

## PAPER

# Energy-Aware Download Method in LTE Based Smartphone

Jie REN<sup>†</sup>, Member, Ling GAO<sup>†a)</sup>, Hai WANG<sup>†</sup>, QuanLi GAO<sup>†</sup>, and ZheWen ZHANG<sup>†</sup>, Nonmembers

**SUMMARY** Mobile traffic is experiencing tremendous growth, and this growing wave is no doubt increasing the use of radio component of mobile devices, resulting in shorter battery lifetime. In this paper, we present an Energy-Aware Download Method (EDM) based on the Markov Decision Process (MDP) to optimize the data download energy for mobile applications. Unlike the previous download schemes in literature that focus on the energy efficiency by simply delaying the download requests, which often leads to a poor user experience, our MDP model learns off-line from a set of training download workloads for different user patterns. The model is then integrated into the mobile application to deal the download request at runtime, taking into account the current battery level, LTE reference signal receiving power (RSRP), reference signal signal to noise ratio (RSSNR) and task size as input of the decision process, and maximizes the reward which refers to the expected battery life and user experience. We evaluate how the EDM can be used in the context of a real file downloading application over the LTE network. We obtain, on average, 20.3%, 15% and 45% improvement respectively for energy consumption, latency, and performance of energy-delay trade off, when compared to the Android default download policy (Minimum Delay).

**key words:** MDP, energy-aware, LTE, mobile devices, download method

## 1. Introduction

From laptop to smartphone, energy conservation always is the key problem in the industry. The studies [1]–[3] have shown that the majority of power consumption of smartphone can be attributed to the radio module when the mobile applications transmit data via internet to guarantee the good running and user experience. In addition, the recent statistics [4] indicate that the Internet usage on mobile devices exceeded that on desktop computers since 2014 and mobile data traffic will grow at a compound annual growth rate (CAGR) of 53 percent from 2015 to 2020, reaching 30.6 exabytes per month by 2020 [5]. There is no doubt that the radio component will increase the battery burden.

The fast penetration of wireless network and richer mobile applications [6] implies a growing need for high-performance and energy-efficient mobile data download strategy to ensure a continued end-user experience. The Android default download scheme is the immediate response when the user make a download request. However, this download method will waste much energy under an adverse network, and many users experience a higher battery

drain when the mobile connected to the network. This paper explores a robust method based on three aspects to reduce the energy expenditure of data download on the smartphone. Firstly, the EDM takes the RSRP (Reference Signal Received Power) and RSSNR (Reference Signal Signal to Noise Ratio) into account, they are two primary factors in LTE signal strength which affect the energy-efficient and user experience. Secondly, the user submit download requests anytime, anywhere with any possible signal strength, dose not think about the battery lifetime. Our goal is to schedule these requests in an energy-efficient way. Thirdly, many applications live with a delay-tolerant download capability. Actually, the users have different delay tolerance toward different tasks, for instance, we want to receive a SMS as soon as possible, even with a poor signal, and also be able to tolerate some delay when downloading a large-sized video in exchange for extending phone lifetime.

This paper presents a novel approach to exploit the energy efficiency for data download. To this end, we carried out the following work. Firstly, we formulate the energy management of data download problem as an optimization formulation which minimizes the per byte energy expenditure and improve the user experience. The MDP framework allows the application to incorporate user preference and user profiles into decisions making at run time. Secondly, we profile the power draw of LTE in different states for MDP model building and evaluation. LTE technology is different from 3G technology in tail time for different states, LTE without any tail energy in fully connected state which named “Activity” and the overhead of “dormant” just accounts for one third of “Activity” where the FACH makes up 50% of DCH [7]. Thirdly, we collect the context history data to train the MDP model and evaluate it against other two approaches. All measurements and evaluations are conducted on the LTE-based SAMSUNG GALAXY S5 smartphone.

The key contribution of this paper is a novel MDP based model that can be used to optimize energy consumption for data download on smartphones. We implement the EDM on an online music player. We then compare it with the state-of-art “concentrated download, low power, stable link selection algorithm” (CLSA) [17] and Android default download scheme. The result shows that the EDM outperforms the other two for reducing energy consumption and latency, and achieves a better energy-delay trade off.

Manuscript received August 17, 2016.

Manuscript revised October 14, 2016.

Manuscript publicized November 18, 2016.

<sup>†</sup>The authors are with the Department of Computer Science, Northwest University, Xi'an, 710000, China.

a) E-mail: gl@nwu.edu.cn (Corresponding author)

DOI: 10.1587/transinf.2016EDP7349

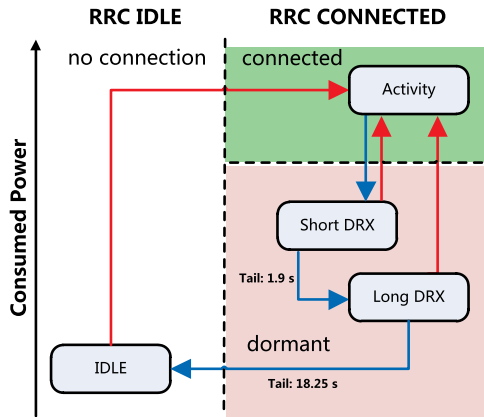


Fig. 1 LTE state machine

## 2. Background and Motivation

The EDM is built on the LTE network, so we provide a brief overview of LTE mechanism, and conduct a series of experiments to illustrate how LTE works (Sect. 2.1). Following that we explain what is important for the user experience (Sect. 2.2).

### 2.1 Power States and Transitions of the LTE Radio

Figure 1 depicts the state machine for all the LTE development which is implemented by the Radio Resource Control (RRC) protocol [9], and the RRC defines the following two states for smartphones to control their radio interfaces.

**RRC IDLE:** The radio remains “IDLE” state in the absence of any network activity and only listening to the control traffic. No radio resources are assigned to the client within the carrier network. The LTE radio draws nearly zero power (8.844mA) in this state as Table 3 shows.

**RRC CONNECTED:** The transition from IDLE to CONNECTED caused by the data transfer event. At first, a request will be issued to the RRC. Then the radio state move to the “Activity”. Finally, a network context is established between the radio tower and the LTE device. Because the “Activity” state requires a significant and highly variable amount of power, multiple sub-states are available for more efficient operations. Once the transmission is completed, the device will move to the “dormant” which contains two sub-states, Short Discontinuous Reception (Short DRX) and Long Discontinuous Reception (Long DRX). The User Equipment only establishes network context without any resources during the “dormant” state.

Figure 2 maps the LTE RRC states to the potential power consumption. This figure plots the instantaneous power consumption and time consuming in each state when downloading a 3 MB file from a remote server to the smartphone. The “Activity” state takes 4.55 seconds which equals to the download time as we measured, and the “dormant” state lasts 20.15 seconds, including 1.9 seconds Short DRX and 18.25 seconds Long DRX. Averaged over 10 iterations,

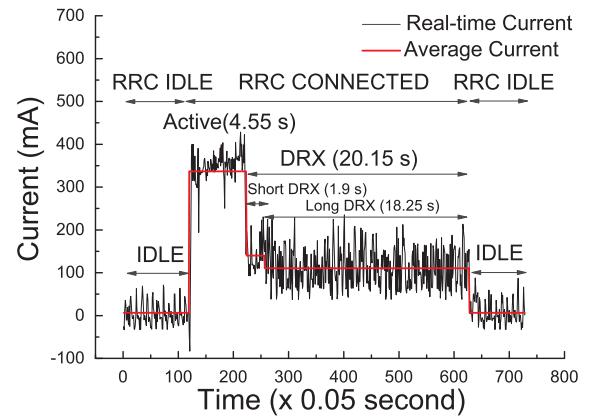


Fig. 2 Power consumption and time consuming in different states of LTE for downloading a 3 MB file

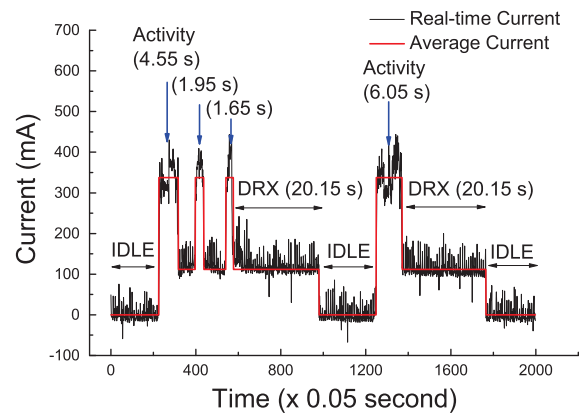


Fig. 3 Power consumption and time consuming for downloading four 3 MB files with different arrival time

the “Activity” state aggregate power is 325 mA. Figure 3 shows the average power consumption for multiple downloads. Due to the little difference between Short DRX and Long DRX, we calculate the average value as “dormant” power. The graph shows that the first task takes 4.55 seconds to complete, and the following two successive requests which only spend 1.95 and 1.65 seconds respectively. Because the LTE device may first require anywhere from 10 to 100 milliseconds of latency to negotiate the required resources with the RRC, this negotiation time (IDLE to CONNECTED) is specified as less than 100 milliseconds, but it takes less than 50 milliseconds from “dormant” to “Activity”. The second and third downloads consume much less time than 4.55 seconds, not only the UE still in the “dormant” state, but also all the files come from the same server. After 19.75 seconds “dormant” state without any other transmissions, the radio releases all the network resources and goes back to the IDLE state, so that the last download request takes 6 seconds, much longer than previous two tasks.

### 2.2 Insights for User Experience and Signal Strength

It is straightforward to relate the user experience (download

Table 1 Signal strength

RSRP Level	RSRP Value	RSSNR Level	RSSNR Value	Signal Level	Signal Value
<i>GREAT</i>	$\geq -95$	<i>GREAT</i>	$\geq 45$	<i>GREAT</i>	4
<i>GOOD</i>	$\geq -105$	<i>GOOD</i>	$\geq 10$	<i>GOOD</i>	3
<i>MODERATE</i>	$\geq -115$	<i>MODERATE</i>	$\geq -30$	<i>MODERATE</i>	2
<i>POOR</i>	<i>otherwise</i>	<i>POOR</i>	<i>otherwise</i>	<i>POOR</i>	1
<i>No Signal</i>	<i>Integer.MAX_VALUE</i>	<i>No Signal</i>	<i>Integer.MAX_VALUE</i>	<i>No Signal</i>	0

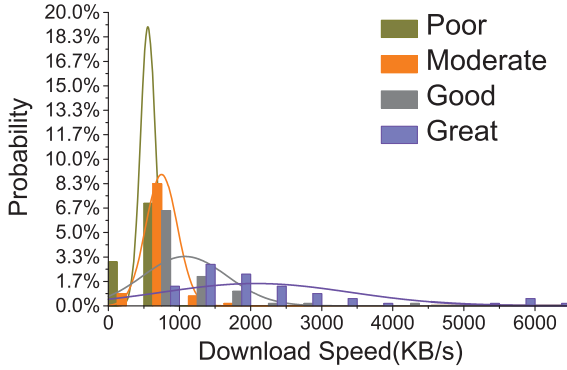


Fig. 4 PDF of download speed for different signal levels

rate) to the signal strength. As the users' intuitive feelings that the adverse network increases the download time compared with the good network environment. In addition, there is a lot of evidence that the mobile users have experienced quicker battery drain with poor network, not only the extra download time rises up the battery expenditure, the mobile devices also have to strengthen the signal transmit power to keep the connection when the signal is weak.

Except the above intuitive understanding, we have rather limited research of the quantitative impact of signal strength on download speed. To study the relationship between them, we consider different scenarios from the user holds outdoor activities with good signal strength to the user stay in the closed environment with poor signal strength. Under different network environments, we download the 3MB file by every 30 seconds and conduct 100 measurements. Figure 4 shows the probability density function (PDF) of download rate for different signal levels. As can be seen from this diagram, the download speed of different signals obeys the gauss distribution. The stronger the signal strength is, the faster the speed will be. For simplicity, we use the Android API to get the LTE signal level that contains 5 different levels (no signal to good signal), and we define the value of no signal level as 0, so that we can reject the download request in such a poor network environment, details are listed in Table 1. The Android operating system defined the signal level depends on RSRP and RSSNR, the source code as follows:

```

if (mRsrp==Integer.MAX_VALUE)
    level=levelRssnr;
else if(mRssnr==Integer.MAX_VALUE)
    level=levelRsrp;
else
    level=(levelRssnr < levelRsrp) ? levelRssnr :
        levelRsrp

```

### 3. MDP System Design

The EDM is based on the MDP for energy-efficient data download. We choose MDP because it provides a mathematical framework for modeling decision making in situations where outcomes are partly random and partly under the control of a decision maker. So the different user patterns can get benefit from the MDP. The input to our model is the current state of the smartphone. The output of our model is an action that indicates the best decision from the global perspective. This section describes the EDM in details. At first, we introduce the general framework of MDP. And then we discuss how we formulate the energy management problem for mobile download as a Markov Decision Process. The goal of EDM is to assign a dynamic download decision to reduce the download overhead and improve the user experience.

#### 3.1 MDP Overview

In the typical definition, a Markov decision process is a 4-tuple  $\langle S, A, P, R \rangle$ , where

- $S$  is a finite set of states.
- $A$  is a finite set of actions.
- $P(s, s', a) = P(s_{t+1} = s' | s_t = s, a_t = a)$  is the probability that the action  $a$  in the state  $s$  at time  $t$  will lead to the state  $s'$  at time  $t + 1$ .
- $R(s, s', a)$  or  $R(s', a)$  is the immediate reward (or expected) received after transition to state  $s'$  from state  $s$  with action  $a$ .

The core problem of MDP is to find a **policy** for the decision maker. The algorithm to calculate this optimal policy requires storage for two arrays indexed by state: value  $V$ , which contains real values, and policy  $\pi$  which refers to actions. Then carry out the two kinds of steps, which are repeated in some order for all the states until no further changes take place. At the end of the algorithm,  $\pi$  will contain the solution and  $V(s)$  will contain the discounted sum of the rewards which is earned (on average) by following that solution from state  $s$ . They are defined as follows:

$$\pi(s) := \arg \max_a \left\{ \sum_{s'} P_a(s, s') (R_a(s, s') + \gamma V(s')) \right\} \quad (1)$$

$$V(s) := \sum_{s'} P_{\pi(s)}(s, s') (R_{\pi(s)}(s, s') + \gamma V(s')) \quad (2)$$

#### 3.2 MDP Model of EDM

Some mobile applications are delay tolerant, they consume a

huge amount of traffic when users perform some operations, such as watching the online YouTube videos and refreshing the Instagram photos, which result in the rapid battery drain. We take the advantage of the delay-tolerant to save the energy of downloading. We describe below how we formulate the energy management problem for downloading data on smartphones as a Markov Decision Process and define the following notations.

$t$ : current time slot.

$T$ : total using time after charging complete.

$I_t$ : time interval of  $T$ .

$e$ : remaining battery (level).

$e_{dd}$ : power for download action over LTE network.

$e_{dy}$ : power for delay action (idle) over LTE network.

$E$ : total energy.

$L_e$ : divide  $E$  into  $L_e$  parts.

$l$ : the level of LTE signal strength.

$q$ : size of current queue backlog.

### 1) State Space

In our model, the state includes the current time, remaining battery, signal strength, and the size of queue backlog.

$$s := (t, e, l, q) \quad (3)$$

In the action space  $A$ , we define  $a_{download}$  as the action for downloading data at current time slot,  $a_{delay}$  as the action for delaying the download request to the next time slot. We formulate the action  $a$  in the action space  $A$  as follows.

$$A := (a_{download}, a_{delay}) \quad (4)$$

$t$ : We divide  $T$  into  $n$  ticks,  $n$  is equal to  $T/I_t$ , for instance, we set  $T$  as 1 hour, and the time interval is 1 minute, so  $t$  includes 60 slots (60 states). The transition of  $t$  is defined as:

$$P(t_j|t_i) = \begin{cases} 1 & \text{if } t_j = t_i + 1 \\ 1 & \text{if } t_j = t_i = \frac{T}{I_t} \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

$e$ : Denote the remaining energy. We divide the  $E$  into  $L_e$  equal parts, so each part contains  $E/L_e$  energy. The transition of  $e$  depends on the action that will be taken. And the energy discrepancy is due to different actions. Similar to the Jigsaw system [8], the probability of the battery level changing from the current level to the next level from time tick  $t_i$  to  $t_{i+1}$  is calculated as:

$$p(download) = \frac{\text{power}(download)}{E/L_e} \times \frac{T}{T/I_t} \quad (6)$$

The transition probability of  $e$  for different actions as follows:

$$P_{delay}(e_j|e_i) = \begin{cases} 1 & \text{if } e_j = e_i \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

$$P_{download}(e_j|e_i) =$$

$$\begin{cases} 1 & e_j = e_i = 1 \\ p(download) & \text{if } e_j = e_i - 1 \\ 1 - p(download) & \text{if } 1 < e_j = e_i < L_e \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

$l$ : We define the LTE signal strength as  $l$ . In Google source code, the signal strength of LTE owns five levels, from 0 (no signal) to 4 (best signal strength), calculated by the RSRP and RSSNR, the  $l$  state transition probability matrix is obtained from users' historical context data (Sect. 5.1).

$q$ : We define three fixed download request probabilities, the value  $p$  from 0.2 (light user) to 0.8 (heavy user) corresponding to the different user patterns. And we assume the task will be finished within one time tick if the action is download:

$$P(q_j|q_i) = \begin{cases} p & \text{if } q_j = q_i - 1 \\ 1 - p & \text{otherwise} \end{cases} \quad (9)$$

To reduce the complexity of the MDP, we assume that the states  $t, e, l, q$  are independent from each other. Thus, we define the overall system transition probability as:

$$P_a(s_j|s_i) = P(t_j|t_i) \times P_a(e_j|e_i) \times P(l_j|l_i) \times P(q_j|q_i) \quad (10)$$

### 2) Reward Function

We define the  $V(t, e, l, q)$  as the optimal value which is the maximum total reward at the current state  $S(t, e, l, q)$  optimized over all possible actions. If the action is download, we define the optimal value as:

$$V_{download}(t, e, l, q) = V(t + 1, e - e_{dd}, l, N - X) + R(X) \quad (11)$$

$R(X)$  is the immediate reward when downloading  $X$  tasks,  $V(t + 1, e - e_{dd}, l, N - X)$  is the future reward. When the action is delay, we have:

$$V_{delay}(t, e, l, q) = V(t + 1, e, l, N + 1) \quad (12)$$

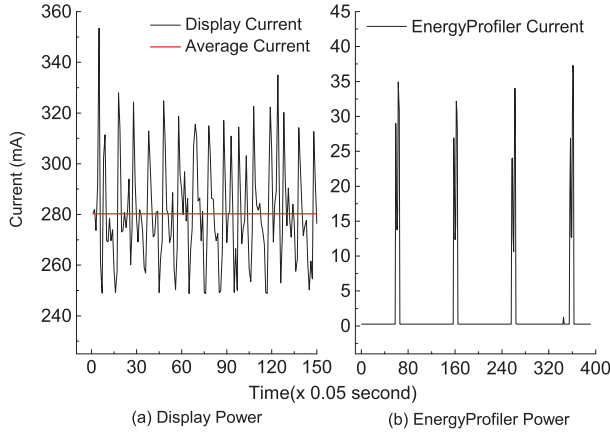
In this paper, the reward function is the key component of MDP and we set  $R(N) = N \times l$ , as it will balance the overhead and user experience. For example, an excellent signal strength offers a good Internet experience and saves a lot of energy as well, we come up with the relevant experiments and evaluate the results in Sect. 5. Based on the above formulation, we can obtain all context transition probabilities using frequency counts and apply the Policy Iteration algorithm [10] to learn the optimal policy.

### 4. Experimental Setup

All of experiments are conducted on the LTE-enabled SAM-SUMG GALAXY S5 which runs the latest Android 6.0 Marshmallow operating system and using the China Mobile LTE-TDD technology. We also develop a context profiling application: EnergyProfiler, a background program to collect run-time data, the collected information are high lighted in Table 2. We use a DC power supply (Agilent 66332A) to

**Table 2** The collected information and description

Probe	Description
<i>Time</i>	<i>Current timestamp</i>
<i>Battery</i>	<i>The remaining battery level</i>
<i>Signal Level</i>	<i>Current signal level</i>
<i>Queue Backlog</i>	<i>The size of download request queue</i>

**Fig. 5** The power consumption for screen and EnergyProfiler**Table 3** The power consumption for the smartphone in different states

Screen State	Radio	Power(mA)	Power with EnergyProfiler(mA)
Screen Off	Airplane	2.773	16.967
Screen On	Airplane	284.480	298.803
Screen Off	LTE	11.617	48.638
Screen On	LTE	298.439	334.662

power the phone instead of the battery, and a multimeter (Agilent 334410A) to get the real time current. For simplicity, we represent power consumption by the current value in milliamper. The actual power draw is the current value multiplied by 5V (DC supply voltage).

Unless otherwise stated, we keep the smartphone's screen on during the experiment, so we need to subtract the display and EnergyProfiler power from the total power consumption. To quantify the display cost, we force the phone into the airplane mode, kill all the background applications, and set the brightness to the maximum (255). After doing above things, the phone goes to the idle state with 284.48 mA. We then disable the display and get another current value (2.77 mA). Averaged over 10 iterations, an average display aggregate power (280.242 mA) as Fig. 5 (a) presents. Figure 5 (b) describes the EnergyProfiler power consumption when the screen is off, the every 5 seconds current pulse caused by the background sampling program. To minimise the impact on battery life and system performance, EnergyProfiler collects the information by using the Android public APIs and the information stored locally, so that no network usage is required. Table 3 presents the power consumption for smartphone in details, when the radio is on (LTE) or off (Airplane mode) with the screen is on or off. In addition, the power consumed in Airplane mode

(16.967mA) is much less than the LTE connection state (48.638mA) when the EnergyProfiler is running in the background, except the power consumption to keep the LTE connection, the additional overhead under LTE network caused by the EnergyProfiler fetches the Signal Strength which is not considered during the Airplane mode.

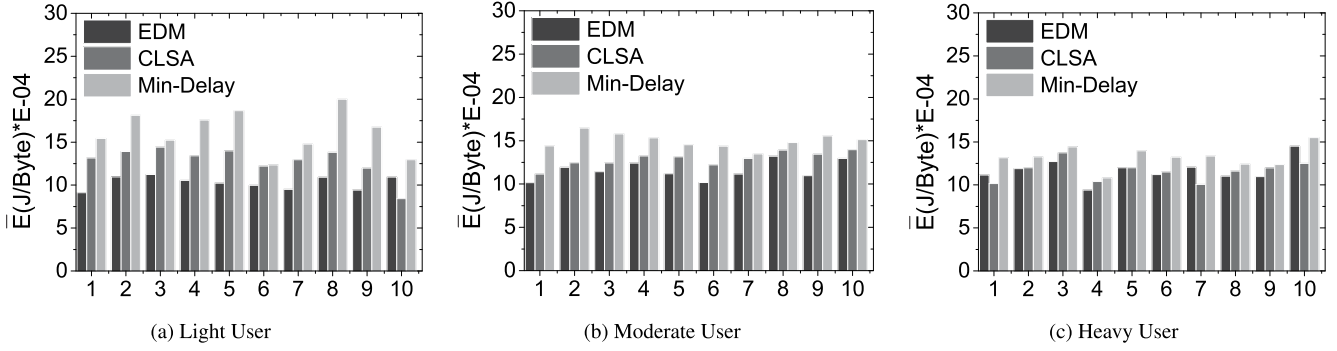
## 5. EDM Modelling and Evaluation

We follow a training-testing procedure to build the model and evaluate the performance of EDM. We collect one month log history for training the MDP model, and apply the off-line built model in the music player for testing. We compare the EDM against CLSA [17] and the Android default download policy (Min-Delay) through the real world for 10 mobile users over one week, the result shows that our approach delivers a better energy reduction than the other two policies. Finally, we analyze the working mechanism of our approach.

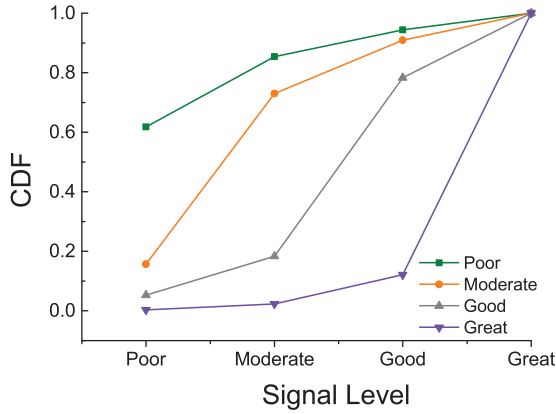
### 5.1 Modelling

The core of our EDM is the MDP model. We use an off-line learning scheme to train the MDP model. We set  $T = 2$  hours and each time tick is 30 seconds. So the total time is divided into 240 parts, the smartphone simulates to discharge from timestamp  $t = 0$  to  $t = 240$ . In the simulation, we use the following parameters from the power measurement on SAMSUNG Galaxy S5 which runs the latest Android 6.0 Marshmallow operating system. We define the initial energy level is 10 unites (10% of the whole battery capacity). For simplicity, the energy impact of signal strength is not considered, and the download power consumption is 325mA as described previously. In order to avoid the impact of network load, we conduct the experiments during the certain period of the day. We collect 10 mobile users historical context data for training, and get the signal level transition for each one. Figure 6 presents the CDF of signal level transition for user No.1. As we can see from this diagram, the signal levels prefer to maintain the last state (over 40% probability) from current time state to the next time state. For the *Good* signal level, there are about 3%, 15%, 55% and 27% probabilities transfer to the *Poor*, *Moderate*, *Good* and *Great* respectively. We collect the users' signal level transition data to build their own MDP model. Given the above mentioned parameters, we can obtain all context transition probabilities and find the optimal decision by Policy Iteration Algorithm [10]. We use Matlab [11] and its Markov Decision Processes Toolbox to build the MDP model. The output is a four-tuple decision table, the mobile application can take the action  $\langle \text{download}, \text{delay} \rangle$  according to the current state  $\langle t, e, l, q \rangle$ . Table 4 lists a piece of the decision table from the No.1 user. Once the MDP model has been built using all the available training data, no further learning takes place.

Our Model can be easily built for many other mobile devices with different user patterns through the on-line MDP



**Fig. 7** EDM average per-byte energy consumption between the light user (a), moderate user (b) and heavy user (c) for downloading the same file compared to the CLSA and the Android default policy Min-Delay.



**Fig. 6** CDF of signal level transition for the No.1 user

**Table 4** MDP decision table

t	e	l	q	Optimal Value	Decision
22	7	3	9	34.370	Download
22	7	3	0	29.033	Delay
22	7	2	15	29.192	Download
22	6	1	12	15.709	Delay
22	6	1	15	21.605	Delay
22	6	1	30	34.552	Download

based optimisation system. The user can download our customized music player application on the application repository. The application will be profiled while users run it and the performance and energy statistics will be periodically sent back to the server when the device is powered. From time to time, our system will generate MDP models based on the collected information and send them back to different users. Furthermore, the MDP models will be constantly updated so that newly uploaded, unseen situations will be optimised.

## 5.2 Deployment and Evaluation

Once we have built the model as described above, we apply it in the on-line music player for Android to decide which action to take for a *new*, *unseen* download request. In the testing stage, 10 users submitted download requests

randomly during the 2 hours. Then the EDM lookups the MDP decision table according to the current time, energy level, signal level and queue backlog size. Finally, the EDM decides what action to take (delay or download).

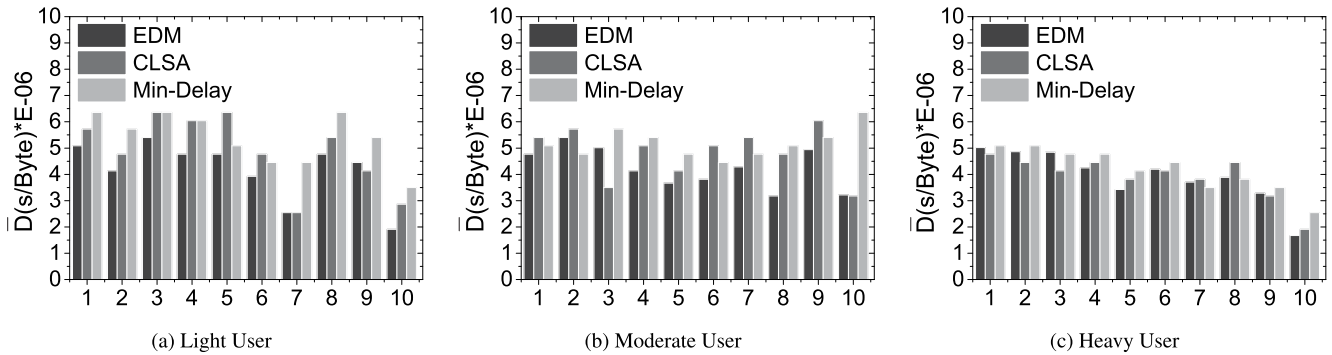
To illustrate the performance of EDM, we compare it with two alternative approaches, a state-of-the-art energy efficiency download mechanism CLSA [17] and Android default download policy Minimum Delay. CLSA uses the Lyapunov optimization framework and optimal consumer model to decide which wireless network interface to use and whether to delay the download tasks. The approach is used to select an energy-efficient and stable link to download data by analyzing the information of smartphone and network state. The core of CLSA is the parameter  $W$ , which influences the energy and latency directly. The small  $W$  will let the CLSA response the download request more quickly but consume more energy. However, the big  $W$  will let the user wait a long time, which causes the poor user experience. In the experiments, we set  $W = 22$  as [17] defined. We simulate various user patterns with different download probabilities, the light user ( $p = 0.2$ ), moderate user ( $p = 0.5$ ) and heavy user ( $p = 0.8$ ). Then we compare the energy ( $\bar{E}$ ), delay ( $\bar{D}$ ) and *performance* for three download schemes.  $\bar{E}$  is the average energy consumption for each byte.  $T(i)$  is the required time to download the task  $i$ .  $\bar{D}$  denotes the average latency for each byte.  $D(i)$  is the time from the user make the download request to finish the request task. Furthermore, we define a new metric *performance* to evaluate the achieved degree of energy-delay trade off, which takes the energy and latency into account by multiplying the  $\bar{E}$  and  $\bar{D}$ .

$$\bar{E} = \frac{\sum_{i=1}^N T(i) \times Power}{FileSize \times N} \quad (13)$$

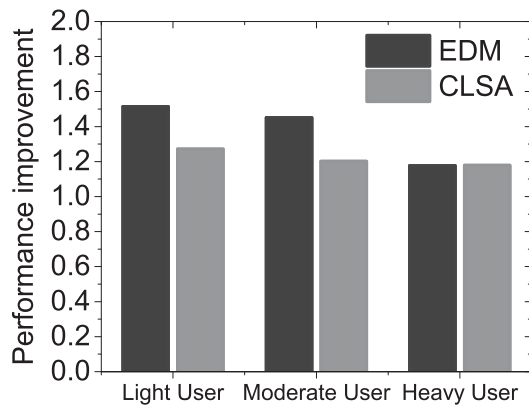
$$\bar{D} = \frac{\sum_{i=1}^N D(i)}{FileSize \times N} \quad (14)$$

$$Performance = \bar{E} \times \bar{D} \quad (15)$$

Figure 7 plots the per-byte energy consumption for 10 smartphone users in different usage patterns. Compared



**Fig. 8** EDM average per-byte latency between the light user (a), moderate user (b) and heavy user (c) for downloading the same file compared to the CLSA and the Android default policy Min-Delay.



**Fig. 9** Achieved performance of EDM and CLSA over Android default download policy Min-Delay

with the Min-Delay, there is a reduction of 28% (light user), 22% (moderate user) and 11% (heavy user) on energy respectively when using the EDM. Compared with CLSA, an average energy savings of up to 18% and 10% are possible by EDM for light and moderate user patterns, and there is no obvious difference in heavy usage pattern. Overall, our approach outperforms CLSA with a better averaged reduction of 20.3% than Min-delay, and 9.3% than CLSA. Figure 8 compares the latency of each byte. In this scenario, adaptive schemes (CLSA and our approach) can reduce the average delay through dynamically adjusting the response time. Here, the EDM takes least time for each byte. Compared with the Min-Delay, the CLSA is able to reduce the latency by 7.5% on average, and it is as not good as our adaptive approach that gives a reduction of 15.3%. Figure 9 shows the performance of energy-delay trade off for EDM and CLSA. Both adaptive schemes achieve improvement on *performance* when compared to the Android default policy. Our approach gives improvement for 1.51x, 1.45x and 1.17x for light user, moderate user and heavy user respectively. The *performance* of CLSA is not as good as EDM, where still achieves 1.27x, 1.20x and 1.16x improvement when compared to the Min-Delay. The reason that the EDM outperforms the CLSA with a better energy-delay trade off is because of the MDP focuses on the global optimization.

The reward function of MDP includes the signal level and the queue backlog size, which considers the energy consumption and user download experience. On the contrary, CLSA is a local optimization algorithm, although it reduces energy and latency to a certain degree by delaying the download requests and concentrating responses, it does not consider the user patterns so that it degrades the performance in some situations. In the adverse network, the light users' few download requests can not incur the CLSA downloads data effectively, the requests will be delayed for a long time until a better network environment is available, or more download requests are proposed by users. However, the reward function of EDM has a significant impact on energy and delay which let the EDM performs more active under the adverse network. With the increasing of download probability, the difference for energy and latency is smaller between EDM and CLSA.

In summary, the experiment shows that the adaptive scheme significantly outperforms a fixed strategy. Furthermore, our approach outperforms CLSA with a better averaged reduction of 20.3% and 15.3% on energy and latency than Min-Delay. CLSA achieves an average improvement of 1.21x on *Performance*, which reflects the energy-delay trade off degree, by contrast, our approach performs better on both energy consumption and latency, and delivers up to 1.38x improvement, compared to the Min-Delay.

## 6. Related Work and Conclusion

Two preliminary pieces of work have inspired our own. Chenren Xu et al. [12] presented a global power management scheme for mobile devices. But they only build a simple MDP model for display and GPS, and it is a high cost project if they build a model for all the components of smartphone. Tang Lung Cheung et al. [13] proposed a WiFi radio power optimization strategy which focused on the energy consumption of interface states, but had not taken the signal strength into consideration which also has great impact on user experience. And Bo Zhao et al. [14] proposed an energy-aware approach for web browsing in 3G based smartphones by reconstructing the computation sequence for opening webpage and building the DOM tree.

Ning Ding [7] conducted the first measurement and modeling study of the impact of wireless signal strength on smartphone energy. Niranjana Balasubramanian et al. [15] described a strategy for exploiting residual energy tail from GPRS/EDGE and 3G radio communication, and then developed the TailEnd, a protocol that reduces energy consumption of common mobile applications by batching scheduling network requests internal to a single delay tolerant application. BreadCrumbs [16] examined WiFi connectivity changes over time and provided mobile connectivity forecasts by building a predictive mobility model. Jie Ren et al. [17] proposed a power aware download algorithm for energy-delay tradeoff using the Lyapunov optimization framework. Gaurav Pande [18] and Riikka Susitaival et al. [19] evaluated the performance of LTE network from the Internet access and Communication aspects. No work so far in the area has used MDP to optimize the energy consumption of data download on the real LTE mobile system. This work is the first to do so.

Other pieces of work, Pathak [20] presented the eprof, the first fine-grained energy profiler for smartphone apps. Eprof could find out the code bugs which cause the energy dissipation in apps and then fix them. Pathak also focused on the apps no-sleep bug that will let the I/O components stay awake for a long time until a force suspend, then he developed better programming language support to avoid no-sleep bugs at programming time [21]. Matt Calder and Mahesh K. Marina [22] described the pitfalls of scheduling applications that mostly run in the background and proposed a general batch scheduling algorithm which can save a significant energy by maximizing sleep time of the phone via overlapping the execution of recurrent applications.

In this paper, we propose a context-aware and low-power model to optimize download process on LTE-enabled smartphone using Markov Decision Process. The download decision takes several factors into account: Time, Remaining Battery, Signal Level and Request Queue Size. And Then, we study the LTE machine state, measure the tail time and power in different scenarios. Compared with the DCH state of 3G with longer tail time for keeping the connection, the “dormant” state in LTE consumed less power. To better understand the effect of signal level on user experience, we test the speed over varied signal level and find out that the speed obeys the gauss distribution. Finally, the experiments performed by 10 users with different download probabilities, and the result shows that the EDM achieves a better trade off between energy and delay, where achieves over 20% improvement over the Android default policy for energy consumption and *performance*. Meanwhile, it consistently outperforms a state-of-the-art power aware download algorithm (CLSA). In future work, we intend to explore further refinement to the energy-delay trade off, and also to exploit the heterogenous architecture to perform optimizations.

## Acknowledgements

This work is supported by the National Sci-Tech Support Plan Project under Grand No.2013BAK01B02; The National Natural Science Foundation of China under Grand No.61373176; The Science and Technology Industrial Research Project of Shaanxi Province under Grand No.2014K05-42.

## References

- [1] T. Pering, Y. Agarwal, R. Gupta, and R. Want, “Coolspots: reducing the power consumption of wireless mobile devices with multiple radio interfaces,” *Proc. of the 4th international conference on Mobile systems, applications and services*, pp.220–232, 2006.
- [2] J. Lee, K. Lee, Y. Kim, and S. Chong, “PhonePool: On energy-efficient mobile network collaboration with provider aggregation,” *Proc. 11th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, IEEE, pp.564–572, 2014.
- [3] A. Carroll and G. Heiser, “An Analysis of Power Consumption in a Smartphone,” *Proc. 2010 USENIX Conf. on USENIX annual technical conference*, pp.21–34, 2010.
- [4] “Mobile Exceeds PC Internet Usage for First Time in History,” <https://searchenginewatch.com/sew/opinion/2353616/mobile-now-exceeds-pc-the-biggest-shift-since-the-internet-began>, 2014.
- [5] “Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2015 White Paper,” <http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html>, 2016.
- [6] “Ericsson Mobility Report,” <http://www.ericsson.com/mobility-report>, 2015.
- [7] N. Ding, D. Wagner, X. Chen, Y.C. Hu, and A. Rice, “Characterizing and modeling the impact of wireless signal strength on smartphone battery drain,” *ACM SIGMETRICS Performance Evaluation Review*, vol.41, no.1, pp.29–40, 2013.
- [8] H. Lu, J. Yang, Z. Liu, N.D. Lane, T. Choudhury, and A.T. Campbell, “The Jigsaw continuous sensing engine for mobile phone applications,” in *Proc. of 8th ACM Conf. on Embedded Networked Sensor Systems ACM*, pp.71–84, New York, 2010.
- [9] “3GPP LTE,” <http://www.3gpp.org/LTE>
- [10] R.A. Howard, “Dynamic programming and markov processes,” *Mathematical Gazette*, 1960.
- [11] Matlab software, <http://cn.mathworks.com/>.
- [12] C. Xu, V. Srinivasan, J. Yang, Y. Hirase, E.M. Tapia, and Y. Zhang, “Boe: Context-aware Global Power Management for Mobile Devices Balancing Battery Outage and User Experience,” in *Proc. of 11th International Conf. on Mobile Ad Hoc and Sensor Systems (MASS)*, Washington, pp.300–308, 2014.
- [13] T.L. Cheung, K. Okamoto, F. Maker III, X. Liu, and V. Akella, “Markov decision process (MDP) framework for optimizing software on mobile phones,” in *Proc. of 7th ACM international conference on Embedded software*, ACM, New York, pp.11–20, 2009.
- [14] B. Zhao, W. Hu, Q. Zheng, and G. Cao, “Energy-Aware Web Browsing on Smartphones,” *IEEE Trans. on Parallel and Distributed Systems*, vol.26, no.3, pp.761–774, 2015.
- [15] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani, “Energy consumption in mobile phones: a measurement study and implications for network applications,” in *Proc. of 9th ACM SIGCOMM conference on Internet measurement conference*, New York, pp.280–293, 2009.
- [16] A.J. Nicholson and B.D. Noble, “Breadcrumbs: forecasting mobile connectivity,” in *Proc. of 14th ACM international conference on Mobile computing and networking*, ACM, New York, pp.46–57, 2008.
- [17] J. Ren, L. Gao, H. Wang, and Y. Chen, “A Method of Power Aware

Large Data Download on Smartphone,” *IEICE TRANSACTIONS on Information and Systems*, vol.E98-D, no.1, pp.131–139, 2015.

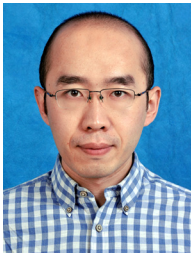
- [18] G. Pande, “Performance Evaluation of Video Communications over 4G Network,” *Advances in Intelligent Systems & Computing*, vol.243, pp.797–809, 2014.
- [19] R. Susitaival, H. Wiemann, J. Östergaard, and A. Larmo, “Internet Access Performance in LTE TDD,” in *Proc. of 7th IEEE Vehicular Technology Conference*, IEEE, Taipei, pp.1–5, 2010.
- [20] A. Pathak, Y.C. Hu, and M. Zhang, “Where is the energy spent inside my app?: Fine Grained Energy Accounting on Smartphones with Eprof,” in *Proc. of 7th ACM European Conf. on Computer System*, New York, pp.29–42, 2012.
- [21] A. Pathak, A. Jindal, Y.C. Hu, and S.P. Midkiff, “What is keeping my phone awake?: Characterizing and Detecting No-Sleep Energy Bugs in Smartphone Apps,” in *Proc. of 10th international Conf. on MobiSys*, New York, pp.267–280, 2012.
- [22] M. Calder and M.K. Marina, “Batch Scheduling of Recurrent Applications for Energy Savings on Mobile Phones,” in *Proc. of 7th Annual IEEE Communications Society Conference on Sensor Mesh and Ad Hoc Communications and Networks (SECON)*, IEEE, pp.1–3, 2010.



**QuanLi Gao** is currently a PhD candidate in the department of Computer Application Technology at Northwest University, Xi’an, China. His research interests are in areas of Recommender system, Artificial Intelligence.



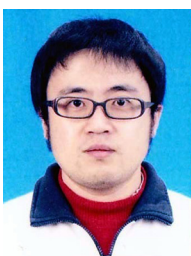
**ZheWen Zhang** is currently a MS candidate in the department of Computer Science at Northwest University, Xi’an, China. Her research interests are in areas of embedded system, Machine Learning.



**Jie Ren** is currently a PhD candidate in the department of Computer Science at Northwest University, Xi’an, China. He received his MS degree in Computer Science from Northwest University, in 2014. His research interests are in areas of heterogeneous system, mobile web browser and mobile computing.



**Ling Gao** is currently a professor of computer science at Northwest University, Xi’an, China. He received his BS degree in Computer Science from Hunan University and his MS degree in Computer Science from Northwest University, in 1985 and 1988 respectively. In 2005, he received PhD degree in Computer Science from Xi’an Jiaotong University, Xi’an, China. His research includes Network Security and Management, Embedded Internet Service. Prof. Gao is the director of China Higher Educational Information Academy, vice chairman of China Computer Federation Network and Data Communications Technical Committee, member of IEEE, CAET (China Association for Educational Technology) director.



**Hai Wang** is currently an associate professor of computer science at Northwest University, Xi’an, China. He received BS, MS, PhD degree in Computer Science from Xi’an Jiaotong University, in 2000, 2004 and 2010 respectively. His research interests are in areas of service computing, semantic Web, and intelligent information system.