Smart Screen Management on Mobile Phones

Hossein Falaki
Center for Embedded Networked Sensing
Computer Science Department
University of California, Los Angeles
Email: falaki@cs.ucla.edu

***Paper type*: Research [$\surd$] Critical survey [ ]**

Main area(s): Mobile Computing, Dynamic Power Management, Context-Aware Computing

***Abstract*—Large and bright screens on today's mobile phones account for significant energy demand on phones' batteries [14]. In this paper we propose an algorithm that, given the energy profile of the screen, finds the optimal schedule to minimize screen energy dissipation when the phone is idle. We profile the screen energy consumption of two popular smartphones, Nokia N95 and E71, through carefully designed micro-benchmarks. Our energy measurement results suggest that the default screen schedules on these phones are far from optimal - on average our algorithm performs 50% better than default. We also find that on the E71 not using the dim state of the screen and directly turning it off is more energy-efficient.**

We improve the performance of our screen scheduling algorithm by considering the history of each user's interaction with his/her phone. We study the interaction patterns of six volunteers with their smartphones. The results suggest that the distribution of the length of idle times for each user does not change over time. Therefore, the screen scheduler can learn this distribution during a learning phase and use it to improve screen management. We show that the probabilistic algorithm can further reduce the screen energy consumption by up to 60% compared to the deterministic algorithm.

***Index Terms*—Mobile Phones, Screen, Dynamic Power Management**

I. Introduction

New functionalities are being added to smartphones at an increasing rate. These include addition of new hardware capabilities, and introduction of new software features. These feature-rich phones are usually referred to as *smartphones*.

However, the increases in battery capacity have not matched increases in functionality. In fact, battery capacities have not been growing more than 10 percent every year, whereas the number of features and applications, and therefore power consumption demand, is growing at a significantly higher rate [14]. As a result, increasingly more powerful processors, multiple wireless interfaces and larger and brighter screens are making phone batteries last shorter than they used to.

The current technology situation makes power management a critical problem on smartphones. Most of the existing research on this topic treat smartphones and traditional cell phones as embedded systems with several components. There has been considerable work on reducing the energy consumption of each component and discovering when they are idle to power them down or turn them off if possible [6]. Dynamically turning off unused components is referred to as *dynamic power management (DPM)*.

We argue that dynamic power management for different components of mobile phones can be improved by considering the usage history of the phone. Mobile phones differ from embedded systems because human users directly interact with them. In this work, we demonstrates how dynamic power management can be improved by considering usage history of the phone user. We use the mobile phone screen and backlight as an example, and propose an energy management strategy that uses the history of user interactions as a priori to achieve better energy saving.

Current mobile phones turn off the screen backlight after a fixed interval that is configurable by the phone user. We show how the optimal value of this interval can be derived from the user's past history of interactions with the phone.

In Section II we show why scheduling the phone screen is not a trivial problem. We model the problem formally in Section III. Section IV details an algorithm that given the energy profile of the screen and the user's past usage history finds the optimal schedule for the backlight. To evaluate the performance of the proposed algorithms we conduct experiments that are presented in Section V. We evaluate the performance of this algorithm using our user traces and compare it with the existing power management strategy in Section VI. In Section VII we present the related work. Finally Section VIII concludes the paper and outlines possible future work.

II. The Screen Scheduling Problem

Large screens have become a de facto requirement for today's smartphones. Large and bright screens account for a significant portion of the energy demand on the phones' batteries [14]. To conserve energy mobile phones turn off their screens when they are idle. On all current smartphones a static timeout value is used to decide when to turn off the screen; most often the user can configure this timeout as part of the phone settings.

In this section we argue why finding the optimal value of the screen sleep timeout is not a trivial decision. First we will show why a universal predefined value is not suitable. Second, we demonstrate that expecting the user to find the right value for himself/herself is also not realistic.

Fig. 1. Picture of a Nokia N95 (left) and a Nokia E71 (right).

State	Description
ON	LCD and backlight are on
DIM	LCD is on, but the backlight is off
OFF	LCD and backlight are off

TABLE I
THREE STATES OF NOKIA SMARTPHONE SCREENS

We performed micro-benchmarks on two popular smartphones: Nokia N95 and E71 phones (Figure 1). The screens on these phones have three power states listed in Table I. Figure 2 plots the average power consumption of both smartphones measured 10 times using the Nokia Energy Profiler tool [5].

In Figure 2, one minute into the experiment the user touches the keyboard of the E71 and the phone turns on its screen and backlight as as a response. 15 seconds after the phone is idle, the backlight turns off (the phone enters the DIM state), and 30 seconds later the LCD is turned off too (the OFF state). The N95 experiment follows a similar scenario, with slight shifts in time.

The spikes in power consumption at times, when the screen state changes, indicate that each state transition incurs a fixed energy cost. We argue that, because of these fixed costs, finding the optimal timeout to change states is non-trivial. Consider a simple situation where a naïve user sets the timeout value to one second. [1] If on average the user *think time* is 1.5 second,s the backlight will turn off right before the next user interaction happens. The user interaction will turn on the screen again and so on. If this scenario happens too often, the phone will end up consuming more energy on the screen compared to the case where the backlight timeout is longer.

This simple example shows why the shorter timeout value is not necessarily the better choice in terms of energy consumption. It also demonstrates how user interaction patterns affect the optimal choice of screen timeout settings. In this paper we will study this problem quantitatively, but first we need a formal model of the problem.

[1]Interestingly, the *Nokia Green Phone* application urges users "set the backlight time out to as little as 1 second" to maximize power saving.

Current State	Run. cost	Next state	Trans. cost
ON	R_1	DIM	C_1
		OFF	C_2
DIM	R_2	ON	C_3
		OFF	C_4
OFF	R_3	ON	C_5
		DIM	C_6

TABLE II
LIST OF POSSIBLE COSTS ASSOCIATED WITH THE SCREEN

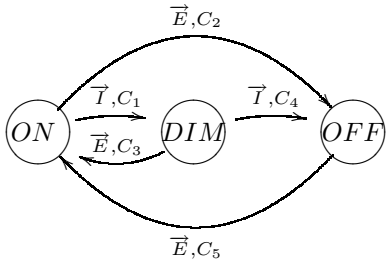


Fig. 3. All possible state transitions in the screen. Characters with arrows indicate the cause of the transition. E represents an event, I and represents being idle.

III. SYSTEM MODELING

In this section we present a formal model of the screen power management problem which is partially based on the theoretical modeling introduced by Augustine, Irani, and Swamy [1]. We simplify that model and use it in a real setting.

Table II summarizes the running and fixed costs associated with each screen state. We assume that the screen power consumption is fixed in each state, and there is a fixed cost associated with each state transition. We will verify these assumptions in Section V.

From the modeling perspective it does not matter whether a fixed cost is incurred when entering a state or when leaving it. To correctly associate fixed costs to state transitions, we need to identify all legitimate state transitions, because in a real system arbitrary state transitions are not possible. Figure 3 demonstrates all possible state transitions of the screen. There are two characters on each transition arrow. The first character indicates the cause of the transition, that could be an idle period ($\overrightarrow{I}$) or an event ($\overrightarrow{E}$). The second character is the fixed cost associated with the transition.

Based on our observations we know that the phone can serve the user only when the screen is ON. Therefore any event (e.g., user interaction) will turn the screen ON. On the other hand, the screen has two possible power saving states. When the screen is idle, it can either directly turn OFF, or turn DIM and then OFF. These two possible scenarios as illustrated in Figure 4. We will refer to the first scenario in this figure, that takes advantage of DIM, as the *three-state scenario*. We will call the alternate scenario the *two-state scenario*.

Figure 5 plots the energy consumed by the screen in each of

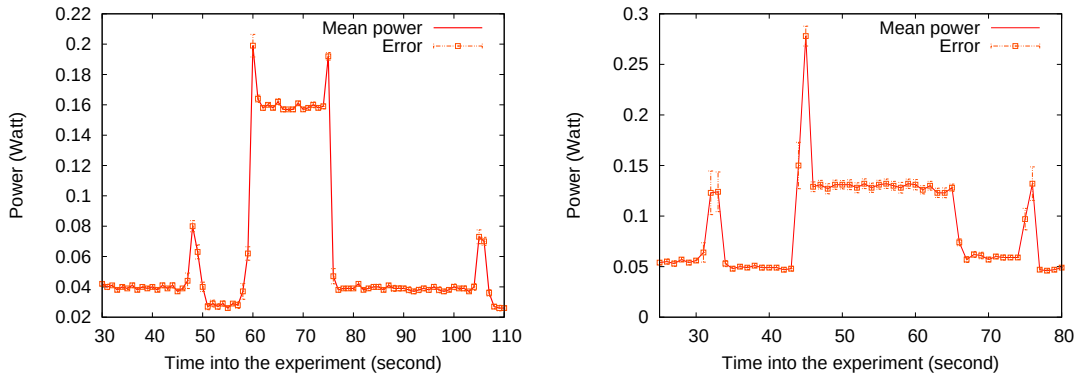


Fig. 2. Energy consumption of two smartphones in three different states of their screens: Nokia E71 (left), and Nokia N95 (right)

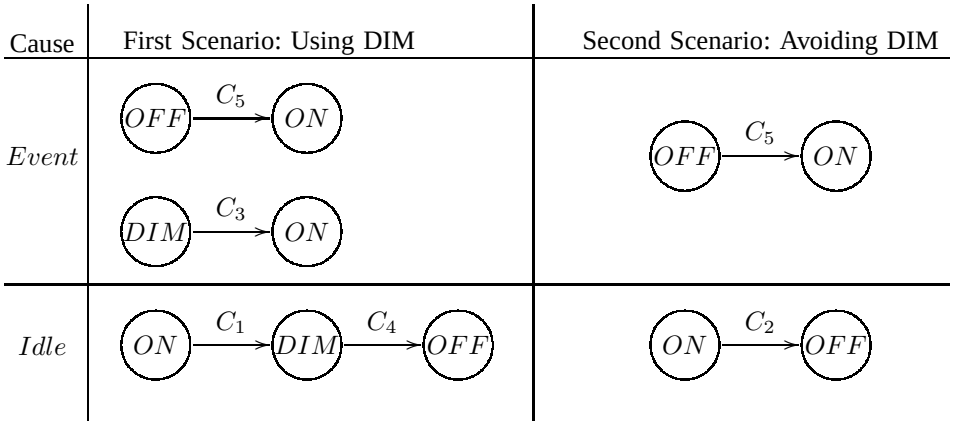


Fig. 4. State transitions in each screen management scenario. The label on each arrow represents the fixed cost associated with the transition.

the three states as a function of the length of the idle period in a hypothetical setting. The three states of the phone screen are represented with three lines. The slope of each line represents the energy consumption rate (power consumption) in that state. The intercept represents the fixed energy cost associated with that state. In this model, initially proposed by Augustine, et al. [1], the fixed costs relative to the next most active state are used. Thus, the line that represents *ON* has zero intercept and its slope is R_1.

When the screen scheduling algorithm decides to make the screen DIM, it should anticipate the fixed cost of entering this state and also the cost of leaving it when a new event arrives. Therefore the intercept of the line representing DIM in Figure 5 is $C_{DIM,ON} = C_1 + C_3$, and its slope is R_2.

Similarly the fixed cost associated with OFF is the sum of fixed costs of entering OFF from the previous state and the fixed cost of entering ON from OFF. Depending on whether the system is operating in the three-state or the two-state scenario, the previous state to OFF is DIM or ON. Therefore, the intercept of the line representing OFF is $C_{OFF,DIM} = C_4 + C_5$ or $C_{OFF,ON} = C_2 + C_5$, respectively. The slope of this line is R_3.

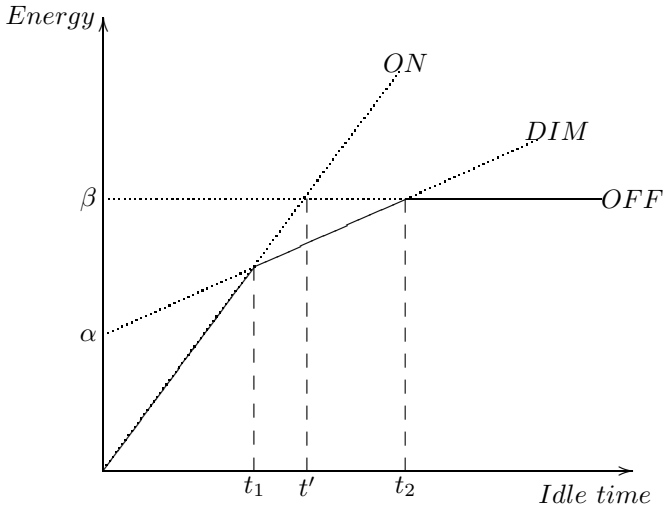


Fig. 5. Energy consumed by the screen in each possible state as a function of the length of idle time.

IV. Algorithm Design

In this section we present an on-line algorithm that gives the optimal schedule for screen management based on the power consumption profile of the screen. We then improve this algorithm to take into account the distribution of the input to achieve better performance in terms of energy consumption.

We define a strategy or schedule for screen management as (T_1, T_2), where T_1 and T_2 are the timeouts for turning the screen DIM and OFF, when the phone is idle. If $T_1 \geq T_2$, the schedule is essentially not using the DIM state of the screen which means the two-state scenario is being used.

The problem of finding the optimal strategy is an on-line problem, because the scheduler does not have the input in advance. A common metric to evaluate the performance of an on-line algorithm is its *competitive ratio*. Competitive ratio is the ratio of the worst case cost incurred by the algorithm to the cost of the optimal off-line algorithm that knows the future.

The algorithms that follow, effectively make two decisions. First they choose the better scenario, and second, they find the best timeout value(s) of that scenario.

A. Deterministic Algorithm

If the fixed costs are additive ($C_{OFF,ON} = C_{OFF,DIM} + C_{DIM,ON}$), the optimal deterministic on-line strategy, called Lower Envelope Algorithm (LEA), follows the lower envelope curve of Figure 5 (plotted in solid lines) at any time. The schedule resulted from their algorithm is (t_1, t_2). where t_1 is the first time that the cost of ON exceeds DIM, and t_2 is when keeping the screen DIM costs more than transitioning to OFF. Irani et al. [8] first proposed this algorithm and proved that it is 2-competitive.

Augustine et al. [1] consider the general case where the fixed transition costs are not additive. They give an algorithm that efficiently enumerates all possible strategies and finds the one with the lowest competitive ratio. They prove that the worst case competitive ratio of this algorithm is $3+2\sqrt{2}$. The number of states in the screen scheduling problem is limited to three, therefor we do not need to run their dynamic programing algorithm to decide which of the two possible strategies has a better competitive ratio. The algorithm in Figure 6 simply compares the worst case cost of the two possible schedules given the cost values of Table II and returns the schedule that has the lowest cost.

B. Probabilistic Algorithm

If the distribution of the length of idle times is known, the on-line algorithm can be augmented to optimize the expected cost of the screen schedule instead of the worst case cost.

Assume that the length of the next user idle time is a random variable that follows a known distribution, $\pi(t)$. If we use T as the timeout value to turn the screen OFF, the expected energy cost of the two-state scenario is:

$$E_{\pi(t)}(Cost_1(T)) = \int_0^T (R_1 t)\pi(t)dt + \int_T^\infty \big(R_1 T + R_3(t-T) + C_{ON,OFF}\big)\pi(t)dt \quad (1)$$

$t' = \frac{C_2+C_5}{R_1}$
$t_1 = \frac{C_1+C_3}{R_1-R_2}$
$t_2 = \frac{C_4+C_5-C_1-C_3}{R_2-R_3}$
$Cost_1 = t'R_1 + (C_2 + C_5)$
$Cost_2 = t_1 R_1 + (t_2 - t_1)R_2 + (C_1 + C_3) + (C_4 + C_5)$
if $Cost_2 < Cost_1$ **then**
  return (t_1, t_2)
else
  return (t', t')
end if

Fig. 6. Deterministic algorithm to find the optimal screen schedule

Obviously, T can be selected to minimize the expected cost.

Augustine et al. [1] proved that in the multi-state scenario the optimal transition times are the optimal transition times of the simplified two-state systems. Therefore the optimal time to transition from ON to DIM in the three-state scenario is T_1 that minimizes the following expected cost.

$$E_{\pi(t)}(Cost'_2(T_1)) = \int_0^{T_1} (R_1 t)\pi(t)dt + \int_{T_1}^\infty \big(R_1 T_1 + R_2(t-T_1) + C_{ON,DIM}\big)\pi(t)dt \quad (2)$$

And the optimal T_2 to transition form DIM to OFF is one that minimizes the following:

$$E_{\pi(t)}(Cost''_2(T_2)) = \int_0^{T_2} (R_2 t)\pi(t)dt + \int_{T_2}^\infty \big(R_2 T_2 + R_3(t-T_2) + C_{DIM,OFF}\big)\pi(t)dt \quad (3)$$

The expected cost of the three-state strategy is neither $E(Cost'_1)$ nor $E(Cost''_2)$. Once the optimal T_1 and T_2 that minimize $E(Cost'_1)$ and $E(Cost'_2)$ are determined, we can compute the expected cost of the three state strategy as following:

$$E_{\pi(t)}(Cost_2(T_1,T_2)) = \int_0^{T_1} (R_1 t)\pi(t)dt + \int_{T_1}^{T_2} \big(R_1 T_1 + R_2(t-T_1) + C_{ON,DIM}\big)\pi(t)dt + \int_{T_2}^\infty \big(R_1 T_1 + R_2(T_2-T_1) + C_{ON,DIM} + C_{DIM,OFF}\big)\pi(t)dt \quad (4)$$

The algorithm in Figure 7 decides which scenario gives the optimal screen schedule and returns the optimal schedule.

V. Measurements

To evaluate the performance of the proposed algorithms we performed measurements on real phones and with real users. In this section we present the methodology and results of our measurements.

$t' = \arg\min E_{\pi(t)}(Cost_1(T))$
$t_1 = \arg\min E_{\pi(t)}(Cost'_1(T_1))$
$t_2 = \arg\min E_{\pi(t)}(Cost''_1(T_2))$
$Cost_1 = E_{\pi(t)}(Cost_1(t'))$
$Cost_2 = E_{\pi(t)}(Cost_2(t_1, t_2))$
if $Cost_2 < Cost_1$ **then**
  return (t_1, t_2)
else
  return (t', t')
end if

Fig. 7. Probabilistic algorithm to find the optimal screen schedule

State	E71	N95
R_1 (ON)	0.132 (W)	0.081 (W)
R_2 (DIM)	0.011 (W)	0.011 (W)
R_3 (OFF)	0.000 (W)	0.000 (W)

TABLE III
POWER CONSUMPTION OF SCREEN STATES OF TWO NOKIA SMARTPHONES

A. Energy Profiling

We performed micro-benchmarks on two popular smartphones, Nokia N95 and E71. On each smartphone we installed the Nokia Energy Profiler [5] and ran the benchmarks while the profiler was running. We repeated each benchmark 10 times. In this section we report the average energy consumption across the 10 measurements. The average of measured energy values are plotted in Figures 8 and 9 along with error bars.

You can see three plots in each of these two figures. Each plot is the result of a different micro-benchmark designed to measure specific fixed costs:

1) The phone is left idle until the screen turns DIM and then OFF.
2) The phone s configured to avoid the DIM sate. It s left to turn the screen OFF, then we press a button to turn the screen ON and wait until it turns off again.
3) The phone s configured to avoid OFF, and only turn DIM. It is left to turn DIM, then we press a button to turn the screen ON and wait until it turned DIM again.

We have labeled the power consumption spikes that indicate fixed energy costs associated with state transitions in Figures 8 and 9 with the corresponding names from Table II. We were unable to design a benchmark to measure C_6, but fortunately this cost is not used in any of the algorithms in Section IV.

Table III summarizes the average power consumption of the screen in each state. We subtract the base power consumption (device power consumption when the screen is OFF) from the total consumption values to get the power that is consumed only by the screen. Table IV summarizes the average fixed transition costs measured for each smartphone.

B. User Study

We gave four N95 and two E71 Nokia smartphones to six volunteers. The subjects agreed to use these handsets as their daily phones. We installed the *Nokia Simple Context (NSC) collector* on these phones, and configured it to sample system information every 10 seconds. The system module of NSC, among other system level information, logs the screen inactive timer. We use the value of this timer to infer the length of screen inactivity times. N95 users participated in this experiment for four weeks, but E71 users continued collecting data beyond the end of one month.

Fixed cost	E71	N95
C_1	0.248	0.049
C_2	0.284	0.243
C_3	0.236	0.174
C_4	0.119	0.096
C_5	0.262	0.360
$C_{DIM,ON}$	0.484	0.223
$C_{OFF,DIM}$	0.381	0.456
$C_{OFF,ON}$	0.546	0.603

TABLE IV
FIXED ENERGY INCURRED DURING SCREEN STATE TRANSITIONS OF TWO NOKIA SMARTPHONES

Figure 10 plots the cumulative distribution functions of the length of inactivity times of two N95 users and the two E71 users. Based on the semi-log plots of the complementary cumulative distribution functions in Figure 11 we conjecture that inactivity times of smartphone users follow truncated exponential distributions. However, the algorithm in Figure 7 is agnostic to the type of the distribution of idle times. But it requires the distribution to be fixed over time (i.e., the idle times be a *stationary* process). A stationary process is a stochastic process whose joint probability distribution does not change over time or space.

We hypothesize that the length of user inactivity times on their phones is a stationary process. We used the KPSS test for stationarity [12] to test our hypothesis. The KPSS test assumes the following model:

$$x_t = r_t + \beta t + \epsilon_t$$

Where r_t is a random walk (i.e., $r_t = r_{t-1} + u_t$, and $u_t \sim i.i.d\ N(0, \sigma_u^2)$). βt is a deterministic trend, and ϵ_t is a stationary error. In this model, to test if x_t is a *level stationary* process, that is the series is stationary around a fixed level, the null hypothesis is $\beta = 0$.

The KPSS test statistic is:

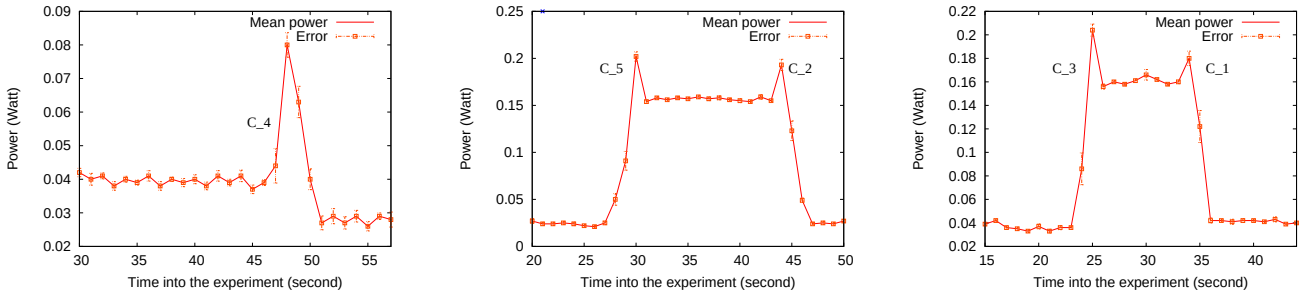
$$KPSS = \frac{\sum_{t=1}^{N} S_t}{N^2 \hat{\sigma}^2} \quad (5)$$

Where $S_t = \sum_{j=1}^{t}(x_t - \overline{x})$ and $\hat{\sigma}^2$ is an estimator of the long-run variance of $(x_t - \overline{x})$.

According to Kwiatkowski et al. [12] under the null hypothesis of level stationarity:

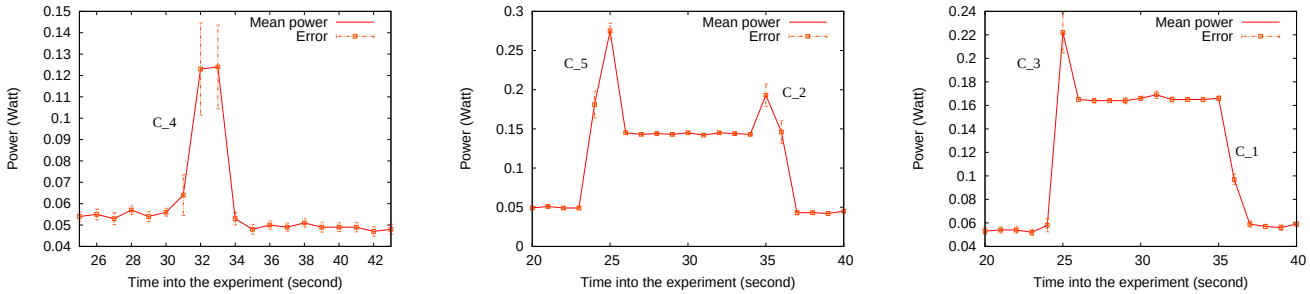
$$KPSS \rightarrow \int_0^1 V_1(r)^2 dr \quad (6)$$

where $V_1(r)$ is a Brownian bridge (i.e., assuming that $B(r)$ is a Brownian motion process on $r \in [0, 1]$, $V_1(r) = B(r) -$



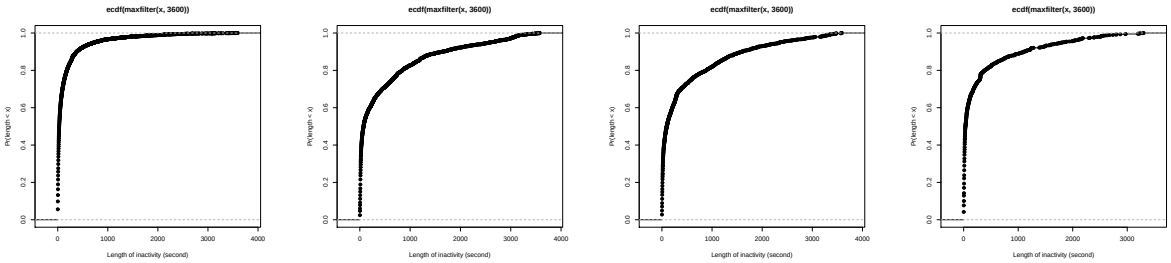
(a) DIM, OFF (b) OFF, ON (c) DIM, ON

Fig. 8. Profiling the power consumption of an E71 smartphone screen in three different states



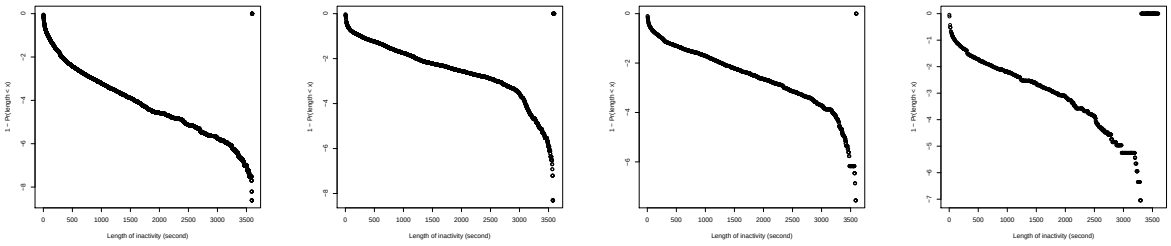
(a) DIM, OFF (b) OFF, ON (c) DIM, ON

Fig. 9. Profiling the power consumption of an N95 smartphone screen in three different states



(a) E71-1 (b) E71-2 (c) N95-1 (d) N95-2

Fig. 10. CDF of the length of inactivity times six mobile users



(a) E71-1 (b) E71-1 (c) N95-1 (d) N95-2

Fig. 11. Semi-log plot of the CCDF of the length of inactivity times

	Upper tail percentiles			
Distribution	**0.1**	**0.05**	**0.025**	**0.01**
KPSS	0.347	0.463	0.574	0.739

TABLE V
UPPER TAIL CRITICAL VALUES FOR THE KPSS STATISTIC ASYMPTOTIC DISTRIBUTION

user	KPSS level	p-value
E71-1	0.428	0.06
E71-2	0.348	0.09
N95-1	0.432	0.06
N95-2	0.725	0.01
N95-3	0.239	> 0.1
N95-4	0.652	0.01

TABLE VI
KPSS TEST FOR STATIONARITY RESULTS

$rB(1)$). Table V lists the upper tail critical values of the asymptotic distribution of the KPSS statistic.

Intuitively, a smaller KPSS value indicates higher stationarity. As an example the KPSS value of a sequence of 10,000 *i.i.d* samples from *Normal(0,1)* is 0.054. The KPSS test implementation in R also reports the *p-value* for the given KPSS value. It approximates the p-values by interpolation from a simulated table of critical values. p-values larger than 0.1 are typically considered to be non-significant and p-values smaller than 0.01 are regarded highly significant.

Table VI summarizes the results of the KPSS test on all our users' data. The p-value is greater than 0.1 only for one user [2]. We can conclude that for five out of six users the length of idle times are stationary distributions with high confidence.

Our explanation for the lack of confidence in stationarity of the distribution of idle times of user N95-3 is that this user did not effectively use the phone as his/her primary phone and used it along with his/her personal phone.

VI. EVALUATION

In this section we evaluate the performance of the proposed algorithms in IV using our measurement results. For all the schedules that will be presented we round the timeout values to the second which is the maximum granularity of our phones' screen timeout setting.

A. Deterministic Schedules

Figure 12 plots the lower energy envelope for both phones using the results of our energy profiling. An interesting observation is that on E71 the lower envelop does not include the DIM line. This is intuitively why the algorithm in Figure 6 does not use the DIM state and returns $(4, 4)$. Four seconds is the time when the lines representing OFF (with intercept $C_{OFF,ON}$) and ON intersect. For N95 Algorithm 6 returns $(3, 20)$ as the optimal schedule.

[2]The KPSS.test() implementation in R does not report the p-value when it is greater than 0.1

Setting	**Phone**	$\mathbf{T_1}$	$\mathbf{T_2}$
Factory Setting	E71	25	45
	N95	10	60
Deterministic Optimal	E71	4	4
	N95	3	20

TABLE VII
DEFAULT AND DETERMINISTIC OPTIMAL SCREEN SCHEDULES FOR E71 AND N95

User	$\mathbf{T}$	$\mathbf{E[Cost_1]}$	$\mathbf{T_1}$	$\mathbf{T_1}$	$\mathbf{E[Cost_2]}$
E71-1	1	0.140	1	10	0.173
E71-2	1	0.138	1	9	0.181
N95-1	1	0.138	1	8	0.134
N95-2	5	0.126	1	13	0.110
N95-3	2	0.128	1	21	0.108
N95-4	3	0.132	1	24	0.113

TABLE VIII
COMPARING THE EXPECTED COST OF THE TWO-STATE AND THREE-STATE SCHEDULES FOR EACH USER.

B. Probabilistic Schedules

We divide the traces from each user into two sets of equal size. We use the first set as *training data* and the second set as *testing data*. We use the training data to compute the histogram of the length of idle times and use this histogram in Algorithm 7 to numerically compute t', t_1, t_2 and the corresponding values of $Cost_1$ and $Cost_2$. These values are listed in Table VIII. For E71 users we use the energy costs of the E71 phone, and similarly for N95 users. We do this, because we observed that the form factor of a phone affects its usage. For example, E71 users interact with their phones more often that N95 users.

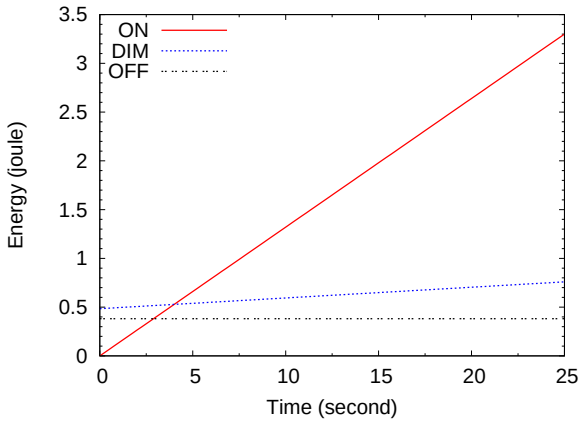
The result of Algorithm 7 for each user based on his/her phone is summarized in Table IX. You can see that similar to the deterministic algorithm, the probabilistic algorithm decides not to use the DIM state of the E71 screen. Also as a result of the short tail of the probability distribution of the length of idle times, the probabilistic algorithm chooses shorter timeout values compared to the deterministic algorithm.

C. Performance Evaluation

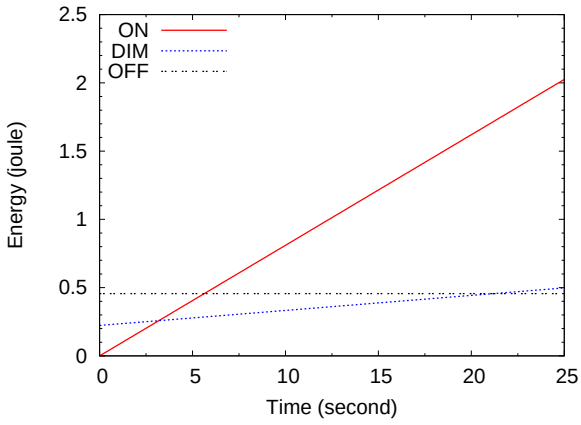
We implemented a trace-based simulator, named *ScreenSim*, to compare the performance of the three screen schedules: default, deterministic, and probabilistic. Given a screen management schedule, the energy profile of a mobile phone screen, and the usage history of a user, ScreenSim computes the total

User	$\mathbf{T_1}$	$\mathbf{T_1}$
E71-1	1	1
E71-2	1	1
N95-1	1	8
N95-2	1	13
N95-3	1	21
N95-4	1	24

TABLE IX
OPTIMAL PROBABILISTIC SCHEDULES FOR EACH USER



(a) E71



(b) N95

Fig. 12. Screen energy lower envelop for the deterministic algorithm

user	Default	Deterministic Opt.	Probabilistic Opt
E71-1	931.611 ±5.572	317.934 ±3.255	223.672 ±2.730
E71-2	305.51 ±3.567	97.731 ±1.976	61.528 ±1.601
N95-1	94.578 ±2.292	55.052 ±1.748	45.069 ±1.582
N95-2	64.496 ±3.035	43.316 ±2.326	36.640 ±2.140
N95-3	72.453 ±2.127	45.661 ±1.689	42.126 ±1.622
N95-4	30.799 ±3.204	18.072 ±2.454	17.298 ±2.401

TABLE X
AVERAGE DAILY ENERGY DISSIPATION OF THE SCREEN BY THREE SCREEN SCHEDULES FOR EACH USER

energy consumed by the screen.

Table X lists the average and standard deviation of the daily energy consumed by each user when using each of the three screen schedules. The average is computed over multiple days of usage. Figure 13 plots the ratio of energy consumed by the deterministic and probabilistic schedules to the default schedule energy consumption for each user.

The deterministic schedule results in 50% average energy reduction compared to the default factory setting. The energy reduction resulting from the probabilistic schedules that take into account the usage pattern of each user is about 40% on average.

Figure 14 plots the histogram of the ratio of energy consumed by the probabilistic schedule of each user to the deterministic schedule. The average ratio is about 80%. The improvement resulted from the probabilistic schedule is minimal for users N95-3 and N95-4. These two users have the lowest number of total interactions, therefore the probabilistic algorithm does not have much input to work with. In other words, because of the infrequent interactions, the probabilistic schedule is close to the deterministic.

VII. RELATED WORK

Prior to the introduction of on-line DPM algorithms, DPM solutions have been following either a predictive approach or a stochastic control approach [2]. The predictive approaches attempt to predict the length of the next idle period using past

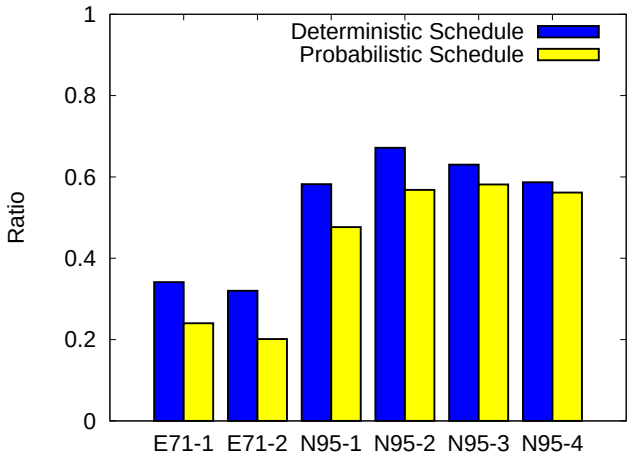


Fig. 13. Ratio of energy consumed by the deterministic and probabilistic schedules to the default schedule energy consumption

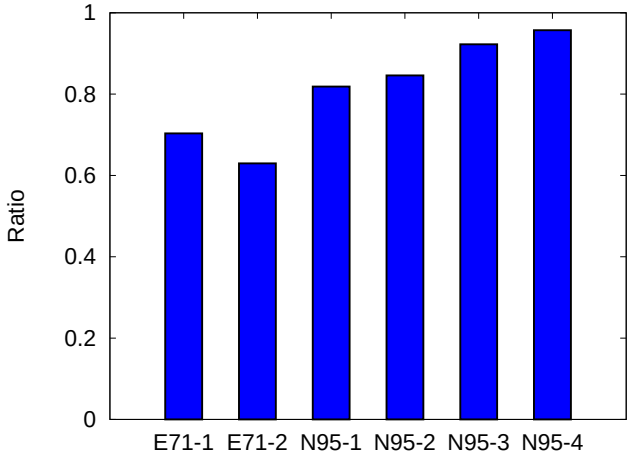


Fig. 14. Ratio of energy consumed by the probabilistic schedules to the energy consumed by the deterministic schedules

history of the system. Based on this prediction, the power management system chooses the optimal power state switching threshold. The prediction algorithm is either adaptive or non-adaptive. Non-adaptive predictive algorithms set the idleness threshold once and forever, whereas adaptive algorithms are affected by the observed input patterns [7] [3] [4] [18] [13]. All these schemes make a single prediction and pay the overhead if the actual idle time is different from the predicted value.

Stochastic approaches [15] [17] [16] make assumptions about the probability distribution of the job request patterns and formulate the DPM problem as an stochastic optimization problem. They are highly sensitive to the underlying assumptions, and proving any theoretical bounds on their performance is difficult.

Recently researchers have studied DPM as an on-line problem [1] [8] [9]. Similar to any on-line algorithm, the power management system should decide about resource allocation before knowing all the input (i.e. the length of the idle period). On-line algorithms are analyzed based on their *competitive ratio*; the ratio of the on-line algorithm cost, to the cost of the optimal off-line algorithm.

The same general setting as the Dynamic Power Management problem, although simpler, exists in many other areas of Computer Science. In a shared memory multiprocessing system, a process that is waiting for a locked resource must decide whether to spin or lock [10]. A gateway between a connection-oriented network and a connectionless network should decide when to drop a connection [11]. These are two-state examples of the same problem for which there is a 2-competitive deterministic on-line algorithm. When the input follows a known probability distribution the on-line algorithm can perform better and achieve $e/(e-1)$ expected competitive ratio.

VIII. Conclusions and Future Work

In this paper we study optimal screen management on mobile phones. We formally model the problem and identify the costs associated with a mobile phone screen. We profile the energy consumption of two popular smartphones (N95 and E71) through carefully designed micro-benchmarks. We also perform a user study with six subjects. Our results show that users' interaction with their phones is not arbitrary - the distribution of the length of idle times of each user does not change over time. We propose two algorithms to find effective values for turning the screen dim and off when the phone is idle. Our deterministic algorithm makes no assumptions about the length of idle times. The probabilistic algorithm improves the deterministic result by considering the distribution of the length user idle times.

We discover that on the E71 turning the screen dim is non-optimal in terms of energy consumption. We evaluate the performance of our algorithms using trace-based simulation. The energy consumption of the deterministic schedule is on average 50% of the default schedule. Also our evaluations show that taking into account the usage history of a phone can result in an extra energy saving of up to 60% compared to the optimal deterministic algorithm.

We conclude that the default screen management on the Nokia smartphones that we studied are non-optimal in terms of energy consumption, and can be improved in two steps:

1) Considering the energy profile of the phone screen
2) Taking into account the usage history

As a future work we intend to implement a simple client that runs on the phone to process the usage history and suggest optimal timeout values to the user.

In our evaluations we did not consider the user experience. One might argue that our suggested timeout values are too short for most mobile applications. While this argument is true, we believe the users can make informed decisions about the settings of their phone when they know the optimal screen schedule. We also think the user experience can be improved by setting the screen schedule for each application separately. Each application knows how its users interact with it. Through carefully designed interfaces, this information can be communicated to the screen scheduler. The scheduler can find the optimal trade-off between energy saving and user satisfaction.

Most modern smartphones use a light sensor to measure the ambient light and set the screen brightness accordingly. During our energy profiling we fixed the screen brightness. The screens on Nokia phones have five levels of brightness. We can perform similar energy profiling for each brightness level and find the optimal schedule for that brightness setting.

IX. Acknowledgments

I would like to acknowledge comments from Prof. Ramesh Govindan and my advisor, Prof. Deborah Estrin, on this work.

References

[1] J. Augustine, S. Irani, and C. Swamy. Optimal power-down strategies. In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 530–539, 2004.

[2] L. Benini, A. Bogliolo, and G. De Micheli. A survey of design techniques for system-level dynamic power management. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 8(3):299–316, 2000.

[3] L. Benini, A. Bogliolo, GA Paleologo, and G. De Micheli. Policy optimization for dynamic power management. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 18(6):813–833, 1999.

[4] E.Y. Chung, L. Benini, A. Bogliolo, Y.H. Lu, and G. De Micheli. Dynamic power management for nonstationary service requests. *IEEE Transactions on Computers*, 51(11):1345–1361, 2002.

[5] G.B. Creus and M. Kuulusa. Optimizing Mobile Software with Built-in Power Profiling. *Mobile Phone Programming and its Application to Wireless Networking, F. Fitzek and F. Reichert, Eds. Springer*, 2007.

[6] Carla Schlatter Ellis. *Controlling Energy Demand in Mobile Computing Systems*. Morgan and Claypool, 2003.

[7] C.H. Hwang and C.H.W. Allen. A predictive system shutdown method for energy saving of event-driven computation. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 5(2):226–241, 2000.

[8] S. Irani, R. Gupta, and S. Shukla. Competitive analysis of dynamic power management strategies for systems with multiple power savings states. In *Proceedings of the conference on Design, automation and test in Europe*. IEEE Computer Society Washington, DC, USA, 2002.

[9] S. Irani, G. Singh, SK Shukla, and RK Gupta. An overview of the competitive and adversarial approaches to designing dynamic power management strategies. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 13(12):1349–1361, 2005.

[10] A.R. Karlin, K. Li, M.S. Manasse, and S. Owicki. Empirical studies of competitve spinning for a shared-memory multiprocessor. *ACM SIGOPS Operating Systems Review*, 25(5):41–55, 1991.

[11] S. Keshav, C. Lund, S. Phillips, N. Reingold, and H. Saran. An empirical evaluation of virtual circuit holding time policies in IP-over-ATM networks. *IEEE Journal on Selected Areas in Communications*, 13(8):1371–1382, 1995.

[12] D. Kwiatkowski, P.C.B. Phillips, P. Schmidt, and Y. Shin. Testing the null hypothesis of stationarity against the alternative of a unit root. *Journal of econometrics*, 54(1-3):159–178, 1992.

[13] Y.H. Lu and G. De Micheli. Adaptive hard disk power management on personal computers. In *VLSI, 1999. Proceedings. Ninth Great Lakes Symposium on*, pages 50–53, 1999.

[14] Y. Neuvo. Cellular phones as embedded systems. In *Solid-State Circuits Conference, 2004. Digest of Technical Papers. ISSCC. 2004 IEEE International*, pages 32–37, 2004.

[15] Q. Qiu and M. Pedram. Dynamic power management based on continuous-time Markov decision processes. In *Proceedings of the 36th ACM/IEEE conference on Design automation*, pages 555–561. ACM New York, NY, USA, 1999.

[16] Q. Qiu, Q. Qu, and M. Pedram. Stochastic modeling of a power-managed system-construction andoptimization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 20(10):1200–1217, 2001.

[17] T. Simunic, P. Glynn, and G. De Micheli. Dynamic power management for portable systems. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 11–19. ACM New York, NY, USA, 2000.

[18] MB Srivastava, AP Chandrakasan, and RW Brodersen. Predictive system shutdown and other architectural techniques forenergy efficient programmable computation. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 4(1):42–55, 1996.