Energy Consumption in Mobile Phones: Measurement, Design Implications, and Algorithms

Niranjan Balasubramanian Aruna Balasubramanian Arun Venkataramani
Department of Computer Science
University of Massachusetts Amherst
{niranjan, arunab, arun}@cs.umass.edu

ABSTRACT

In this paper, we present a measurement study of the energy consumption characteristics of three widespread mobile networking technologies: 3G, GSM, and WiFi. We find that 3G and GSM incur a high *tail energy* overhead because of lingering in high power states after completing a transfer. Based on these measurements, we develop a model for the energy consumed by network activity for each technology. Armed with this model, we seek to reduce the energy consumption of common mobile applications.

Towards this goal, we develop TailEnder, a protocol that schedules transfers so as to minimize the cumulative energy consumed while meeting user-specified delay-tolerance deadlines. We show that the TailEnder algorithm is within a factor $1.28\times$ of the optimal and show that no deterministic online algorithm can achieve a better competitive ratio. For applications like web search that can benefit from prefetching, TailEnder can aggressively prefetch several times more data and improve user-specified response times while consuming *less* energy. We evaluate the benefits of TailEnder for three different case study applications—email, news feeds, and web search—based on real user logs and show significant reduction in energy consumption in each case. Experiments conducted on the mobile phone shows that TailEnder can download 60% more news feed updates and download search results for more than 50% of web queries, compared to using the default policy. Our model-driven simulation shows that TailEnder can reduce energy by 35% for email applications, 52% for news feeds and 40% for web search.

1. INTRODUCTION

Mobile phones are ubiquitous today with an estimated cellular subscription of over 4 billion worldwide [2]. Most phones today support one or more of 3G, GSM, and WiFi for data transfer. For example, the penetration of 3G is estimated at over 15% of cellular subscriptions worldwide and is over 70% in some countries [1]. Recent measurement studies report that in the daily lives of urban users, cellular and WiFi availability is about 99% and 50% respectively.

How do the energy consumption characteristics of network activity over 3G, GSM, and WiFi on mobile phones compare with each other? How can we reduce the energy consumed by common applications using each of these three technologies? To investigate these questions, we first conduct a detailed measurement study to quantify the energy consumed by data transfers across 3G, GSM, and WiFi. We find that the energy consumption is intimately related to the characteristics of the workload and not just the total transfer size, e.g., a few hundred bytes transferred intermittently on 3G can consume more energy than transferring a megabyte in one shot. The key findings of our measurement are summarized below. These findings remain consistent across three different cities, diurnal variation, mobility patterns, and devices.

1. In 3G, a large fraction (nearly 60%) of the energy, referred to as the *tail energy*, is wasted in high-power states after the completion of a typical transfer. In comparison, the *ramp energy* spent in switching to this high-power state before the transfer is small. Tail and ramp energies are constants that amortize over larger transfer sizes or frequent successive transfers.

2. In GSM, although a similar trend exists, the time spent in the high-power state after the transfer, or the *tail time*, is much smaller compared to 3G (6 vs. 12 secs). Furthermore, the lower data rate of GSM implies that more energy is spent in the actual transfer of data.

3. In Wifi, the association overhead is comparable to the tail energy of 3G, but the data transfer itself is significantly more efficient than 3G for all transfer sizes.

Based on these findings, we develop a simple model of energy consumption of network activity for each of the three technologies. Armed with these models, we ask how we can reduce the energy consumption of network activity induced by common mobile applications. To this end, we design TailEnder, an energy-efficient protocol for scheduling data transfers. TailEnder considers two classes of applications: 1) delay-tolerant applications such as email and RSS feeds, and 2) applications such as web search and web browsing that can benefit from aggressive prefetching.

For delay-tolerant applications on 3G and GSM, TailEnder schedules outgoing transfers so as to minimize the overall time spent in high energy states after completing transfers, while respecting user-specified delay-tolerance deadlines. We show that the TailEnder scheduling algorithm is provably within a factor $1.28\times$ of the energy consumed by an optimal offline algorithm that knows the complete arrival pattern of transfers a priori. Furthermore, we show that no deterministic online algorithm can be better than 1.28-competitive with respect to an optimal offline adversary, i.e., the upper and lower bounds on the competitive ratio are tight.

For applications that can benefit from prefetching, TailEnder determines what data to prefetch so as to minimize the overall energy consumed. Prefetching useful data reduces the number of transfers and their associated cumulative tail energy, while prefetching useless data incurs additional transmission energy. TailEnder uses a probabilistic strategy to balance these concerns. Somewhat counterintuitively, for applications such as web search, TailEnder fetches several times more data and improves user-perceived response times, but still consumes *less* energy.

We evaluate the performance of TailEnder for three different applications: email, news feeds and web search. For each of these applications, we collect real user traces including arrival times and transfer sizes. We evaluate TailEnder by conducting experiments on the mobile phone and find that TailEnder can download 60% more news feed updates and download search results for more than 50% of web queries, compared to using the default policy. Our model-driven simulation shows that TailEnder can reduce energy by 35% for email applications, 52% for news feeds and 40% for web search. Further, we find that, opportunistic WiFi access substantially reduces energy consumption compared to only using 3G. Even when WiFi in only available 50% of the time, sending data over WiFi when available reduces the energy consumption by over 3 times for all three applications.

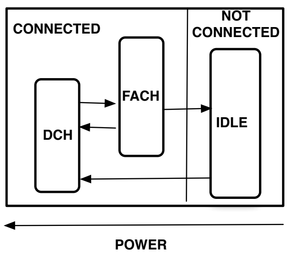
2. BACKGROUND AND RELATED WORK

2.1 Cellular power management

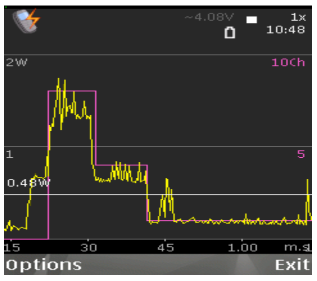
Two factors determine the energy consumption due to network activity in a cellular device. First, is the transmission energy that is proportional to the length of a transmission and the transmit power level. Second, is the *Radio Resource Control* (RRC) protocol that is responsible for channel allocation and scaling the power consumed by the radio based on inactivity timers.

Figure 1(a) shows the state machine [16] implemented by the RRC protocol for GSM/EDGE/GPRS (2.5G) as well as UMTS/WCDMA (3G) networks that follow the 3GPP [4] standard. The radio remains in the IDLE state in the absence of any network activity. The radio transitions to the higher power states, DCH (Dedicated Channel) or FACH (Forward Access Channel), when the network is active. The DCH state reserves a dedicated channel to the device and ensures high throughput and low delay for transmissions, but at the cost of high power consumption. The FACH state shares the channel with other devices and is used when there is little traffic to transmit and consumes about half of the power in the DCH state. The IDLE state consumes about one percent of the power in the DCH state.

The transition between the different states is controlled by



(a)



(b)

Figure 1: : (a) The radio resource control state machine for 3GPP networks consisting of three states: IDLE, DCH and FACH (b) Instantaneous power measurements for an example transfer over 3G showing the transition time between high to low power state

inactivity timers [16]. Figure 1(b) shows the instantaneous power measurements for an example transfer. The graph shows the time taken to transition from a high power to a low power state. Instead of transitioning from the high to the low power state immediately after a packet is transmitted, the device transitions only when the network has been inactive for the length of the inactivity timer. This mechanism serves two benefits: 1) it alleviates the delay incurred in moving to the high power state from the idle state, and 2) it reduces the signaling overhead incurred due to channel allocation and release during state transitions. Since lingering in the high power state also consumes more energy, network operators estimate the value of the inactivity timer based on this peformance/energy trade-off [16, 11], with typical values being several seconds long.

The 3GPP2 standard [3] used by the CDMA2000 technology is another standard for 3G networks, and 3GPP and 3GPP2 are the most prevalent standards today. Though the state machine for the radio resource control in the 3GPP2 standard is different from that shown in Figure 1, several features are similar. In particular, the 3GPP2 standard also uses an inactivity timer to transition from the high power to the low power state for performance reasons [16, 23].

2.2 WiFi power management

In comparison, WiFi incurs a high initial cost of associating with an access point (AP). However, because WiFi on phones typically uses the Power Save Mode (PSM), the cost of maintaining the association is small. When associated, the energy consumed by a data transfer is proportional to the size of the data transfer and the transmit power level. Our measurements (Section 3) confirm that the transmission energy consumed by WiFi is significantly smaller than both 3G and GSM[1], especially for large transfer sizes.

2.3 Related work

[1]We use the terms GSM and 2.5G interchangeably in this paper.

Energy consumption of network activity in mobile phones has seen a large body of work in recent times. To our knowledge, our paper presents the first comparative study of energy consumption characteristics of all three technologies—3G, GSM, and WiFi—that are under widespread use today. In particular, our study of the energy consumption characteristics of 3G reveals significant and nonintuitive implications for energy-efficient application design.

Analytical modeling. Prior work [16, 23, 11] has studied the impact of different energy saving techniques in 3G networks using analytical models. These works analyze the impact of the inactivity timer by modeling the delay and energy utilization for different values for the inactivity timer. Their goal is to determine the optimal value of the inactivity timer from the network operator's perspective. Yeh *et al.* [23] analytically compare the impact of the inactivity timer in both 3GPP and 3GPP2 networks. In comparison, our study treats the inactivity timer value as a given and develops algorithms for energy-efficient application design based on real measurements and application traces.

Measurement. Gupta *et al.* [13] present a measurement study of the energy consumption of VoIP applications over WiFi-based mobile phones. The authors find that intelligent scanning strategies and aggressive use of PSM in WiFi can reduce power consumption for VoIP applications. Xiao *et al.* [22] measure the energy consumption for Youtube-like video streaming applications in mobile phones using both WiFi and 3G. Their focus is on the energy utilization of various storage strategies and application-level strategies such as delayed-playback and playback after download. Nurminen *et al.* [18] measure the energy consumption for peer-to-peer applications from mobile phones over 3G. In comparison to these application-specific studies, our focus is on reducing the energy consumption of general network activity across 3G, GSM, and WiFi.

Energy-efficient mobile network activity. Several previous studies [5, 20, 21, 6] have investigated strategies for energy-efficient network activity in mobile phones supporting multiple wireless technologies. Pering et al.[20] develop strategies to intelligently switch between WiFi and Bluetooth. Agarwal et al.[5] propose an architecture to use the GSM radio to wake up the WiFi radio upon an incoming VoIP call to leverage the better quality and energy-efficiency of WiFi while keeping its scanning costs low.

Rahmati *et al.* [21] show that intelligently switching between WiFi and GSM reduces energy consumption substantially as WiFi consumes less transmission power. However, in order to avoid the cost of unnecessary scanning in the face of poor WiFi availability, the authors design an algorithm that predicts WiFi availability, and the device scans for WiFi access points only in areas where WiFi is available with high probability. Trevor *et al.* [6] present application-level modifications to reduce energy consumption for updates to dynamic web content. Their key ideas include using a proxy to 1) only push new content when the portion of the web document of interest to the user is updated, 2) batch updates to avoid the overhead of repeated polling, and 3) use SMS on GSM to signal the selection of WiFi or GSM based on the transfer size for energy-efficient data transfer. In addition to confirming these prior findings about GSM and WiFi power consumption, our measurement study also investigates 3G that reveals significantly different energy consumption characteristics, which lead us to develop novel energy-efficient and provably near-optimal data transfer algorithms for 3G.

Algorithms. Prior theoretical works [10, 12, 14, 7] study the problem of energy minimization while meeting job deadlines in processors that transition between the sleep and active state. The model assumed in most of these works is that a device incurs a large *ramp energy* overhead in transitioning from the low power state to the high power state, so the goal is to minimize the number of transitions. As explained in Sections 3 and 4, this model cannot be applied *as is* to mobile phones because their transition characteristics are different and include a significant *tail energy* component. For this model, we develop a provably near-optimal online algorithm to minimize energy, and further show that no deterministic online algorithm can achieve a better competitive ratio.

3. MEASUREMENT

The main goals of our measurement study are to:

1. Compare the energy consumption characteristics of 3G, GSM and WiFi and measure the fraction of energy consumed for data transfer versus overhead.
2. Analyze the variation of the energy overhead with geographic location, time-of-day, mobility, and devices.
3. Develop a simple energy model to quantify the energy consumption over 3G, WiFi and GSM as a function of the transfer size and the inter-transfer times.

Next, we describe our measurement methodology and then present our findings.

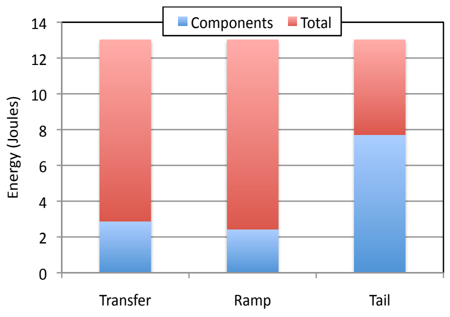
3.1 Devices and Tools

Our experiments are performed using four Nokia N95 phones[2]. Two of the phones are 3G-enabled AT&T phones that use HSDPA/UMTS technology and two are GSM-enabled AT&T phones that use EDGE. All four phones were equipped with an 802.11b WiFi interface. We use Python, PyS60 v1.4.2, developed for the Symbian OS 3rdEd FP 1 to conduct data transfer experiments.

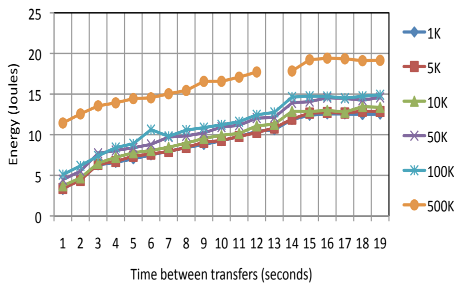
To measure energy consumption, we use Nokia's energy profiling application, the Nokia Energy Profiler (NEP) v1.1[3]. NEP provides instantaneous power measurements sampled once every 250 milli-seconds. Using the power measurements, we estimate the energy consumed by approximating the area under the power measurement curve over a time interval. We subtract the idle power from the energy consumption

[2]http://en.wikipedia.org/wiki/Nokia_N95

[3]http://www.forum.nokia.com/Resources_and_Information/Tools/Plug-ins/Enablers/Nokia_Energy_Profiler/



(a)



(b)

Figure 2: : 3G Measurements: (a) Ramp, transfer and tail energy for 50K 3G transfer. (b) Average energy consumed for transfer against the time between successive transfers

estimation when appropriate. Unless stated, all measurement results are averaged over 20 trials.

3.2 Measurement Methodology

3.2.1 3G and GSM

We conduct measurement study to quantify the: 1) *Ramp energy*: energy required to switch to the high-power state, 2) *Transmission energy*, and 3) *Tail energy*: energy spent in high-power state after the completion of the transfer.

We conduct measurements for data transfers of different sizes (1 to 1000 KB) with varying intervals (1 to 20 seconds) between successive transfers. We measure energy consumption by running NEP in the background while making data transfers. For each configuration of (x,t), where x $\in$ [1K, 1000K] and t $\in$ [1, 20] seconds, the data transfers proceed as follows: The phone initiates an x KB download by issuing a http-request to a remote server. After the download is completed, the phone waits for t seconds and then issues the next http request. This process is repeated 20 times for each data size. Between data transfer experiments for different intervals, the phone remains idle for 60 seconds. We conduct a similar experiment to upload data to a remote server.

We extract the energy measurements from the profiler for analysis. We use the time-stamps recorded by NEP to mark the beginning and end of data transfer as well as the beginning and end of the *Ramp time* and the *Tail time*. Then, the energy consumed by each data transfer is computed by approximating the energy under the power-curve between the end of *Ramp time* and the start of *Tail time*.

3.2.2 WiFi

Our WiFi measurements quantify: 1) Energy to scan and associate to an access point and 2) transfer energy. We conduct two sets of measurements. In the first set of measurements, for each data transfer, we first scan for Wifi access points, associate with an available AP and then make the transfer. In the second set of measurements, we only make one scan and association for the entire set of data transfers to isolate the transfer energies.

In addition, all three networks, 3G, GSM and WiFi, incur a maintenance energy, which is the energy used to keep the interface up. We estimate the maintenance energy per second by measuring the total energy consumed to keep the interface up for a time period.

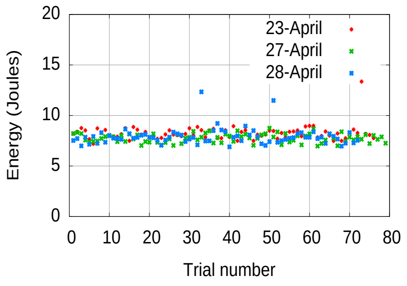
3.2.3 Accounting for idle power

For all measurements, we configure the phone in the lowest power mode and turn off the display and all unused network interfaces. The energy profiler itself consumes a small amount of energy, which we include in the idle power measurement. We measure idle energy by letting the energy profiler run in the background with no other application activity. The average idle power is less than 0.05 W and running the energy profiler at a sampling frequency of 0.25 seconds increases the power to 0.1 W.

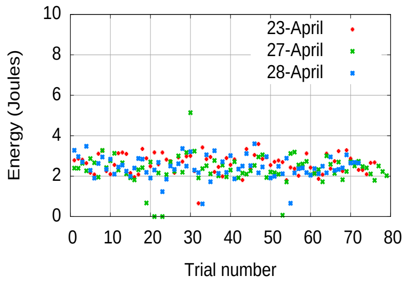
3.3 3G Measurements

Figure 2(a) shows the energy consumption for a typical 50KB download over 3G. We find that the *Tail energy* is more than 60% of the total energy. The *Ramp energy* is significantly small compared to the tail energy, and is only 14% of the total energy. 3G also incurs a maintenance energy to keep the interface on, and is between 1-2 Joules/minute (not shown).

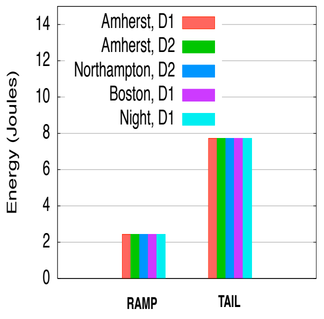
Figure 2(b) shows the average energy consumed for data transfer when the time between successive transfers is varied. We ignore the idle energy consumed when waiting to transfer the next packet. Consider the data points for transferring 100 KB data. The energy increases from 5 Joules to 13 Joules as the time between successive transfers increases from 1 second to 12 seconds. When the time between successive transfers is greater than 12.5 seconds, the energy consumed for 100 KB transfers plateaus at 15 Joules. When the device waits less than the *Tail time* to send the next packet, each data transfer does not incur the total *Tail energy* penalty, reducing the average energy per transfer. This observation suggests that the *Tail energy* can be amortized using multiple transfers, but only if the transfers occur within *Tail time* of each other. We use this crucial observation to design TailEnder, a protocol that reduces the energy consumed by network applications running on mobile phones.



(a) Temporal variation: Tail energy

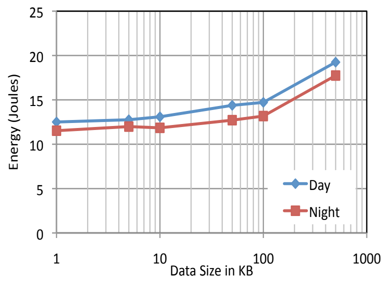


(b) Temporal variation: Ramp energy

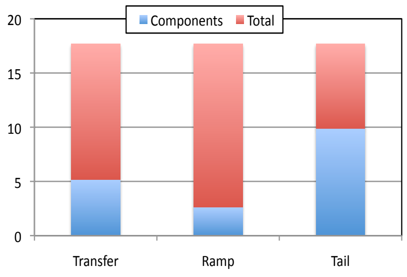


(c) Geographical variation

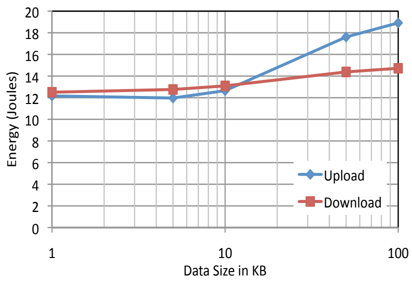
Figure 3: : 3G: 50 Energy measurements on three different days: (a) Tail energy (b) Ramp energy. (c) Tail and Ramp energies measured in different locations with two devices, including a night time measurement.



(a) Download: Day versus Night



(b) Upload measurements



(c) Upload versus Download

Figure 4: : 3G: (a) Comparison of energy consumption of day versus night time transfers. x-axis is in logarithmic scale. (b) Transfer, Tail and Ramp energy for upload experiments (c) Comparison of energy consumption of download versus upload experiments.

3.3.1 Geographical and Temporal variations

We measure the energy consumption across different days and in different geographical regions. The objective of this experiment is two-fold. First, we wish to verify that mobile phones in different cell tower areas are also affected by the *Tail time* overhead. Second, we want to evaluate the consistency of *Tail time*.

Figure 3(a) shows that the *Tail energy* remains consistent across three days. On the other hand, Figure 3(b) shows that the *Ramp energy* is about 2 and 4 Joules for the measurements conducted on different days.

We conducted 3G energy measurements in three different cities, Amherst, Northamption and Boston, in Massachusetts, USA using two different devices, D1 and D2. Figure 3(c) shows that *Tail energy* is consistent across different locations and for two different Nokia devices. The figure also shows that the *Tail energy* and *Ramp energy* do not vary across day (9:00 am to 5:00 pm) and night (8:00 pm to 6:00 am). Based on these measurements, it appears that the *Tail time* or the inactivity timer is configured statically by the network operators and can be inferred empirically. In Section 4, we use the value of the inactivity timer to design TailEnder.

Figure 4(a) compares the total energy consumed for data transfers during the day versus night, averaged over three days. Even though the ramp and tail energies are similar during night and day (shown in Figure 3(c)), the total energy consumed during the night is up to 10% lower than during the day. This is likely due to lower congestion during the night leading to lower transfer energy.

3.3.2 Uploads

Figure 4(b) shows the tail, ramp and transfer energy for upload experiments. As observed in the download experiments, the *Tail energy* consumes more than 55% of the total energy. Figure 4(c) shows that the transfer energy for uploads is higher than downloads for larger data sizes. For example, the transfer energy for upload is nearly 30% more compared to that for downloads when transferring 100 KB. This difference is because upload bandwidths are typically smaller than the download bandwidth.

3.3.3 Mobility

Figure 5 compares energy consumption under mobility within the town of Amherst, MA for 50K data transfers. Mo-

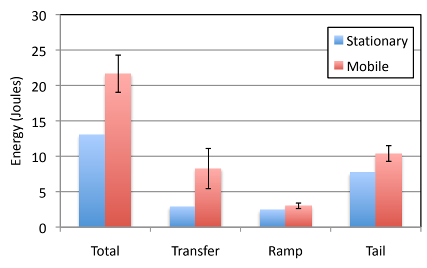
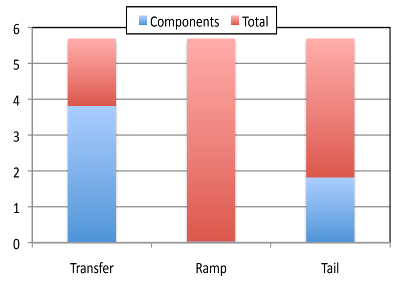


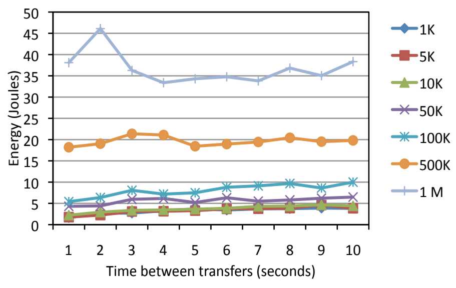
Figure 5: 3G Mobility Measurements: Energy components of 50K 3G transfers with confidence intervals. (Averaged over 35 mobility trials)

bility in outdoor settings affect transfer rates due to factors such as signal strength and hand-offs between cell towers resulting in varying transfer times [17]. Despite the large variances in the transfer energy compared to the stationary measurements, we observe that the *Tail energy* accounts for nearly 50% of the total energy.

3.4 GSM Measurements



(a) GSM: Energy components



(b) GSM: Varying data sizes

Figure 6: : GSM Measurements: (a) Proportional energy consumption for *Ramp energy*, *Tail energy*, and transfer energy. (b) Energy consumption for different data sizes against the inter-transfer time.

We conducted a set of measurements using the two Nokia phones equipped with GSM. Figure 6(a) shows the energy consumption in GSM networks as a proportion of the *Tail energy*, *Ramp energy* and transfer energy for a 50K download. Unlike in 3G, the *Tail energy* only accounts for 30% of the transfer energy. However, similar to 3G, the *Ramp energy* in GSM is small compared to the *Tail energy* and the transfer energy. We also observed that the *Tail time* is 6 seconds and GSM incurs a small maintenance energy between 2-3 J/minute (not shown in figure).

Due to the small *Tail time* in GSM (unlike 3G), data sizes dominate energy consumption rather than the inter-transfer times. Figure 6(b) shows the average energy consumed when varying the time between successive transfers. The average energy does not vary with increasing inter-transfer interval. For example, for data transfers of size 100 KB, the average energy consumption is between 19 Joules to 21 Joules even as the time between successive transfers is varied. In comparison, Figure 2(b) shows that the average energy consumption varies significantly in 3G with varying inter-transfer interval, until the inter-transfer interval grows to more than the *Tail time*.

3.5 Wi-Fi Measurements

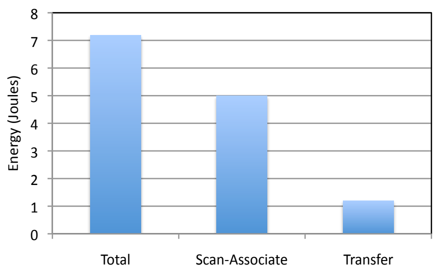
Figure 7(a) shows the energy consumption in WiFi composed of scanning, association and transfer, for a 50 K download. We observe that the scanning and association energy is nearly five times the transfer energy. Our results confirm previous measurements by Rahmati *et al.* [21].

Figure 7(b) shows that for WiFi, energy consumption increases when time between successive transfer increases. Interestingly, energy consumption does not plateau after a threshold inter-transfer interval like in 3G (Figure 2(b)). The reason for increasing energy consumption with increasing inter-transfer interval is the high maintenance energy in WiFi. We measured the maintenance overhead (not shown) for keeping the WiFi interface on to be 3-3.5 Joules per minute.

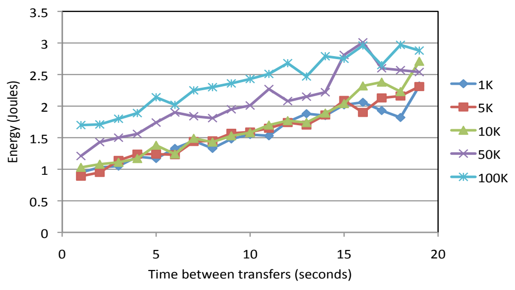
3.6 3G vs GSM vs WiFi

Figure 8 compares the energy consumption of 3G, GSM and WiFi. 3G consumes significantly more energy to transfer data of all sizes (12-20J) compared to GSM and WiFi. GSM consumes 40% to 70% less energy compared to 3G. This is due to two reasons – (1) GSM radios typically operate at a lower power level than 3G radios and (2) the *Tail energy* set for GSM is around 6 seconds, much lower than the 12.5 seconds set for 3G.

Wi-Fi is more energy efficient than both cellular networks once it is connected to an access point (AP). We find that the transfer energy for Wifi grows nearly three times slower compared to the cellular networks. For a transfer of size 10K, Wifi consumes one-sixth of 3G's energy and one-third of GSM's energy. With increasing data sizes Wifi's efficiency increases dramatically. The graph shows that when the cost of scan and transfer is included (marked in the graph as WiFi + SA), WiFi becomes inefficient for small sized transfers compared to GSM (as also observed by Rahmati *et al.* [21]). Surprisingly, when compared to 3G, WiFi is energy efficient even when the cost of scanning and association is included. We exploit this observation in Section 5.2.4.



(a)



(b)

Figure 7: Wifi Measurements: (a) Proportional energy consumption for scan/associate and transfer. (b) Energy consumption for different transfer sizes against the inter-transfer time.

Figures 9(a) and 9(b) is a qualitative comparison of the instantaneous power measurements over 3G and GSM respectively. The measurements corresponds to a single 50K transfer. The lines marked *Uplink* and *Downlink* show the network activities in the uplink and downlink direction respectively (corresponding to request and download). The profiles show both that GSM radios operate at lower power levels and that the device returns to the low power mode in a shorter time in GSM compared to 3G.

Figure 9(c) shows a similar snapshot for a 50K transfer over WiFi. The initial spikes corresponds to energy consumed for scanning and association. The WiFi radio frequently checks the AP for incoming packets, and is shown by the power spikes corresponding to uplink activity after the initial scanning and association is complete. Since we set WiFi to operate in the power-save mode, these checks are terminated after a fixed idle time of 40 seconds.

3.6.1 Energy model

One of the goals of our measurement study is to obtain accurate energy models for energy consumption in 3G, GSM and WiFi. The model enables us to empirically estimate the energy consumption of different applications using the measurement results (Section 5). We model energy consumption as a function of the size and the time between successive transfers. Figure 7(b) and Figure 2(b) show that time between transfers significantly effects energy consumption.

Table 1 shows the energy model we derive from the measurement study. The energy spent to send x bytes of data is consists of three components: 1) *Ramp energy* 2) transmission energy and 3) *Tail energy*. In case of 3G and GSM, we use $R(x)$ to denote the sum of the *Ramp energy* and the transmission energy to send x bytes and E to denote the *Tail energy*. For WiFi, we use $R(x)$ to denote the sum of the transfer energy and the energy for scanning and association, and we set the *Tail energy* E to zero. M denotes the maintenance energy per second and T denotes the *Tail time*. In addition to $R(x)$ and E, the total energy to transmit a packet also depends on the time the interface is on. The last row in Table 1 shows an example computation for the average energy spent to transmit 50K of data with a 20 second interval.

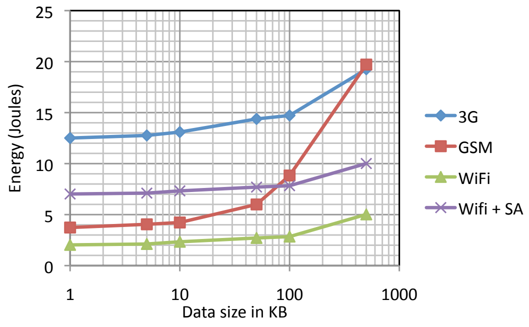


Figure 8: : Energy consumption of transfers of different sizes: WiFi versus cellular networks. x-axis is in logarithmic scale.

3.7 Summary

We summarize our key measurement findings as follows:

1. In 3G, nearly 60% of the energy is *tail energy*, is wasted in high-power states after the completion of a typical transfer. In comparison, the *ramp energy* spent in switching to this high-power state before the transfer is small. The tail and ramp energies can be amortized over frequent successive transfers, but only if the transfers occur within *Tail time* of each other.

2. In GSM, although a similar trend exists, the *tail time* is much smaller compared to 3G (6 vs. 12 secs). Furthermore, the lower data rate of GSM implies that more energy is spent in the actual transfer of data compared to in the tail.

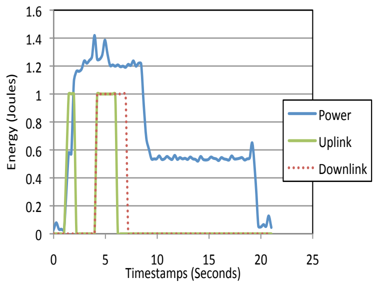
3. In WiFi, the association overhead is comparable to the tail energy of 3G, but the data transfer itself is significantly more efficient than 3G for all transfer sizes. When the scan cost is included, WiFi becomes inefficient for small sized transfers compared to GSM, but is still more energy efficient than 3G.

4. PROTOCOL

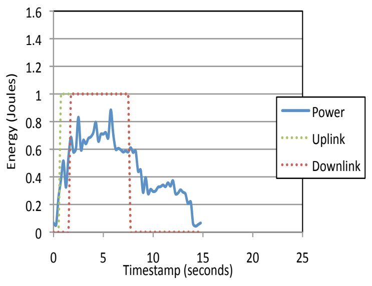
Informed by our measurement-driven model, we develop TailEnder, a protocol whose end-goal is to reduce energy

		3G	GSM	WiFi
Transfer Energy	$R(x)$	$0.025(x) + 3.5$	$0.0357(x) + 1.7$	$0.007(x) + 5.9$
Tail energy	E	0.62 J/sec	0.25 J/sec	NA
Maintenance[4]	M	0.02 J/sec	0.03 J/sec	0.05 J/sec
Tail time	T	12.5 seconds	6 seconds	NA
Energy for 50K (20 second interval)		12.5 J	5.0 J	7.6 J

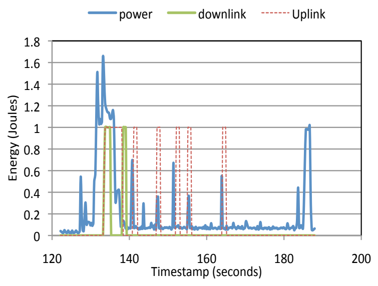
Table 1: Energy Model for data transfers for the different network technologies. All values except the Maintenance values for 3G and GSM, are averaged over more than 50 trials.



(a) 3G: Power Profile - 50K



(b) GSM: Power Profile - 50K



(c) Wifi: Power Profile - 50K

Figure 9: Power profiles of WiFi, 3G and GSM networks

consumption on mobile phones. Common network applications on mobile phones include e-mail, news-feed, software updates and web search and browsing. Many of these applications can be classified under two categories: 1) applications that can tolerate delays, and 2) applications that can benefit from prefetching. E-mail, news-feeds and software updates can be classified as applications that can tolerate a small user-specified delay. For example, a user may be willing to wait for a short time before e-mail and news-feeds are received, if it results in substantial energy savings[5]. Web search and browsing can be classified as applications that benefit from prefetching. Several studies [8, 15, 19] have shown that prefetching web documents can significantly improve search and browsing experience for users.

TailEnder uses two simple techniques to reduce energy consumption for the two different class of applications. For delay tolerant applications, TailEnder schedules transmissions such that the total time spent by the device in the high power state is minimized. For applications that can benefit from prefetching, TailEnder determines the number of documents to prefetch, so that the expected energy savings is maximized.

4.1 Delay tolerant applications

First, we present a simple example to illustrate how applications can exploit delay tolerance to reduce energy utilization. Assume a user sends two emails within a span of a few minutes. The default policy is to send the emails as they arrive, and as a result the device remains in the high power state for two inactivity timer periods. However, if the user can tolerate a few minutes delay in sending the emails, the two emails can be sent together, and the device remains in the high power state for only one inactivity timer period. Our measurement study shows that for low to moderate email size, the second strategy halves the energy consumption.

Several theoretical studies consider the problem of optimally scheduling transmissions to minimize energy, called the *Gap minimization Problem* [10, 12]. Gap minimization problem assumes the following model – a device incurs a large overhead in transitioning from the low power state to the high power state and the goal is to minimize the number of times the device transitions between states. Baptiste *et al.* [10] show that the Gap minimization problem can be solved optimally in polynomial time, under certain simplistic assumptions. Researchers [12, 14, 7] have also designed online algorithms and shown that any online algorithm can at best achieve an optimal competitive ratio of 2.

Our measurement study shows that the Gap minimization energy model is not compatible with that of data transfers on mobile phones: The energy required to transition from the low power to the high power state is not the primary overhead during data transfers; rather a significant overhead is the energy wasted in the high power state after the completion of transmission, due to the inactivity timer settings. Therefore, the goal of TailEnder is to minimize the total time spent in the high power state.

4.1.1 *Scheduling transmissions to minimize energy*

We adapt the Gap minimization problem formulation to our setting as follows – Consider n equal-sized requests,

[5]Commodity phones such as the iPhone explicitly request the user to specify a delay-tolerance limit to improve battery life.

where each request r_i is associated with an arrival time a_i and a deadline d_i by which it needs to be transmitted. When the request r_i is scheduled to be transmitted at time s_i, the radio transitions to the high power state, transfers request r_i instantaneously, and remains in the high power states for T time units, equal to the *Tail time*. Also, based on our measurements, we ignore the energy overhead to switch to the high-power state. Note that when multiple requests are transmitted at the same time, the device is in the high power state only for T time units. Let Z denote the the total time spent in high-power states for a given schedule of requests. The problem is to compute a schedule $s_1, s_2, \cdots, s_n$ that minimizes Z, such that $a_i \leq s_i \leq d_i$.

In practice, we need to solve an online version of the problem, where arrivals and their deadlines are not known in advance.TailEnder uses a simple online algorithm to schedule transmission of an incoming request r_i at time t. The main idea of TailEnder is to transmit a request r_i if either

- the request arrives within a time $x \cdot T$ from the previous deadline d', or
- the request's deadline is reached.

We show that the algorithm provably achieves a competitive ratio of 1.28 compared to the optimal if the value of x is set to 0.57. Further, we show that no deterministic online algorithm can achieve a competitive ratio lower than 1.28. Thus, the TailEnder algorithm achieves the optimal competitive ratio(see Appendix for details). Figure 10 presents the TailEnder algorithm.

An alternate solution to the scheduling problem is to *wait until deadline*, where incoming requests are batched until one of the requests reaches its deadline [9]. It is easy to show that the *wait until deadline* scheme is at best 2-competitive compared to the optimal. Further, the *wait until deadline* scheme can delay requests even if the delay does not provide any energy benefit. For example, if a new request arrives immediately after another request is scheduled, the *wait until deadline* scheme will delay scheduling the new request, even though delay provides no energy benefit.

```
TailEnder scheduler $(t, r_i)$:
  1. set $d' = 0$
  2. if $(t < a_i)$, return
  3. If $(t < d_i)$
      (a) if $(d' + 0.57T < a_i)$, transmit.
      (b) else, add the request to queue Q.
  4. If $(t == d_i)$
      (a) Transmit $r_i$
      (b) Transmit all requests in $Q$ and set $Q = null$
      (c) Set $d' = d_i$
```

Figure 10: TailEnder scheduler that makes a decision about transmitting a request r_i at time t

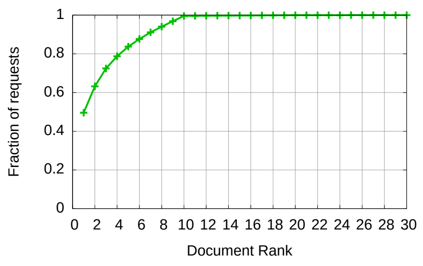


Figure 11: CDF of the fraction of times a user requests a web document at a given rank, for Web search application. The figure shows the CDF over more than 8 million queries collected across several days.

4.2 Applications that benefit from prefetching

Previous work [8] shows that *aggressive* prefetching can reduce delays in networks with intermittent connectivity. However, it is not straightforward to design a prefetching strategy whose end goal is to reduce energy consumption. On the one hand, in the absence of any prefetching, the application needs to fetch the user-requested documents sequentially. If the user think-time is greater than the value of the inactivity timer, then the application incurs a large energy overhead in fetching each document. On the other hand, if the application aggressively prefetches documents and the user does not request any of the prefetched documents, then the application wastes a substantial amount of energy in prefetching. Clearly, user behavior is key in determining the effectiveness of prefetching for reducing energy consumption.

We consider the prefetching problem in the context of Web search and browsing. The information retrieval community and search engine providers collect large amounts of data to study user behavior on the web. Our goal is to exploit the user-behavior statistics to make prefetching decisions. We formulate the prefetching problem as follows: *Given user behavior statistics, how many documents should be prefetched, in order to minimize the expected energy consumption?.*

4.2.1 Maximizing expected energy savings

Figure 11 shows the distribution of web documents that are requested by the user when searching the web. The graph is generated using Microsoft Search logs [6]. The logs contain over 8 million user queries and were collected over a month. Figure 11 shows that 40% of the time, a user requests for the first document presented by the search engine. A user requests for a document ranked 11 or more, less than 0.00001% of the time.

Based on the user-behavior statistics, we estimate the expected energy savings as a function of prefetched documents

[6]obtained from Microsoft Live Labs

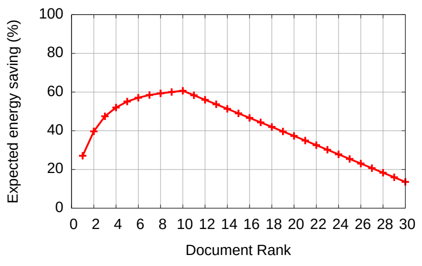


Figure 12: Expected percentage energy savings as a function of the number of documents prefetched.

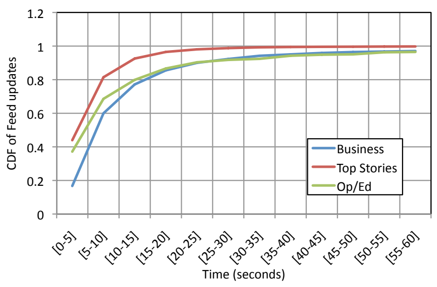


Figure 13: Arrival times distribution for 3 news feed topics.

size. We assume that the user think-times are greater than the value of the inactivity timer. Let k be the number of prefetched documents, prefetched in the decreasing rank order and $p(k)$ be the probability that a user requests a document within rank k. Let E be the *Tail energy*, $R(k)$ be the energy required to receive k documents, and TE be the total energy required to receive a document. TE includes the energy to move from the low to the high power state, the energy to receive a requested document and the *Tail energy*. The expected fraction of energy savings if the top k documents are prefetched is

$$\frac{E \cdot p(k) - R(k)}{TE}$$

Figure 12 shows the expected energy savings for varying k as estimated by Equation 4.2.1. The value of $p(k)$ is obtained from statistics presented in Figure 11, and E, $R(k)$ and TE are obtained from the 3G energy measurements (in Table 1). We set the size of a document to be the average web document size seen in the search logs.

Figure 12 shows that prefetching 10 web documents maximizes the energy saved. When more documents are prefetched, the cost of prefetching is greater than the energy savings. When too few documents are prefetched, the expected energy savings is low since the user may not request a prefetched document. Therefore, for Web search applications, TailEnder prefetches 10 web documents for each user query. In Section 5, we show that TailEnder can save substantial amount of energy when applied to real Web search sessions.

5. EVALUATION

We evaluate TailEnder using a model-driven simulation and real experiments on the phone. The goal of our evaluation is to quantify the reduction in energy utilization when using TailEnder for different applications, when compared to a Default protocol.

To show the general applicability of TailEnder, we evaluate its performance for three applications: emails, news-feeds and Web search. Both email and news-feeds are applications that may be able to tolerate a small delay, while Web search is an application that benefits from prefetching. For all three applications, the impact of TailEnder for energy minimization largely depends on the application traffic and user behavior. For example, if a user receives an email at a periodic interval of once per hour, or if news-feeds are updated once per hour, TailEnder is unlikely to provide energy benefits. In our experiments, we collect real application level traces to evaluate TailEnder.

5.1 Application-level trace collection

To collect traces for evaluating e-mail application, we monitor the mailboxes of 3 graduate students for 10 days and log the size and time-stamps of incoming and outgoing mails. Table 2 tabulates the statistics of the resulting email logs.

	User 1	User 2	User 3
Incoming	446	405	321
Incoming size	214MB	162MB	161MB
Outgoing	219	183	354
Outgoing size	107MB	66MB	178MB

Table 2: Characteristics of collected e-mail

To collect news feed traces, we polled 10 different Yahoo! RSS news feeds[7] once every 5 seconds for a span of 3 days. We log the arrival time and size of each new story or an update to an existing story Table 3 lists the news-feeds we crawled. The traces cover major news topics, both critical (e.g., Business, Politics) and non-critical (e.g., Entertainment). Figure 13 shows the inter-arrival times of updates for three example news topics – *Top stories, Op/Ed and Business*. Updates to the Top stories topic arrive with higher frequency than the Op/Ed and Business topics and nearly 60% of the updates for the Top stories topic arrive with an inter-arrival time of 10–15 seconds.

For Web search, we use the Microsoft Search logs that contains more than 8 million queries sampled over a period of one month. The search log contains the time and the urls that were clicked for each query, as well as the size of the requested document. We extracted a random subset of 1000 queries from this data set. Figure 11 characterizes the distribution of clicks for each query.

[7]http://news.yahoo.com/rss

Feed	Total stories
Opinion/Editorials	507
Health	2177
Technology	4659
Business	5616
Sports	7265
Politics	12069
US News	12389
Entertainment	16757
World	21006
Top Stories	23232

Table 3: News Feeds Collection

5.2 Model-driven evaluation

To quantify the performance of TailEnder for varying parameters and settings, we conduct a trace-driven evaluation using the application traces we collected. We obtain an energy model for per-byte data transmission from our measurement study. We also obtain the *Tail time*, *Tail energy* and *Ramp energy*form our measurement (see Section 3.7). We use the energy values obtained from our Amherst, day time, stationary measurements, shown in Table 1.

The application trace consists of a sequence of arrivals of the form (s_i, a_i), where s_i is the size of the request and the a_i is the time of arrival. For example, for the news feed application, a_i is the time a topic is updated and s_i is the size of the update. Requests could be downloads as in news feeds or uploads as in outgoing emails. The Default protocol schedules transmissions as requests arrive. For delay tolerant applications, TailEnder schedules transmissions using the algorithm shown in Figure 10. For applications that benefit from prefetching, TailEnder schedules transmissions for all prefetched documents. For both protocols, we estimate the energy consumption as a sum of the *Ramp energy* (if the device is not in high power state), transmission energy and the energy consumed because of staying in the high power state after transmission. If a request is scheduled for transmission before the *Tail time* of the previous transmission, the previous transmission does not incur an overhead for the entire *Tail time*.

5.2.1 News-feeds

Figure 14 shows the improvement in energy using TailEnder for each of the news feed topics. We set the deadline for sending the news feeds update to 10 minutes; i.e., a newsfeed content needs to be sent to the user with a maximum delay of 10 minutes since the content was updated. The average improvement across all news feeds is 42%. The largest improvement is observed for the Tech news feed at 52% and the smallest improvement for the Top story news feed at 36%. One possible reason for the top story news feed to yield lower performance improvement is that 60% of the top story updates arrive within 10 seconds, which is the less than the *Tail time* of 12.5 seconds (see Figure 13). Therefore, Default does not incur a *Tail energy* penalty for a large portion of the updates.

Figure 15 and 16 show the expected energy consumption for *business* news feeds using TailEnder and Default for varying deadline settings over 3G and GSM respectively. Figure 15 shows that as deadline increases to 25 minutes(1500 seconds), TailEnder's energy decreases to nearly half of the energy consumption of Default, decreasing from 10 Joules to 5 Joules per update. When sending data over GSM, the energy decreases from 6 Joules to 4 Joules when using TailEnder, compared to Default, yielding a 30% improvement. The improvements of TailEnder over default are smaller in GSM compared to 3G. However, the improvements are substantial when considering the relative proportions of GSM's *Tail energy*and transfer energies.

5.2.2 E-mail

Figure 17 shows the energy reduction using TailEnder as percentage improvement over default, for incoming and outgoing emails for a set deadline of 10 minutes. We obtain an improvement of 35% on average for all three users. We note that we use download energy model for incoming emails and upload energy model for outgoing emails.

Figure 18 and Figure 19 show the energy consumed by TailEnder and Default as the deadline increases, when data is sent on the 3G and GSM networks respectively. This experiment is conducted using the incoming email traces of *User 2*. Increasing the deadline improves energy benefits of TailEnder in both 3G and GSM. Over 3G, TailEnder reduces energy consumption by 40% when the deadline is set to 15 minutes. As before, TailEnder provides a lower energy benefit in GSM compared to 3G. For a 15 minute deadline, TailEnder decreases energy by only 22% when data is sent over GSM.

5.2.3 Web search

Figure 20 shows the energy improvement using TailEnder for Web search application, when sending data over 3G and GSM. For Web search, TailEnder prefetches the top 10 documents for each requested query. Default only fetches documents that are requested by the user. TailEnder reduces energy by nearly 40% when data is sent over 3G and by about 16% when data is sent over GSM.

To understand the distribution of energy savings per query, we plot the CDF of the energy improvement in Figure 21. The plot shows that about 2% of the queries see little energy improvement. TailEnder reduces energy for 80% of the queries by 25–33%. For the remaining 18% of the queries, TailEnder reduces energy by over 40%. We find that the 18% of the queries that benefits most by TailEnder's prefetching are queries for which the user requested 3 or more documents.

5.2.4 Using WiFi for energy savings

Figures 22, 23 and 24 show how utilizing WiFi availability can reduce energy consumption for news feed, email and

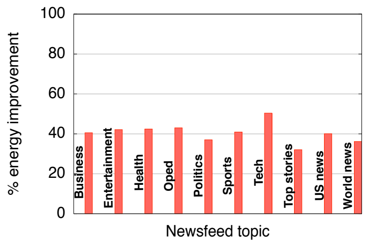


Figure 14: News feed: Energy improvement in using TailEnder for different news feed topics

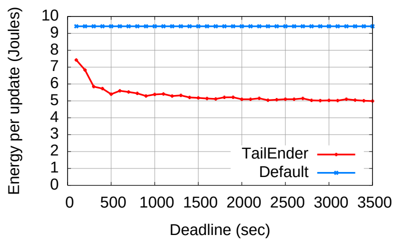


Figure 15: Newsfeed: The energy consumption of TailEnder and Default for varying deadline over 3G

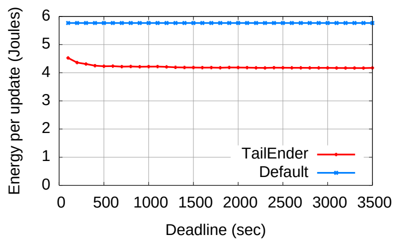


Figure 16: News feed: The energy consumption of TailEnder and Default for varying deadline over GSM

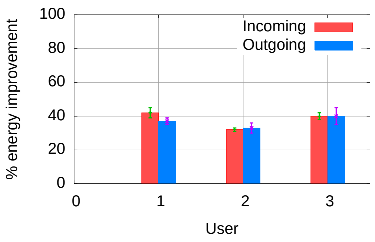


Figure 17: Email: Energy improvement using TailEnder for email application

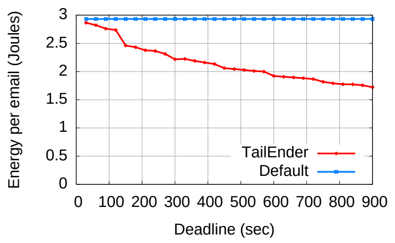


Figure 18: Email: The energy consumption of TailEnder and Default for varying deadline over 3G

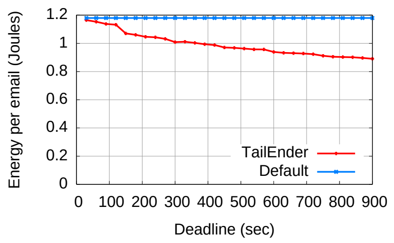


Figure 19: Email: The energy consumption of TailEnder and Default for varying deadline over GSM

Web search applications respectively. We assume that the WiFi interface is switched on only when WiFi is available, to avoid unnecessary scanning. Related work [21] show that WiFi availability can be predicted accurately, especially in stationary scenarios.

Keeping the WiFi interface on incurs a maintenance energy, as we observe in our measurement study (see Table 1). Therefore, our algorithm switches the WiFi interface off x seconds after a transfer if no data packet arrives. We estimate x as the ratio of the energy required for scanning/association and the per-second maintenance energy.

When WiFi is always available, the energy consumption is 10 times lower compared to Default and more than 4 times lower compared to TailEnder for all three applications. Even when WiFi is available only 50% of the time, sending data over WiFi reduces energy consumption by 3 times compared to Default for all three applications. Previous works [21] do not observe such substantial energy savings when using WiFi to augment cellular networks. The primary reason is that previous studies are based on the GSM network. However, 3G consumes more energy than WiFi, even when accounting for the WiFi scan and association costs. Therefore, opportunistic usage of WiFi can provide substantial energy benefits.

5.3 Experiments on the mobile phone

We conduct data transfer experiments on the phone using application-level traces. We convert an application trace into a sequence of transfers $S = \{<s_1, a_1>, <s_2, a_2>, \cdots, <s_n, a_n>\}$, such that data of size s_i is downloaded by the mobile phone at time a_i. Then, from a fully charged state, we repeatedly run this sequence of transfers until the battery drains completely.

We run two sequences of transfer, one generated by TailEnder and the other by Default. Given an application trace, TailEnder schedules the transfers according to whether the application is delay tolerant or can benefit from prefetching. Default schedules transfers as they arrive. We conduct the experiments for two applications: downloading Tech news feeds and Web search. For the news feed application, the metric is the number of stories downloaded and for Web search the metric is the number of queries for which all user requested documents were delivered.

Table 4 shows the results of the news feeds experiment. TailEnder downloads more than 60% news feed updates compared to Default, and the total size of data downloaded by protocol increases from 127 MB to 240 MB providing a 47% improvement. Our model-based evaluation showed that for the Tech news feed, TailEnder reduces energy by 52% compared to Default.

	Default	TailEnder
Stories	1411	3900
Total transfer size	127 MB	240 MB

Table 4: News feeds experiment. TailEnder downloads more than twice as many news feeds compared to Default

Table 5 shows results for the Web search experiment. By prefetching, TailEnder sends responses to 50% more queries

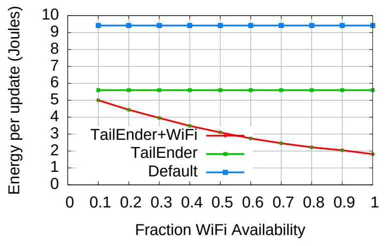


Figure 22: News feed. Vertical bars show 95% confidence interval

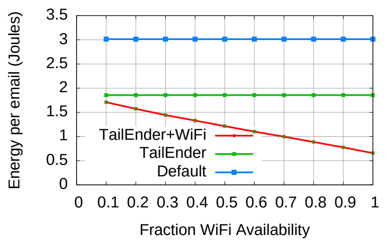


Figure 23: E-mail. Vertical bars show 95% confidence interval

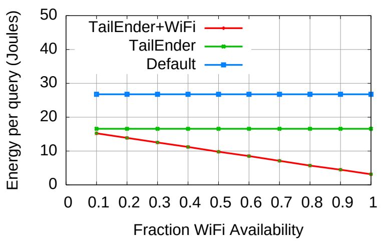


Figure 24: Web Search. Vertical bars show 95% confidence interval

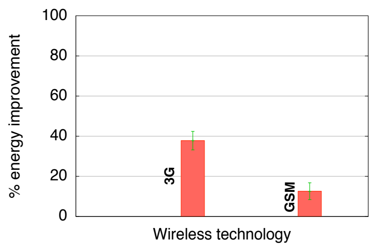


Figure 20: Web search: Energy improvement using TailEnder for Web search over 3G and GSM

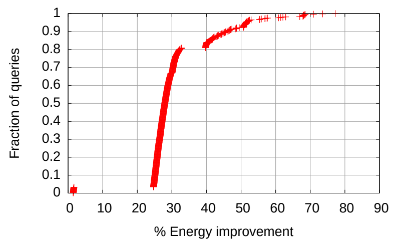


Figure 21: Web search: CDF of the Energy improvement using TailEnder for each query

for the same amount of energy and the average number of transfers decreases by 45%. Prefetching is energy efficient, even though it sends ten times more data for each transfer on an average. Our model-driven evaluation showed that for Web search, TailEnder decreases energy consumption by 40% compared to Default.

	Default	TailEnder
Queries	672	1011
Documents	864	10110
Transfers	1462	1011
Average transfers per query	9.3K	147.5K

Table 5: Web search experiment. TailEnder downloads 50% more queries compared to Default

6. CONCLUSIONS

Energy on mobile phones is a precious resource. As phones equipped with multiple wireless technologies such as 3G, GSM, and WiFi become commonplace, it is important to understand their relative energy consumption characteristics. To this end, we conducted a detailed measurement study and found a significant tail energy overhead in 3G and GSM. We developed a measurement-driven model of energy consumption of network activity for each technology.

Informed by the model, we develop TailEnder, a protocol that provably minimizes energy consumption while meeting user-specified delay-tolerance deadlines. For applications that can benefit from prefetching, TailEnder aggressively prefetches data, including potentially useless data, and yet reduces the overall energy consumed. We evaluate the performance of TailEnder for three case study applications—email, news feeds, and web search—based on real user logs and find significant savings in energy in each case. Experiments conducted on the mobile phone shows that TailEnder can download 60% more news feed updates and download search results for more than 50% of web queries, compared to using the default policy. Our model-driven simulation shows that TailEnder can reduce energy by 35% for email applications, 52% for news feeds and 40% for web search.

APPENDIX

Here we provide optimality analysis for TailEnder. We first prove that any online algorithm can at most be 1.28 competitive with an optimal offline algorithm. Next, we prove that the online algorithm TailEnder(SCHED) we describe in Figure 10 achieves this competitive ratio;

Let OPT be an optimal adversary that has complete knowledge of packet arrivals. OPT schedules transmissions to minimize Z, the time spent by the mobile device in the high power state. Let T be the *Tail time*.

THEOREM 1. *Any online algorithm ALG can at most be* 1.28*-competitive with the offline adversary, OPT, with respect to the time spent in the high energy state.*

PROOF. We prove the theorem by constructing the offline adversary, OPT, that incrementally generates new requests after observing the actions of ALG. OPT generates a new request at time $x \cdot T$ after the ALG schedules a previous request. We show that $\frac{Z(ALG)}{Z(OPT)}$ is maximized when x is set to 0.57; in other words, when OPT generates a new request at $0.57T$ after ALG schedules a request, OPT can force the ALG schedule to remain in the high power state for the longest time compared to its own schedule.

Let the first request be $r_1 = (a_1, d_1)$ and let both ALG and OPT schedule to transmit r_1 at time s_1. As a first step, OPT generates a second request $r_2 = (a_2, d_2)$, where $a_2 = s_1 + x \cdot T$ and $d_2 \gg T$.

ALG can schedule r_2 in the following ways:

Choice 1: If ALG schedules r_2 at a_2, ALG remains in the high power state for $x \cdot T$ time units. In response, OPT generates a third request $r_3 = (a_3, d_3)$ such that $a_3 = d_2$. OPT schedules the transmission of r_2 and r_3 at d_2. Therefore $Z(OPT) = 2T$. ALG is forced to schedule r_3 at some time between a_3 and d_3, incurring an overhead of T. In total, $Z(ALG) = (2 + x) \cdot T$ and $Z(OPT) = 2T$.

Choice 2: If ALG schedules r_2 at d_2, OPT schedules r_2 at a_2 and generates no more requests. Therefore $Z(ALG) = 2T$ while $Z(OPT) = (1 + x) \cdot T$.

Choice 3: If ALG schedules r_2 at time s_2, where $a_2 < s_2 < d_2$, OPT generates a third request $r_3 = (a_3, d_3)$ such that $a_3 = d_2$ (as in *Choice 1*). Thus, using the same argument as *Choice 1*, $Z(OPT) = 2T$, while $Z(ALG) \geq (2 + x) \cdot T$.

The competitive ratio for the different choices is

$$\begin{aligned} \frac{Z(ALG)}{Z(OPT)} &= \frac{2+x}{2} (\textit{Choices 1 \& 3}) \\ &= \frac{2}{(1+x)} (\textit{Choice 2}) \end{aligned} \tag{1}$$

The lower bound of the competitive ratio is $\mathit{min}(\frac{2+x}{2}, \frac{2}{(1+x)})$. We compute the value of x that maximizes the lower bound of the competitive ratio by solving:

$$\frac{2+x}{2} = \frac{2}{(1+x)} \tag{2}$$

Solving Eq. 2, we get $x = 0.57$ and plugging it in Eq. 1, we get $\frac{Z(ALG)}{Z(OPT)} = 1.28$; i.e., when OPT introduces a new request at time $0.57T$ after the previous schedule, ALG is forced to be in the high energy state 1.28 times longer than OPT for any choice it makes. If $x > 0.57$, then ALG can schedule according to *Choice 2* and reduce the competitive ratio to less than 1.28. Similarly, if $x < 0.57$, then ALG can schedule according to *Choice 1* and reduce the competitive ratio.

In summary, OPT can generate new requests such that ALG is forced to be in the high energy state 1.28 times longer than OPT. Therefore any online algorithm ALG can at most be 1.28 competitive with an offline adversary.

□

THEOREM 2. *SCHED is* 1.28*-competitive with the offline adversary OPT with respect to the time spent in the high energy state.*

PROOF. Let the set of incoming requests be $r_1, r_2 \cdots r_m$.

Construction: We first construct a sequence $S = \{a'_1, d'_1, a'_2, d'_2 \cdots\}$ as follows. a'_i is the arrival time of a request r_i that occurs after the previous deadline d'_{i-1} such that, of all arrivals that occur after d'_{i-1}, a'_i has the earliest deadline. The deadline for r_i is d'_{i+1}. By definition, the sequence S is an alternating sequence of a'_i and d'_i. Figure 25 shows an example construction.

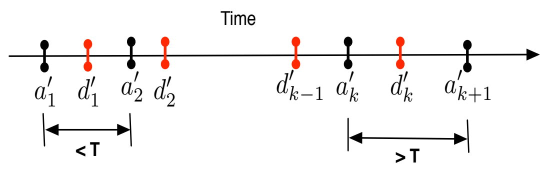


Figure 25: Construction for Theorem 2.

Let a'_k be the first arrival time, such that a'_{k-1} occurs more than T time units after a'_k. If no such request exists, then we set a'_k to be the last arrival time in the sequence.

It is easy to show that both SCHED and OPT schedule the first request at its deadline. For simplicity, let the first request be scheduled at time 0.

Claim 1: $Z(OPT) \geq a'_k + T$

Since an arrival a'_j is at most T time units from the subsequent arrival a'_{j+1} for $j < k$, the interspersed deadlines are also at most T time units apart. The first request is scheduled at its deadline. Therefore, the schedule of the first request and the deadline of the second request are at most T time units apart, and so on. This implies that the device remains in the high power state at least until a'_k. The request that arrives at a'_k is scheduled after its arrival, and remains in the high power state for at least T time units. Therefore, $Z(OPT) \geq a'_k + T$.

Claim 2: $Z(SCHED) \leq a'_k + 1.43T$.

Two cases can occur.

Case 1: If $a'_k < d'_{k-1} + 0.57T$, SCHED transmits the request r_k, and $Z(SCHED) = a'_k + T$

Case 2: If $a'_k > d'_{k-1} + 0.57T$, then SCHED transfers request r_k at its deadline d'_{k-1}, and remains in the high energy state for at most T time units. SCHED schedules the first $k-1$ requests and incurs an overhead of at most $a'_k + 0.43T$. Note that the latest that SCHED can schedule request r_{k-1} is at its deadline d'_{k-1}. Since, a'_k is at a distance of at least 0.57T from d'_{k-1}, the device remains in the high power state for at most 0.43T time units after a'_k. Adding together, we get $Z(SCHED) \leq a'_k + 1.43T$.

If *Case 1* occurs $Z(OPT) = Z(SCHED)$.

If *Case 2* occurs, note that $a'_k \geq 0.57T$, since a'_k is at least 0.57T time units from d'_{k-1}. Therefore, $Z(SCHED) \leq 0.57T + 1.43T$ and $Z(OPT) \geq 0.57T + T$ and $\frac{Z(SCHED)}{Z(OPT)}$ is at most 1.28. We can repeat this argument for requests from $r_{k+1}, \cdots r_m$.

Therefore, SCHED is 1.28 competitive with OPT. □

A. REFERENCES

[1] 3g: Wikipedia. http://en.wikipedia.org/wiki/3G.

[2] International telecommunication union press release.

http://www.itu.int/newsroom/press_releases/2008/29.html.

[3] Third generation partnership project 2 (3gpp2). http://www.3gpp2.org.

[4] Third generation partnership project (3gpp). http://www.3gpp.org.

[5] Y. Agarwal, R. Chandra, A. Wolman, P. Bahl, K. Chin, and R. Gupta. Wireless wakeups revisited: energy management for voip over wi-fi smartphones. In *MobiSys '07: Proceedings of the 5th international conference on Mobile systems, applications and services*, pages 179–191, New York, NY, USA, 2007. ACM.

[6] T. Armstrong, O. Trescases, C. Amza, and E. de Lara. Efficient and transparent dynamic content updates for mobile clients. In *MobiSys '06: Proceedings of the 4th international conference on Mobile systems, applications and services*, pages 56–68, New York, NY, USA, 2006. ACM.

[7] J. Augustine, S. Irani, and C. Swamy. Optimal power-down strategies. In *FOCS '04: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 530–539, Washington, DC, USA, 2004. IEEE Computer Society.

[8] A. Balasubramanian, B. N. Levine, and A. Venkataramani. Enabling Interactive Applications in Hybrid Networks. In *Proc. ACM Mobicom*, September 2008.

[9] N. Banerjee, J. Sorber, M. D. Corner, S. Rollins, and D. Ganesan. Triage: balancing energy and quality of service in a microserver. In *MobiSys '07: Proceedings of the 5th international conference on Mobile systems, applications and services*, pages 152–164, New York, NY, USA, 2007. ACM.

[10] P. Baptiste. Scheduling unit tasks to minimize the number of idle periods: a polynomial time algorithm for offline dynamic power management. In *SODA '06: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 364–367, New York, NY, USA, 2006. ACM.

[11] X. Chuah, M.; Wei Luo; Zhang. Impacts of inactivity timer values on umts system capacity. In *Wireless Communications and Networking Conference (2002)*, volume 2, pages 897–903. IEEE, 2002.

[12] E. D. Demaine, M. Ghodsi, M. T. Hajiaghayi, A. S. Sayedi-Roshkhar, and M. Zadimoghaddam. Scheduling to minimize gaps and power consumption. In *SPAA '07: Proceedings of the nineteenth annual ACM symposium on Parallel algorithms and architectures*, pages 46–54, New York, NY, USA, 2007. ACM.

[13] A. Gupta and P. Mohapatra. Energy consumption and conservation in wifi based phones: A measurement-based study. In *Sensor and Ad Hoc Communications and Networks (SECON)*, pages 121–131, Washington, DC, USA, 2007. IEEE Computer Society.

[14] S. Irani, S. Shukla, and R. Gupta. Algorithms for power savings. In *SODA '03: Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 37–46, Philadelphia, PA, USA, 2003. Society for Industrial and Applied Mathematics.

[15] Z. Jiang and L. Kleinrock. Web prefetching in a mobile environment. In *IEEE Personal Communications*, volume 5, pages 25–34, September, 1998.

[16] C.-C. Lee, J.-H. Yeh, and J.-C. Chen. Impact of inactivity timer on energy consumption in wcdma and cdma2000. In *Proceedings of the Third Annual Wireless Telecommunication Symposium (WTS)*. IEEE, 2004.

[17] X. Liu, A. Sridharan, S. Machiraju, M. Seshadri, and H. Zang. Experiences in a 3g network: interplay between the wireless channel and applications. In *MobiCom '08: Proceedings of the 14th ACM international conference on Mobile computing and networking*, pages 211–222, New York, NY, USA, 2008. ACM.

[18] J. Nurminen, J.K.; Noyranen. Energy-consumption in mobile peer-to-peer - quantitative results from file sharing. In *Consumer Communications and Networking Conference (CCNC)*, pages 729–733, Washington, DC, USA, 2008. IEEE Computer Society.

[19] V. Padmanabhan and J. Mogul. Using Predictive Prefetching to Improve World Wide Web Latency. In *Proc. ACM Sigcomm*, pages 22–36, July 1996.

[20] T. Pering, Y. Agarwal, R. Gupta, and R. Want. Coolspots: Reducing the power consumption of wireless mobile devices with multiple radio interfaces. In *MobiSys 2006: Proceedings of the 4th international conference on Mobile systems, applications and services*, pages 220–232, New York, NY, USA, June 2006. ACM Press.

[21] A. Rahmati and L. Zhong. Context-for-wireless: context-sensitive energy-efficient wireless data transfer. In *MobiSys '07: Proceedings of the 5th international conference on Mobile systems, applications and services*, pages 165–178, New York, NY, USA, 2007. ACM.

[22] Y. Xiao, R. S. Kalyanaraman, and A. Yla-Jaaski. Energy consumption of mobile youtube: Quantitative measurement and analysis. In *NGMAST '08: Proceedings of the 2008 The Second International Conference on Next Generation Mobile Applications, Services, and Technologies*, pages 61–69, Washington, DC, USA, 2008. IEEE Computer Society.

[23] J.-H. Yeh, J.-C. Chen, and C.-C. Lee. Comparative analysis of energy-saving techniques in 3gpp and 3gp2 systems. In *Transactions on Vehicular Technology*, volume 58, pages 432–448. IEEE, 2009.