A Measurement Study of Vehicular Internet Access Using *In Situ* Wi-Fi Networks

Vladimir Bychkovsky, Bret Hull, Allen Miu, Hari Balakrishnan, and Samuel Madden
MIT Computer Science and Artificial Intelligence Laboratory
{vladb, bwhull, aklmiu, hari, madden}@csail.mit.edu
http://cartel.csail.mit.edu

ABSTRACT

The impressive penetration of 802.11-based wireless networks in many metropolitan areas around the world offers, for the first time, the opportunity of a "grassroots" wireless Internet service provided by users who "open up" their 802.11 (Wi-Fi) access points in a controlled manner to mobile clients. While there are many business, legal, and policy issues to be ironed out for this vision to become reality, we are concerned in this paper with an important technical question surrounding such a system: can such an unplanned network service provide reasonable performance to network clients moving in cars at vehicular speeds?

To answer this question, we present the results of a measurement study carried out over 290 "drive hours" over a few cars under typical driving conditions, in and around the Boston metropolitan area (some of our data also comes from a car in Seattle). With a simple caching optimization to speed-up IP address acquisition, we find that for our driving patterns the median duration of link-layer connectivity at vehicular speeds is 13 seconds, the median connection upload bandwidth is 30 KBytes/s, and that the mean duration between successful associations to APs is 75 seconds. We also find that connections are equally probable across a range of urban speeds (up to 60 km/hour in our measurements). Our end-to-end TCP upload experiments had a median throughput of about 30 KBytes/s, which is consistent with typical uplink speeds of home broadband links in the US. The median TCP connection is capable of uploading about 216 KBytes of data.

Our high-level conclusion is that grassroots Wi-Fi networks are viable for a variety of applications, particularly ones that can tolerate intermittent connectivity. We discuss how our measurement results can improve transport protocols in such networks.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Wireless communication; C.4 [**Performance of Systems**]: Measurement techniques

General Terms

Measurement, Performance, Experimentation, Design

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
MobiCom'06, September 24–29, 2006, Los Angeles, California, USA.
Copyright 2006 ACM 1-59593-286-0/06/0009 ...$5.00.

Keywords

Mobile networks, mobility, vehicular mobility, connectivity, wireless LAN, 802.11, Wi-Fi

1. INTRODUCTION

The penetration of 802.11 (Wi-Fi) in homes and offices around the world has been phenomenal. For instance, Jupiter Research estimates the number of home-deployed Wi-Fi access points (APs) in the United States to be 14.3 million (65% of online households) and growing. Many home users deploy Wi-Fi in their homes and connect to the Internet over broadband cable modem, DSL, or even fiber. Because these Wi-Fi networks and upstream broadband access links are often idle, they could potentially be used to offer Internet access to other users (for the moment, imagine homes being able to function as "micro-ISPs"). We are interested in understanding what sort of performance one might expect from these unplanned community networks. In particular, could the blanket of radio connectivity provided by home Wi-Fi networks actually provide reasonable network coverage and performance, even for highly mobile users (*e.g.*, those traveling in cars)?

Before describing the technical questions and contributions of this paper, we must note that there are several important issues concerning policy, business decisions, and law [10] that must be resolved before this vision of an "open Wi-Fi" Internet access network can become real. We believe, however, that with the increasing deployment of open urban Wi-Fi networks in many cities around the world [24], community mesh networks (*e.g.*, Roofnet (`http://pdos.csail.mit.edu/roofnet`), Champaign-Urbana Community Wireless (`http://www.cuwireless.net`) and commercial activity (*e.g.*, `http://fon.com`, `http://wibiki.com`) in this space, such networks may become real in the next few years.

This paper is not concerned with whether such open Wi-Fi networks *could* get deployed, or even whether they *should*, but primarily with what their performance *would* be *if* they were deployed and permitted open access (*e.g.*, by providing controlled access to subscribers of an "open Wi-Fi service"). We briefly discuss the important issues of security, viability, and deployment in Section 6.

We now turn to the central question addressed in this paper: *What is the expected performance of open Wi-Fi networks for mobile users, particularly for users in automobiles, as they move in urban and suburban areas where APs are currently widely deployed?* We focus on vehicular mobility because that is precisely where the answers to the performance questions raised above are largely unknown and where we believe connectivity and performance problems are most likely to arise.

Some of the questions addressed in this paper are:

1. What is the distribution of the duration of connectivity per AP? What is the distribution of the duration of disconnectivity (*i.e.*, the time between associations to different APs)? How long does it take for a client to scan, associate, and obtain an IP address?
2. What is the distribution of the coverage region of an AP?
3. What is the distribution of packet loss and data transfer rates?
4. What is the effect of a car's speed on these metrics?

We answer these questions by running a measurement study over a set of *in situ* open APs deployed currently in and around the Boston metropolitan area (some of our data also comes from a small area in and around Seattle). Nine distinct cars, each outfitted with a mobile embedded computer, collect data about Wi-Fi networks during the course of their owners' normal driving. These computers attempt to associate with open APs deployed nearby. If the association succeeds, then the mobile node attempts to obtain an IP address, and then initiates an end-to-end ping (with several retransmissions until the first success or a timeout) to a well-known IP address. If the ping succeeds, then the node starts a set of periodic local AP pings to the first-hop IP router. In addition, for a subset of associations, the mobile node initiates a TCP transfer (upload) to the Internet site (being careful not to "steal" bandwidth or disrupt performance for the owners of these APs). This experimental apparatus, described in more detail in Section 3, allows us to gauge both radio channel conditions and end-to-end wide-area connectivity from moving cars.

In this paper, we focus on data uploads from cars rather than downloads to cars. There are two reasons for this. First, several emerging applications treat cars as data sources in mobile sensor networks (*e.g.*, the CarTel project [12]), where a variety of sensors (GPS, cameras, on-board diagnostics, etc.) acquire and deliver data about cars and the surrounding environment. Second, it is very likely that the download performance will be at least as good as uploads, because most broadband links have more bandwidth in the download direction. In any case, many of our results concern the bidirectional properties of the radio channel itself, and these findings should apply equally to both data transfer directions.

We analyzed over 290 "drive hours" of data collected over 232 different days over nearly a year. Figure 1 shows the geographic area covered by our study.

We divide our results into two broad categories: connectivity (Section 4) and data transfer performance (Section 5). First, we analyze the link-layer and end-to-end connectivity properties of our measurements, finding that the distribution of the duration of link-layer per AP (as measured by the time between the first and last successful local AP pings) is 13 seconds. We find that acquiring IP addresses using DHCP has a median delay of 1.83 seconds and that a simple caching scheme that uses the AP's MAC address and honors DHCP lease times can reduce this latency to less than 350 ms (but the cache hit rate is only around 20%, leaving room for considerable improvement). We find that our cars were able to successfully associate with APs and also transfer data with a uniform probability at all speeds between 0 and 60 km/hour, showing that urban vehicular speeds need not preclude Wi-Fi use. In addition, we find that the mean inter-arrival duration between open, associable APs is about 75 s. These APs appear in clusters and are not uniformly distributed.

We then turn to the factors affecting data transfer performance, investigating both link-layer packet loss rates and end-to-end TCP throughput. Our end-to-end TCP upload experiments had a median throughput of about 30 KBytes/s, which is consistent with typical uplink speeds of home broadband links in the US. The median TCP connection is capable of uploading about 216 KBytes of data.

After describing our experimental method and discussing our findings in detail, we describe the implications of our findings on the design of data dissemination protocols for such intermittently connected networks (Section 6). We also discuss various issues concerning the viability of an open Wi-Fi Internet service.

2. RELATED WORK

The performance of TCP and UDP in wireless network scenarios from stationary clients has been fairly well-studied [1]. For the most part, however, few previous measurement studies have attempted to characterize wireless "LAN" network performance from moving vehicles. Ott and Kutscher [16] study the behavior of network connections that are initiated over an IEEE 802.11b channel from a moving car. This study involved a small number of bidirectional measurements over both UDP (RTP) and TCP. The goal was to understand the impact of the car's speed, transmission rate, 802.11 bit-rate, and packet size on throughput and delay. The experiments used a fixed, carefully planned test-bed, one in their lab and one in a limited two-AP deployment.

The authors break a TCP connection into three phases: the "entry" phase, the "production" phase, and the "exit" phase. During the entry and exit phases, the car is too far from the AP and throughput is low. Once the car is within a certain range ($\approx$ 200 m in their experiments), throughput rises as the car enters the production phase. The rise is rather dramatic, but the throughput is highly variable. Although a significant volume of data can be transferred, the authors believe that proxies or changes to protocols may improve performance further. For example, they show in their more recent work [17] that it is possible to avoid startup overheads in transport protocols like TCP by using a proxy to hide short periods of disconnection from the transport layer.

Gass *et al.* [9] demonstrate the feasibility of using off-the-shelf IEEE 802.11b wireless networks for TCP and UDP transfers to and from a moving car. Their experiments are also conducted in a planned environment—they measure performance from an "in-motion" client to a single access point in the California desert, where interference from other radios and vehicles is non-existent and there are no obstacles. This environment allows them to measure performance in a controlled mobile setting.

In the first part of the paper, the authors measure the performance between the client and the AP only. As in [16], they conclude that packet losses are low within 150 m of the AP and that for a wide speed range (5–75 mph), there is a region of good connectivity and throughput. Then, they measure the end-to-end performance of various protocols by connecting a server to the AP and changing the delay and bandwidth of that link, finding that Web performance degrades when a delay of 100 ms is added to the end-to-end link. They suggest that aggregating multiple HTTP requests will mitigate the impact of this delay; we believe that this delay could be avoided by simply issuing pipelined requests, an option already present in HTTP/1.1. (The authors use `wget`, an HTTP/1.0 client that does not appear to support pipelined requests.)

Similar performance studies have also been performed for cellular networks [19, 18]. Cellular networks, however, have characteristics that are very different from the urban and suburban *in situ* IEEE 802.11b deployments we are examining in our work.

Several "war driving" studies have mapped APs in various areas around the world (*e.g.*, `http://placelab.org`, `wifimaps.com`, `http://wardriving.com` or `http://wigle.net`), but these studies have generally not gone beyond characterizing the location of APs. Akella *et al.* [2] also measure the supported bitrates, whether encryption is turned on, and (more recently) [3] the coverage region of each AP for a few thousand APs in Pittsburgh.

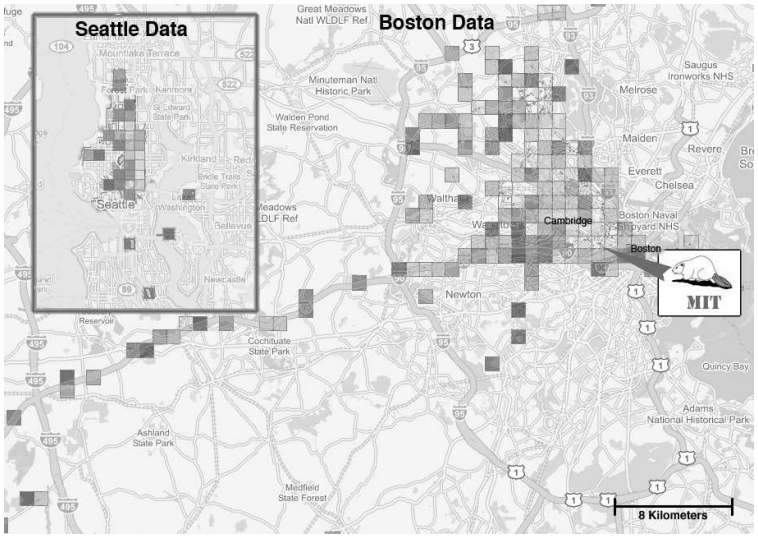


Figure 1: Map of the greater Boston area (Seattle inset), with 1.4 x 1.6 km grid cells indicating locations where cars were able to successfully establish a connection to an access point. Cells are shaded to indicate the most "popular" cells, with darker colored cells having fewer access points connections (including repeat connections to the same access point.) Total number of covered cells shown in Boston is 160. We have a small amount of additional data in outlying areas not shown here.

3. EXPERIMENTAL METHOD AND DATA SUMMARY

Our measurement study uses 9 cars belonging to people who work at MIT's CSAIL. We instrumented these cars with an embedded computer running the CarTel system software [12], which includes a set of programs, `Scanping`, to measure Wi-Fi performance. The 9 cars maintained their normal driving patterns and schedule during the course of our study, and did not do anything out of the ordinary. All the data collected and analyzed are therefore subject to the constraints of real traffic and network conditions, a marked departure from the previous studies discussed in Section 2.

This paper analyzes 290 total hours of driving over 232 distinct days between July 29, 2005 and July 20, 2006. Section 3.2 describes the high-level features of our deployment and the data.

3.1 The Experiments

The CarTel embedded computer (Figure 2) has a high-powered 802.11b miniPCI Wi-Fi card[1] with a 5.5 dBi gain omni-directional rubber-duck antenna, a Rayming TN200 GPS unit, 128 MBytes of RAM, and 1 GByte of flash memory for storage. It runs Linux 2.4.31. The `Scanping` programs run as CarTel applications on the embedded device.

During the experiments, the Wi-Fi card was used solely to perform measurements and not for any other communication. `Scanping` probes the GPS device once per second to obtain the current latitude, longitude, altitude, and speed; the GPS device also serves as the time reference when the computer boots up.

The embedded computer draws power from the car, boots up when the ignition turns on, and launches `Scanping`. It shuts down when the ignition is turned off. `Scanping` continuously loops through the following sequence of operations:

1. *Scan.* The Wi-Fi interface performs an active scan, which consists of sending probe requests and waiting for responses over all 11 802.11b channels. For each AP that is discovered, `Scanping` logs its ESSID (a human-readable string that identifies a network), BSSID (a 48-bit bit-string that uniquely identifies an AP), radio frequency, and the received signal strength of the AP's response to the probe. In addition, `Scanping` logs the AP's advertised "Privacy Bit" value, which hints if an AP has enabled 802.11 WEP encryption or other security settings[13]. The scan operation repeats in a tight loop until the interface detects at least one AP. When `Scanping` finds an AP, it proceeds to the next step.

2. *Association.* `Scanping` issues a command to the Wi-Fi interface to associate with an AP that responded to its probe. If multiple APs respond, `Scanping` associates with the AP whose response had the highest received signal strength. We patched the Linux HostAP Wi-Fi driver (v.0.2.4) [14] to override the default roaming procedure to give `Scanping` full control over initiating and terminating associations with an AP, and to propagate feedback to `Scanping` about the success or failure of the association attempt. `Scanping` logs the feedback status along with the start time and the measured duration of this operation.

   `Scanping` then jumps back to Step 1 (Scan) if the association fails. Otherwise, it launches `tcpdump` on the Wi-Fi interface to monitor and log all subsequent networking activity involving `Scanping`, and proceeds to the next step. Running `tcpdump` has proved invaluable in debugging and understanding observed performance.

3. *Address configuration.* `Scanping` uses `dhcpcd` to obtain an IP address. As explained in Section 4, caching IP addresses speeds up connection establishment substantially. `Scanping` therefore uses the AP's BSSID value to query a local cache for the AP's IP configuration information obtained from a previous drive. If an entry exists and has not expired according to the previous DHCP lease, then `Scanping` uses the cached parameters, and proceeds to the next stage in the program loop. Otherwise, it invokes `dhcpcd` to obtain a DHCP lease from the DHCP server running on the AP's network. If `dhcpcd` fails to acquire an ad-

[1]Senao NL-2511MP; Prism 2.5 chipset flashed with secondary firmware v1.7.4.

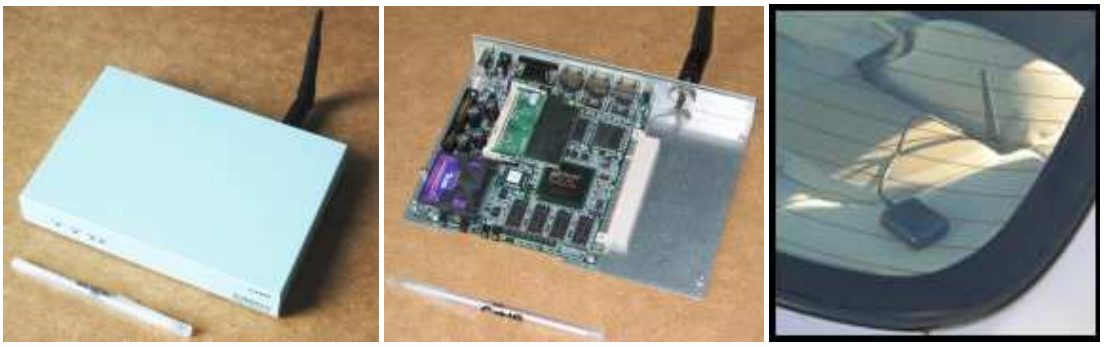


Figure 2: The measurement rig placed in each car.

dress, the client times out after $T = 5$ seconds and returns to Step 1 (Scan).

4. *Single end-to-end ping.* At this point, `Scanping` has established connectivity with the wireless network. Certain types of APs grant association rights and DHCP addresses to any client, but refuse to forward traffic for them without proper authentication. Examples of such APs include commercial hot-spot networks and wireless LANs configured with MAC address filters or firewalls. Our device can connect to such an AP and potentially waste valuable scan opportunities; moreover, by including such APs in our analysis, we might be over-estimating the connectivity available in today's unplanned Wi-Fi deployments.

   Our connectivity test simply pings our central server's IP address (to avoid a time-consuming DNS lookup). To combat potential packet losses, which can be especially high while entering an AP's coverage area [16], `Scanping` attempts this end-to-end ping every 200 ms, until the first successful one, or until 2 seconds elapse. `Scanping` only considers APs for which this end-to-end test succeeds in estimating end-to-end connectivity durations.

5. *Connectivity and uploads.*

   If the previous step succeeded, `Scanping` starts two separate processes that run in parallel. The first one pings the first-hop router, while the second one attempts TCP uploads.

   (a) *AP pings.* Our primary interest is in understanding the nature of vehicular connectivity and estimating what the capacity of unplanned Wi-Fi networks is likely to be.

   To measure the properties of the wireless channel, `Scanping` sends ping packets to the first-hop router every 100 ms. We picked this periodicity because it is similar to that of 802.11 AP beacons, but it is fine-grained enough to understand the characteristics of the channel as observed by the car.

   `Scanping` logs the time at which each AP ping was done and whether the ping succeeded or failed.

   (b) *TCP uploads.* In a subset of cases, `Scanping` opens a TCP connection to the central server and delivers data. During this upload, the car can move out of range an AP and cause the wireless interface to disassociate. At this point, `Scanping` needs to terminate the TCP connection as soon as possible, so that it can proceed to discover new APs in the next iteration of the loop. The default TCP connection termination time is many minutes, which is too long. Unfortunately, the low-level mechanism for detecting AP dis-

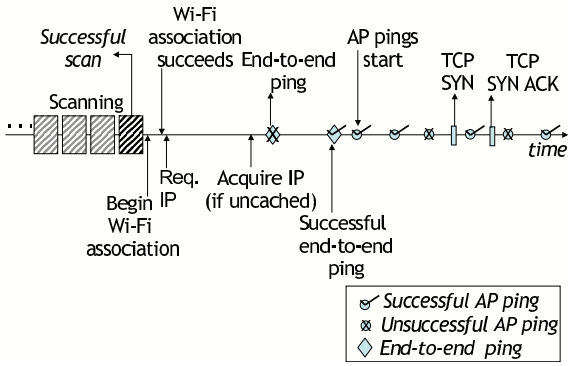


Figure 3: Timeline of network activity as the car nears an AP.

association events from our Wi-Fi card and driver is unreliable. Hence, `Scanping` uses the ping connectivity probes described above, and terminates the entire association (and hence the upload) if no responses are heard for 3 seconds.

`Scanping` logs the time at which each TCP upload was done, the total number of bytes uploaded, and the duration of the connection. `Scanping` currently attempts no downloads.

All the results reported in this paper use an 802.11 transmission bit-rate of 1 Mbit/s, the lowest bit rate. We picked this rate because it tends to have a bigger region of coverage than higher rates, and because we expect most home broadband access links to have upload bandwidths that are smaller than this number. We configured the Wi-Fi interface to use the maximum transmit power (200 mW) and used a link-layer retransmission limit of 8.

3.2 Data Summary

Figure 3 shows the typical timeline of various events of interest as a vehicle moves into the region of coverage of an access point. The salient features of our data set are shown in Table 1. This data spanned a large geographic area, as shown in Figure 1 (each box there is a cell 1.4 km $\times$ 1.6 km in size; boxes here are relatively large so they can be seen). Figure 4 shows the same information as a CDF of successful pings grouped by smaller cells of size 260 m $\times$ 270 m (this size is roughly the "typical" AP range). The CDF shows that our data set samples about 700 total cells (corresponding to an area of 49 square kilometers or a "distinct linear

No. of cars	9
Drive hours analyzed	290
Start date of analyzed data	July 29, 2005
End date of analyzed data	July 20, 2006
No. distinct days analyzed	232
No. of traces (drives)	1605
Traces with non-empty scans	1176
No. of non-empty scans	154981
No. of APs discovered	32111
No. of join attempts	75334
Successful join attempts	19576
Joins that acquired IP address	6410
Joins responding to e2e ping	4997
Distinct APs joined	5112
Distinct APs that gave IP	2259
Distinct APs that responded to AP ping	793
Distinct APs that responded to e2e ping	1023

Table 1: Data summary.

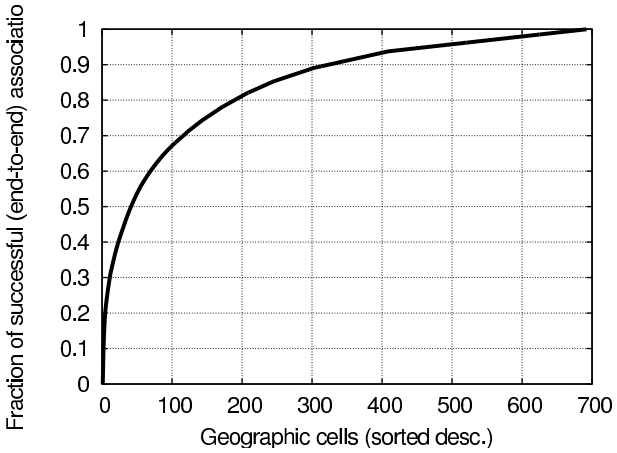


Figure 4: CDF of the geographic coverage of associations with successful end-to-end pings in our data set. We divide the world into cells of size 260 m $\times$ 270 m, sort the cells in descending order of the number of distinct successful associations, and plot the resulting CDF. The top 42 most popular geographic cells accounted for 50% of all successful associatios; the remaining 50% came from over 650 other cells.

distance"—assuming we traveled along the diagonal of each cell once—of more than 260 km). 50% of the successful pings come from 42 cells (about 15 km of distinct linear distance).

`Scanping` stores all collected data in a Postgres database on each car; another process periodically sends the data to a centralized back-end database and analysis system when no experiments are being done (specifically, when a car entered an MIT parking garage, the CarTel software in the car used the Wi-Fi network there to deliver the data collected to the central site; that Wi-Fi network was not included in our data set). The back-end also uses Postgres and includes a map-based visualization framework that produces graphs and location-based pictures over Google maps.

The database stores information about all non-empty scans and association attempts in corresponding tables. Information about observed access points is normalized out of the scan records and stored in a separate table. Location information produced by the GPS is also stored into a separate table. Wi-Fi and GPS data is cross-referenced through time stamps.

3.3 Lessons Learned

We learned several lessons from our measurement and analysis work. First, we found that our experimental setup stretches the limits of existing software and hardware, because they are not designed to operate in a challenging environment where power disruption is frequent and where the low-level Wi-Fi operations of scanning and association are controlled by a user-level process and occur at such high rates. To cope with failures while our experiments were running "in the field", we instrumented `Scanping` to detect and recover from database corruptions (in particular, we found that frequent power disruptions and flash failures can corrupt the filesystem). `Scanping` also proactively reloads the wireless interface driver whenever there is over 60 seconds of inactivity, because the Wi-Fi subsystem does get "wedged" occasionally at high rates of scanning and association.

Second, the CarTel embedded computer has a high process overhead; it often takes multiple seconds to start new processes (*e.g.*, new Perl programs) when the rest of the system (including Postgres) is running. As a result, the two programs started in Step 5 above do not start as soon as they are initiated. We did not realize this shortcoming until after all the data was collected. This problem does not significantly affect the results we report and analyze, but is nonetheless a shortcoming.

On the positive side, our use of CarTel and the reliance on a database (as opposed to flat files with scripts for analysis) was a good decision. Nearly all of our graphs are generated through short, declarative SQL statements, and Postgres (and the associated PostGIS packages) make it easy to perform relatively sophisticated analyses over geo-coded data.

Data analysis in this environment proved to be extremely time consuming, both in terms of running sanity checks to ensure consistent and explainable results, and to remove a variety of outliers. For example, when the GPS on our devices acquires a satellite fix, it automatically sets the time on the in-car device. If we are in the middle of a connection when this happens, it can cause connections to appear to be extremely long (or have a negative duration.) In the middle of a drive, this behavior can also cause the disconnection duration sample to be wrong. Many outliers are also generated by AP associations and end-to-end connections that partially complete or timeout, leaving some state in the database that must be filtered out. In addition, data analysis was also complicated by our experiments evolving with time and being refined iteratively.

4. CONNECTIVITY RESULTS

This section investigates the time it takes to perform the different steps of the experiment specified in Section 3.1. For the first three steps—Scan, Association, and IP address acquisition—we show the distribution of latencies. We then show the distribution of the time between a successful association and a successful end-to-end ping reception, and the distribution of latency between the first and last successful AP ping. This latency captures the duration over which the car maintains Wi-Fi connectivity (the time between the end of Step 2 through the end of Step 5).

Figure 5 shows four CDFs: the latency after a successful association to (1) acquire an IP address, (2) receive a response to the end-to-end ping, (3) receive the first successful AP ping response, and (4) receive the last successful AP ping response. The data used to plot each line in this graph consists only of those associations that successfully reached the corresponding phase of the association.

Figure 5 shows that the median time between AP association and IP address acquisition is about three seconds (this includes associations with and without the DHCP caching optimization discussed in Section 4.1 below.) The median time to first AP ping from the time

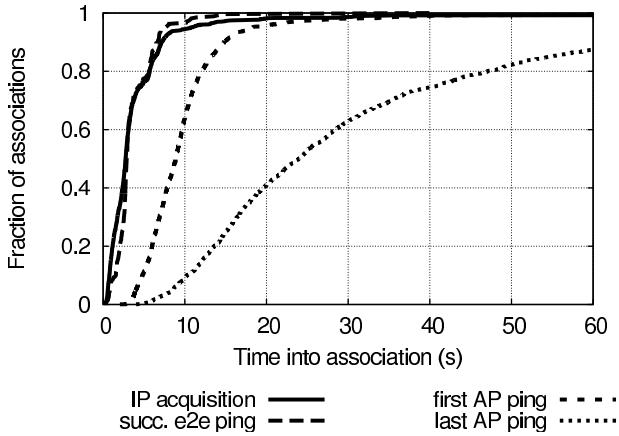


Figure 5: CDF showing the distribution of times for various phases of an association following a successful association with an AP. The small offset between IP acquisition and a successful end-to-end ping shows that, if the end-to-end ping succeeds, then the response usually comes quickly. The delay between the end-to-end ping response and first AP ping response is surprisingly high, and is an artifact of the high process initiation overhead of our experimental setup.

of association is about 8 seconds and the minimum is about 5 seconds. This several-second delay is surprisingly high. Upon investigation, we found that the primary reason is the high overhead on our embedded platform for starting a separate Perl process in Step 5 to launch the AP pings. The platform has only a small amount of memory (128 MB), and several other processes (including Postgres) use up fair amounts of it. A secondary factor are the end-to-end pings, but that CDF is almost the same as the IP address acquisition CDF (end-to-end pings are launched from the same process), suggesting that they do not contribute significantly to this delay.

Figure 5 also shows CDF of the time at which `Scanping` receives the last successful AP ping. The median is about 24 seconds. Thus, although an artifact of our setup is the inability to usefully probe or transfer data during the first few seconds following an association, we are able to carefully characterize and analyze performance over the remainder of the association. The distribution of overall association times has a heavy tail and ranges from just a few seconds to several minutes (presumably representing times when cars were connected at stop lights or in heavy traffic).

In the rest of this section, we analyze the Wi-Fi association and IP address acquisition times, overall connectivity duration (including the effects of car movement), periods without connectivity, and the distribution of an AP's coverage region in more detail.

4.1 Wi-Fi Association and IP Address Acquisition

Scan and association latency. Figure 6 shows the CDF of scan and association times. The minimum, mean, and maximum scan times are 120 ms, 750 ms, and 7030 ms, respectively. The minimum, mean, and maximum association times are 50 ms, 560 ms, and 8970 ms, respectively. Both distributions don't have heavy tails; in both cases, the 95th percentile is not significantly higher than the median. The outliers in the scan times are probably anomalies, while the outliers in the association data are probably due to the retries attempted after failures to associate.

The scan duration depends on the card but our measured numbers are consistent with previously published microbenchmarks. Our association times seem to be slightly higher than those reported

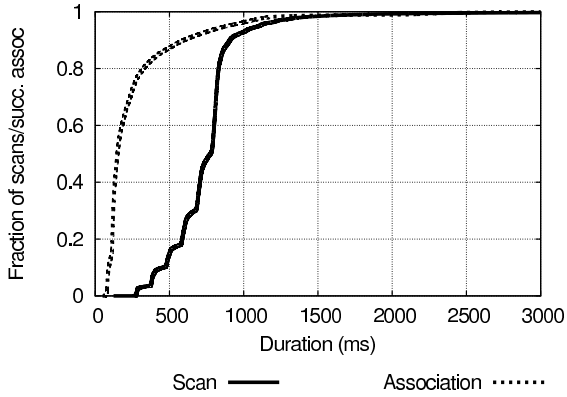


Figure 6: Distribution of scan and association times observed by our moving cars.

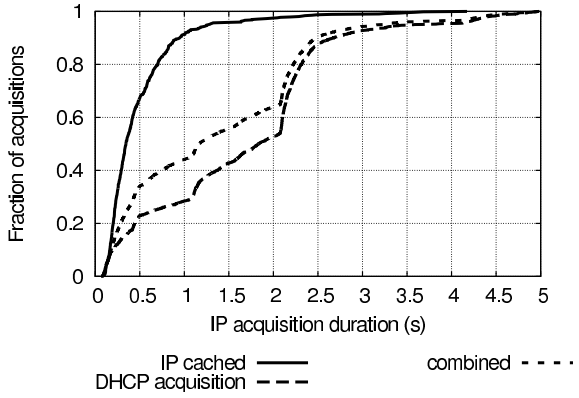


Figure 7: CDF of IP address acquisition latency, with curves showing schemes using DHCP, caching, or both. The median latencies are 1.83 s for a cache miss, 346 ms for a cache hit, and 1.38 s for the combined case. The cache hit rates we observed are between 17% and 22%.

in some previous studies [15], possibly because our network conditions are less controlled and harsher.

DHCP latency. We found that the time to acquire an IP address is often significantly higher than the scanning and association latencies. The "DHCP acquisition" curve in Figure 7 shows the CDF of the time to acquire an IP address after a successful association when using DHCP (without the caching optimization discussed below).

The main reason for the relatively high median and tail DHCP latencies is that when the car successfully associates with an AP, it may still be at the "edge" of the area covered by the AP. As a result, some link-layer frames corresponding to the DHCP request or its response may be lost. (The spikes in the CDF at 0.5, 1, and 2 seconds are artifacts of experimental runs when we set the DHCP timeout to those values.)

Accelerating initialization. These results show that when a car encounters an AP for the first time, it takes a non-negligible amount of time before it can actually send or receive data using that AP. We now investigate whether caching can reduce this latency.

To reduce scan and association latency, we considered a scheme that would cache mappings between GPS coordinates and a set of APs (with their channel parameters) based on past drives, and use that to avoid re-scanning when the car is near the same location again. Our experimental setup, however, obtains GPS coordinates only once per second, and so the location will be slightly out of date.

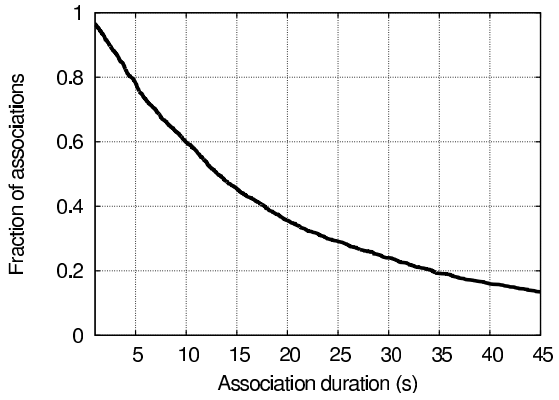


Figure 8: Complementary CDF of association duration as measured by the time between the first and last successful AP pings. The median duration of connectivity is 13 seconds and the mean is 24 seconds.

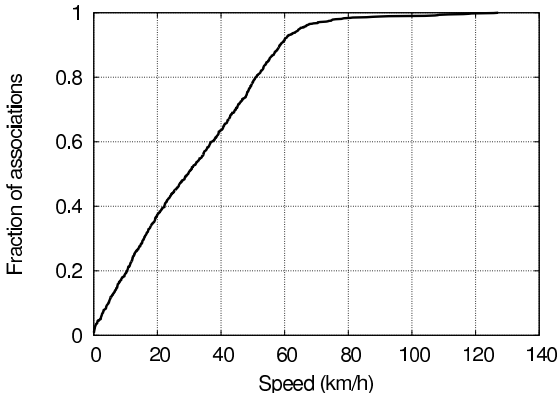


Figure 9: CDF of average speeds for associations. Associations are equally likely to be made at surface street speeds (< 60km/h). Not surprisingly, we have very few associations at highway speeds.

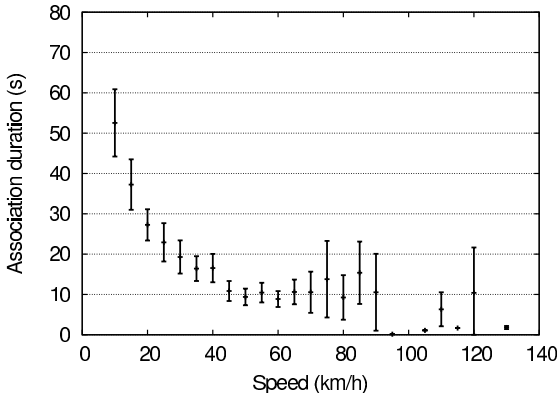


Figure 10: A plot of association durations vs. speed. Error bars indicate 95% conf. intervals. The maximum association duration is shorter at higher speeds, as expected.

It is possible to develop techniques that predict the likely current location using the past few location samples, but we didn't experiment with that idea because the scan and association times are currently a relatively small fraction of the overall idle time. However, such an optimization may be even more beneficial for 802.11a than for 802.11b, because of the larger number of channels in 802.11a.

Caching the mapping between an AP's MAC address ("BSSID") and an IP address granted by the corresponding DHCP server from the previous drive should be a significant win, because a cache hit can eliminate the DHCP delay altogether. Caching the IP address that worked the last time, while honoring the DHCP lease rules, avoids the need to negotiate a DHCP address. The "IP cached" line of Figure 7 shows that a successful cache hit greatly reduces IP address acquisition latency (the median reduces from 1.83 seconds to 346 milliseconds). We also found that for our drives the cache hit rate was between 17% and 22%.

This caching scheme can be improved in several ways. For instance, we might tag the cache with flags that show to what extent the address worked (successful end-to-end ping, successful AP ping, etc.). Such tagging would be useful because cached addresses in the current scheme turn out to have a higher probability of not leading to successful AP or end-to-end pings than when DHCP provides an address. In addition, we might also try to use a cached address even after the expiration of the lease, after running `arping` to check if the IP is currently being used. Investigating these optimizations is a topic for future work.

4.2 Connectivity Duration

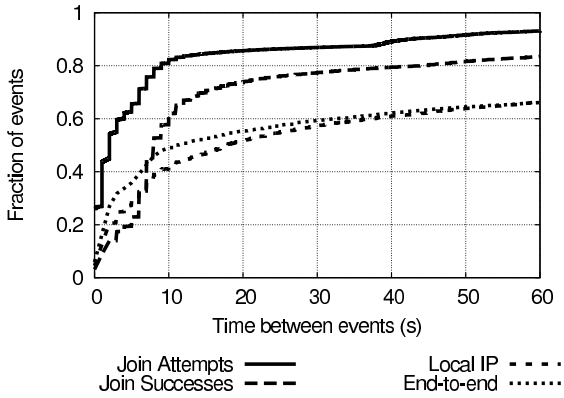
We estimate the duration of connectivity by calculating the time duration between the first and last successful AP pings for each association (Figure 8). As explained earlier, owing to the process overheads of our platform, the time to successfully send the first AP ping is several seconds long, and that is reflected in the durations being a few seconds smaller than suggested by Figure 5. We have verified that the duration between a successful end-to-end ping and the last successful AP ping is typically about five seconds longer than between the first and last successful AP pings.

One might wonder whether the connectivity results obtained so far are biased by the amount of time our cars spend traveling slowly. For example, does the bulk of our successful associations occur when the cars travel slowly or when they are stationary (*e.g.*, at a busy intersection or a traffic light)?

To answer this question, we analyzed the probability of a successful association as a function of car speed. Figure 9 shows the results. Somewhat surprisingly, the number of associations is fairly uniform across all speeds between 0 and 60 km/hour. Note that only a few of the associations were made at speeds above 70 km/hour, and in particular the number of successful associations at highway speeds is small.

There are three reasons for this lack of success at higher speeds. First, most of the driving was done on surface roads in these measurements. Second, open wireless APs are common to residential areas that are often distanced from major highways. Third, we have not optimized our measurement system to achieve sub-second associations, which would improve connectivity at highway speeds (that said, we still noticed a few successful associations at speeds above 80 km/h). Hence, we believe that the possibility of using unplanned open Wi-Fi APs from highways might still hold promise, especially if the penetration of Wi-Fi APs near highways rises.

Even though association establishment success is not directly affected by speed, associations made at lower speeds tend to have longer durations (see Figure 10). This result is as expected, and implies that more data would be transfered per AP at lower speeds.



Type of connectivity	Mean (sec)	Std. dev.(sec)
End-to-end conn.	260	642
Local IP conn.	162	447
Successful assoc.	75	300
Assoc. attempts	23	151

Figure 11: CDF of the time between connectivity events for four types of events: successful end-to-end pings, successful local IP connectivity, successful associations, and association attempts. Whereas today only the first type can actually be used, the other three events show improvements in connectivity that can arise if more access networks participate. Ultimately, observe that the mean time between association attempts is comparable to the mean time for which connectivity lasts (24 seconds from Figure 5), although connectivity is not uniformly distributed in space.

4.3 Periods without Connectivity

This section investigates the inter-arrival time observed between connectivity events, for different types of connectivity. We start with end-to-end connectivity as observed *in situ* in our data set. The mean time between successful end-to-end pings in our traces is only 260 seconds. The corresponding CDF is shown on the "end-to-end" line of Figure 11. The median is considerably lower than the mean, and the CDF shows that end-to-end connectivity events occur in clusters. In fact, the standard deviation of this duration is large, about 642 seconds. Thus, some intervals without connectivity are extremely long (witness the heavy tail in the CDF), but when successful connectivity occurs, it does so in quick succession. A cursory analysis suggests that the long durations correspond to times when drivers are in areas of low human population or traveling at high speeds (a more systematic correlation with census data is forthcoming).

We are also interested in knowing what would happen to the time duration -between connectivity events as more APs participate. To answer this question, we successively refine the set of APs to include ones for which our mobile nodes successfully obtained a local IP address, successfully associated, and simply encountered (whether open or not). The CDFs for these inter-arrival durations, as well as the mean times between events and their standard deviations are shown in the remaining lines of Figure 11. If all APs were to be used, the mean time between connectivity events reduces to just 23 seconds, which is actually comparable to the mean duration of connectivity (Figure 8).

Of course, because these events occur in bursts, one cannot conclude that the system would provide continuous connectivity. One can conclude, however, that even when restricted to the number of APs that currently permit associations or even those that provide

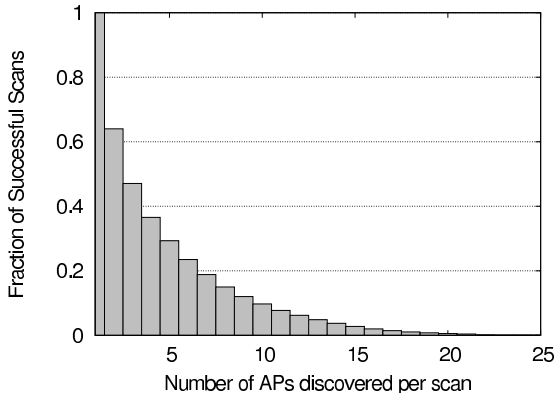


Figure 12: Complementary CDF of the number of APs discovered on a successful (non-empty) scan. Two or more APs are encountered over 65% of the time, implying that an algorithm to select one from a set of APs is necessary.

local IP connectivity, cars traveling in areas in and around cities are likely to encounter several usable APs on any drive that lasts more than a few minutes. This finding implies that it may be possible to get timely updates from (and to) vehicles in urban areas using an unplanned open Wi-Fi infrastructure.

Section 6 discusses some incentives that might enable more participation in a potential open Wi-Fi network. Although is is highly unlikely that all, or even a majority, of these APs would participate, even a 20% participation rate (*e.g.*, those APs that allow successful associations today) would reduce the mean time between connectivity events by a factor of over $3\times$.

4.4 AP Coverage

Figure 12 shows the number of APs discovered per successful AP scan (*i.e.*, any scan that found one or more APs). More than two APs are discovered over 65% of the time, attesting to the high density of APs in many areas. APs are generally found in clusters. Also note that our measurements provide a lower bound on the number of APs, because `Scanping` does not scan when it is either trying to associate or has associated successfully with an AP.

Figure 13 shows a CDF of the fraction of all associations that were made to each of the 5112 APs that we associated with, where the APs are sorted in descending order of the number of associations. The top 100 APs account for about 20% of all associations (these are the darker geographic cells in Figure 1).

We are also interested in the *coverage* of an AP. To compute this quantity, we take the set of GPS coordinates at which the car received an AP ping response. We then compute the smallest bounding box around those coordinates and log the length of the diagonal of this bounding box. This definition does not account for possible coverage holes within the region formed by these coordinates (for that reason, we don't report an area but rather the diameter).

Figure 14 shows the complementary CDF of the AP coverages observed in our data. The median AP has a coverage of 96 meters and the top 10% of associations have a coverage of more than 300 meters (recall that our mobile nodes transmit at a relatively high power level of 200 mW). We note that the coverage shown in this graph will often be a lower bound on the true coverage of the APs, because our experiments do not attempt to find the maximal coverage region. Moreover, it is very likely that the coverage is not circular or symmetric about the AP, but is likely to be quite erratic and non-convex. Hence, these numbers should be treated with these caveats in mind.

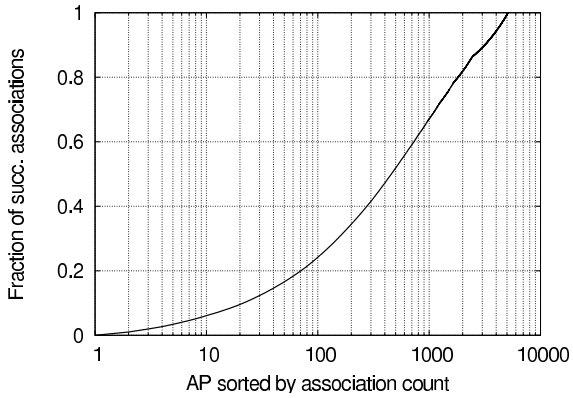


Figure 13: CDF showing the fraction of associations to any given AP. The top 100 access points account for over 20% of our associations.

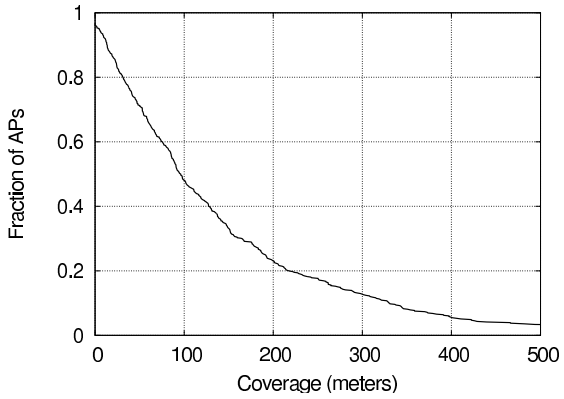


Figure 14: Complementary CDF of connection coverage. The coverage of an AP is the length of the diagonal of the smallest bounding box that encloses all GPS points from which some communication was possible with the AP.

Our results are generally consistent (except for the shape of the tail) with Akella *et al.*'s recent Pittsburgh study [3], but are considerably lower than the maximum 802.11 AP range of 800 meters reported in previous work [16] for a single access point placed directly along a road. This difference is not surprising given that our access points are likely in buildings and could be far from the road; moreover, we do not include the distance from an AP to a car in these estimates because we don't attempt to precisely localize APs. These coverages would also reduce at higher radio bit-rates.

These relatively large coverages suggest that a relatively small number of access points can cover a large area, if we are willing to tolerate some holes. For example, the city of Boston is about 130 square km, so in theory, according to our measurements, the entire city could be covered with just 2,000 (properly placed) APs. Of course, the real number is likely to be bigger because of the vagaries of real-world radio propagation.

5. PACKET LOSSES AND DATA TRANSFERS

This section describes observed wireless packet loss rates and the end-to-end performance of TCP uploads.

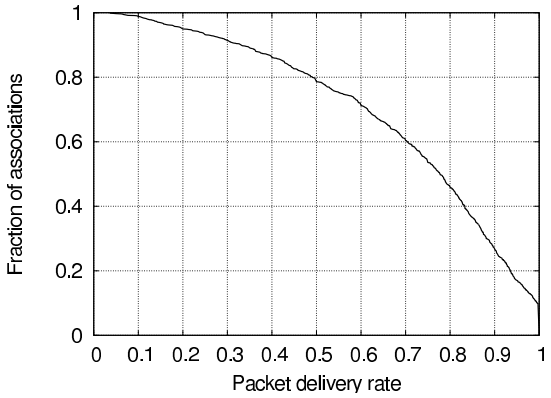


Figure 15: Complementary CDF of the fraction of AP pings that succeeded per connection. The median delivery rate is 78%.

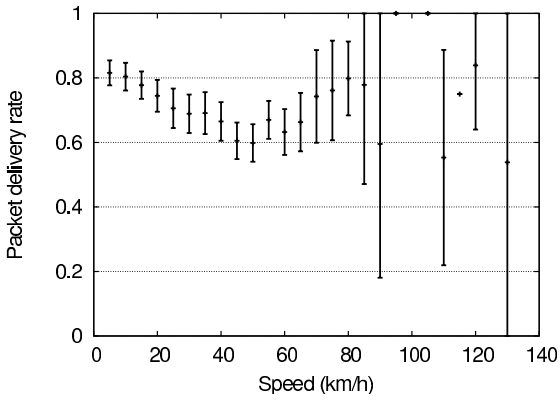


Figure 16: Mean Wi-Fi packet delivery rate vs. car speed. We partitioned the data into 5 km/hour bins and for each bin computed the average packet delivery rate of all AP pings that were initiated when traveling in the speed range represented by the bin. Error bars show 95% confidence intervals.

5.1 Wi-Fi Packet Loss Rates

We now examine the bi-directional loss rates of AP ping packets to assess the channel quality as cars move into and out of range of an AP. Figure 15 shows a complementary CDF of the probability of successful AP ping over all connections, for connections in which at least one end-to-end ping succeeded. The median connection has a delivery rate of 78%; assuming symmetric losses, this translates into an approximate uni-directional delivery rate of about 90%.

The better understand the distribution of losses throughout the connection, we studied the loss rate of our AP pings in different phases of long connections. We found that end-to-end loss rates are only about 10% at the time AP pings begin (recall that this is several seconds into the connection, as shown in Figure 5), and that loss rates remain at this level until the last few seconds of the connection, when they become quite high, often exceeding 80% as the car moves out of range of the AP. We speculate that if we did not have our initial address acquisition delay or end-to-end ping, we would have seen high loss rates at the beginning of connections as well, as has been observed in prior work [16].

Figure 16 plots the mean AP ping success rate observed at different speeds. Our data set has little data at higher speeds, which is why some speeds do not have points. There appears to be no correlation between speed and packet delivery rates. This result is not

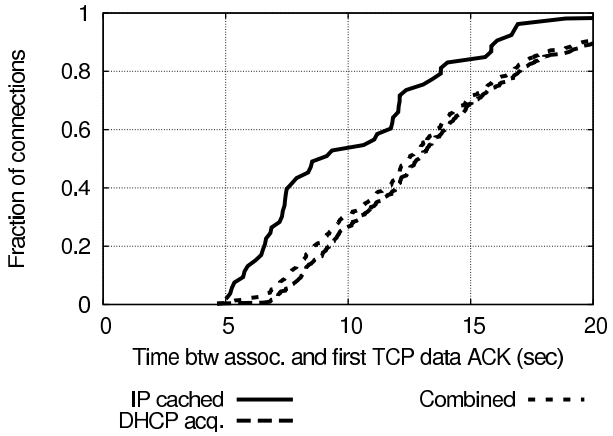


Figure 17: CDF of duration between association and first TCP data ACK reception at the client, broken out by IP acquisition method. Our results show that for APs whose IP addresses we were able to cache, the median time to the first payload data ACK is reduced by about 4 seconds (to 9.13 seconds from 12.9 seconds). Only 53 connections used caching here, so we consider these numbers to be preliminary.

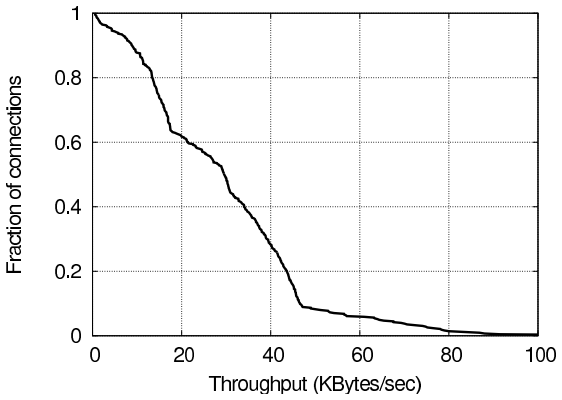


Figure 18: Per-connection end-to-end throughput complementary CDF. The median throughput is about 30 KBytes/sec, which is consistent with the upstream bandwidth of most cable modem backed APs.

too surprising; because the car-to-AP latency is only a few milliseconds, the distance traveled by a car in this time is small.

5.2 TCP Throughput

This section analyzes the performance of TCP uploads from moving cars. These uploads were done only on a small subset of all AP associations. We begin by looking at the time from initial association until we receive the first TCP acknowledgment (ACK). This time distribution, shown in Figure 17 (broken down by IP address acquisition method), captures the time taken for short single-packet reliable transfers. Observe that the minimum time until any data is successfully transmitted is about 5 seconds, which as mentioned before is mainly an artifact of our resource-constrained implementation platform. This time also includes other overheads, including first-hop router address resolution using ARP (about 1.5 seconds in our experiments) and DHCP (if used).

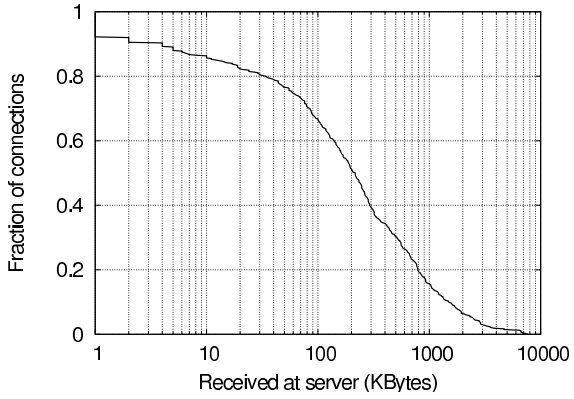


Figure 19: Per-connection bytes received at the server complementary CDF. The median bytes received is about 216 KBytes.

Once a connection has been established, `Scanping` begins transmitting data from the car to the server. Figure 18 shows the complementary CDF of the per-connection end-to-end throughput. The median connection throughput is about 30 KBytes/s. 80% of all the connections (between the 10th and 90th percentiles) observe TCP performance between 10 KBytes/s and 50 KBytes/s, a fairly narrow range). These results are not surprising in light of the fact that in urban Wi-Fi networks, most APs reside on home networks that receive their Internet connectivity through cable modems or DSL links. Typically, these types of connections offer upstream bandwidths that are roughly similar to the values we observe in this data. Because we use a 1 Mbit/s radio rate, we don't see any throughput results that are higher than 100 KBytes/s.

The relatively high packet loss rates shown in the previous section may also impair TCP throughput, although the connection attempts that actually succeeded in transferring some data over TCP probably had a lower average loss rate than those that only succeeded in transmitting AP pings. Moreover, as we discussed the average reported in the previous section is not uniform over the duration of the connection since the middle of a connection, when most data is sent, has a lower packet loss rate than the end.

Figure 19 shows the CDF the total number of bytes transferred via TCP per connection. The median connection transfers about 216 KBytes of data, which, at 30 KBytes/sec, suggests a connection duration of about 8 seconds. This number is consistent with our analysis of server logs, which show that the typical connection has packets arriving at the server for about this much time. Some connections (about 10%) transfer only one packet, and in some of those cases the TCP ACK was not received by the mobile unit. Notice from the CDF that the number of bytes delivered has a heavy tail, presumably corresponding to times when a car is at a stop light or is in heavy traffic.

6. DISCUSSION

The measurements in previous sections show that there is a surprising amount of connectivity in metropolitan areas even with the relatively small fraction of access points that are currently open. Our data suggests that the idea of individual users banding this connectivity together to form a home-grown, community-based, nearly-ubiquitous network holds much promise. While the idea is appealing, there are a number of technological, social, and legal questions that arise in this context, namely:

- How can we incentivize users to open their networks and opt into such a service? How do we convince ISPs to allow this?

- What will connectivity in such networks look like as more APs join?
- How will the network capacity scale as more mobile clients (*e.g.*, cars) participate?
- What transport protocol features will such a network require?
- How do we prevent roaming mobile users from monopolizing the resources of users who donate their Internet connections?

We address some of these questions below.

6.1 Toward Open Wi-Fi Networks

For these kinds of open Wi-Fi networks to become a reality, it is important to provide users and service providers with incentives for opening their APs. There are two potential incentives: promising participants free service, and giving participants money. For example, the recently proposed Fon network provides users who open their APs free access to all other users' APs, but charges users who have not opened their APs to use the system. Fon then keeps some of the money and proposes to redistribute the rest to the ISPs involved, and perhaps also to the owners of the APs (especially those that do not wish to become clients of other owners' APs).

This approach is not the only possible economic model, and we are certain that other models will emerge in the future (only time will tell what might succeed in the marketplace). We also note that Fon does not target highly mobile users moving in cars, while that is our main focus in this paper.

A tiered security model for Wi-Fi APs would also be a good idea; home users should be able to encrypt their traffic, while leaving APs usable by subscribers. A simple approach would be to use three tiers, one for the owners, one for subscribers, and possibly a third one for everyone else.

To reduce the impact on an AP owner's data transfers in an open network of this kind, owners should be able to set different rate limits for these tiers. Furthermore, as we discuss below, there are a number of optimizations that can be made to further reduce the potential impact of greedy mobile users.

ISPs will presumably be willing to participate in such an arrangement provided they are given sufficient financial incentives. An ISP's primary concern is likely that users will use the open network rather than paying for an Internet connection of their own; a proper fee schedule or connection time limits on the open network can obviate these concerns.

Fortunately, the legal issues in this case favor users who open the networks, at least in the US. Here, so-called "safe harbor" laws protect participants from being liable for the actions of users of their access points, just as they protect ISPs (as long as the participants agree to comply with law-enforcement officials in stopping or tracking down malicious users on their APs).

As an alternative to community-driven open networks, it is possible that municipalities will simply install a large number of open access points, as they are starting to do in some cities [24]. Though in some cases these networks will be free, it seems very likely that many cities will charge some fee to partially subsidize the cost of deployment and maintenance. Furthermore, such networks are likely to span only the dense cores of cities, rather than smaller communities or suburban areas. Hence, we expect that the future metropolis will consist of a mix of commercial and municipal Wi-Fi networks, as well as community-driven "open" models. Many of the results reported in this paper apply to municipal and community Wi-Fi networks.

6.2 Connectivity and Network Transport in Open Wi-Fi Networks

In either the open community or municipality-driven model, it is interesting to explore what connectivity will look like from the perspective of a mobile user. One possibility is that connectivity will be continuous. Based on our analysis in Section 4.3, however, we believe that it is unlikely that at vehicular speeds it will be possible to continuously maintain connectivity with at least one AP.

If connectivity for mobile users does turn out to be continuous, the transport layer solutions will most likely consist of Mobile IP with some set of techniques for fast connection handoff [5, 20]. If connectivity is discontinuous, however, there are a number of open issues. First, the API that applications use to connect to each other will change, since disconnectivity suggests that connection-oriented protocols are no longer appropriate. Instead, a natural solution is to expose the non-continuous nature of the underlying connectivity by means of *application level framing* [7], where applications express their transmissions in application defined data units (ADUs.) The exact nature of this interface is an important area for future exploration. Recent work on delay-tolerant networks can inform the design of such a stack [8, 17, 21, 22, 12].

6.3 Fast, Friendly Connection Establishment

Regardless of whether such networks are continuous or intermittent, an essential feature of future network stacks tuned for these mobile environments is that they provide fast connection establishment that provides fair sharing of available bandwidth. We explore three possible optimizations in this area, related to: (1) timing of TCP connection initiation, (2) time-based fairness, and (3) aggregating bandwidth across multiple access points.

Connection initiation timing: There has been substantial work on using TCP over wireless [4]. Most of this work focused on wireless clients with little or no mobility. As previous work suggests [16], wireless LAN connections at vehicular speeds go through a set of phases as the client moves relative to the AP. High losses at the beginning of a TCP connection could dramatically reduce the overall throughput of the connection. In our experiments, TCP connections start only a few seconds after a successful association. Optimizations to the transport protocol (*e.g.*, [17]) might be able to better address this issue.

Fairness: One way to improve fairness and avoid over-utilizing the connections of users who donate their access is to use rate limiting at APs or on clients, or to use cooperative transport protocols like TCP Nice [25] (appropriately tuned for wireless access).

Another fairness related issue is that most Wi-Fi interfaces are designed to adapt their bit-rates according to the link condition, selecting lower bit-rates to overcome frame errors and signal-to-noise ratio in a lossy link. Our experiments show that a significant fraction of connections suffer from high loss rates. Therefore, the wireless interfaces on the urban network clients are likely to operate in low bit-rates most of the time.

Unfortunately, although lower bit-rates help reduce link losses, they have the side effect of reducing the capacity of Wi-Fi networks [11, 23]. In certain situations, a home client's throughput might reduce to an eighth of its original value as a result of an urban client's connection. Such anomalies can occur even when there are no packet losses in any of the links. Thus, the anomaly is strictly a link-layer problem that cannot be addressed by higher layers. To solve this problem, Tan and Guttag [23] suggest that time-based fairness be used for scheduling access to the channel. We believe that using a similar mechanism for Wi-Fi clients will limit the impact of bit-rate differences on throughput in open Wi-Fi networks.

Aggregating Bandwidth over Multiple Access Points: The authors in [19, 18] describe different systems that aggregate bandwidth across multiple cellular towers of several different cellular networks. Their measurements show that bandwidth aggregation can provide a several-fold improvement in TCP throughput over connections that use only a single cellular tower.

Our measurement results suggest that the same techniques can be used to improve throughput for unplanned Wi-Fi networks. In 65% of the AP scans, we find at least two APs. When an urban network client finds multiple APs after a scan, it can use a system like MultiNet [6] or multiple wireless interfaces to simultaneously associate with different APs and use them.

One premise of bandwidth aggregation is that the concurrent connections should operate in orthogonal channels so that simultaneous transmissions do not interfere with each other. Indeed, a Wi-Fi measurement study discovered that over 60% of open AP deployments are configured to operate in 3 channels so that it is very likely that any two nearby APs may operate in an overlapping channel [2]. In this case, the concurrent uplink transmissions would be serialized at the link-layer, as the multiple interfaces contend to use the same or overlapping radio channel.

Despite serialized transmissions, connecting to multiple APs can still improve throughput as long as the connections sharing the overlapping channels do not saturate the wireless link. In Section 5.2, we show that the median throughput of 30 Kbytes/sec remains well below 802.11b's saturation point for a TCP connection at the lowest bit-rate of 1 Mbit/s [23]. Thus, the wireless link may have spare capacity to support concurrent connections using overlapping channels.

7. CONCLUSION

This paper was motivated mainly by curiosity: given the proliferation of Wi-Fi networks in residential areas in and around cities, would an unplanned, community-driven "open Wi-Fi" wireless network composed of *in situ* access points be feasible and perform well? We are primarily interested in understanding these issues for clients moving at vehicular speeds in and around cities.

Our findings are that, even if only about 3.2% of all currently deployed APs participate in the system (as in our experiments, in which we discovered 32111 APs and were able to obtain end-to-end ping connectivity from just 1023), clients can potentially remain connected for several seconds (24 seconds on average) and send tens of kilobytes of data per second each time. The mean period of disconnectivity in our data set was 260 seconds, a number that will reduce dramatically with higher participation. We also found that the number of connections established at different speeds was the same across a range of urban car speeds (from 0 to 60 km/hour).

Of course, there are commercial, legal, and policy issues to be ironed out for this open Wi-Fi network vision to become reality. Our technical conclusion is that such grassroots Wi-Fi networks are viable for a variety of vehicular applications, particularly ones that can tolerate some intermittent connectivity including mobile sensor networks with cars, e-mail, messaging applications, Web access using pre-fetching and caching, etc. We are optimistic that such alternate networks will appear in the near future supporting both peripatetic clients and users in vehicles.

Acknowledgments

We are grateful to Eugene Shih for his contributions to this project. We thank Michael Walfish and the Mobicom reviewers for their many useful comments. This work was supported by the T-Party Project, a joint research program between MIT and Quanta Computer Inc., Taiwan, and by the National Science Foundation under grants CNS-0205445 and CNS-0520032.

8. REFERENCES

[1] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris. Link-level measurements from an 802.11b mesh network. In *Proc. ACM SIGCOMM*, Portland, OR, Aug. 2004.

[2] A. Akella, G. Judd, S. Seshan, and P. Steenkiste. Self-management in Chaotic Wireless Deployments. In *Proc. ACM Mobicom*, Cologne, Germany, Aug. 2005.

[3] A. Akella, G. Judd, S. Seshan, and P. Steenkiste. Self-management in Chaotic Wireless Deployments: Extended Version. *Wireless Networks (WINET)*, 2006. Special issue on selected papers from MobiCom 2005. To appear.

[4] H. Balakrishnan. *Challenges to Reliable Data Transport over Heterogeneous Wireless Networks*. PhD thesis, Univ. of California, Berkeley, Aug. 1998.

[5] R. Caceres and V. N. Padmanabhan. Fast and Scalable Handoffs in Wireless Internetworks. In *Proc. 2nd ACM MOBICOM Conf.*, Nov. 1996.

[6] R. Chandra, V. Bahl, and P. Bahl. MultiNet: Connecting to Multiple IEEE 802.11 Networks Using a Single Wireless Card. In *Proc. IEEE INFOCOM*, Hong Kong, Mar. 2004.

[7] D. Clark and D. Tennenhouse. Architectural Considerations for a New Generation of Protocols. In *ACM SIGCOMM*, pages 200–208, 1990.

[8] K. Fall. A delay-tolerant network architecture for challenged internets. In *Proc. ACM SIGCOMM*, pages 27–34, 2003.

[9] R. Gass, J. Scott, and C. Diot. Measurements of In-Motion 802.11 Networking. In *Proc. WMCSA*, Apr. 2006.

[10] R. V. Hale. Wi-Fi Liability: Potential Legal Risks in Accessing and Operating Wireless Internet. *Santa Clara Computer and High Technology Law Journal*, 21:543, 2005.

[11] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda. Performance anomaly of 802.11b. In *INFOCOM*, April 2003.

[12] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, E. Shih, H. Balakrishnan, and S. Madden. CarTel: A Distributed Mobile Sensor Computing System. In *Proc. ACM SenSys*, Nov. 2006. `http://cartel.csail.mit.edu`.

[13] IEEE93. *Draft Standard IEEE 802.11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. IEEE P802.1-93/20b0.

[14] J. Malinen et al. HostAP Linux driver for Intersil Prism 2/2.5/3 wireless LAN cards and WPA Supplicant. `http://hostap.epitest.fi/`.

[15] A. Mishra, M. Shin, and W. Arbaugh. An Empirical Analysis of the IEEE 802.11 MAC layer handoff process. *SIGCOMM Comput. Commun. Rev.*, 33(2):93–102, 2003.

[16] J. Ott and D. Kutscher. Drive-thru Internet: IEEE 802.11b for Automobile Users. In *INFOCOM*, 2004.

[17] J. Ott and D. Kutscher. A Disconnection-Tolerant Transport for Drive-thru Internet Environments. In *INFOCOM*, 2005.

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

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

[20] S. Seshan, H. Balakrishnan, and R. H. Katz. Handoffs in Cellular Wireless Networks: The Daedalus Implementation and Experience. *Kluwer Journal on Wireless Personal Communications*, Jan. 1997.

[21] A. Seth, S. Bhattacharyya, and S. Keshav. Application Support for Opportunistic Communication on Multiple Wireless Networks. `http://blizzard.cs.uwaterloo.ca/keshav/home/Papers/data/05/ocmp.pdf`, 2005.

[22] A. Seth, P. Darragh, S. Liang, Y. Lin, and S. Keshav. An Architecture for Tetherless Communication. In *DTN Workshop*, 2005.

[23] G. Tan and J. Guttag. Time-based Fairness Improves Performance in Multi-rate Wireless LANs. In *USENIX*, 2004.

[24] B. Tedeschi. Big Wi-Fi Project for Philadelphia. *New York Times*, Sept. 2004.

[25] A. Venkataramani, R. Kokku, and M. Dahlin. System Support for Background Replication. In *Proc. 5th USENIX OSDI*, Boston, MA, Dec. 2002.