Cool-Tether: Energy Efficient On-the-fly WiFi Hot-spots using Mobile Phones

Ashish Sharma* [1], Vishnu Navda‡, Ramachandran Ramjee‡,
Venkata N. Padmanabhan‡, Elizabeth M. Belding*

*University of California, Santa Barbara
{asharma, ebelding}@cs.ucsb.edu

‡Microsoft Research India
{navda, ramjee, padmanab}@microsoft.com

ABSTRACT

We consider the problem of providing ubiquitous yet affordable Internet connectivity to devices at home, at work, and on the move. In this context, we take advantage of two significant technology trends: the commoditization of WiFi WLAN technology and the rapid growth of cellular data services. We propose an architecture called Cool-Tether that harnesses the cellular radio links of one or more mobile smartphones in the vicinity, builds a WiFi hotspot on-the-fly, and provides energy-efficient, affordable connectivity.

Prior approaches to supporting such a *tethered* mode operation have relied on the WiFi ad hoc mode, which impedes the key goal of conserving battery energy on mobile phones. To address the challenges of energy efficiency, Cool-Tether carefully optimizes the energy drain of the WAN (GPRS/ EDGE/ 3G) and WiFi radios on smartphones. In particular, Cool-Tether employs a cloud-based gatherer and an energy-aware striper that exploit the unique energy characteristics of the WAN radio. Cool-Tether also uses a novel reverse-infrastructure mode for WiFi, where the client host serves as a WiFi access point while the mobile phone gateway serves as a WiFi client. We prototype Cool-Tether on smartphones and, experimentally demonstrate savings in energy consumption between 38%-71% compared to prior energy-agnostic solutions.

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: Wireless communication; C.2.1 [**Network Architecture and Design**]: Energy Efficiency

General Terms

Design, Experimentation, Measurement, Performance

Keywords

Energy Efficient, Web browsing, Mobile phones, Tethering, 3G, WiFi Hotspot, Wireless

[1]The author was an intern at Microsoft Research India during the course of this work.

CoNEXT'09, December 1–4, 2009, Rome, Italy.

1. INTRODUCTION

Mobile phones have evolved from merely being phones to full-fledged computing and communications devices, as embodied by smartphones. Driven by the near-ubiquitous worldwide reach of the cellular infrastructure and the phenomenal growth in the number of cellular subscribers, smartphones are rapidly evolving into users' primary Internet access device. In fact, the number of wireless data subscribers in some countries already exceeds the number of wireline data subscribers. For example, in India as of June 2009, there were about 127 million wireless Internet subscribers but only 14 million wireline Internet subscribers [1].

As more and more users access the Internet on their phones, it may be difficult to justify an additional broadband subscription for their home computers, especially in the emerging markets of the world. Even in developed countries where users subscribe to wireless Internet access on their smartphones as well as wireline Internet access for their home computers, users must rely on the spotty availability of WiFi hotspots while on the go. It is in situations like this, where multiple users, such as family members or colleagues, do not have access to a wired or WiFi Internet connection at home or while traveling, that a system like Cool-Tether would have the most utility, by allowing users to create high-speed on-the-fly WiFi hot-spots using multiple smartphones.

Cool-Tether is an architecture that provides *energy-efficient*, affordable connectivity by leveraging one or more WiFi-equipped and Internet-enabled smartphones. The key idea is to harness the available smartphones to build an on-the-fly WiFi hotspot that is both energy-efficient and easy to use. Using Cool-Tether, a user's laptop, at home or on the move, can obtain Internet connectivity via the user's smartphone, thereby avoiding the need for a separate wide-area (WAN) connection and the attendant subscription costs. Smartphones are a natural fit for serving as a communication gateway for other devices, given that they are typically equipped with both local-area radios (e.g., Bluetooth or WiFi) and wide-area radios (e.g., GPRS, EDGE, 3G).

A common solution adopted today is to use the *tethered mode* operation of mobile phones, allowing a dedicated phone to be used as a modem to provide connectivity to another device. This typically involves making a wired connection (e.g., using USB) or to use WiFi in ad hoc mode or Bluetooth to connect the client to the smartphone gateway. However, neither of these approaches are satisfactory. The WiFi ad hoc mode solution is not designed to be energy efficient and ends up draining the battery of the smartphone very quickly. The Bluetooth solution incurs a high latency

for discovery and connection setup and is less energy efficient than WiFi for bursty data traffic such as Web browsing (Section 3.1). The USB cable solution is not convenient and does not support the use of more than one gateway device or smartphone, whereas, in many situations, more than one smartphone is available for use as a gateway (Section 3.1).

To address the two key challenges of energy efficiency and multiple gateway support, Cool-Tether employs several novel techniques that focus on optimizing the energy drain of the WAN and WiFi radios on the smartphones. In the case of GPRS/EDGE and 3G, we identify *two unique energy characteristics of the WAN radio* and design solutions that leverage these characteristics (Section 5). First, we find that the WAN radio exhibits an energy tail that translates into a significant fixed setup energy cost for using a phone as a gateway. Cool-Tether exploits this knowledge by first computing the optimal number of gateway phones to use for a given workload before performing energy-aware striping of data over the selected gateway smartphones. Second, we find that in both GPRS/EDGE and 3G the radio has a non-linear energy profile that necessitates a bursty mode of operation for higher energy efficiency. Thus, Cool-Tether employs a proxy in the cloud that first gathers all necessary data before commencing a bursty transmission over the WAN link. The key insight is that for maximum energy efficiency, the radio should be used for as long a burst as can be sustained at the full data rate.

In the case of WiFi, Cool-Tether adopts a novel *reverse-infrastructure mode* of operation to accomplish tethering. In contrast to the typical WiFi infrastructure setting, where the gateway device serves as the access point (AP), the gateway device (smartphone) plays the role of a WiFi client and associates with the laptop client that acts as a WiFi Access Point in order to establish the tethering connection. Compared to either WiFi's infrastructure mode or ad hoc mode, the reverse-infrastructure mode used by Cool-Tether offers greater energy efficiency since it allows the gateways (i.e., smartphones) to put their WiFi interfaces to sleep more effectively when not in use.

In this paper, we demonstrate the feasibility of a smartphone-based architecture for Internet access and make the following specific contributions:

1. We identify unique energy characteristics of WAN radios and present a proxy design that employs bursty transmissions and energy-aware striping algorithms to exploit these characteristics and optimize the energy drain on mobile phone gateways.

2. We present a novel reverse-infrastructure WiFi mode in order to tether laptop clients to multiple phone gateway devices in an energy-efficient manner.

3. Using an implementation of Cool-Tether on laptops, Windows Mobile smartphones and proxies running on servers (Section 6), we show through extensive experiments (Section 7) that our architecture delivers 38%-71% savings in energy for a web access workload compared to COMBINE, a recently proposed energy-agnostic system for wireless collaborative downloading [3].

Finally, note that while the Cool-Tether system design and implementation in this paper is focused primarily on energy-efficient support for *web downloads*, the gather/striper algorithms can be implemented at a lower layer in the stack, for example the transport layer, in order to provide broader support for all application types.

2. RELATED WORK

Related work falls under two broad categories, viz., optimizing information access on mobile devices over wireless links and techniques for improving access performance by striping data over multiple paths.

The Coda [13] project and the related Odyssey platform [16] pioneered the work on supporting mobile information access over wireless links. These systems were designed to address the challenges of intermittent connectivity and limited bandwidth of wireless links and made extensive use of novel optimistic concurrency control and adaptation techniques. The Rover [11] toolkit proposed the use of relocatable dynamic objects and queued RPC in order to enhance the performance of mobile web browsing and other applications over intermittently connected wireless links. The WebExpress [8] system specifically focused on improving HTTP performance over wireless links and proposed novel differencing and caching optimizations. The Mowgli [14] architecture targeted improving web performance over GSM-based circuit data connections (8Kbps links) and proposed a custom MHTTP protocol that performs data/header compression and image transcoding to reduce bandwidth usage. GPRSWeb [6] built on many of the early proposals and was designed to comprehensively address the challenges of accessing the web from a mobile device using an always-on GPRS link. The authors proposed a proxy-based solution that includes caching, parsing-and-pushing of embedded web objects, compression and a UDP-based transport for improving web access performance.

While the above research efforts have focused on optimizing the use of a single wireless access link, other efforts have focused on improving download performance by using multiple wireless links [3, 5, 9, 12, 15, 18, 19, 21, 22]. Proposals such as [9, 18, 22] address TCP specific problems, such as packet reordering, that occur while using multiple wireless links. The MAR [19] commuter router system is a gateway device with multiple wide-area wireless links that provides Internet access in moving vehicles. These solutions are targeted towards multi-homed devices and, thus, do not address issues of locating and utilizing gateways in an energy-efficient manner. Other systems such as [3, 5, 12, 15, 21] have proposed solutions that take advantage of wireless links from multiple devices in order to enhance performance of a given client. However, these systems have focused mainly on the performance benefits of using multiple gateways and do not take into account energy efficiency. The COMBINE [3] system is the closest to Cool-Tether in terms of the setting. While COMBINE is focused on improving download performance by using collaborative downloading over GPRS links of multiple mobile smartphones in the vicinity, the primary focus of COMBINE is performance and not energy-efficiency. In this paper, we use COMBINE as our baseline system for comparison.

In summary, while many of the techniques proposed in the literature are complementary and can be used to enhance our solution, none of these techniques directly addresses the problem space that is the focus of this paper, namely, the design of an *energy-efficient* system that uses *wireless WAN links* of *multiple mobile phones* serving as gateways for im-

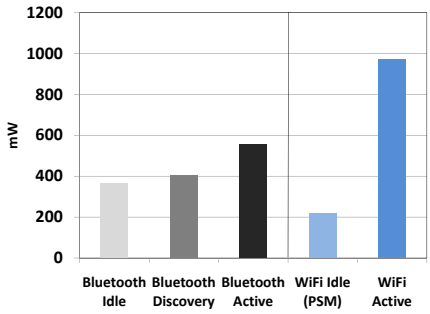


Figure 1: Power consumption of Bluetooth and WiFi radios in different states.

proving web access performance. In particular, we believe that we are the first to identify and design energy efficient transport techniques that leverage the unique energy characteristics of WAN link radios.

3. PROBLEM CONTEXT

The problem context we consider is as follows. One or more client devices seek Internet access. Such devices could be netbooks, laptops, PCs, etc., although for ease of exposition, we term them all as "laptops" in the discussion that follows.

In their vicinity are one or more mobile smartphones, each of which includes both a local-area radio, specifically WiFi, and a wide-area radio such as EDGE/GPRS or 3G. These smartphones are available for opportunistic use by the laptops, as gateways for Internet access. However, to avoid interfering with the owner's use of their device, a smartphone is considered to be available for use as a gateway only when it is turned on but idle. Further, in order to minimize disruption to the owner's future usage of their device, the primary focus of Cool-Tether is minimizing energy consumption on the smartphones while they serve as opportunistic Internet access gateways for other clients.

3.1 Motivation

In this section, we motivate a few aspects of our problem setting. The smartphone gateways are typically on battery power (e.g., when in a user's pocket), although they might be plugged in to a charger on occasion (e.g., when in the user's home or office). Likewise, the laptop clients might also be on battery. However, in the case of smartphones, unlike laptops, the radio energy consumption dominates the overall energy consumption. For example, in the HP iPAQ 6965 smartphone, our measurements indicate that the energy consumption of the base device ranges in 200-700mW depending on the intensity of the backlight while the base power of a laptop class device is on the order of 20W. In comparison, for both devices, the WiFi radio consumes between 1-2W while transmitting. Thus, it is critical to efficiently use the WiFi and WAN radios of a smartphone, the primary motivation behind Cool-Tether.

It is well-known that WiFi exhibits a superior energy-per-bit performance compared to alternatives such as Bluetooth [17]. To validate this observation, we measured the power consumption on iPAQ smartphone (details of the setup are in Section 7). Figure 1 shows the average power consumed by Bluetooth 1.2 and WiFi in different states without the base cost of the device. In terms of energy per bit performance, WiFi at 11Mbps achieved 0.2mW/Kbps compared to 1.24mW/Kbps for Bluetooth. Since Bluetooth has lower active cost, it is suitable for low bandwidth application such as VoIP, but for bursty traffic such as web browsing, WiFi is a clear winner.

Cool-Tether optimizes smartphone energy consumption for the common case of a single smartphone gateway serving a single client. However, our work is also motivated by collaboration in settings such as a family or a group of friends, where multiple smartphones are available to serve as gateways. Such scenarios are not uncommon, as various studies on human mobility patterns have shown. For example, human contact traces [10] collected from different parts of the world and in different settings (conferences, colleges, etc.) show that in 60-70% of cases where two people come in contact, the contact duration lasts for 1000 seconds or more. These contact durations provide ample opportunities for sharing and aggregation of the limited bandwidth available on WAN radios.

Note that, in this paper, we do not consider the issues of incentives for the owner of a smartphone to make their device available to serve as a gateway nor do we consider the privacy implications of traffic routed through the smartphone. While these are important questions, we can lean on existing work and techniques to address them, e.g., using energy accounting for incentives [3] and SSL-based end-to-end encryption for privacy. Furthermore, in settings such as a family or a group of friends, incentives and privacy considerations may not be as important.

3.2 Workload Model

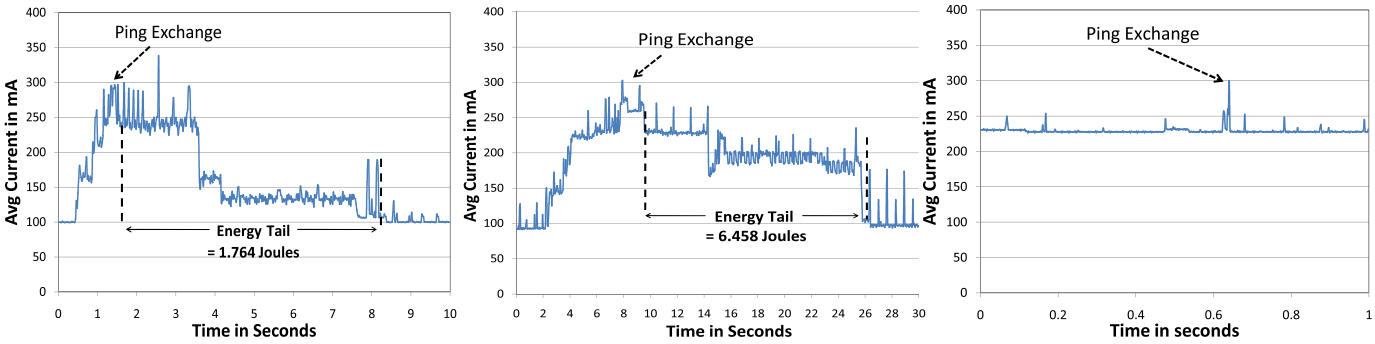
We consider a web access workload, wherein a client accesses multiple web pages in a session. A web page access involves downloading the base HTML page and the embedded objects. There is a gap, called the think time, between the completion of one page download and the initiation of the next download.

The above model corresponds to the closed loop model for web traffic that has been proposed in the literature [20] and is indeed the basis of synthetic web traffic generators such as SURGE [4]. It also is in agreement with intuition: the time at which the user initiates the download of a page would likely depend on when the download of the previous page completed rather than being anchored at a fixed point in time. For example, the download of a new page may result from perusing and following a hyperlink on the previously downloaded page.

3.3 Metrics

The primary metric we use in our evaluation is the *overall energy cost* of a session, which is defined as the sum of the battery drain on all of the smartphone gateways, from the start of the first download in the session to the end of the last download. The time period includes both the download time for the individual web pages in the session and the intervening think times. Also, the metric includes the battery drain on *all* of the available smartphone gateways, whether any traffic was routed through them or not. This reflects the observation that the battery drain on even an unused potential gateway is an opportunity cost, for if the gateway had been used, the session may have completed sooner, with a lower overall energy cost.

A second metric we consider is the *session completion time*, which is defined as the duration from the start of the



(a) GPRS/EDGE (b) 3G (c) WiFi

Figure 2: Energy-tail experiment.

first download in the session to the end of the last download. As indicated above, a shorter duration may mean a lower overall energy cost since the session completes more quickly.

4. AN ENERGY-AGNOSTIC APPROACH: COMBINE

We use a recently proposed energy-agnostic approach for wireless collaborative downloading called COMBINE [3] as a baseline system for comparison with Cool-Tether. In COMBINE, the web clients (i.e., the laptops) and the gateways (i.e., the smartphones) form a WiFi ad hoc network. Each gateway runs an HTTP proxy. To download a webpage, a client issues an HTTP GET request for the base HTML via the proxy running on one of the gateways. Once the HTML has been received, the client parses it to determine the URLs of the embedded objects and then proceeds to issue requests. In order to fully utilize the WAN bandwidth available at the gateways, the client stripes the requests for the embedded objects across all available gateways. The striping in COMBINE can happen at the level of objects or at a finer granularity, for instance, using the HTTP byte-range mechanism.

We now make a few observations on various facets of the COMBINE design to help introduce the design of Cool-Tether. First, download and striping of data across phones is driven entirely by the client in COMBINE. As we shall see in the next section, the unique energy characteristics of the WAN radios make such a design choice particularly inefficient, motivating the need for an energy-aware striping proxy in Cool-Tether. Second, COMBINE relies on WiFi ad hoc mode, that was indeed specifically designed for enabling such peer-to-peer communication scenarios. However, WiFi ad hoc mode can be significantly expensive from an energy perspective. Instead, Cool-Tether employs a novel reverse-infrastructure mode for its LAN communication needs.

5. DESIGN OF COOL-TETHER

The mobile phone gateways have two main energy-intensive components, namely, the GPRS/EDGE/3G gateway radio and the WiFi local-area radio. We consider the properties of each of these radios in detail and identify specific architectural elements that can address the inadequacies of the COMBINE architecture.

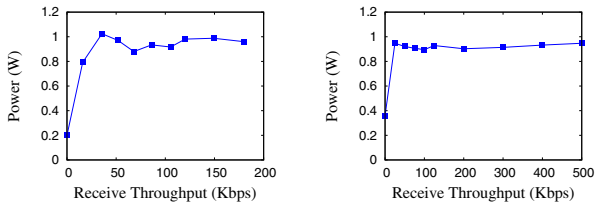
5.1 WAN Link: GPRS/EDGE/3G

The WAN link from the mobile phone serves as the gateway link in the Cool-Tether architecture. We first detail two unique characteristics of the WAN link and then describe their implications for Cool-Tether design.

5.1.1 GPRS/EDGE/3G Energy Tail

Unlike WiFi, WAN links exhibit a "tail" in energy consumption. In other words, there is a residual energy cost, that is sustained for a short time interval even after the final packet reception, before the WAN link goes back into its low power state. The reason for the sustained energy spike in GPRS/3G is due to the fact that the radio is maintained in high power active/ready state by the network in anticipation of subsequent transmissions, in order to amortize for the signaling costs in a cellular network. Only when there is no transmission for a couple of seconds, do we see the radio moved back into a low power idle state.

Figures 2(a), 2(b) and 2(c) plot the current drawn versus time (seconds) for a mobile device using a GPRS/ EDGE radio, a 3G radio and a WiFi radio, respectively, when a single ICMP ping packet is transmitted to the device and a ping response packet is subsequently received. We observe that while EDGE/GPRS and 3G has a much lower idle current draw than WiFi (100mA vs 230mA), transmission and reception activity results in a much larger *aggregate* drain in energy in the case of EDGE/GPRS and 3G compared to the case of WiFi. In the case of EDGE/GPRS and 3G, the spike in energy is sustained for almost three seconds, followed by another lower energy plateau of about four and twelve seconds, respectively. In contrast, in the case of WiFi, the spike in energy lasts only for a fraction of a second. Note the difference in x-axis scales in the figures — the largest spike in Figure 2(c) corresponds to the transmission of the ping response.



(a) GPRS/EDGE (b) 3G

Figure 3: Receive energy.

5.1.2 WAN Radio Receive Energy Profile

Figures 3(a) and (b) show the energy consumption versus UDP reception data rates for GPRS/EDGE and 3G, respectively. It is clear that the energy consumption for the WAN radios is a non-linear function of the received data

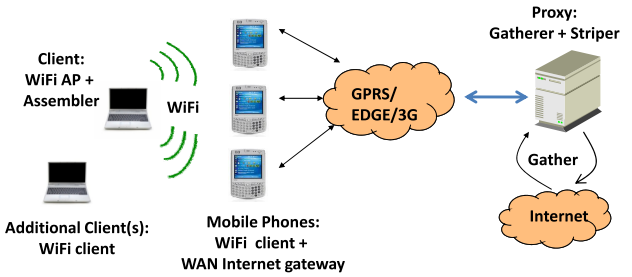


Figure 4: Cool-Tether architecture.

rate, unlike the linear energy profile seen with a typical WiFi radio [2]. There are several characteristics of WAN link energy consumption that can be seen from the figure. First, a high energy cost is entailed even for reception at very low data rates. Second, the energy cost of receiving at 10kbps and 500kbps in these radios is virtually identical. Thus, the incremental cost of sending at higher data rates is negligible.

We now highlight the implications of the above identified WAN link characteristics and the corresponding design elements that address the energy efficiency requirements of the Cool-Tether system. As we shall see in Section 7, the Cool-Tether system delivers 38%-71% savings in energy for a web access workload compared to the COMBINE approach. Figure 4 depicts the overall Cool-Tether architecture which is described in detail in Section 6.

5.1.3 Implication #1: Gatherer

The receive energy cost profile of WAN radios imply that dribbling data to mobile phones at low bit rates is not energy efficient and it is better to send data to a mobile phone at the highest possible receive rate. The tail behavior of the WAN radio implies that sending data to the mobile phone in intermittent bursts, even if each burst is sent at the highest receive rate, would incur a far greater cost (due to repeated tails) as compared to sending the data in a single burst, thereby minimizing the occurrence of the tail to a single episode after all the data is downloaded.

On the other hand, a typical web page has many embedded data objects and a browser usually first downloads the main web page before downloading each of the embedded objects, possibly from different servers. The COMBINE design presented in Section 4 would also experience a similar traffic pattern, which is clearly not a good fit with either the receive energy profile or the tail behavior of the WAN radio. Further, even the use of persistent connections [7] or a simple web proxy is not sufficient to fully exploit the WAN radio receive energy profile and tail behavior (refer to Section 7.3.1).

In order to address this problem, the Cool-Tether design employs a *gatherer* proxy in the cloud that first gathers all the embedded data objects in a web page and then sends the data in a single burst. At the client side, a simple *assembler* is employed that ensures that any request to the embedded data objects by the browser is matched to the forthcoming data from the gatherer, suspending any premature embedded object requests from the browser as necessary.

As we shall see in Section 7.3.1, the deployment of a simple gatherer results in energy savings of 25.9% and reduction in session completion time of 18.5% over the ad hoc network based design of COMBINE.

5.1.4 Implication #2: Energy-aware Striper

The WAN radio tail energy behavior also implies that there is a setup cost that needs to be taken into account when deciding whether to stripe data over one or more additional gateway phones in order to speed up a given download. Simply using all available gateway phones in an energy-agnostic manner, as the COMBINE proposal in Section 4 does, could be costly. However, this policy of using all available phones is representative of much of the related work that seek to improve download performance by using multiple wireless links [3, 5, 9, 12, 18, 19, 22]. We first describe the *energy-agnostic* policy and then describe an *energy-aware-balanced* policy, that computes the optimal number of phones that are chosen for striping, given the setup costs of the WAN link due to the energy tail.

The **energy-agnostic** policy is designed for minimizing completion time and stripes data across all available phones, ignoring both the energy efficiency of the data transfer and the residual battery of the phones. The striping can be either client-based, as in the COMBINE design, or server-based and could happen at the level of objects or byte-ranges. Although the energy-agnostic policy does well to improve the performance in terms of session completion time, it is very likely that some phones will run out of battery much quicker than others, since the lifetime of a phone depends on several parameters unique to the phone, such as capacity of the battery, battery percentage remaining, the base drain rate of the phone, etc. Given that each phone may be owned by individual users, a desirable property of the striping policy is to keep each phone active for as long as possible. Next, we look at an energy aware policy, which addresses these issues.

The **energy-aware-balanced** policy is designed to stripe data across phones such that the skew in the battery levels of the phones are minimized. Periodically, say every 30 seconds, a cloud-based striper learns the remaining battery percentage for all phones. It then eliminates phones whose residual battery life is below that of the phone with the highest battery life by a certain threshold, say 5% of battery capacity. Only the remaining subset of phones are chosen as *eligible phones* during the current interval. Assuming a workload where the number of requests is unknown ahead of time, this policy is designed to maximize the number of requests serviced until any one phone runs out of battery.

Given a set of eligible phones, it is not clear if all of them should be used to speed up the download. There is a trade-off between the fixed setup cost of using a phone for striping due to the GPRS/EDGE/3G energy tail versus the ability of the phone to speed up the transfer, thereby resulting in overall lower base energy expenditure in all phones for the duration of the download. We now derive an expression for the optimal number of phones that optimizes this trade-off.

Optimal number of phones to use for striping. Let n be the total number of eligible phones and let n_s be the number of phones that are used for striping. Let P_{Base} be the power consumed (energy consumed per second) by a device when it is not used for striping and let P_{Stripe} be the power consumed by the device (excluding the radio cost) when it is used for striping data. Note that, in general, $P_{Stripe} > P_{Base}$ given the extra CPU cost of processing and forwarding packets from the WAN link to the WiFi link. For simplicity, let us assume that P_{Base} and P_{Stripe} are the same for all the phones (i.e., the phones are homogeneous). We will relax this homogeneity assumption later.

Now let us consider the radio cost of the phones that are used for striping. Let E_{setup} be the fixed component of the cost of using a phone to stripe data (i.e., the WAN tail

cost discussed earlier) and let B be the speed of the WAN link. We assume that the overall download speed depends on WAN radio rather than WiFi. Let E_{bit} be the total energy cost of receiving a bit on the WAN radio and transferring the bit onto the WiFi link. While the receive energy cost on the WAN radio has a non-linear energy profile (see Figure 3), we use the receive energy cost corresponding to the average data rate over the link for computing E_{bit}.

Now, the total cost of transmitting S bits of data in time t using n_s phones is simply the sum of the base powers of the unused phones for time t and the radio energy cost of the phones that are used for striping, or:

$$[(n - n_s).P_{Base} + n_s.P_{Stripe}].t + n_s.[E_{setup} + (\frac{S}{n_s}).E_{bit}]$$

Assuming a linear speed up with striping as shown in [3], we can substitute $t = S/(n_s.B)$. Simplifying, the total energy cost is

$$\frac{n.P_{Base}.S}{n_s.B} + n_s.E_{setup} + S.(\frac{P_{Stripe} - P_{Base}}{B} + E_{bit})$$

In order to minimize the total energy, differentiating the above expression with respect to n_s and setting the result to zero, we get

THEOREM 1. *The number of phones to be used for striping to minimize the total energy expenditure is*

$$n_s = min(n, \sqrt{\frac{(n.P_{Base}.S)}{(E_{setup}.B)}})$$

If the gateway phones are heterogeneous, let the base energy consumption of the phones be $P_{base}(i)$. The above expression still works with a small modification. Instead of $n.P_{Base}$, we simply substitute $\sum_{i=1}^{n} P_{base}(i)$. Similarly, one can account for different values of $P_{stripe}(i)$. Note that B is likely to be similar for the various phones since the phones are in close vicinity to one another; if necessary, one could extend the derivation above to take into account different values of $B(i)$.

Note that the proxy gathers data before striping. Thus, it has an accurate estimate of the total size of the download, S. It simply needs to obtain the number of eligible phones, n, by first applying the energy-aware-balanced policy and then, using the above expression, stripe data over a subset of the eligible phones, thereby delivering the requested data in an energy efficient manner.

As we shall see in Section 7.3.2, always choosing all four available phones for striping can result in up to 20% higher overall energy cost compared to the optimal case.

5.2 LAN Link: WiFi

We now focus on the WiFi connectivity between the laptop client(s) and mobile phone gateway(s).

5.2.1 Architecture

The traditional approach for opportunistic communication using WiFi, similar to the one used in the COMBINE design in Section 4, is to use the WiFi ad hoc mode of operation that allows peers to join/leave the network as needed. However, the ad hoc mode is designed with the assumption that all nodes are peers and thus, all nodes share responsibility equally. For example, all nodes take turns sending beacons and the node sending the beacon is required to stay awake for the entire beacon duration, with the attendant energy costs. This results in a significant energy drain as shown in Section 7. Furthermore, such a symmetric ad hoc mode of operation is a mismatch with the needs of Cool-Tether since we are focused on reducing the energy on the mobile smartphone, even if at some cost to the laptop.

5.2.2 Implications #3: Reverse-infrastructure mode

Clearly, the asymmetry of the WiFi infrastructure mode matches the constraints of Cool-Tether better than the symmetric WiFi ad hoc mode. However, we want the gateway phone devices to benefit rather than pay the cost of the asymmetry. Thus, we adopt a novel reverse-infrastructure mode for WiFi connectivity. One of the laptop clients acts as a WiFi access point to which the mobile phones (and other laptop clients, if any) associate as WiFi clients. The choice of the laptop that acts as the AP can be made by a simple consensus based protocol among the participating client-laptops. Unlike the traditional infrastructure-based WiFi setting where the access point serves as the gateway to the external network, in Cool-Tether, the mobile phone WiFi clients serve as the gateway. Thus, the burden of sending beacons and staying awake is borne by the laptop while the mobile phones operate in low-power WiFi adaptive Power Save Mode (PSM), switching to active mode only when there is traffic. Further, the mobile phone gateways can join/leave the network by simply associating/disassociating with the access point. Finally, by allowing one laptop to serve as the access point, traffic from all laptop clients is consolidated, enabling centralized optimization of gateway bandwidth and energy resources.

Optimal Beacon Interval: We now discuss how the beacon interval can be adapted in order to further optimize energy drain on the mobile phones. As described in Section 3, we consider the problem of minimizing the total mobile phone energy cost of finishing a workload of a fixed set of web page downloads, with fixed think times between each download. Given this setting, there is an inherent trade-off in the choice of the beacon interval. A short beacon interval increases beacon processing costs on the mobile phone. On the other hand, since a download begins after half a beacon interval on average, a shorter beacon interval allows faster completion of downloads, and consequently lower expenditure of a mobile phone's base energy.

b	Beacon period for a given distribution of think times
α	Wake up cost for each beacon reception (Joules)
θ	Base power of phone, excluding radio costs (Watts)
$d_1, ..., dn$	Think times between successive downloads
D_{avg}	Average think time
$C_1, ..., C_n$	WiFi CAM energy consumed for download n
$E_{T1}, ..., E_{Tn}$	Energy consumed at the end of download n

Table 1: Parameters in optimal beacon period derivation.

In order to derive the optimal beacon period, we first define a few parameters of interest in Table 1. If a task T1 (download) for a client arrives at the AP at time d_1, then the second task (download) arrives d_2 seconds after the first task is finished. We assume that once the task has arrived, the client switches to Continuously Active Mode (CAM) until the task (download) is complete. Such an adaptive PSM approach has been shown to be effective in improving both

energy and performance [2]. Then, the total energy consumed at the end of the first download is simply the sum of three terms: the cost of the beacons during the think time, the cost of the base energy during the think time and half the beacon interval (on average) until download begins, and the cost of the WiFi radio in CAM while downloading. Thus,

$$E_{T1} = \frac{d_1}{b}.\alpha + (d_1 + \frac{b}{2}).\theta + C_1 = [(1 + \frac{\alpha}{b\theta}).d_1 + \frac{b}{2}].\theta + C_1$$

$$E_{T2} = [(1 + \frac{\alpha}{b\theta}).d_2 + \frac{b}{2}].\theta + C_2 + E_{T1}$$

$$E_{Tn} = [(1 + \frac{\alpha}{b\theta}).\sum_{i=1}^{n} d_i + n\frac{b}{2}].\theta + \sum_{i=1}^{n} C_i$$

$$E_{Tn} = [(1 + \frac{\alpha}{b\theta}).D_{avg} + \frac{b}{2}].n.\theta + \sum_{i=1}^{n} C_i$$

Minimizing for energy by differentiating the above expression with respect to b and setting the resulting term to zero, we get

THEOREM 2. *The optimal beacon interval is given by*

$$b_{optimal} = \sqrt{\frac{2D_{avg}\alpha}{\theta}}$$

The expression for $b_{optimal}$ agrees with intuition. The beacon interval should be longer if think times are long (or beacon processing costs are high) and the beacon interval should be shorter if base energy cost of the device dominates (since the phone is assumed to be turned on and draining base energy, irrespective of whether it is being used for download as described in Section 3).

Finally, note that the same expression can be used to determine a (different) beacon interval for inactive periods between browsing sessions. For example, if a typical user has an idle interval of 15 minutes between two workloads, the Cool-Tether system could adopt a separate beacon interval, say, optimized for 15 minute idle periods and switch to a different beacon interval, say, optimized for 5 second think times, once activity is detected.

As we shall see in Section 7.3.3, the use of the reverse-infrastructure WiFi PSM mode with the optimal beacon interval in Cool-Tether results in energy savings of 50% over the COMBINE design of using WiFi ad hoc PSM mode.

6. COOL-TETHER SYSTEM

Based on the discussions in the previous section, we now present the architecture of the Cool-Tether system, as shown in Figure 5. Cool-Tether consists of three key components:
(a) *Cloud-based gatherer*
(b) *Energy-aware striper*
(c) *Reverse-infrastructure mode WiFi LAN*

The reverse-infrastructure mode WiFi network is setup by one of the laptop clients that acts as an access point (AP) to which other laptop clients and mobile phone gateways associate as WiFi clients. The client-based WiFi access point transmits beacons at the optimal beacon interval that minimizes the energy consumption of the mobile phones when they are not being used for downloading. The client laptop acting as the AP also runs a proxy to which all client requests are directed. Once a web page request arrives at the proxy running on the AP laptop, it is forwarded on a first-come first-serve basis via a gateway phone to the cloud-based gatherer. The gatherer gathers the full web page, including all embedded objects, before handing off the data to the energy-aware striper to commence a bursty transmission that makes efficient use of the non-linear energy profile of the GPRS/EDGE/3G backhaul radio of each mobile phone gateway. The energy-aware striper in the cloud implements the energy-aware-balanced policy, identifies the eligible phones for striping, computes the optimal number of phones from within the eligible set based on the workload size and commences bursty striping of data over these phones. Finally, an assembler at the laptop acting as the AP assembles data sent by the gatherer and the striper before serving the HTTP responses to the connected clients.

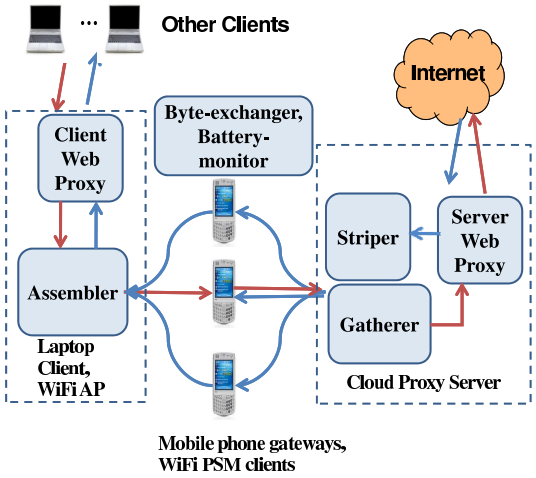


Figure 5: Cool-Tether implementation.

Thus, the Cool-Tether architecture is designed to support energy-efficient on-the-fly WiFi Hotspots, using the WAN connection available on multiple mobile stations, as backhaul. Cool-Tether implementation has the following requirements:

- The system should not require any changes to the client browser and should present a transparent interface to the client web applications.
- The system should require minimal support from mobile phones, both in terms of computation complexity and software changes. This is an important consideration from the viewpoint of not only energy conservation but also deployment on a wide-variety of mobile phone platforms.
- The system should allow dynamic addition and removal of mobile phones based on their availability. This requires that the initial setup overhead should be low.

As shown in Figure 5, the components of the Cool-Tether architecture can be divided into three broad categories, namely, components on the client machine, mobile gateway and the cloud proxy. On the client side, the system is comprised of two main components - the *client HTTP proxy* and the *assembler*. The mobile phone includes two components - the *energy monitor* and the *byte-exchanger*. Finally, the cloud proxy consists of the *gatherer*, the *striper* and the *server HTTP proxy*. We now describe the design and implementation of each of these components in detail.

Cool-Tether is able to support any application that can operate via an HTTP proxy. Incoming HTTP requests from the laptop client that acts as the access point (AP) as well as

other laptop clients are first routed through the HTTP client proxy running on the laptop AP. On receiving the requests, the client proxy queries its local assembler for the requested URL. If the URL response is present with the assembler, it is served to the client browser. Otherwise, if the requested URL is for a new page, it forwards the request to the mobile gateway. If the requested URL is for an embedded object of a previously requested page, it buffers the request, awaiting a response from the gatherer.

Each mobile gateway runs a *byte-exchanger* utility that acts as a byte-level forwarder, moving data arriving on one wireless link to the other. In addition, each mobile runs an *energy-monitor* utility that monitors the battery level of the device and can either piggyback this data on existing messages or respond to server queries.

The gatherer on the cloud proxy receives the URL request from the client and initiates download of the web page and the embedded objects. This is achieved by running a browser object instance with the exact URL request including the cookie information and other parameters. The response from the server is mirrored back to the striper, which then stripes the response to the pre-selected optimal set of phones. Each URL object is partitioned into fixed sized chunks tagged with a sequence number and sent along to different gateways, which then forward the data to the client.

On the receiving end, the client assembler aggregates the chunks corresponding to an object, and when all the chunks are received, it does one of two things. If there is an outstanding request from the client proxy for this aggregated object, the assembler serves the object to the client proxy, which then serves the data to the client browser. In case the response received at the client proxy is for an embedded object contained in an earlier requested page, the assembler stores the object temporarily and waits for the client browser to issue an explicit request for this object.

The Cool-Tether architecture has low setup overhead, as the addition or removal of a mobile gateway is easily achieved by managing the WiFi association with the software access point running on the client machine. The removal of the laptop AP is also easily handled by another laptop taking over the role of the AP; the impact on the mobile phones would simply be the equivalent of a WiFi handoff which can be executed quickly. Hence, Cool-Tether is able to efficiently meet all three implementation goals.

7. EVALUATION

In this section we carefully evaluate the performance of the Cool-Tether system. As described in Section 3, we use total energy consumption of all phones, session completion time, and network lifetime as our performance metrics. First, we describe the experimental setup and then present macro-results that compare the COMBINE proposal with the Cool-Tether system. We then evaluate each of the design components of Cool-Tether individually in order to get a microscopic understanding of the overall gains of our system.

7.1 Experimental Setup

Figure 6 depicts our setup for the experiments. We use four HP iPAQ hw6965 Windows Mobile 5 Smartphones that connect to the Internet using GPRS/EDGE links. While we used a HTC 8525 Windows Mobile 6.1 phone for benchmarking purposes that highlighted the similarity between 3G and GPRS/EDGE energy characteristics in Section 5,

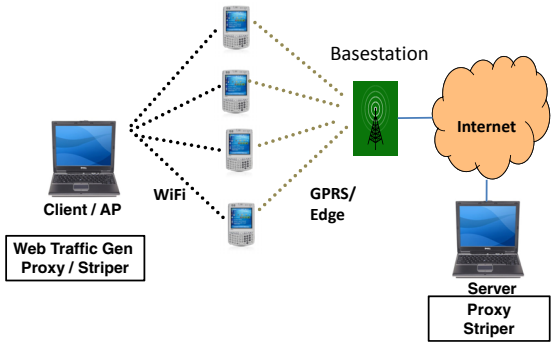


Figure 6: Experiment setup.

we had only limited access to this phone and thus do not include this phone as part of further experiments. The WiFi (802.11b) interface of each phone is configured to communicate with a client laptop in infrastructure mode. The power management settings of WiFi interface is set to Power Save Mode (PSM) or Continuously Active Mode (CAM) mode as specified in each experiment. All client side components of Cool-Tether, as well as the Web traffic workload generator, are hosted on this laptop. The server side proxy runs on another laptop that is connected to the Internet via a DSL line.

Energy measurements are made on each phone using a software API, which can be invoked as needed, to measure the energy drain on the battery. The *GetSystemPowerStatusEx2()* system API in the Windows Mobile SDK provides two ways of accessing energy characteristics: one function call returns the battery percentage remaining while another pair of function calls return the measure of voltage and current drawn, averaged every ten seconds. While the battery remaining function call is typically accurate in most phones, it is too coarse-grained for our purpose since its resolution is only 1%, encompassing several minutes of usage. So, we used the voltage/current function call, sampled every five seconds and compute the total energy in joules for the duration of the experiment.

In the experiments for network lifetime evaluation, letting the experiment run until the battery drains completely takes a long time, and is quite inconvenient since the phones need to be recharged before running the next experiment. For ease of running multiple experiments, we assume the battery capacity to be T units (e.g., 10%) of battery level. During an experiment, whenever a phone drains T units of battery from the start of the experiment, we assume that it has used up the entire battery and can no longer be used for the remainder of the experiment. Finally, we generate web request workload as detailed in Section 3 and use a fixed think time of 5 seconds in-between downloads.

7.2 Macroscopic View

We first start with a high-level comparison of the Cool-Tether system against the COMBINE design. We use all three architectural elements of Cool-Tether in this evaluation. In the case of the COMBINE design, which requires WiFi ad hoc mode of connectivity, none of our phones support PSM in WiFi ad hoc mode. Thus, we compare Cool-Tether against two versions of the COMBINE system, one called Ad-hoc-COMBINE, where ad hoc CAM mode is used, and another called Infrastructure-COMBINE, where we use Cool-Tether's reverse-infrastructure PSM mode for WiFi connectivity. Note that the true performance of COMBINE

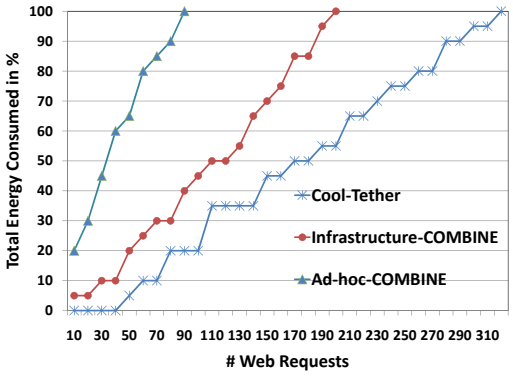


Figure 7: Comparison of Cool-Tether with two COMBINE approaches.

with ad hoc PSM would be somewhere in between the two COMBINE's that we were able to experimentally evaluate. In Section 7.3.3, using simulations, we evaluate the performance of WiFi ad hoc PSM and find that Cool-Tether's infrastructure PSM results in 50% energy savings as compared to WiFi ad hoc PSM.

We provision four fully-charged mobile phones as gateways and start a web workload on a client. The workload alternates between downloads of a webpage with embedded objects (average total size of webpage = 40KB) and idle think times. The workload is repeated until all the gateway phones are deemed to be without battery capacity (for timeliness, we assume the loss of a threshold $T = 10\%$ of battery capacity is equivalent to losing full charge).

Figure 7 depicts the performance of Cool-Tether and the ad-hoc and infrastructure-COMBINE approaches. We note that ad-hoc-COMBINE is the first to lose battery capacity in all the phones, while having completed only 94 web requests. Infrastructure-COMBINE is the next to end, having completed 203 requests while the Cool-Tether system is able to serve 322 requests (3.4/1.6 times more requests than the ad hoc/infrastructure-COMBINE designs) before losing charge in its gateway phones. In other words, *Cool-Tether delivers 38%-71% savings in energy for web access workloads as compared to the COMBINE tethering solution.*

Another way to demonstrate Cool-Tether's effectiveness is as follows. We run a browsing workload at a laptop client that uses two mobile phones as gateways and then compare the battery drain on the two phones to the case where the phones were not used as gateways. For a typical browsing session of about 15 minutes, the difference in battery percentage between the phones being used as gateways and the phones not being used as gateways is only 2%.

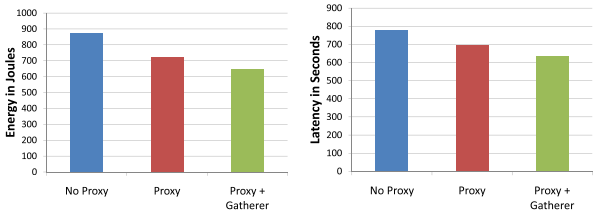
We will now explore how the architectural elements of Cool-Tether help to deliver these substantial energy gains.

7.3 Microscopic View

We evaluate each of the design choices of Cool-Tether separately in order to obtain a fine-grained understanding of the overall gains of Cool-Tether. First, we show the benefit of using a gatherer in the Cool-Tether system. Second, we study the impact of striping policy on energy consumption of individual mobile phones by comparing two policies: energy-aware-balanced and energy-agnostic. Finally, we compare the performance of different WiFi modes, such as WiFi ad hoc CAM and WiFi ad hoc PSM, with Cool-Tether's reverse-infrastructure mode.

7.3.1 Proxy Server and Gatherer

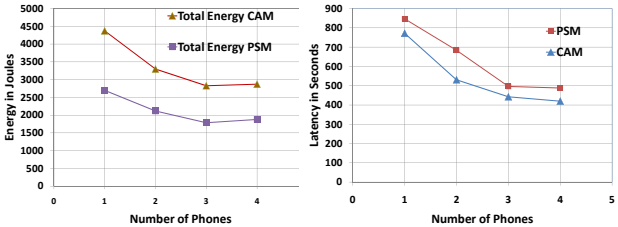
Let us now focus on the impact of having a server proxy implemented with gatherer optimization. We conduct experiments using a single phone since we want to isolate the gatherer's performance from the striper. We compare three schemes: no proxy, proxy and proxy with gatherer. In order to run the baseline scenario, which is no proxy, we simply have to connect the phone as a modem to the laptop. But, the Windows Mobile 5 platform exposes the modem facility over USB and not WiFi. As a result, to compare the three schemes fairly, we connected the phone and client laptop using USB. To prevent the phone from charging during the experiments, we disabled USB charging option on the phone. We measure the total energy consumed by the phone after the workload completes and also record the total time to complete the entire workload.



(a) Energy consumed (b) Completion time

Figure 8: Impact of Server Proxy and Gatherer on energy consumed and workload completion times.

Figures 8(a) and 8(b) plot the energy consumed and workload completion times, respectively, for the above three schemes. The simple proxy already provides an improvement over the baseline scheme of 10.7% and 17.6% for the completion time and total energy metrics, respectively, for a couple of reasons. First, the HTTP proxy on the server eliminates all DNS lookups at the client. Second, all responses are now transferred over a single TCP connection between the client and the server instead of opening multiple TCP connections for each object embedded in the web page, thus providing an opportunity for some burstiness in the download. When the proxy with the gatherer optimization is used, we see further gains in performance. This is achieved due to the fetching of all embedded objects of a webpage by the *gatherer* proxy, as explained in Section 5.1.3. Compared to the baseline scheme, completion time is reduced by 18.5% while total energy consumed is reduced by 25.9%, demonstrating the significance of the gatherer for WAN links.



(a) Energy consumed (b) Completion time

Figure 9: Impact of number of phones on energy consumed and workload completion times.

7.3.2 Striper

We now proceed to the scenario with multiple phones and evaluate the different striping policies, and how they can

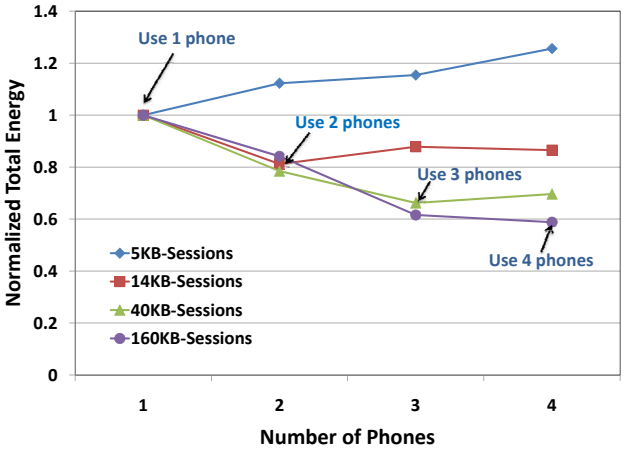


Figure 10: Optimal number of phones for different workloads.

GPRS Energy tail, E_{setup}	1.764J
P_{Base} (PSM)	0.48W
GPRS average bandwidth, B	40Kbps
Eligible phones, n	4
n_s for workload, $S = 5KB$	1.04
n_s for workload, $S = 14KB$	1.75
n_s for workload, $S = 40KB$	2.95
n_s for workload, $S = 160KB$	4 (min(4,5.90))

Table 2: Optimal number of phones, n_s, for striping.

impact the energy drain on the phones. We first conduct experiments to understand how the number of phones can impact the energy and latency numbers. All four phones are connected to the client laptop in reverse infrastructure-mode. We varied the number of phones used from one to four, and for each set of phones we ran the web workload twice, once in Power Save Mode (PSM) and once in Continuously Active Mode (CAM). Note that the WiFi power-save settings of the phones play a role in the energy consumption and latency to complete the workload. Figure 9(a) shows the total energy consumed by all phones used for that run. Figure 9(b) shows the workload completion times for the same experiment. For this workload, the optimal number of phones to be used for striping in order to minimize total energy is three phones. We now highlight how workload size, S, impacts the choice of optimal number of phones.

Figure 10 shows the energy performance for different workloads when different numbers of phones are striped. The y-axis in the figure is the total energy of mobile phones normalized to the case when one phone is being used. From the figure, we can see that using the optimal number of phones for striping can lower the total energy by between 20-40% compared to the worst choice of number of phones for striping. Furthermore, always choosing all four available phones can result in up to 20% higher energy compared to the optimal case.

Finally, we would like to compare these experimental results with the theoretically calculated optimal value using the results of Theorem 1. Our measured values on the HP iPAQ 6965 for various parameters of interest, as well as the computed optimal number of phones using Theorem 1 for different workloads, are shown in Table 2. Notice the close agreement between the predicted optimal number of phones in Table 2 and the optimal value for the corresponding workloads in Figure 10.

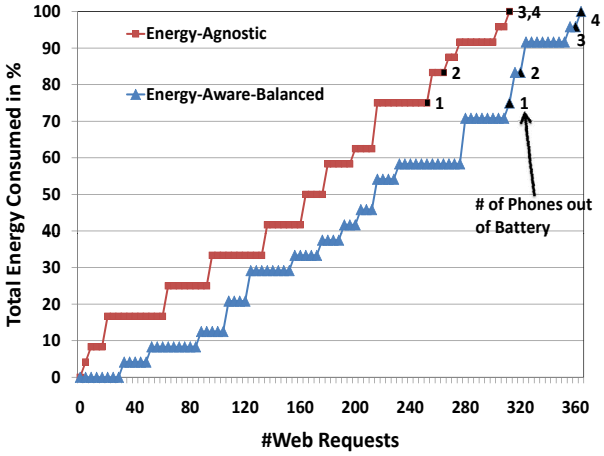


Figure 11: Lifetime comparison.

From these results, we conclude that the Cool-Tether striper can dynamically compute the optimal number of phones for each given workload and can deliver up to 40% gains over a naïve striper that may choose incorrect numbers of phones for striping.

Having analyzed the impact of number of phones, we evaluate two striping policies: energy-agnostic and energy-aware-balanced. Figure 11 shows the lifetime of the network metric for the two schemes. The times at which individual phones run out of battery are shown for each scheme. The energy-aware-balanced scheme achieves approximately 14% improvement in overall energy consumption compared to the energy-agnostic scheme. In addition, 20% more requests are serviced by the energy-aware scheme before the first phone drains completely.

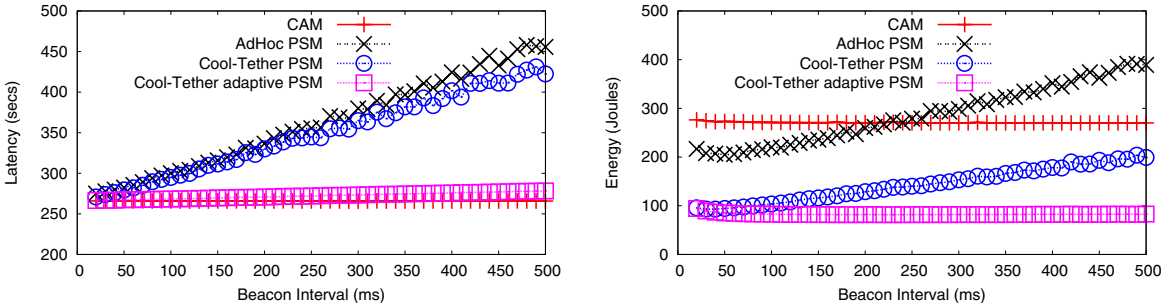
Finally, we also evaluate the impact of using a phone for striping while it is being charged. Does its charging rate change? From our experimental results (not shown), we found that the charging rate was indeed unaffected due to striping. Thus, phones that are being charged can automatically be enrolled for striping needs.

7.3.3 *WiFi Reverse-infrastructure Mode*

We now seek to quantify the effect of different modes of WiFi communication between the client laptop and the gateway mobile phone. Due to the unavailability of ad hoc-PSM implementation on most mobile phones, we use the Qualnet simulator to simulate the different WiFi modes of operation. Although a simulation environment may be insufficient to completely capture the energy drain model of the mobile phone (that includes the energy spent on the GSM interface), our simulation setup helps us analyze and model the effect of various WiFi communication modes in isolation. Note that in the previous section, we already evaluated the reverse-infrastructure PSM and CAM, and WiFi ad hoc CAM experimentally as these modes are supported on most phones, including our Windows Mobile 5 phones.

Figure 12: Simulation setup.

Figure 12 shows the simulation setup. In infrastructure mode, the HTTP client acts as the access point to which



(a) Completion time (b) Total energy cost

Figure 13: Total energy consumed and time taken to finish a workload of 50 HTTP URL requests.

Tx Cost	1.87 W
Rx Cost	1.62 W
Idle Cost	0.99 W
Sleep Cost	0.23 W
Ad Hoc ATIM window	5 ms
DTIM interval	1
Bit rate	2 Mbps

Table 3: Simulation parameters.

the mobile gateway is connected. The metric used in our simulations is the total time taken and energy consumed by the mobile phone to finish a given workload. The workload is comprised of 50 HTTP page downloads, where each download consists of a main web page and multiple embedded objects. Note that each page download is separated by fixed think times during which there are no outstanding requests. Table 3 shows the parameters used in our simulations, which are derived from actual measured values on the HP iPAQ smartphone. Figures 13(a) and 13(b) show the session completion time and the total energy cost for a workload of 50 HTTP requests for CAM, ad hoc PSM, Cool-Tether's reverse-infrastructure PSM and Cool-Tether's reverse-infrastructure adaptive PSM modes. In the adaptive PSM mode, the mobile gateway switches from PSM to CAM at the start of each HTTP session and immediately switches back to the low power PSM mode during think times when there is no traffic in the network. As seen in Figure 13(a), as the beacon interval increases, the PSM mode in both ad hoc and Cool-Tether settings takes much longer than CAM or adaptive PSM to finish the workload.

Examining Figure 13(b), we find that the *total energy cost of the Cool-Tether's reverse-infrastructure PSM or adaptive PSM modes is 50% lower than the total energy cost of the ad hoc PSM mode* at their respective optimal values. This huge performance gap is because of several reasons. First, in ad hoc mode, nodes indicate the availability of data for other nodes in a short duration called the ATIM window. All ad hoc nodes have to stay active for at least the ATIM window duration at the beginning of each beacon interval. Furthermore, all nodes share the responsibility of beacon transmission, and the node that transmits the beacon stays awake during the entire beacon period (as mentioned in the IEEE 802.11 standard). Thus, in ad hoc mode, both the laptop client and the mobile gateway have similar energy consumption. On the other hand, in Cool-Tether's reverse-infrastructure mode, the onus of beacon transmissions and staying awake lies entirely on the laptop client, which acts as the access point, allowing the mobile gateway to conserve energy.

We also find that the choice of the optimal beacon interval plays a much bigger role in static PSM mode as compared to the adaptive PSM mode. Based on the results in Theorem 2, using a think time, $D_{avg} = 5$ seconds, base power consumption, $\theta = 0.23W$, and beacon reception cost, $\alpha = 1.3J$, the optimal beacon interval for Cool-Tether adaptive PSM is $238.1ms$, which agrees with our simulation results. However, the choice of optimal beacon value provides an energy gain of only a few percent as compared to choosing a default beacon interval of, say, 100ms.

7.4 Summary

In this section, we demonstrated that Cool-Tether provides significant energy savings of 38%-71% over COMBINE, a recently proposed system for wireless collaborative downloading [3]. A closer look at the three architectural elements of Cool-Tether reveals that the cloud-based gatherer provides energy gains of 25%, the energy-aware striper provides gains of up to 40% over an energy-agnostic striper, and the reverse-infrastructure WiFi mode provides gains of 50% over the traditional WiFi Ad hoc PSM mode. Thus, we believe Cool-Tether has demonstrated the feasibility of smartphone-based architecture for energy-efficient, affordable Internet access.

8. DISCUSSION

Cool-Tether is designed with ease of deployment as a key criteria. The WiFi infrastructure model used between the laptops and the mobile phones allows any of the gateway phones to join or leave without affecting the rest of the network. Even if the laptop client that serves as the WiFi AP leaves the network, other laptop clients can detect the absence of the beacon and can takeover as the new WiFi AP by implementing simple leader election techniques. Most mobile phones today support WiFi client mode with adaptive PSM support (in contrast, we could not find a phone that supported WiFi ad hoc PSM mode). The forwarding function on the mobile phone is a simple user-level proxy and can easily be ported to many phone platforms.

However, one key component of Cool-Tether is the need for a proxy server in the cloud that performs prefetching and striping. While the proxy server is not strictly essential for functionality, a significant portion of the energy savings of Cool-Tether will not be realizable without a proxy server as part of the infrastructure. For example, prefetching eliminates roundtrips and allows bursty delivery of data that optimizes the WAN link. Prefetching also enables compu-

tation of the optimal number of gateway phones to be used for striping. We believe that the need for a proxy server in the cloud is not a significant hurdle to deployment today.

Finally, although Cool-Tether system is primarily designed for a multi-phone tethering scenario, the underlying components of the system are generic and can be used independently to achieve savings in other scenarios as well. One such example is to employ the gatherer proxy independent of the striper to achieve energy savings for web browsing on a single mobile phone.

9. CONCLUSION

We have presented Cool-Tether, an architecture for providing Internet access by banding together a group of mobile smartphones to create a WiFi hotspot on the fly. The Cool-Tether architecture incorporates key elements to address the central concern of energy efficiency: (a) an infrastructure proxy that performs gather and stripe operations to modulate web access traffic to better match the characteristics of a GPRS/EDGE wireless WAN link, and (b) a novel reverse-infrastructure mode of operation for the WiFi network, with the smartphone gateways functioning as WiFi clients. Based on a prototype of Cool-Tether built on Windows Mobile smartphones, we have demonstrated very significant energy savings of 38%-71% compared to a state-of-the-art but energy-agnostic system for tethered operation.

Cool-Tether lies at the intersection of two significant trends: the emergence of smartphones as *the* pipe into many homes and small businesses, especially in the developing world, and the growth of cloud-based services that make computing relatively inexpensive for clients, including smartphones. In future work, we plan to explore how smartphones combined with an intelligent cloud infrastructure could form the connectivity glue not just for PCs and laptops but also for the emerging world of ubiquitous sensors.

10. ACKNOWLEDGMENTS

This work was funded in part by NSF Career Award CNS-0347886.

11. REFERENCES

[1] The Indian Telecom Services Performance Indicator Report for the Quarter ending June 2009. http://www.trai.gov.in/reportpre.asp?id=48, Oct 2009.

[2] Anand, M., Nightingale, E., and Flinn, J. Self-Tuning Wireless Network Power Management. In *MobiCom* (September 2003).

[3] Ananthanarayanan, G., Padmanabhan, V. N., Ravindranath, L., and Thekkath, C. A. COMBINE: Leveraging the Power of Wireless Peers through Collaborative Downloading. In *MobiSys* (June 2007).

[4] Barford, P., and Crovella, M. Generating Representative Web Workloads for Network and Server Performance Evaluation. In *SIGMETRICS* (June 1998).

[5] Carter, C., and Kravets, R. User Devices Cooperating to Support Resource Aggregation. In *WMCSA* (June 2002), IEEE, pp. 59–69.

[6] Chakravorty, R., Clark, A., and Pratt, I. GPRSWeb: Optimizing the Web for GPRS Links. In *MobiSys* (May 2003).

[7] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and Berners-Lee, T. Hypertext Transfer Protocol – HTTP/1.1. RFC 2616, June 1999.

[8] Housel, B. C., and Lindquist, D. B. WebExpress: A System for Optimizing Web Browsing in a Wireless Environment. In *MobiCom* (November 1996).

[9] Hsieh, H.-Y., and Sivakumar, R. A Transport Layer Approach for Achieving Aggregate Bandwidth on Mutli-homed Mobile Hosts. In *MobiCom* (September 2002).

[10] Hui, P., and Crowcroft, J. Bubble Rap: Forwarding in Small World DTNs in Ever Decreasing Circles. Tech. rep., UCAM-CL-TR-684, 2007.

[11] Joseph, A. D., Delespinasse, A. F., Tauber, J. A., Gifford, D. K., and Kaashoek, M. F. Rover: A Toolkit for Mobile Information Access. In *SOSP* (December 1995).

[12] Kim, K.-H., and Shin, K. G. Improving TCP Performance over Wireless Networks with Collaborative Multi-homed Mobile Hosts. In *MobiSys* (June 2005).

[13] Kistler, J. J., and Satyanarayanan, M. Disconnected Operation in the Coda File System. *ACM Transactions on Computer Systems, Volume 10* (February 1992), 3–25.

[14] Liljeberg et al., M. Mowgli WWW Software: Improved Usability of WWW in Mobile WAN Environments. In *GlobeCom* (November 1996).

[15] Luo, H., Ramjee, R., Sinha, P., Li, L., and Lu, S. UCAN: A Unified Cellular and Ad-hoc Network Architecture. In *MobiCom* (September 2003).

[16] Noble, B. D., and Satyanarayanan, M. Experience with Adaptive Mobile Applications in Odyssey. *Mobile Networks and Applications, Volume 4* (December 1999), 245–254.

[17] Pering, T., Agarwal, Y., Gupta, R., and Want, R. CoolSpots: Reducing the Power Consumption of Wireless Mobile Devices with Multiple Radio Interfaces. In *MobiSys* (June 2006).

[18] Qureshi, A., and Guttag, J. Horde: Separating Network Striping Policy from Mechanism. In *MobiSys* (June 2005).

[19] Rodriguez, P., Chakravorty, R., Chesterfield, J., Pratt, I., and Banerjee, S. MAR: A Commuter Router Infrastructure for the Mobile Internet. In *MobiSys* (2004).

[20] Schroeder, B., Wierman, A., and Harchol-Balter, M. Closed versus Open System Models: a Cautionary Tale. In *NSDI* (May 1996).

[21] Sharma, P., Lee, S.-J., Brassil, J., and Shin, K. G. Handheld Routers: Intelligent Bandwidth Aggregation for Mobile Collaborative Communities. In *BroadNets* (October 2004).

[22] Snoeren, A. C. Adaptive Inverse Multiplexing for Wide-Area Wireless Networks. In *IEEE Global Internet Symposium* (December 1999).