time duration that a device is connected to an AP. Inter-AP time is the time duration after a user leaves an AP coverage area, until it returns to a coverage area.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2016.2604565, IEEE Transactions on Multimedia

point (AP). We denote the set of access networks by $\mathcal{A}$, where $\mathcal{A} = \{1, 2\}$ (1 denotes the cellular networks, and 2 denotes the WiFi networks). The data rate of network $a \in \mathcal{A}$ at time slot t is denoted by $r_a(t)$. Users can estimate data rates using a low-power rate prediction algorithm that exploits signal strength, location information or past data rates. We refer readers to [33]–[35] for rate prediction algorithms.

Unlike cellular networks, data rate of a WiFi network can be zero, $r_2(t) = 0$, when the connection is unavailable to any WiFi network. We denote the received video size until time slot t by $b(t)$. The received video size should be greater than the amount of video chunks that is needed to play the video without buffering, i.e., $b(t) \geq r_p(t - t_0) + B_{\min}$, where $B_{\min}$ is the minimum buffer size of a streaming system. We assume that the minimum buffer size $B_{\min}$ is chosen to be sufficiently large to ensure that the playback never stops in the middle of playback (i.e., no buffering). For instance, we can apply the Chebyshev's inequality[6] to bound the video stalling probability, using the knowledge about the variance of random data rates from rate prediction.

We model that the inter-AP and AP connection times follow the geometric distribution that is the discrete analogue of the exponential distribution. We find that from our public transportation traces, this modeling is realistic as shown in our measurement in Section V. The inter-AP time is distributed with parameter λ_{off} and AP connection time is distributed with parameter λ_{on}.[7] Our scheduler is assumed to obtain these parameters a priori by analyzing the previous history of AP connection events over the same mobility path. For a newly visited location, we may use crowdsourced AP connection times of other users who visited the area or the mobility path. We denote a WiFi connection indicator at time slot t by $\omega(t)$, where $\omega(t) = 1$ when the user is connected to a WiFi AP, and $\omega(t) = 0$ otherwise. The user churning time is denoted by T_c, which is a random variable and the distribution of T_c is given by the audience retention probability. We define a churning (i.e., abandoning) probability at time t by $p_c(t)$, where $p_c(t)$ is a failure rate of T_c, i.e., $p_c(t) = \mathbb{P}[T_c = t | T_c \geq t]$. We denote the remaining tail duration of cellular networks at time t by $\tau(t) \leq \tau^{\text{tail}}$, where this value is set to be an integer value τ^{tail} (unit is time slot) after completing a cellular data transfer, and decreased by 1 at each time slot if the user does not download a video chunk through cellular networks. When $\tau(t)$ becomes zero, the cellular module goes into *idle*.

2) State transition model: We denote a system state s as a tuple of time t^s, received duration $\frac{b^s(t)}{r_p}$, remaining tail $\tau^s(t)$, WiFi connection indicator $\omega^s(t)$. We omit the superscript s unless confusion arises. Note that the received duration $\frac{b(t)}{r_p}$ should be greater than the playback time t. When the playback ends at $t = L + t_0$ or the user abandons watching the video with probability $p_c(t)$ at time t, the system moves to the terminating state, s_∞, with an instantaneous cost of remaining tail energy consumption, $\tau(t)$.

[6]For a random variable X, $\Pr\{\mathbb{E}[X] - X \geq \delta\} \leq \frac{Var(X)}{\delta^2}$

[7]The average inter-AP time is $1/\lambda_{\text{off}}$ and the average AP connection time is $1/\lambda_{\text{on}}$.

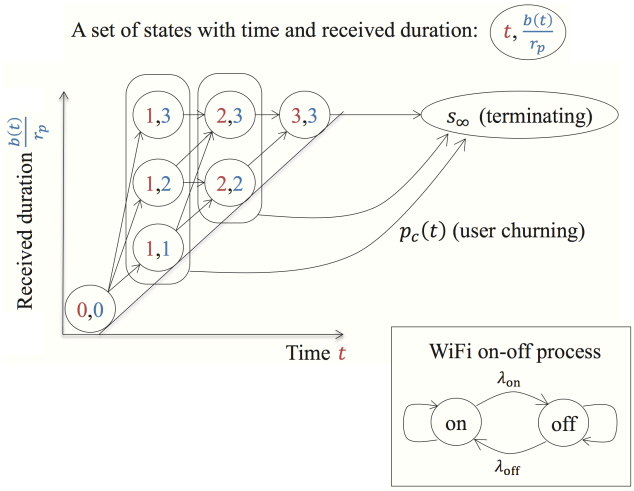


Fig. 3. An example of the state transition model ($L = 3$).

We illustrate an example of the state transition model when $L = 3$, $t_0 = 0$, and $B_{\min} = 0$ in Fig. 3. States are depicted by circles and each arrow indicates the possible transitions between states. We denote by s_t the system state at slot t. The system state s_{t+1} is dependent on the previous state s_t and a control $\mu(s_t)$. We define $\mu : s \mapsto \mu(s)$ as a scheduling policy, which is a set of controls for each state s, where a control $\mu(s) \in \{0\} \cup \mathcal{A}$ selects an access network to download a video chunk or does not download any data (when $\mu(s) = 0$). We denote by $\mu(s_t)$ a control in system state s_t and policy μ. We denote U as the set of all feasible policies, where $U(s)$ is a set of feasible controls in state s, or $\mu(s) \in U(s)$. When the user is connected to a WiFi AP, $U(s) = \{0, 1, 2\}$ and $U(s) = \{0, 1\}$ otherwise.

We define the transition probability from state i to state j under a control u by $p_{i,j}(u)$. We assume that $t_0 = 0$ and $B_{\min} = 0$ for simplicity. When the next state is the terminating state, s_∞, $p_{i,j}(u)$ becomes:

$$p_{i,s_\infty}(u) = \begin{cases} p_c(t^i) & \text{if } t^i < L \\ 1 & \text{if } t^i = L \end{cases}$$

If j is not the terminating state, s_∞, and $t^i < L$,

$$p_{i,j}(u) = \begin{cases} (1 - p_c(t^i))\lambda_{\text{off}} & \text{if } w^i(t^i) = 0, w^j(t^j) = 1 \\ (1 - p_c(t^i))(1 - \lambda_{\text{off}}) & \text{if } w^i(t^i) = 0, w^j(t^j) = 0 \\ (1 - p_c(t^i))\lambda_{\text{on}} & \text{if } w^i(t^i) = 1, w^j(t^j) = 0 \\ (1 - p_c(t^i))(1 - \lambda_{\text{on}}) & \text{if } w^i(t^i) = 1, w^j(t^j) = 1 \end{cases}$$

where $t^j = t^i + 1$, $b^j(t^j) = b^i(t^i) + r_u(t)\mathbf{1}_{\{u \geq 1\}}$, and $\tau^j(t^j) = \max(\tau^i(t^i) - 1, 0)$ if $u \neq 1$, and $\tau^j(t^j) = \tau^{\text{tail}}$ if $u = 1$. $\mathbf{1}_{\{\cdot\}}$ is the indicator function.

An instantaneous cost in a state i and during state transition from i to j under a control u is denoted by $g_{i,j}(u)$, which is the summation of an instantaneous cellular cost, and instantaneous energy consumption multiplied by a trade-off parameter γ. The trade-off parameter γ is for the unit conversion between cellular data cost and energy, where this parameter depends on the user's preference. When the next state j is not the terminating state, s_∞, $g_{i,j}(u)$ is written as:

$$g_{i,j}(u) = r_1(t)\mathbf{1}_{\{u=1\}} + \gamma\left(E^{\text{tran}}_{\text{LTE}}\mathbf{1}_{\{u=1\}} + E^{\text{tail}}_{\text{LTE}}\mathbf{1}_{\{\tau^i(t^i)>0\}} + E^{\text{tran}}_{\text{WiFi}}\mathbf{1}_{\{u=2\}}\right).$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2016.2604565, IEEE Transactions on Multimedia

The instantaneous cellular cost is the amount of cellular traffic transmitted during state transition which occurs only when $u = 1$ meaning that the cellular network is used. We assumed that the monetary cost of WiFi data usage is zero, which is true in common. Often, WiFi networks are complementary services provided by cellular providers or some organizations (e.g., government). The instantaneous energy consumption is either unit cellular transfer energy, unit cellular tail energy, or unit WiFi transfer energy, which we explained in Section III-B, depending on the control u and remaining tail time $\tau(t)$. When the next state is the terminating state, s_∞, the tail energy term ($E^{\text{tail}}_{\text{LTE}}\mathbf{1}_{\{\tau^i(t^i)>0\}}$) changes to $E^{\text{tail}}_{\text{LTE}}\tau^i(t^i)$ to consider all remaining tail energy:

$$g_{i,s_\infty}(u) = r_1(t)\mathbf{1}_{\{u=1\}} + \gamma\left(E^{\text{tran}}_{\text{LTE}}\mathbf{1}_{\{u=1\}} + E^{\text{tail}}_{\text{LTE}}\tau^i(t^i) + E^{\text{tran}}_{\text{WiFi}}\mathbf{1}_{\{u=2\}}\right).$$

B. Problem Definition

We assume that $t_0 = 0$ and $B_{\min} = 0$ for simplicity. It is trival to generalize our results to non-zero start-up delay (t_0) and minimum buffer size ($B_{\min}$). Our objective is to minimize the *expected sum cost* of cellular data and energy consumption where the expectation is with respect to the joint distribution of the random variables (such as T_c and $\omega(t)$) involves:

$$\min_{\mu\in U}\mathbb{E}\Big[g_{s_{T_c},s_\infty}(\mu(s_{T_c})) + \sum_{t=1}^{T_c-1} g_{s_t,s_{t+1}}(\mu(s_t))\Big], \tag{1}$$

where we recall that T_c is the churning time that a user abandons watching the video. Note that $T_c = L$ when the user watches the entire video (i.e., no churning).

C. Optimal Algorithm

The value iteration algorithm that yields an optimal policy is as follows [36]:

$$V^{(n+1)}(i) = \min_{u\in U(i)}\sum_{j=1}^{m} p_{i,j}(u)\left[V^{(n)}(j) + g_{i,j}(u)\right], \tag{2}$$

where $V^{(n)}(s)$ is the value of state s, which is the expected sum of the instantaneous cost from state s to the terminating state s_∞, at n-th iteration. Initially, $V^{(0)}(s) = 0$ for all state s. In each step, the algorithm updates the value function of each state, which will converge to the expected sum cost until termination (i.e., the user churns or watches the entire video). When $\max_s |V^{(n+1)}(s) - V^{(n)}(s)| < \epsilon$, for a sufficiently small ϵ, the scheduling policy μ which attains minimum at every state s is an optimal scheduling policy. Intuitively, as the iteration runs, the value function of each state captures the probabilistic effect of more distant states (i.e., states that take longer transitions to visit). However, it is well known that this optimal policy is practically impossible to use due to the curse of dimensionality [37].

D. Heuristic: Maximum Buffer Control

We develop a heuristic algorithm that mimics an optimal policy by controlling the maximum buffer size (i.e., a threshold) considering (i) data wastes from user churning, (ii) WiFi offloading from future WiFi contacts, and (iii) tail energy from turning off the cellular interface. Our heuristic algorithm (maximum buffer control (MBC)) is described in Table IV. We denote $u(t)$ as the schedule (control) at time slot t.[8] We denote the random variable of inter-AP time as T_{IAP}.

TABLE IV
MAXIMUM BUFFER CONTROL (A HEURISTIC SCHEDULER)

At the first time slot $t = 1$,
if $\omega(t) = 1$ (under a WiFi AP) and $r_2(t) > r_p$,
  $u(t) = 2$ (download a chunk through WiFi networks).
else $u(t) = 1$ (download a chunk through cellular networks).
At each time slot $t > 1$,
if $\omega(t) = 1$,
  if $b(t-1) + r_2(t) > r_p \cdot t$,
    $u(t) = 2$.
  else $u(t) = 1$.
else if $\omega(t) = 0$ (out of WiFi coverage) and $u(t-1) = 0$ (did not download any chunk at the previous slot),
  if $b(t-1) < r_p \cdot t$,
    $u(t) = 1$.
  else $u(t) = 0$ (does not download any chunk).
else if $\omega(t) = 0$ and $u(t-1) = 1$ (downloaded a chunk through cellular networks at the previous slot),
  $C_1 = \mathbb{P}\Big[T_c \le \lfloor\frac{b(t)}{r_p}\rfloor \Big| T_c \ge t\Big] \times (r_1(t) + \gamma E^{\text{tran}}_{\text{LTE}})$,
  $C_2 = \Big(1 - \mathbb{P}\Big[T_c \le \lfloor\frac{b(t)}{r_p}\rfloor \Big| T_c \ge t\Big]\Big) \times \mathbb{P}\Big[T_{\text{IAP}} \le \lfloor\frac{b(t)}{r_p}\rfloor - t\Big] \times \Big(r_1(t) + \gamma\Big(E^{\text{tran}}_{\text{LTE}} - E^{\text{tran}}_{\text{WiFi}}\frac{r_1(t)}{\mathbb{E}[r_2(t)]}\Big)\Big)$,
  $C_3 = \Big(1 - \mathbb{P}\Big[T_c \le \lfloor\frac{b(t)}{r_p}\rfloor \Big| T_c \ge t\Big]\Big) \times \Big(1 - \mathbb{P}\Big[T_{\text{IAP}} \le \lfloor\frac{b(t)}{r_p}\rfloor - t\Big]\Big) \times \gamma E^{\text{tail}}_{\text{LTE}}\tau^{\text{tail}}$.
  if $C_1 + C_2 \le C_3$,
    $u(t) = 1$.
  else $u(t) = 0$.

We first myopically prefer opportunistic WiFi networks to reduce cellular costs. This strategy is reasonable since per-bit transfer energy of WiFi networks is comparable to or better than that of cellular networks in practice, unless the signal strength is extremely weak or interference is too severe. We also emphasize that the tail energy of WiFi networks is negligible. Thus, if the user is connected to a WiFi AP, it downloads data through WiFi networks as long as the data rate of WiFi is fast enough to continue video streaming. When WiFi APs are not available, the scheduler does not download data cellular networks, as long as the video buffer is sufficient enough to play the video at the next time slot. If the remaining buffered video is not sufficient, the scheduler starts downloading data through cellular networks. The scheduler will stop downloading data if the buffer is sufficiently large in order to (i) avoid cellular data waste and (ii) not to lose future

[8] We recall that when $u(t) = 0$, buffering (i.e., downloading) stops at time slot t.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2016.2604565, IEEE Transactions on Multimedia

WiFi connection opportunities.

To determine how much the sufficient buffer size is, we compare the costs (C_1 and C_2) and gain (C_3) while downloading data from cellular networks. C_1 and C_2 are projected data waste cost and offloading failure cost when downloading data through cellular networks at time slot t, respectively. The data waste cost is the product of the probability that the user churns before watching out the currently buffered video ($\mathbb{P}\left[T_c \leq \lfloor \frac{b(t)}{r_p} \rfloor \middle| T_c \geq t\right]$) and the sum of cellular cost and energy cost ($r_1(t) + \gamma E_{\text{LTE}}^{\text{tran}}$). There can be another cost, that is the offloading failure cost which is incurred if the user meets a WiFi AP before watching out the currently buffered video (with probability $\mathbb{P}\left[T_{\text{IAP}} \leq \lfloor \frac{b(t)}{r_p} \rfloor - t\right]$), which is inflated by the difference between the cellular cost and WiFi cost ($r_1(t) + \gamma\left(E_{\text{LTE}}^{\text{tran}} - E_{\text{WiFi}}^{\text{tran}} \frac{r_1(t)}{\mathbb{E}[r_2(t)]}\right)$). C_3 is the projected tail energy gain when downloading a chunk through cellular networks at time slot t. When the user will not churn and will not meet a WiFi before watching the current buffer, then the tail energy of $E_{\text{LTE}}^{\text{tran}} \tau^{\text{tail}}$ will be incurred if the user does not download chunks further from cellular networks. Here, we assume that the remaining time of the received video size is generally longer than the tail period. If the sum of projected costs is greater than the tail energy gain ($C_1 + C_2 > C_3$), the scheduler stops downloading through cellular networks ($u(t) = 0$).

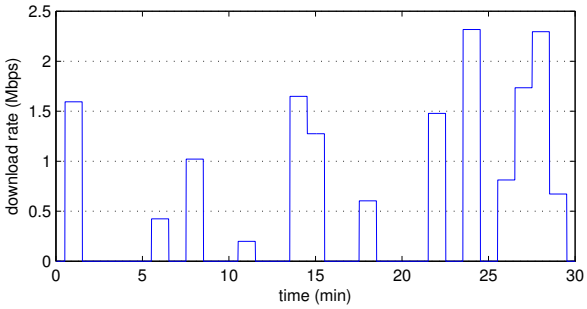
V. Trace-driven Numerical Analysis

A. Setup

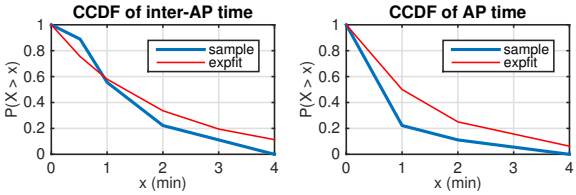
We consider a 10-minute long video ($L = 600$) with playback rate $r_p = 0.69$Mbps.[9] This is a typical rate of a SD (standard definition, 480p) video in YouTube. We let $t_0 = 0$ and $B_{\min} = 0$. The data rate of cellular networks of LTE is assumed to be constant as $r_1(t) = 6.9$Mbps. We first measure the connectivity and data rates of WiFi networks over a Seoul subway trace as depicted in Fig. 4. We choose to use a subway trace since people actively watch videos in public transportation, supported by the fact that 36.5% of people primarily use their mobile data while commuting by public transportation [38]. In subway stations of many countries including South Korea, there exist WiFi APs deployed by cellular carriers with the purpose of cellular traffic offloading. We exploit such APs for our stream scheduling. The inter-AP connection time in those public transportation traces is similar to the difference between arrival times of two consecutive stations, which is just a few minutes in general. Thus, a scheduler can offload some parts of video traffic through WiFi networks as long as the video length is longer than the inter-station time. The average WiFi data rate observed from the trace is 1.24Mbps. We use this average WiFi data rate as the data rate of WiFi in deriving the optimal and heuristic policies. The average inter-AP connection time observed is 2 minutes and the average AP connection time is 1.4 minutes, where they fit well with exponential distributions. We use the audience retention graph of video 1 in Fig. 5. These audience retention

[9]The total video size is 52MB.

(a) Mobility trace in Seoul metro line 2.



(b) Download data rates of WiFi APs.



(c) CCDF of inter-AP and AP connection times.

Fig. 4. Download data rates and CCDF of inter-AP and AP connection times in a 30-min Seoul subway trace.

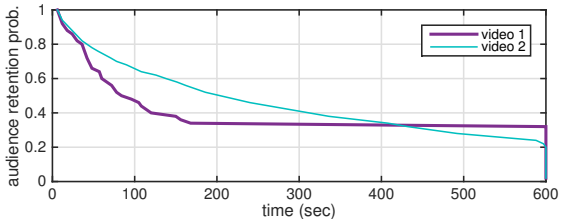


Fig. 5. Audience retention probabilities of video 1 and 2 [14].

graphs are extracted by Hoque et al. from YouTube videos in [14]. We omit the result for video 2 for brevity, since it has similar results as video 1.

We classify streaming policies into four categories: *existing LTE only streaming*, *WiFi offloading + existing streaming*, *optimal and heuristic streaming*, *oracle*. In *existing LTE only streaming* policies (YouTube and Vimeo), large cellular costs are generated since cost-free WiFi networks are not utilized. Also, in YouTube, energy consumption is high because of throttling. In *WiFi offloading + existing streaming*, the video is always downloaded through WiFi networks when the device

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2016.2604565, IEEE Transactions on Multimedia

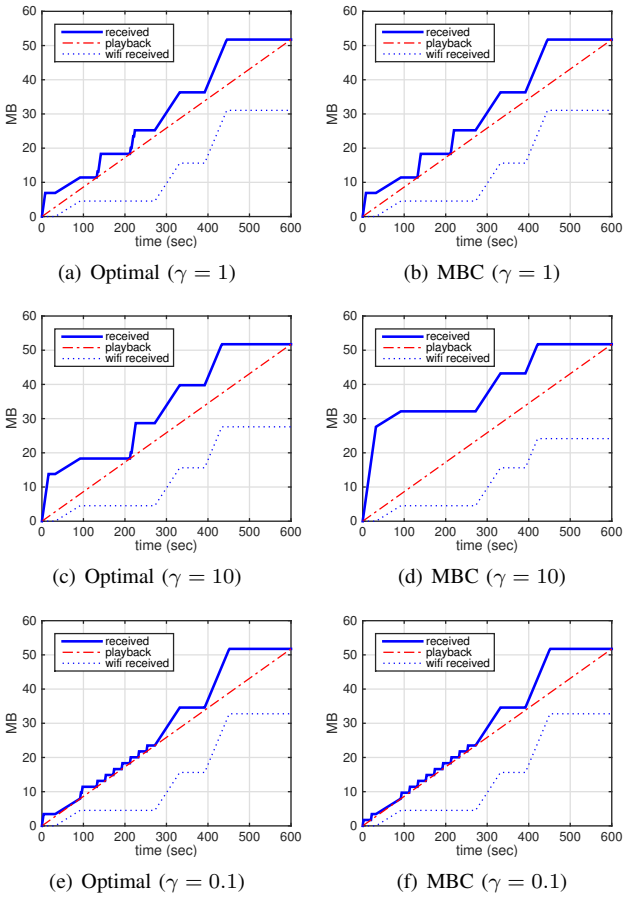
is connected with WiFi networks. When the device is not connected with WiFi APs, existing streaming policies are used. For example, in the *WiFi offloading + YouTube* policy, when the device is connected to a WiFi AP, the video is downloaded with an achievable WiFi data rate, and when the device is not connected to a WiFi AP, the video is downloaded at the playback rate multiplied by 1.25 through cellular networks. An infeasible policy, *oracle*, assumes that the device knows exactly when the user will abandon watching the video and when the user contacts with WiFi APs, a priori. *Oracle* maximally uses WiFi connectivity to minimize the cellular cost.

B. Comparison between an optimal and our heuristic (MBC) scheduling algorithms

We depict total received sizes and received sizes from WiFi networks in an optimal scheduling algorithm with $\gamma = 1, 10, 0.1$ and our heuristic (MBC) scheduling algorithm with $\gamma = 1, 10, 0.1$ in Fig. 6, when the 10-minute long video is requested at 17 minutes of Seoul metro line 2 trace in Fig. 4. In the optimal scheduling algorithm with $\gamma = 1$ in Fig. 6(a), buffer sizes are kept small when the user churning probability is high (see 0-150 seconds for video 1 in Fig. 5), and buffer sizes become larger when the user churning probability is negligible (see after 150 seconds). Our heuristic scheduling algorithm (MBC) closely mimics the optimal scheduling algorithm. Thus, similar cost and energy savings are obtained by using our simple heuristic algorithm. The buffer sizes in the optimal and MBC scheduling algorithm become smaller when cellular costs are more valuable than energy, and γ is set to be small such as 0.1, whereas the buffer sizes become larger when battery lifetime is more valuable than cellular costs, and γ is large such as 10. Thus, as γ becomes smaller, the download schedule results in higher cellular cost saving and smaller energy saving, and vice versa, which will be shown in the next subsection.

C. Cost and energy saving

For various streaming policies, we run simulations 100 times at randomly selected video request points in our subway trace, and depict average energy consumption and cellular cost in Fig. 7. We calculate the expected energy and cellular cost at each time slot in conjunction with the retention probability by multiplying the energy/cost and the retention probability. Existing streaming policies of YouTube and Vimeo have high cellular costs, as they do not adaptively choose access networks, where YouTube also has high energy cost from throttling. We can see that WiFi offloading results in large cellular cost saving, as it prioritizes WiFi networks. In *optimal and heuristic streaming* policies, cost savings are larger than *WiFi offloading + existing streaming* policies and energy savings are larger than the *WiFi offloading + YouTube* policy. In optimal scheduling policies, as γ becomes smaller, the scheduling results in higher cellular cost saving and smaller energy saving, and vice versa. Thus, the best operating point between energy and cellular cost is achievable by controlling the trade-off parameter γ, depending on users' priority. Our simple heuristic algorithm, maximum buffer control (MBC), obtains similar cost and energy savings as an optimal scheduling algorithm.



(a) Optimal ($\gamma = 1$) (b) MBC ($\gamma = 1$)

(c) Optimal ($\gamma = 10$) (d) MBC ($\gamma = 10$)

(e) Optimal ($\gamma = 0.1$) (f) MBC ($\gamma = 0.1$)

Fig. 6. Received sizes in the optimal policy with $\gamma = 1, 10, 0.1$ (a, c, e) and a heuristic (MBC) policy with $\gamma = 1, 10, 0.1$ (b, d, f).

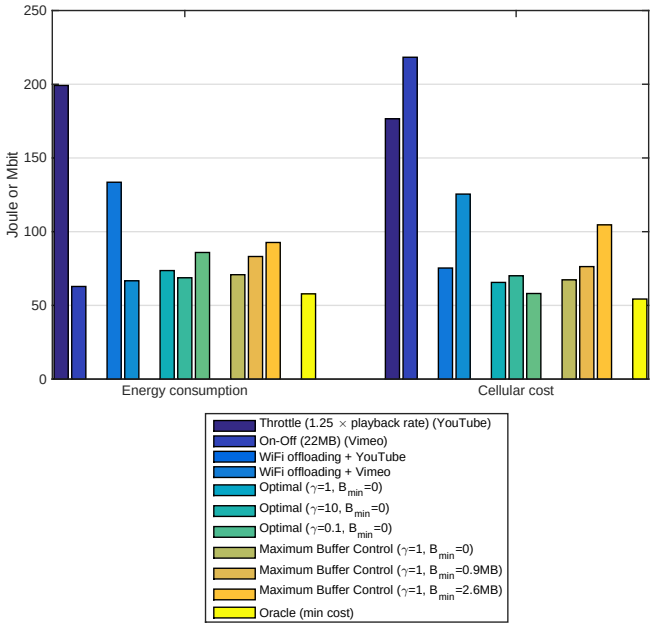


Fig. 7. Energy consumption and cellular costs in various streaming policies. The unit of the y-axis is Joule for energy and Mbit for cellular data cost.

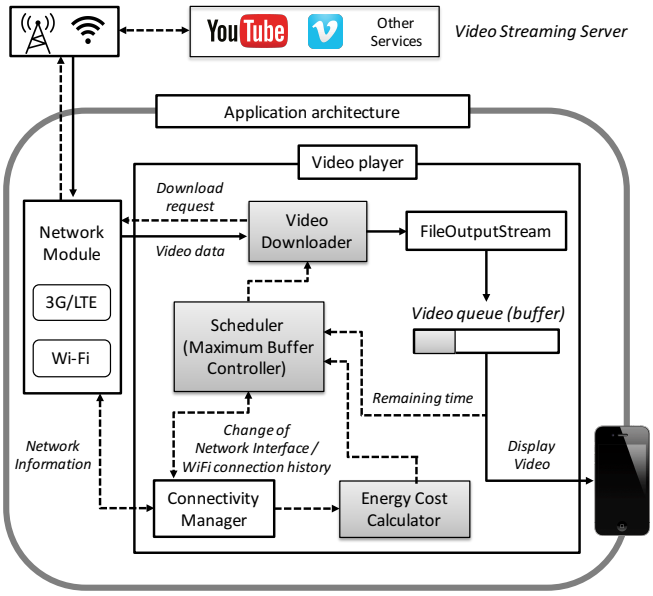


Fig. 8. The overall architecture of our Android video player implementation.

Even with consideration of WiFi scanning energy for periodic scan (e.g., every 20 seconds) that consumes 3 Joules per scan, our MBC algorithm reduces the cellular cost by 59% and the energy consumption by 41% compared to YouTube default streaming. Note that WiFi scanning is performed only when the device is not connected to a WiFi AP. We recall that the *oracle* streaming policy assumes that the device knows when the user will abandon watching the video and when the user will be connected with WiFi APs, a priori. *Oracle* maximally uses WiFi connectivity to minimize the cellular cost and energy, but it is infeasible.

D. Effect of minimum buffer size

In this subsection, we run our MBC algorithm for non-zero minimum buffer size, $B_{\min}$, and discuss its effect on performance gains. We let $B_{\min} = 0.9$MB and 2.6MB, which correspond to 10 and 30 seconds of playback, respectively. In Fig. 7, we plot the results for non-zero minimum buffer sizes. The energy and cost savings are decreased by 6% and 5% for $B_{\min} = 0.9$MB and 11% and 21% for $B_{\min} = 2.6$MB, respectively. The video stalling probability is dependent on the data rate distribution and the buffer size. We remind that for independent and identically distributed data rates, we can bound the video stalling probability by a Chebyshev's inequality, $\Pr\{\mathbb{E}[X] - X \geq \delta\} \leq \frac{Var(X)}{\delta^2}$, where δ increases with the amount of video buffer and the stalling probability (i.e., video buffer becomes empty) quadratically decreases with δ. Thus, as we have a larger minimum buffer size, the video stalling probability will decrease in dynamic network conditions, with the cost of reduced energy and cost savings.

VI. Andriod Implementation

We depict the architecture of our streaming scheduler implementation on Android in Fig. 8. Two helper modules, *connectivity manager* and *energy cost calculator* estimate the network and energy parameters. Based on the previous history of cellular/WiFi data rates and the density of WiFi APs in the projected mobility path (e.g., a route from one station to another in a metro line), the *connectivity manager* estimates the arrival and departure rates of WiFi APs, as well as the average data rates of cellular and WiFi networks ($r_1(t)$ and $r_2(t)$). The *energy cost calculator* estimates the unit transfer and tail energy of cellular and WiFi networks and reports them to the *scheduler* in advance. The network energy can be also acquired from the manufacturer or the video server, as the network energy is almost similar for the same device model.

When a user requests a video clip, the scheduler first collects the video length (L), the playback rate (r_p), and the retention graph from the video streaming server to calculate the churning probability ($p_c(t)$). The *connectivity manager* scans WiFi APs periodically (e.g., every 20 seconds) when the device is not connected to a WiFi AP, and reports the available access networks to the *scheduler*. Then, at each time slot, the *scheduler* (e.g., *maximum buffer control (MBC)*) decides whether or not to download data, and which access network to use. When the scheduler decides to download the data, the *video downloader* executes streaming download in regular order through the chosen access network. The *video downloader* can pause and resume video downloading at any specific point (i.e., sequence number) of the video. It tracks the last downloaded sequence number, and restarts from the next sequence number. The *video queue* reports the amount of buffer size (i.e., received size) to the *scheduler*.

A. Results from indoor experiments

We conduct a proof-of-concept experiment in our laboratory. We excluded outdoor testing because the energy measurement device is not handheld.[10] We install our video player implementation on a Samsung Galaxy Note 2 smartphone. The achievable rate of cellular networks in our lab is about 65Mbps. To be compatible with such a high data rate, we use the video with playback rate $r_p = 6.9$Mbps and video length $L = 600$ seconds. The total video size is 518MB. We use our own video server for stable connection. We note that our implementation can run over any videos on the Internet. For emulation of WiFi connectivity while being mobile, we use the Seoul subway trace in Fig. 4 and turned on and off the AP accordingly. For diversity, we fragment our trace of 30 minutes into three to make each part matches with the test video length: WiFi 1 (from 0 to 10 mins), WiFi 2 (from 10 to 20 mins), and WiFi 3 (from 20 to 30 mins). The average WiFi data rate in our lab is 8.4Mbps. We use the audience retention graph of video 1 in Fig. 5. We let the minimum buffer size $B_{\min} = 8.6$MB, which corresponds to 10 seconds of playback.

[10]The precision of integer-valued battery level reported by Android is not sufficient for our test.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2016.2604565, IEEE Transactions on Multimedia

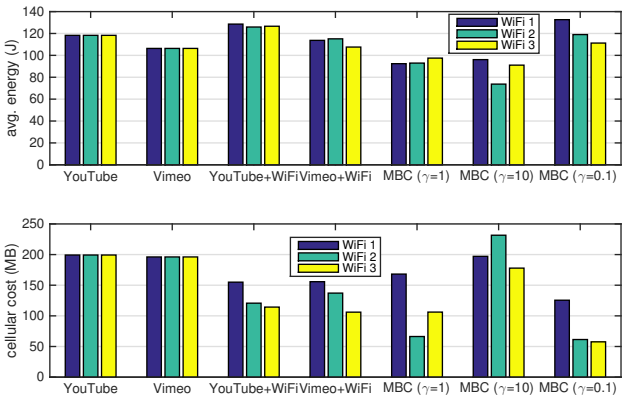


Fig. 9. Average energy consumption and cellular cost of various streaming policies in our in-lab experiment.

During each experiment, we log the amount of data transferred through LTE and WiFi networks and measure power consumption separately. In every experiment, power consumption from screen and video playback, which is about 1064mW (mJ/s), is subtracted. In Fig. 9, we compare the performance of our heuristic (MBC) algorithm with *existing LTE only streaming* (YouTube and Vimeo), *WiFi offloading + existing streaming* (YouTube+WiFi and Vimeo+WiFi). As in the simulation, we calculate the energy and cost at each point, and multiply it by the retention probability to take into account user churning behaviors.

In YouTube and Vimeo streaming policies that do not utilize WiFi networks, cellular costs are high irrespective of WiFi opportunities. Note that the cellular cost is always less than or equal to the video size, as we only take into account the amount of downloaded traffic until a user churns for evaluation. With WiFi offloading, we can see that the cellular cost is reduced by 34.8% in YouTube and 32.2% in Vimeo. However, since the data rate of WiFi is much slower than LTE and the bit per energy is lower in WiFi, the energy consumption of WiFi offloading policies becomes even higher than without WiFi offloading. In our MBC algorithm, this effect is mitigated since it takes account the energy efficiency of access networks. When $\gamma = 1$, the cellular cost is decreased by 43% and the energy consumption is reduced by 20.3% compared to YouTube. Again, the reduction in energy consumption is smaller than our simulation results, because of relatively smaller WiFi data rate. As we decrease γ, the cellular cost reduces, and as we increase γ, the energy consumption decreases. When $\gamma = 0.1$, the cellular cost is decreased by 59% compared to YouTube, with the cost of increased energy consumption. When $\gamma = 10$, the energy consumption is decreased by 26.5% compared to YouTube, with the cost of increased cellular cost. In all cases, our MBC algorithm results in the better operating point in terms of both cellular cost and energy consumption.

VII. Conclusion

In this paper, we suggested an optimal scheduling policy for non-live video streaming traffic based on dynamic programming, which minimizes the sum objective of cellular cost and smartphone energy consumption by exploiting audience retention probability and WiFi contact statistics (i.e., inter-AP and AP time distribution). We further developed a heuristic scheduling policy, Maximum Buffer Control (MBC), which mimics the optimal policy. Through trace-driven simulation over real mobility and video traces, we show that our heuristic scheduling policy obtains 59% cellular cost saving and 41% energy saving (considering WiFi scanning energy) compared to an existing streaming technique of YouTube, where optimal policies obtain 56-63% cellular cost saving and 32-42% energy saving depending on the trade-off parameter γ. We also implemented our streaming algorithm on an Android platform, and experimentally evaluated the performance of MBC. Our implementation and evaluation validate that MBC is easy to adopt and gives real user benefit in practice.

Acknowledgement

This work was supported in part by Institute for Information & communications Technology Promotion (IITP) grants funded by the Korea government (MSIP), (B0126-16-1064, Research on Near-Zero Latency Network for 5G Immersive Service) and (B0190-16-2017, Resilient/Fault-Tolerant Autonomic Networking Based on Physicality, Relationship and Service Semantic of IoT Devices). This work was also supported in part by Ministry of Science, ICT & Future Planning (NRF-2014R1A1A1006330) and Ulsan National Institute of Science and Technology (1.160085.01).

References

[1] Cisco Systems Inc., "Cisco visual networking index: Global mobile data traffic forecast update, 2011-2016," 2012, http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.pdf.

[2] AT&T, "Pricing plans of AT&T," http://www.att.com/shop/wireless/data-plans.html.

[3] X. Chen, N. Ding, A. Jindal, Y. C. Hu, M. Gupta, and R. Vannithamby, "Smartphone energy drain in the wild: Analysis and implications," in *Proceedings of ACM SIGMETRICS*, 2015, pp. 151–164.

[4] Computer World, "4G LTE networks hit battery life on some smartphones, Metrico finds," 2012, http://www.computerworld.com/article/2503760/smartphones/4g-lte-networks-hit-battery-life-on-some-smartphones--metrico-finds.html.

[5] Connect Magazine Report, "Battery Life Competition," 2014, http://www.spirent.com/-/media/Reports/Connect-Magazine_Battery_life_competiton_report.pdf.

[6] A. Finamore, M. Mellia, M. M. Munafò, R. Torres, and S. G. Rao, "Youtube everywhere: Impact of device and infrastructure synergies on user experience," in *Proc. of ACM SIGCOMM conference on Internet measurement conference*, 2011.

[7] YouTube, "YouTube Analytics and Reporting API - Audience retention metric," https://developers.google.com/youtube/reporting/v1/reports/metrics#Audience_Retention_Metrics.

[8] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Transactions on Multimedia*, vol. 8, no. 2, pp. 390–404, 2006.

[9] D. Jurca and P. Frossard, "Media flow rate allocation in multipath networks," *IEEE Transactions on Multimedia*, vol. 9, no. 6, pp. 1227–1240, 2007.

[10] Y. Go, O. C. Kwon, and H. Song, "An energy-efficient http adaptive video streaming with networking cost constraint over heterogeneous wireless networks," *IEEE Transactions on Multimedia*, vol. 17, no. 9, pp. 1646–1657, 2015.

[11] R. Guruprasad and S. Dey, "Battery aware video delivery techniques using rate adaptation and base station reconfiguration," *IEEE Transactions on Multimedia*, vol. 17, no. 9, pp. 1630–1645, 2015.

[12] S. Almowuena, M. M. Rahman, C.-H. Hsu, A. A. Hassan, and M. Hefeeda, "Energy-aware and bandwidth-efficient hybrid video streaming over mobile networks," *IEEE Transactions on Multimedia*, vol. 18, no. 1, pp. 102–115, 2016.

[13] M. A. Hoque, M. Siekkinen, J. K. Nurminen, S. Tarkoma, and M. Aalto, "Saving energy in mobile devices for on-demand multimedia streaming–a cross-layer approach," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 10, no. 3, p. 25, 2014.

[14] M. A. Hoque, M. Siekkinen, and J. K. Nurminen, "Using crowd-sourced viewing statistics to save energy in wireless video streaming," in *Proc. of ACM MobiCom*, 2013.

[15] M. Siekkinen, M. A. Hoque, and J. K. Nurminen, "Using viewing statistics to control energy and traffic overhead in mobile video streaming," *IEEE/ACM Transactions on Networking*, vol. PP, no. 99, pp. 1–15, 2015.

[16] X. Li, M. Dong, Z. Ma, and F. C. Fernandes, "Greentube: power optimization for mobile videostreaming via dynamic cache management," in *Proc. of ACM international conference on Multimedia*, 2012.

[17] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: Evidence from a large video streaming service," in *Proc. of ACM SIGCOMM*, 2014.

[18] C. Singhal, S. De, R. Trestian, and G.-M. Muntean, "Joint optimization of user-experience and energy-efficiency in wireless multimedia broadcast," *IEEE Transactions on Mobile Computing*, vol. 13, no. 7, pp. 1522–1535, 2014.

[19] A. Bokani, M. Hassan, and S. Kanhere, "Http-based adaptive streaming for mobile clients using markov decision process," in *IEEE 20th International Packet Video Workshop (PV)*, 2013.

[20] J. Chen, A. Ghosh, J. Magutt, and M. Chiang, "Qava: quota aware video adaptation," in *Proc. of ACM CoNEXT*, 2012.

[21] J. Chen, R. Mahindra, M. A. Khojastepour, S. Rangarajan, and M. Chiang, "A scheduling framework for adaptive video delivery over cellular networks," in *Proc. of ACM MobiCom*, 2013.

[22] S.-P. Chuah, Z. Chen, and Y.-P. Tan, "Energy-efficient resource allocation and scheduling for multicast of scalable video over wireless networks," *IEEE Transactions on Multimedia*, vol. 14, no. 4, pp. 1324–1336, 2012.

[23] S. Sharangi, R. Krishnamurti, and M. Hefeeda, "Energy-efficient multicasting of scalable video streams over wimax networks," *IEEE Transactions on Multimedia*, vol. 13, no. 1, pp. 102–115, 2011.

[24] A. Schulman, V. Navday, R. Ramjeey, N. Spring, P. Deshpandez, C. Grunewald, K. Jain, and V. N. Padmanabhan, "Bartendr: A practical approach to energy-aware cellular data scheduling," in *Proc. of ACM MobiCom*, 2010.

[25] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong, "Mobile data offloading: How much can wifi deliver?" *IEEE/ACM Transactions on Networking*, vol. 21, no. 2, pp. 536–550, 2013.

[26] A. Balasubramanian, R. Mahajan, and A. Venkataramani, "Augmenting mobile 3g using wifi," in *Proc. of ACM MobiSys*, 2010.

[27] S. Dimatteo, P. Hui, B. Han, and V. O. Li, "Cellular traffic offloading through wifi networks," in *Proc. of IEEE MASS*, 2011.

[28] Y. Moon, D. Kim, Y. Go, Y. Kim, Y. Yi, S. Chong, and K. Park, "Practicalizing delay-tolerant mobile apps with cedos," in *Proc. of ACM MobiSys*, 2015.

[29] Tcpdump, http://www.tcpdump.org.

[30] Monsoon Power Monitor, http://www.msoon.com.

[31] M. A. Hoque, M. Siekkinen, J. K. Nurminen, M. Aalto, and S. Tarkoma, "Mobile multimedia streaming techniques: Qoe and energy consumption perspective," *arXiv preprint arXiv:1311.4317*, 2013.

[32] A. Wang, J. Yu, D. Liu, and Y. Cui, "A measurement study for transmission energy on mobile device," in *Proc. of IEEE ICCSNT*, 2012.

[33] Q. Xu, S. Mehrotra, Z. M. Mao, and J. Li, "Proteus: Network performance forecast for real-time, interactive mobile applications," in *Proc. of ACM MobiSys*, 2013.

[34] J. Yao, S. S. Kanhere, and M. Hassan, "An empirical study of bandwidth predictability in mobile computing," in *Proc. of ACM WiNTECH*, 2008.

[35] ——, "Improving qos in high-speed mobility using bandwidth maps," *IEEE Transactions on Mobile Computing*, vol. 11, no. 4, pp. 603–617, 2012.

[36] D. P. Bertsekas, *Dynamic Programming and Optimal Control, Vol. II*. Athena Scientific, 2007.

[37] W. B. Powell, *Approximate Dynamic Programming: Solving the curses of dimensionality*. John Wiley & Sons, 2007, vol. 703.

[38] DMC report, "Analysis of smartphone usage patterns and consumer awareness investigation of mobile advertisements," http://www.dmcreport.co.kr/content/ReportView.php?type=Survey&id=1340&gid=10.

Joohyun Lee (S'11-M'14) received his B.S. and integrated M.S./Ph.D degree in the Department of Electrical Engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2008 and 2014, respectively. He is currently a postdoctoral researcher in the Department of Electrical and Computer Engineering at the Ohio State University. He received IEEE William R. Bennett Prize Paper Award in 2016, given to the best original paper published in IEEE/ACM Transactions on Networking in the previous three calendar years. His research interests are in the areas of context-aware networking and computing, mobility-driven cellular traffic offloading, energy-efficient mobile networking, protocol design and analysis for delay-tolerant networks, and network economics and pricing.

Kyunghan Lee (S07-M'10) is an associate professor in the school of electrical and computer engineering at UNIST (Ulsan National Institute of Science and Technology), Ulsan, South Korea. His research interests include low latency networking, human mobility modeling, disruption tolerant networking and context-aware mobile system design. He received two IEEE ComSoc William R. Bennett Prize Paper Awards in 2013 and 2016, given to the best original paper published in IEEE/ACM Transactions on Networking in the previous three calendar years. He received his B.S., M.S., and Ph.D. degrees in Electrical Engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2002, 2004, and 2009, respectively.

Choongwoo Han received his B.S. in Electrical and Computer Engineering from Ulsan National Institute of Science and Technology (UNIST), Ulsan. He is currently studying for a master degree at Korea Advanced Institute of Science and Technology (KAIST), Daejeon. His research interests include the areas of software engineering, bug finding, and program analysis.

Taehoon Kim received his B.S. degree in Electrical and Computer Engineering from Ulsan National Institute of Science and Technology (UNIST), Ulsan, Korea, in 2015. He is currently a software engineer in Devsisters, a mobile game company in Korea, and building an automatic game balancing based on deep learning and reinforcement learning. His research interests focus on deep learning, reinforcement learning, and language understanding.

Song Chong (M'93) is a Professor in the School of Electrical Engineering at Korea Advanced Institute of Science and Technology (KAIST). He is the Head of Computing, Networking and Security Group of the school since 2009, and the Founding Director of KAIST 5G Mobile Communications & Networking Research Center. Prior to joining KAIST, he was with the Performance Analysis Department, AT&T Bell Laboratories, Holmdel, New Jersey, USA, as a Member of Technical Staff. His current research interests include wireless networks, mobile systems, performance evaluation, distributed algorithms and data analytics. He is on the editorial boards of IEEE/ACM Transactions on Networking, IEEE Transactions on Mobile Computing and IEEE Transactions on Wireless Communications. He was the Program Committee Co-Chair of IEEE SECON 2015 and has served on the Program Committee of a number of leading international conferences including IEEE INFOCOM, ACM MobiCom, ACM CoNEXT, ACM MobiHoc, IEEE ICNP and ITC. He serves on the Steering Committee of WiOpt and was the General Chair of WiOpt 2009. He received two IEEE William R. Bennett Prize Paper Awards in 2013 and 2016, given to the best original paper published in IEEE/ACM Transactions on Networking in the previous three calendar years, and the IEEE SECON Best Paper Award in 2013. He received the B.S. and M.S. degrees from Seoul National University and the Ph.D. degree from the University of Texas at Austin, all in electrical engineering.