PAPER

A Method of Power Aware Large Data Download on Smartphone

Jie REN[†], ***Member***, **Ling GAO**[†a)], **Hai WANG**[†], ***and*** **Yan CHEN**[†], ***Nonmembers***

SUMMARY The endurance time of smartphone still suffer from the limited battery capacity, and smartphone apps will increase the burden of the battery if they download large data over slow network. So how to manage the download tasks is an important work. To this end we propose a smartphone download strategy with low energy consumption which called CLSA (Concentrated Download and Low Power and Stable Link Selection Algorithm). The CLSA is intended to reduce the overhead of large data downloads by appropriate delay for the smartphone, and it based on three major factors: the current network situation, the length of download requests' queue and the local information of smartphone. We evaluate the CLSA using a music player implementation on ZTE V880 smartphone running the Android operation system, and compare it with the other two general download strategies, Minimum Delay and WiFi Only. Experiments show that our download algorithm can achieve a better trade-off between energy and delay than the other two.

key words: *smartphone, download strategy, CLSA, energy-delay trade off*

1. Introduction

Nowadays, the market share for the battery-powered portable equipment keeps increasing [1], [2], and the app market will explode exponentially to $38 billion industry by 2015 [3]. Despite the incredible market penetration of smartphones and exponential growth of the app market, their utility has been and will remain severely limited by the battery life.

For example, multimedia applications such as video playing and gaming are very much resource intensive in terms of processing and data transfer rates. Consequently, they consume much energy and drain smartphone battery very quickly [4]. An investigation has proved the feasibility of Multimedia Cloud Computing to provide the Energy-as-a-Service (EaaS), so that can reduce the local processing [5]. But the overhead of large data transfer has not been solved yet. Some researchers have found that the communication subsystem contributes much to the energy consumption in most embedded system [6], [7], and there are many apps exchange large data by WNI (Wireless Network Interface) of smartphone. At present, the general download strategy of mobiles is downloading data from web server to local (or upload data from local to web server) as soon as the download request arrives. But the immediate response may consume a lot of energy and reduce the battery duration over poor network. This paper explores a robust method to reduce the overhead of smartphone in three aspects. First, modern smartphones have multiple wireless interfaces (3G and WiFi) for data transfer, and they also have different data rates and transmission power [8], [9]. Second, the user usually does not consider the network environment and make download requests depending on their requirements which maybe discontinuous and scattered. Third, the download tasks can be delayed because of user's delay tolerance. This paper describes a Concentrate Download and Low Power and Stable Link Selection Algorithm (CLSA) based on the Lyapunov optimization framework. The algorithm decides which WNI to use and whether to delay the download tasks after analyzing the local information and network state. The CLSA was validated by a music player in the campus of Northwest University, and the result shows that our algorithm could achieve the goal of reducing energy consumption within the user's delay tolerance, a good trade-off relationship between energy and delay.

2. The Study of Centralized Tasks Strategy

This paper proposes the download strategy that originally from traffic lights at crossroad, green light go, red light stop. Such an orderly and concentrated management can control people and cars effectively. For example, in order to reduce the interrupt times of CPU and prolong the idle state of it [10], the wake-up source needed to be managed to avoid scattered waking. We adopt a controllable and concentrated waking-up strategy to minimize the wakeup times, delay the wakeup when it's possible, so as to maintain the system in low power state. Our download algorithm can use this method to defer the download requests when the smartphone with poor network and reduce the times of WNI statues switching between idle and working, as shown in Fig. 1.

We apply the concentrated strategy to the PCM (Pro-

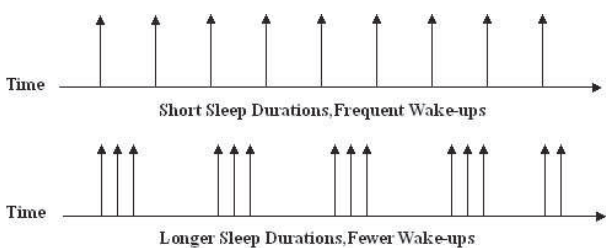

Fig. 1 Concentrated waking-up strategy.

Manuscript received June 4, 2014.
Manuscript revised August 25, 2014.
Manuscript publicized October 15, 2014.

[†]The authors are with the Department of Computer Science, Northwest University, Xi'an, 710000, China.
a) E-mail: lg@nwu.edu.cn

DOI: 10.1587/transinf.2014EDP7178

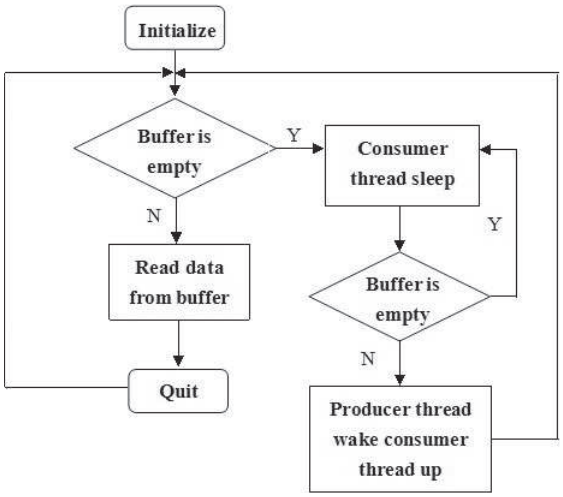

Fig. 2 Original consumer model.

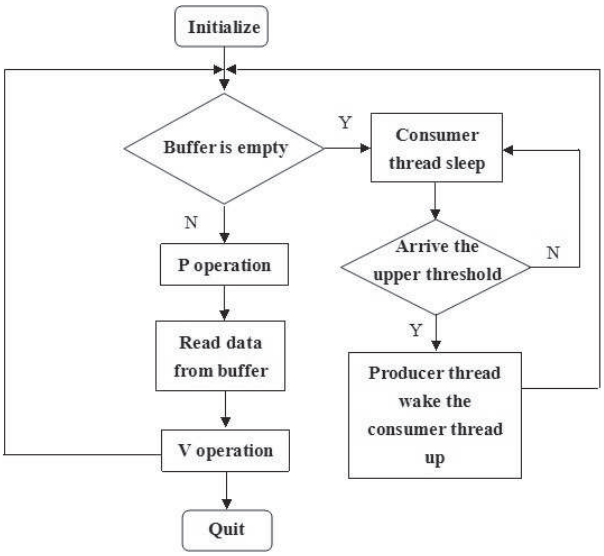

Fig. 3 Optimal consumer model.

ducer/Consumer Model). In the original PCM, two threads share a public buffer zone. The producer always put the data into the buffer and the consumer process the data. There are three threads, the main thread allocate and provide a series of system resources such as buffer zones and physical memory, the producer thread write data and the consumer thread read data.

Further control those threads which have a collaborative relationship, we optimize the original PCM, and take the consumer (offer download service) as the task that could be delayed to process. In the original PCM, the consumer will be waked up to process date even though there is only one data in buffer zone, resulting in much task switching, as shown in Fig. 2. We set a upper threshold to control the inappropriate wakeup behaviors, thus the consumer will be waken up to process data only when the number of data reach the threshold. Figure 3 shows the optimal consumer model that can reduce some potential wake-up behaviors, so as to reduce the waste in the system and achieve low energy consumption. But at the same time, it will create some new problems. First, if the smartphone in the good network environmet, only a small amount of download requests does not reach the upper threshold, thus the consumer will not provide download service. Second, if we set a small threshold and the network environment is poor, so that the number of download requests will reach the upper threshold easily as well as consume much energy. In order to solve these problems, we propose the CLSA download strategy in which the threshold is a variable not a constant value.

3. A Simple Example

In order to illustrate the energy-optimal download strategy, we consider the following example with two wireless network interfaces, WiFi and 3G, to download data. The system operates in slotted time, and the network state of every slot is taken into consideration.

$U[t]$ denotes the length of current download request queue, and no power should be allocated for downloading data when $U[t] = 0$. We must decide whether to allocate power on the current slot or wait for a more energy-efficient future network state. In this example, we set WiFi transmit power to 1 watt and 3G transmit power to 1.5 watt. $S[t]$ denotes radio network state is among 'Good', 'Medium' and 'Bad':

$$P_{wifi}[t] = 1, P_{3G}[t] = 1.5, S_{wifi}[t], S_{3G}[t] \in \{G, M, B\}$$

And we set $\mu[t]$ as the download rate, given by:

$$u[0, S_{All}] = 0 \text{ units/slot for all } S_{wifi}(t), \quad S_{3G}(t) \in \{G, M, B\}$$
$$u_{wifi}[1, G] = 3, \quad u_{wifi}[1, M] = 2, \quad u_{wifi}[1, B] = 1$$
$$u_{3G}[1.5, G] = 3, \quad u_{3G}[1.5, M] = 2, \quad u_{3G}[1.5, B] = 1$$

The radio network can download 3 units of data with 'Good' state, 2 units with 'Medium', 1 units with 'Bad'. There is no difference between WiFi and 3G.

Let $A[t]$ represent the number of newcome download requests during slot t. Queueing dynamics proceed according to the equation:

$$U[t + 1] = U[t] - u\{P_{All}[t], S_{All}[t]\} + A[t]$$

We assume $A[t]$ and $S[t]$ for the first 9 timeslots $t \in \{0, 1, \ldots, 8\}$ are given in Table 1, and the largest rate backlog product $U[t]u[t]$ as the policy one (p1) of WNI selection. This policy can be shown to stabilize the system whenever possible [11], [12], although it is not necessarily energy-efficient. We can get that the queue backlog is 0 at the beginning slot, so p1 will not select any WNI. Because $A[0] = 4$, so the $U[1] = 4$ at slot 1, and wifi or 3G means the selected interface at present slot. The WNI states at slot 1 are $\{S_{wifi}[t], S_{3G}[t]\} = \{G, M\}$, so the product of $U[1]u_{wifi}[1, G] = 12$, $U[1]u_{3g}[1.5, M] = 8$ the Max $U[t]u[t]$ policy select WiFi to download data, and so on for other

Table 1 Example.

t	0	1	2	3	4	5	6	7	8
$A[t]$	4	0	5	0	3	0	2	1	0
$S_{wifi}[t]$	G	G	B	M	G	B	B	G	G
$S_{3G}[t]$	M	M	M	G	B	M	M	M	M
$P1\_U[t]$	0	4(wifi)	1(3G)	4(3G)	1(wifi)	1(3G)	0	2(wifi)	0
$P2\_U[t]$	0	4(wifi)	1	6(3G)	3(wifi)	3	3	5(wifi)	3(wifi)

slots. According to the Table 1, almost every slot needs to select a WNI to download data except slot 0 and 6, and the time average power consumption:

$$P_{av}^{1} = \frac{1\times3+1.5\times3}{9} = 0.83 \text{ watt}$$

Then we consider a better policy two (p2) that downloads data with good WNI state, so the time average power consumption:

$$P_{av}^{2} = \frac{1\times4+1.5}{9} = 0.61 \text{ watt}$$

Although the finite time cannot show the full advantage of the better policy very well, the example still illustrates that the energy can be saved by appropriate strategy. In the next section, we focus on the stability of the queue backlog, and then we develop a more intelligent download strategy.

4. Lyapunov Optimization Framework

Lyapunov optimization framework [13] is an architecture that guarantees the stability of all queues in the network and optimizes other objects (Power, Fairness and Throughput).

4.1 Stable Queue

First, we need to give the concept of stable queue. Let $U[t]$ denote the queue backlog at beginning of timeslot t, $\mu[t]$ is the download speed and $A[t]$ is the app download tasks' length during timeslot t. The condition for the stable network is the increased data less than the transferred data during slot t ($A[t] \le u[t]$) [14]. Over time, the queue backlog evolves as follows:

$$U[t+1] = U[t] - u[t] + A[t] \tag{1}$$

The queue is stable if:

$$\overline{U} = \lim_{t\to\infty} \sup \frac{1}{t} \sum_{\tau=0}^{t} E\{U[\tau]\} < \infty \tag{2}$$

Under the finite length of stable queue, the time average transmission power defined as:

$$\overline{P} = \lim_{t\to\infty} \sup \frac{1}{t} \sum_{\tau=0}^{t} E\{P[\tau]|U[\tau]\} < \infty \tag{3}$$

4.2 Lyapunov Drift for Queueing Networks

Lyapunov drift is a well-known technique used to analyze the stability of queue. It involves defining a nonnegative, scalar function, called a Lyapunov function whose value depends on the queue backlog $U[t]$. If there are N waiting queues in the network, and define the vector of queue backlogs at time t by:

$$\vec{U[t]} = (U_1[t], U_2[t], \ldots, U_N[t]) \tag{4}$$

For each slot t, define Lyapunov function $L[t]$ as the sum of the squares of the current queue backlogs (divided by 2 for convenience later):

$$L[t] \triangleq \frac{1}{2} \sum_{i=1}^{N} U_i[t]^2 \tag{5}$$

This function is a scalar measure of the total queue backlog in the network. It is called quadratic Lyapunov function on the queue state. Define the Lyapunov drift as the change in this function from one slot to the next:

$$\Delta(t) = L[t+1] - L[t] \tag{6}$$

4.3 Bounding the Lyapunov Drift

From (1) we can suppose the queue backlog i change over time according to the following equation:

$$U_i[t+1] = \max\{U_i[t] + A_i[t] - u_i[t], 0\} \tag{7}$$

This equation can be used to compute a bound on the Lyapunov drift for any slot t:

$$\begin{aligned} U_i[t+1]^2 &= \max\{U_i[t] + A_i[t] - u_i[t], 0\}^2 \\ &\le \{U_i[t] + A_i[t] - u_i[t]\}^2 \end{aligned} \tag{8}$$

Rearranging this inequality, summing over all i and dividing by 2 leads to:

$$\Delta(t) \le B[t] + \sum_{i=1}^{N} U_i[t]\{A_i[t] - u_i[t]\} \tag{9}$$

Where $B[t]$ is defined:

$$B[t] = \frac{1}{2} \sum_{i=1}^{N} \{A_i[t]^2 + u_i[t]^2 - 2A_i[t]u_i[t]\} \tag{10}$$

Suppose the second moments of arrivals and service in each queue are bounded, so that there is a finite constant $B > 0$ such that for all t and all possible queue vectors $U[t]$ the following property holds:

$$E\{B[t]|U[t]\} \le B \tag{11}$$

Taking conditional expectations of (9) leads to the following bound on the conditional expected Lyapunov drift:

$$E\{\Delta(t)|U[t]\} \le B + \sum_{i=1}^{N} U_i[t]E\{A_i[t] - u_i[t]|U[t]\} \quad (12)$$

4.4 A Basic Lyapunov Drift Theorem

In most cases, the network can be controlled so that the difference between arrivals and service at each queue satisfies the following property for some real number [15], [16]:

$$E\{u_i[t] - A_i[t]|U[t]\} \ge \varepsilon(\varepsilon > 0) \quad (13)$$

Bring (13) into (12) we can get:

$$E\{\Delta(t)|U[t]\} \le B - \varepsilon \sum_{i=1}^{N} U_i[t] \quad (14)$$

Summing the above expression over $\tau \in \{0, 1, \ldots, t-1\}$ and using the law of telescoping sums gives:

$$E\{L[t]\} - E\{L[0]\} \le Bt - \varepsilon \sum_{\tau}^{t-1} \sum_{i=1}^{N} E\{U_i[\tau]\} \quad (15)$$

Then for all slots $t > 0$ the time average queue size in the network satisfies:

$$\frac{1}{t}\sum_{\tau}^{t-1}\sum_{i=1}^{N} E[U_i(\tau)]\frac{1}{t}\sum_{\tau}^{t-1}\sum_{i=1}^{N} E[U_i(\tau)] \le \frac{B}{\varepsilon} + \frac{E[L(0)]}{\varepsilon t} \quad (16)$$

Under the condition of $E\{u_i[t] - A_i[t]|U[t]\} \ge \varepsilon(\varepsilon > 0)$, it can be proved that the average length of queue backlog of all nodes in the network cell is bounded, so that the network is stable. We can get the same result on a single node by analogy analysis.

5. Concentrate Download and Low Power and Stable Link Selection Algorithm

The core of CLSA is the optimized PCM, and focus on a single node, smartphone. The goal is to stabilize the queueing network while minimizing the average transmits power.

5.1 CLSA

The link selection algorithm:

$$\tilde{l}[t] = \max(U[t] \times E\{u[t]|l, S_l[t]\} - W \times P_l[t]) \quad (17)$$

It chooses a link $\tilde{l}[t]$ to download data during timeslot t by calculate the $\max\{\tilde{l}[t]\}(\tilde{l}[t] > 0)$, the current request queue backlog is $U[t]$. $E\{u[t]|l, S_l[t]\}$ is the estimation of download rate which can be achieved on link l and the current channel condition $S_l[t]$. $P_l[t]$ is the transmite power, W is a weight parameter that used for energy and delay tradeoff.

The CLSA is an extended branch of M.J. Neely's algorithm [17], it will not select any link to download data when $\tilde{l}[t] \le 0$. The formula (17) derived as follows.

Let quadratic Lyapunov function $L[t]$ as the sum of the squares of the current download request queue backlogs:

$$L\{U[t]\} \triangleq \frac{1}{2}\{U[t]\}^2 \quad (18)$$

Lyapunov drift:

$$\begin{aligned}\Delta(U[t]) &\triangleq E\{L\{U[t+1]\} - L\{U[t]\}|U[t]\}\\ &= \frac{1}{2}U[t+1]^2 - \frac{1}{2}U[t]^2\\ &= \frac{1}{2}\{U[t] - u[t] + A[t]\}^2 - \frac{1}{2}U[t]^2\\ &= \frac{1}{2}\{u[t]^2 + A[t]^2 - 2u[t]A[t]\} - U[t]\{u[t] - A[t]\}\\ &= \frac{1}{2}(u[t] - A[t])^2 - U[t]\{u[t] - A[t]\}\\ &= \frac{1}{2}(u[t] + A[t])^2 - U[t]\{u[t] - A[t]\} - 2U[t]A[t]\\ &\le \frac{1}{2}(u[t] + A[t])^2 - U[t]\{u[t] - A[t]\} \quad (19)\end{aligned}$$

Take conditional expectations given $U[t]$:

$$\begin{aligned}\Delta(U[t]) \le{}& \frac{1}{2}E\{(u[t] + A[t])^2|U[t]\}\\ &-U[t]E\{u[t] - A[t]|U[t]\} \quad (20)\end{aligned}$$

Let $B[t]$ equal to:

$$C[t] \triangleq \frac{1}{2}\{E\{(A[t] + u[t])^2|U[t]\}\} \quad (21)$$

And then bring (21) into (20), we can obtain:

$$\Delta(U[t]) \le C[t] - U[t]E\{u[t]|U[t]\} + U[t]A[t] \quad (22)$$

Now we take the minimum power consumption objective into consideration, and add a weighted cost to achieve the goal of low power and the stability of request queue:

$$\begin{aligned}&\Delta(U[t]) + WE\{P[t]|U[t]\}\\ &\le C[t] - U[t]E\{u[t]|U[t]\} + U[t]A[t] + WE\{P[t]|U[t]\}\\ &= C[t] - U[t]E\{E\{u[t]|U[t], l, S_l[t]\}\}\\ &+U[t]A[t] + WE\{P[t]|U[t]\}\\ &= C[t] - E\{(U[t]E\{u[t]|l, S_l[t]\}\\ &-WP[t])|U[t]\} + U[t]A[t] \quad (23)\end{aligned}$$

We assume the data arrival speed $A[t]$, and the data service speed $\mu[t]$ have finite variance, exist constant to satisfy $A[t] < A$ and $E[u[t]|U[t]] < u$, so we can get:

$$C[t] < C \quad \forall t, C = (A + u)^2 \quad (24)$$

From (23) we can know that minimizing the right side hand of formula will guarantee the stability of request queue with minimum power consumption. However, we cannot

control the application data arrival process, and hence cannot do much about the term $U[t]A[t]$. In order to minimize the right hand side of (23), CLSA maximizes the negative term on the right hand side of (23), so we get the control decision in (17).

From (17), we can get that the CLSA will stop the download service if the smartphone under poor network condition. Corresponding to the consumer with idle state in the PCM, and it will be waken up until the number of data reach the upper threshold which produced by the user. The threshold is not a constant value in the CLSA, it either small in good network environment or large when the download speed is low. The key of the threshold is the parameter W, it control power and delay trade-off in (17). The energy consumption will not be taken into consideration if W is small enough, and the algorithm will offer the minimum delay download service. On the contrary, the algorithm will save much energy if the W is big, it will delay the download service as long as possible until $U[t] \times E\{u[t]|U[t]\}$ is bigger than $W \times P_l[t]$. So a good W is important for the algorithm to control power and delay trade-off. We can set the W as a configurable for the users, so they can set it for different goals, but the users do not need nor want to know the internal theory. We will provide a fixed W which can achieve a good performance by a serious of simulations in Sect. 6.3. The concentrated download strategy accomplish three goals. First, the CLSA will reduce the energy dissipation which caused by switching the state of WNI between working and idle frequently. Second, deferring the download request until the smartphone with the good network, so that will prevent downloading data with poor speed which consume much energy. Third, dynamic threshold guarantees the stability of request queue, avoiding the infinitely increasing of queue backlogs.

5.2 Average Queue Backlog and Power Consumption

M.J. Neely has defined the average length of queue backlog and the minimum average power consumption in a network cell. We defined the same thing of a single node with parameter $W > 0$:

$$\overline{U} = \limsup_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t} E\{U[\tau]\} < C + WP^*/\varepsilon \tag{25}$$

$$\overline{P} = \limsup_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t} E\{U[\tau]\} < P^* + \frac{C}{W} \tag{26}$$

The derivation process of the above formulas is similar to [17], but (25) (26) are only for single node. P^* is a theoretical lower bound on the time average power consumption, $\varepsilon > 0$ is a constant that means the distance between arrival pattern and the capacity region boundary. We can let the average power consumption close to theoretical lower bound P^* by increasing the value of W, but the cost of it is the linear increase in length of request queue, prolonging the wait time for users. On the contrary, we can also reduce the latency of download service by decreasing the value of W. The relationship between energy and delay satisfies $[O(1/W), O(W)]$.

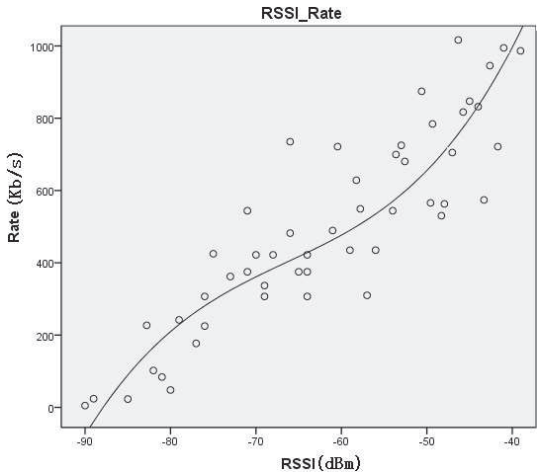

Fig. 4 The relationship between RSSI and download speed.

5.3 Estimation of Download Rate

At present, there is no general strategy nor related software can predict the download speed of smartphone accurately with few energy. This paper proposes a forecast method by the Received Signal Strength Indicator (RSSI) of different APs. We store the AP's SSID, current RSSI and download speed as one item and refresh it when a new download service comes and calculate the average speed. The result shows that the *error* < 8%, satisfying the experiment requirement. Figure 4 shows the relationship between RSSI and download rate, and they are almost proportional.

6. Validation

We develop a music player based on open source MP3 decoder Libmad on the Android 2.2, and implement the CLSA in this app. The experiment is conducted on ZTE V880 and table 2 shows the related parameters. The Agilent 66332A provide DC power supply, two Agilent 334410A measure voltage and current respectively, as shown in Fig. 5.

6.1 Experiment Object

The experiment conducted in Northwest University where equipped many WiFi hotspots, and collect data around four locations, Dormitory, Dining Hall, Library and Laboratory, as shown in Fig. 6. Based on our measurement of V880, we get the download power of 3G and WiFi are 0.925W and 1.387W, respectively, as shown in Table 3. Table 4 lists WiFi signal strength.

6.2 Android WiFi Linking Strategy

We conducted three tests to verify the Android embedded linking strategy.

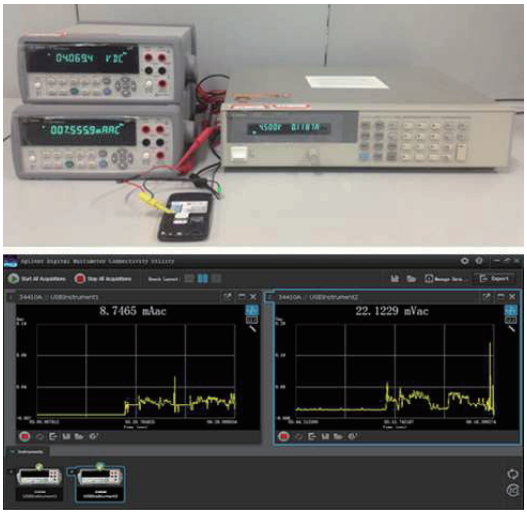

Fig. 5 Energy measurement tools.

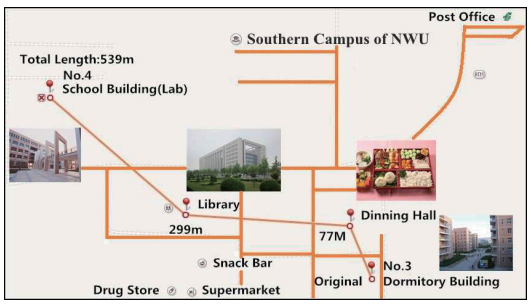

Fig. 6 Southern campus of NWU.

Table 2 ZTE V880.

	ZTE V880
CPU	600MHZ
RAM	256M
ROM	512M
Micro-SD Card	2GB
Network	GSM+WCDMA+Wifi
OS	Android 2.2
Battery	Li-ion 1250mAh

Table 3 Power of radio network.

Platform	WiFi	3G
ZTE V880	0.925W	1.387W

Table 4 Signal strength of WiFi.

	Dorm	Lib	Lab	Din
Hotspot	Laptop	Router	Router	None
Strength	Weak	Normal	Strong	None
Distance	5–10m	15–20m	30–40m	0m

Test one:
1) Open WiFi interface.
2) Connect to WiFi hotspot AP1, and then connect to AP2.
3) Close WiFi interface, and then open WiFi interface ($AP1\_RSSI > 0$, $AP2\_RSSI > 0$).
The WiFi linking strategy will select AP2, the lately hotspot.

Test two:
1) Open WiFi interface.
2) Connect to WiFi hotspot AP1, and then connect to AP2.
3) Close WiFi interface, and then open WiFi interface ($AP1\_RSSI > 0$, $AP2\_RSSI == 0$).
The WiFi linking strategy will select AP1, the hotspot with WiFi signal.

Test three:
1) Open WiFi interface.
2) Connect to WiFi hotspot AP1, and connect to AP2, then AP3.
3) Close WiFi interface, and then open WiFi interface ($AP1\_RSSI == 3$, $AP2\_RSSI == 2$, $AP3\_RSSI == 0$).
The WiFi linking strategy will select AP2, the hotspot with WiFi signal and lately connected, not select the hotspots with strongest signal. This problem still exists till the latest Android version 4.4.

Our download strategy can solve this problem, select the hotspot with strongest signal by RSSI and download speed, resulting in low energy consumption.

6.3 Experiment and Analysis

The paper compares CLSA against other two general algorithms, Minimum Delay (always make an immediate response) and WiFi Only (only use WiFi to transmit data). We also measured the overhead of different strategies over HTTP and FTP protocols via 3G and WiFi interfaces. We download music at four locations, suppose the arrival process $A[t]$ obeying the uniform distribution, and then hang around 50 minutes at each site, record the experiment data. Let $S_i(i \in \{1 \ldots N\})$, $E_i(i \in \{1 \ldots N\})$ and $D_i(i \in \{1 \ldots N\})$ denote the length, overhead and delay of song i, then calculate the average energy consumption and delay per byte by (27).

$$\overline{E} = \frac{\sum_{i=1}^{N} E_i}{\sum_{i=1}^{N} S_i}, \overline{D} = \frac{\sum_{i=1}^{N} D_i}{\sum_{i=1}^{N} S_i} \tag{27}$$

We use the product of $\overline{E}(J)$ and $\overline{D}(s)$ to evaluate the performance of different W. The evaluation based on two factors, energy and delay, the higher the product, the worse the performance. If there is no WiFi hotspots and the user in a relatively closed environment, the W should be higher to delay the downloads longer because of poor network environment. So we can make a settable W for users to deal with different environments, but they do not have specialized knowledge and do not need to know how it works. Maybe the battery will be drained faster after the user set a new W, therefore we set W as a constant in this paper. We conducted a serious of simulations to choose a good W in the range from 10 to 50. And the result shows that when $W = 22$, our algorithm can reach the best performance from Fig. 7

Table 5 $\overline{E}(J)$ and $\overline{D}$(s) of three different download strategy by FTP.

	Minimum Delay		WiFi Only		CLSA	
	$\overline{E}(J)\times10^{-6}$	$\overline{D}$(s)	$\overline{E}(J)\times10^{-6}$	$\overline{D}$(s)	$\overline{E}(J)\times10^{-6}$	$\overline{D}$(s)
Dorm	5.41	187	1.15	192	1.88	346
Lib	1.45	40	0.564	118	1.23	62
Lab	3.7	66	0.38	86	1.12	32
Din	1.4	45	0.578	4025	1.29	233
Average	2.99	84	0.668	1105	1.38	168

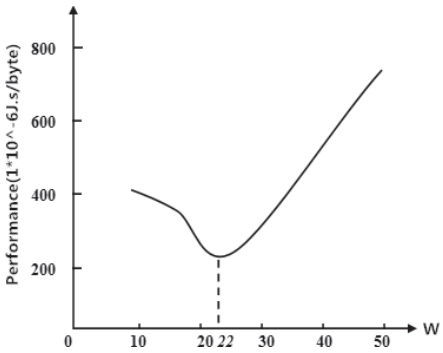

Fig. 7 CLSA performace with different W.

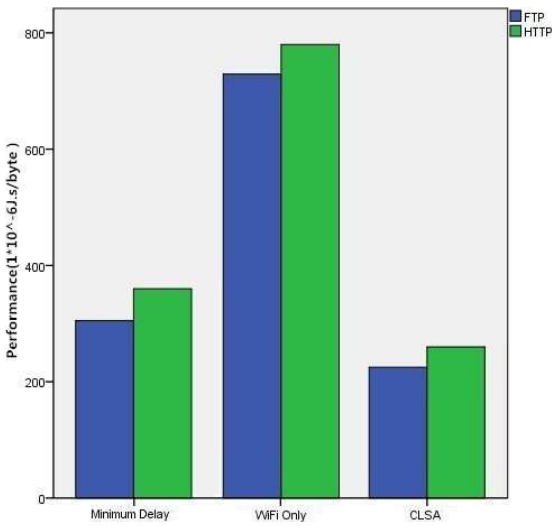

Fig. 8 Performance comparison between different strategies.

Table 5 shows the result after data processing over FTP. We know that the Minimum Delay with the minimum latency and consume the most energy per byte. The Minimum Delay will not take the network state into consideration and make an immediate response when the download request arrive, so it will cause high energy consumption if the smartphone in the poor network environment. WiFi only with the maximum latency and consume the least energy per byte. The aforementioned download power of WiFi interface is lower than 3G interface and the transmit rate is faster than 3G, so it consumes the least energy per byte. But the length of the request queue will increase infinitely when there is no WiFi signal around. The CLSA provides a good trade-off between energy and delay by formula (17). Under the poor network condition, CLSA will not select a link to download unless the request queue backlog is long enough, so that will save some energy and avoid infinite latency. For example, the average download speed of 3G is 20Kb/s and WiFi is 100Kb/s in dormitory. The smartphone cannot get WiFi signal when it is far away from the hotspot, then the CLSA will make a decision by detecting RSSI and local queue backlogs. If the queue backlog do not reach the upper threshold with slow speed, the CLSA will not offer download service to save energy, but it will offer download service when the length of queue is long enough, guaranteeing the stability of queue.

We compare the performance of different download strategies by the product of $\overline{E}(J)$ and $\overline{D}(s)$ in Fig. 8. WiFi Only owns minimum $\overline{E}(J)$, but it does not achieve the best performance because of the maximum $\overline{D}(s)$. Only if both $\overline{E}(J)$ and $\overline{D}(s)$ have better value, the download strategy can get good performance. CLSA neither reduce the most energy, nor have the lowest latency, but it owns the best comprehensive properties which consider the energy/delay trade off in the algorithm. From Fig. 8 we can see that the CLSA does not get great advantage compare with Minimum Delay, because we conduct our experiments in a relatively open environment with many WiFi hotspots, and the radio signal is strong in most cases. If the user always in a good network environment (strong radio signal, quickly network speed) the CLSA will not offer significant help at this time, but if there is no WiFi hotspots and the 3G signal is poor (weak radio signal), the Minimum Delay will show a worse performance, it will consume more energy to finish the download as soon as possible, and the CLSA can balance the energy and delay to get a good performance. The CLSA will perform better in a relatively poor network environment.

7. Related Work & Conclusion

A number of specialized energy saving techniques on mobiles have been proposed. Pathak [18] presented the eprof, the first fine-grained energy profiler for smartphone apps. Eprof can find out the code bugs that cause the energy dissipation in apps and then fix them. Abhinav P. also focused on the apps no-sleep bug [19] which 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 [20]. Other specialized energy saving techniques, e.g., for specific applications on mobile system [9], [21], for a specific protocol [8], [22], via offloading [23], [24], and via delaying communication [25].

This paper describes an energy/delay trade off ap-

proach for data download. We propose an optimal PCM model, and then we use Lyapunov optimization framework and optimal PCM model to develop the download algorithm CLSA for smartphone. The CLSA reduce unnecessary overhead and achieve energy-efficient to a certain degree, it can get great advantage in poor network condition. We implement our algorithm in the ZTE V880 smartphone running the Android operating system. Experiments show that our algorithm can get better performance than prior works. The low power download algorithm provides smartphone users with better usage experiences.

Acknowledgements

This work is supported by the Doctoral Scientific Fund Project of the Ministry of China under Grant No.20116101110016; The National Natural Science Foundational of China under Grand No.61373176; Nature Science Foundation of Shaanxi Province under Grand No.2012JQ8047; Shaanxi Province Department of Education Fund under Grand No.11JK1059; Major Science and Technology Project of Shaanxi Province under Grand No.2012ZKC05-2.

References

[1] Mobile Marketing Statistics 2012, http://snaphop.com/2012-mobile-marketing-statistics/, 2013.
[2] Smartphone sales overtake pcs for the first time, http://mashable.com /2012/02/03/smartphone-sales-overtake-pcs/, 2013.
[3] Mobile app internet recasts the software and services landscape, http://tinyurl.com/5s3hhx6, 2013.
[4] 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.
[5] M. Altamimi, R. Palit, K. Naik, and A. Nayak, "Energy-as-a-Service (EaaS): On the Efficacy of Multimedia Cloud Computing to Save Smartphone Energy," Proc. 2012 IEEE 5th International Conference on Cloud Computing (CLOUD), pp.764–771, Honolulu, 2012.
[6] M. Stemm and R.H. Katz, "Measuring and reducing energy consumption of network interfaces in hand-held devices," IEICE Trans. Commun., vol.E80-B, no.8, pp.1125–1131, Aug. 1997.
[7] V. Raghunathan, C. Schurgers, S. Park, and M.B. Srivastava, "Energy aware wireless microsensor networks," IEEE Signal Process. Mag., vol.19, no.2, pp.40–50, March 2002.
[8] Y. Agarwal, R. Chandra, A. Wolman, P. Bahl, K. Chin, and R. Gupta, "Wireless wakeups revisited: Energy management for VoIP over WiFi smartphones," Proc. 5th International Conf. on MobiSys, pp.179–191, New York, 2007.
[9] S. Kang, J. Lee, H. Jang, H. Lee, and Y. Lee, "SeeMon: Scalable and energy-efficient context monitoring framework for sensor-rich mobile environments," Proc. 6th International Conf. on MobiSys, pp.267–280, New York, 2008.
[10] J. Ren, L. Gao, H. Wang, and Y.J. Lu, "Research on low power application software in Linux System," Proc. 7th International Conf. on Computing and Convergence Technology, pp.949–954, Seoul, 2012.
[11] L. Tassiulas and A. Ephremides, "Dynamic server allocation to parallel queues with randomly varying connectivity," IEEE Trans. Inf. Theory, vol.39, no.2, pp.466–478, March 1993.
[12] M. Andrews, K. Kumaran, K. Ranmanan, A. Stolyar, P. Whitiing, and R. Vijayakumar, "Providing quality of service over a shared wireless link," IEEE Commun. Mag., vol.39, no.2, pp.150–154, Feb. 2001.
[13] L. Georgiadis, M.J. Neely, and L. Tassiulas, Resource Allocation and Cross-Layer Control in Wireless Networks, now publisher, Hanover, 2006.
[14] M.J. Neely, "Dynamic Power Allocation and Routing for Satellite and Wireless Networks with Time Varying Channels," PhD thesis, Massachusetts Institute of Technology, LIDS, 2003.
[15] M.J. Neely, Stochastic Network Optimization with Application to Communication and Queueing Systems, Morgan & Claypool, 2010.
[16] E. Leonardi, M. Mellia, F. Neri, and M. Ajmone Marsan, "Bounds on average delays and queue size averages and variances in input-queued cell-based switches," Proc. 20th IEEE INFOCOM, vol.2, pp.1095–1103, Anchorage, 2001.
[17] M.J. Neely, "Energy optimal control for time varying wireless networks," IEEE Trans. Inf. Theory, vol.52, pp.2915–2934, 2006.
[18] A. Pathak, Y.C. Hu, and M. Zhang, "Where is the energy spent inside my app?: Fine grained energy accounting on smartphones with Eprof," Proc. 7th ACM European Conf. on Computer System, pp.29–42, New York, 2012.
[19] A. Pathak, Y.C. Hu, and M. Zhang, "Bootstrapping energy debugging for smartphones: A first look at energy bugs in mobile devices," Proc. 10th ACM Workshop on Hotnets, no.5, New York, 2011.
[20] 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," Proc. 10th International Conf. on MobiSys, pp.267–280, New York, 2012.
[21] Y. Wang, J. Lin, M. Annavaram, Q.A. Jacobson, J. Hong, B. Krishnamachari, and N. Sadeh, "A framework of energy efficient mobile sensing for automatic user state recognition," Proc. 7th International Conf. on MobiSys, pp.179–192, New York, 2009.
[22] F. Qian, Z. Wang, A. Gerber, Z.M. Mao, S. Sen, and O. Spatscheck, "Profiling resource usage for mobile applications: A cross-layer approach," Proc. 9th international Conf. on MobiSys, pp.321–331, New York, 2011.
[23] E. Cuervo, B. Aruna, D. ki Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: Making smartphones last longer with code offload," Proc. 8th International Conf. on MobiSys, pp.49–32, New York, 2010.
[24] B.G. Chun and P. Maniatis, Augmented Smartphone Applications Through Clone Cloud Execution, in HotOs, 2009.
[25] C. Schurgers, O. Aberthorne, and M.B. Srivastava, "Modulation scaling for energy aware communication systems," Proceedings of the 2001 international symposium on Low power electronics and design, pp.96–99, ACM, 2001.

Jie Ren is currently with PhD candidate in the department of Computer Science at Northwest University, Xi'an, China. He received BS degree in Computer Science from Northwest University in 2014. His research interests are in areas of machine learning, low power of embedded system 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 Xi'an, 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 include Network Security and Management, Embedded Internet Service. Prof. Gao is 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 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.

Yan Chen is currently with MS candidate in the department of Computer Science at Northwest University, Xi'an, China. Her research interests are in areas of embedded system, data mining.