Design and Performance Considerations for Vehicular Network Access

A Dissertation Presented
by
Pralhad Dinesh Deshpande

to
The Graduate School
in Partial Fulfillment of the
Requirements
for the Degree of
Doctor of Philosophy
in
Computer Science
Stony Brook University

August 2012

Stony Brook University
The Graduate School

Pralhad Dinesh Deshpande

We, the dissertation committee for the above candidate for the
Doctor of Philosophy degree, hereby recommend
acceptance of this dissertation.

Dr. Samir R. Das, Dissertation Advisor
Professor, Department of Computer Science

Dr. Himanshu Gupta, Chairperson of Defense
Associate Professor, Department of Computer Science

Dr. Jennifer Wong, Committee Member
Assistant Professor, Department of Computer Science

Dr. K. Wendy Tang, External Committee Member
Associate Professor, Department of Electrical and Computer Engineering,
Stony Brook University.

This dissertation is accepted by the Graduate School

Charles Taber
Interim Dean of the Graduate School

Abstract of the Dissertation

Design and Performance Considerations for Vehicular Network Access

by

Pralhad Dinesh Deshpande

Doctor of Philosophy

in

Computer Science

Stony Brook University

2012

The 'Connected Car' concept has gained considerable interest in recent years from the car manufacturers. The 'Connected Car' is envisioned to be connected to the Internet over wireless data networks enabling services like remote vehicle monitoring and diagnostics, real-time navigational assistance, sensor data collection and access to Internet media on the go. Although cellular data connectivity is largely ubiquitous in the developed world, it provides limited bandwidth and is quite expensive. Also, licensed spectrum is a scarce resource and cellular network providers are constantly faced with the challenge of providing good performance to an ever increasing user base. We posit that road-side WiFi networks have the potential to provide high bandwidth and inexpensive wireless connectivity to moving vehicles and can be beneficially used for offloading data away from the cellular data networks. In this dissertation we perform measurement analysis to show feasibility of such offloading. We also develop techniques to show how such offloading could be performed. We follow up with further techniques to improve vehicular WiFi access.

First, we investigate a hybrid wireless access network design that integrates 3G and WiFi networks under vehicular mobility. The goal is to shift the load from the expensive 3G networks to the less expensive WiFi network without hurting the user experience. Instead of simply striping data over two network connections, we develop a utility and cost-based formulation that decides the right amount of load that can be put on the 3G network to maximize users benefit. We develop and experiment with a scheduler to do this. We show via extensive measurements on a metro-scale WiFi network and a nationwide 3G network that the hybrid design is able to deliver much superior mobile video streaming experience for the user while reducing the load on the 3G network by three-fourth.

Then, we focus our attention to improving vehicular WiFi performance. We argue that mobility and connectivity information along drives can be predicted with good accuracy using historical information such as GPS tracks and RF fingerprints. We exploit such information to develop new handoff and data transfer strategies to reduce connection establishment latency and to improve download performance.

Next, we develop Brave – an SNR-based rate adaptation algorithm for vehicular WiFi access environments. Because of the highly dynamic nature of the outdoor vehicular WiFi link, BRAVE only considers short history to make rate selection decisions and out-performs other well-known frame-based and SNR-based algorithms.

Finally, we show how a multi-radio multi-vehicle system can improve the perceived coverage and throughput performance of vehicular WiFi clients. Using a metro-scale WiFi deployment we experimentally demonstrate that with intelligent access point filtering, a single multi-radio vehicular client can effectively mask connection establishment latencies completely and using another such vehicle as a relay can mask coverage holes to a large extent.

Dedicated to Aai, Baba and Nabha.

Contents

List of Figures

List of Tables

Acknowledgements

I am greatly indebted to my advisor, Professor Samir R. Das for his guidance in the course of my Ph.D. The key ideas and concepts in this dissertation came out of innumerable meetings we had over the course of the last 4 years. His guidance has been invaluable in developing my research methodology; and writing and presentation skills. I also thank his wife June for hosting several get-togethers, which I always looked forward to.

At the start of my Ph.D., the Government of Goa, India awarded me a substantial scholarship for which I shall remain forever indebted. I also thank the Chief Minister of Goa, Shri Manohar Parrikar for his incessant efforts in promoting education.

Pandurang Kamat and Arati Baliga advised me through each and every stage of my graduate career; right from applying to grad schools to job hunting. They are both very dear to me and I thank them profusely.

I thank Vishnu Navda, Ramachandran Ramjee and Venkat Padmanabhan for having me over at Microsoft Research as an Intern. Their invaluable advice and constructive criticisms during the course of my internship helped improve my perspective on systems research.

Over my stay at Stony Brook, several friends and colleagues helped me in different ways. I will cherish these friendships and professional associations forever. In the following I take great pleasure to mention at least a few of them.

It was a pleasure and an extremely enriching experience to work with Anand Kashyap and Anand Prabhu Subramanian as an upstart Ph.D. student. They went above and beyond the call of duty to handhold me through

my initial learning curve. I fondly remember my mid-day trips to the cafeteria with Anand Kashyap, Anand Prabhu Subramanian, Shweta Jain, Ritesh Maheshwari and Bin Tang, which very often initiated long brainstorming sessions. Utpal Paul and I started our Ph.Ds. together and although there isnt any overlap in our work we had several wonderful discussions about life and work, which often served very well to kill boredom. Towards the end of my Ph.D., I got to work with Zafar Qazi who I believe is a very promising student. The thoroughness with which he approaches problems will serve him very well in his academic pursuits. I wish him all the very best. I was fortunate to work with two very hard working and focused exchange students from China, Jing Cao and Zhibin Dou. Their extraordinary work ethic always amazed me. Akshay Athaley has been around during the entire course of my Ph.D., and I thank him for letting me use his lab equipment whenever needed.

I enjoyed the company of my friends Omkar Aphale, Saurabh Joglekar, Akshay Patil, Sujan Dhabolkar, Santosh Kulkarni, Prasad Kerkar, Ravi Dessai, Ashish Lohia and Shishir Das. I could always count on them for any help at any time of the day.

Finally, I would like to thank Aai, Baba and my sister Nabha for believing in me and always being just a phone call away. This dissertation is dedicated to them.

Chapter 1

Introduction

Over the past decade and a half, there has been a significant research and development effort to provide wireless connectivity to moving vehicles. The focus has been on both vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) wireless communication. Even though the original motivation of networked vehicles was to promote traffic safety, it is becoming increasingly clear that many more applications are possible. Already, OnStar Corporation which is a subsidiary of General Motors provides subscription-based communications, in-vehicle security, hands free calling, turn-by-turn navigation, and remote diagnostics systems throughout the United States, Canada and China. The service boasts of over 6 million customers. Other car manufacturers are also following suit and are buiding prototypes with even more applications like dashboard access to maps, Internet media and social networking.

Researchers have attempted to answer several questions in this new domain of networking using various methodologies. Several research groups have proposed the use of road-side WiFi networks as infrastructure for V2I communication. As such [24] conducts a measurement study to characterize vehicular WiFi access using open WiFi networks.

Some questions such as routing in vehicular ad-hoc networks (VANETs) [68] and impact of VANET protocols on road traffic congestion [33, 40] can be answered via large scale simulations. To this end, traffic and network simulators such as TraNS [91] have been developed. TraNS links a road traffic simulator, SUMO [19] with a network simulator, ns2 [77] allowing the information exchanged in VANET protocols to influence the vehicle behavior in the mobility model.

While some research questions can only be answered via simulations, other questions such as those relating to vehicular wireless access protocol design are better addressed by following experimental approaches. Vehicular WiFi access is characterized by short durations of WiFi connectivity and frequent handoffs. Several academic papers [35, 24, 41] have attempted to address the problem of reducing connection establishment latency in order to increase the duration of connectivity. Transport layer protocols that maintain sessions transparently to changing IP addresses [35, 88] have also been proposed.

Significant research effort has been done in improving the link layer performance during vehicular WiFi access. [82] and [94] have addressed the problem of improving link layer performance by using directional antennas. In the ViFi project [17] the link layer performance is improved by exploiting macro-diversity (using multiple APs simultaneously), and opportunistic receptions by nearby APs.

In this dissertation, we focus on designing protocols to improve V2I wireless communication. We demonstrate that a hybrid network access system that integrates 3G and WiFi networks under vehicular mobility provides better performance, reduces load on the cellular network and is more cost effective than a 3G only system. Then, we focus on improving vehicular WiFi access. The handoff techniques we develop help in reducing WiFi outages and our data transfer and bit-rate selection strategies help in improving vehicular WiFi access performance when connected.

1.1 Augmenting Cellular Data Networks with Metro-scale WiFi

Cellular data networks are a natural ally to the ‘Connected Car’. The ubiquitous coverage provided by these networks helps vehicles in remaining connected all the time. However, the bandwidth provided by these networks is quite limited. Also, pricing policies have changed. The days of unlimited cellular data access for a flat fee seem to be past us. The licensed spectrum in which such networks operate is also scarce and network providers continuously struggle to provide good performance to an ever increasing user base. Thus, offloading data from cellular to WiFi networks has been previously proposed by both networking [34, 120] and policy researchers [67, 119, 46] as well as by cellular network providers. From a more scientific point of view, the competition here is between licensed and unlicensed spectrum. The latter can only provide short range links due to power limitations (regulatory mandate) to encourage spatial reuse. The former, however, can provide much longer ranges, but likely lower bit rates due to spectrum limitations. Thus, moving bits from long range links occupying licensed spectrum to shorter range links on unlicensed spectrum is likely to remain an important question in the foreseeable future.

In the last few years, metro-scale WiFi networks are cropping up all over USA. These networks consist of thousands of access points and cover large metropolitan areas. Such networks consist of both indoor and outdoor access points. We posit that such metro-scale WiFi networks have the potential to provide high bandwidth and inexpensive wireless connectivity to moving vehicles and can be beneficially used for offloading data away from the cellular data networks.

We design and implement a hybrid network access scheme that uses both a 3G cellular data network and a metro-scale WiFi network. Our hybrid network access mechanism uses a scheduling technique based on network utility and cost functions with the aim of maximizing the benefit that a user can extract from the system.

1.2 Reducing WiFi Outages

WiFi access points have a range of a few hundred meters. This introduces frequent handoffs at the vehicular client. Absence of connectivity is a common phenomenon even in very dense WiFi deployments because of inefficient handoff policies employed in mobile clients. Previous works have addressed this problem by trying to reduce connection establishment latency [35, 24] and by proposing better handoff strategies [41]. We take a two pronged approach to this problem. First, we argue that mobility and connectivity information along drives can be predicted with good accuracy using historical information such as GPS tracks and RF fingerprints. We exploit such information to develop a new handoff strategy that out-performs previous research prototypes. Next, we experiment with a multi-radio multi-vehicle system to mask connection establishment latencies and coverage holes that may exist in WiFi deployments.

1.3 Improving Performance when Connected

Apart from improving the duration over which a vehicular client is connected to WiFi access points it is also important to make sure that it fully exploits the WiFi link when it is available. To this end, we first propose a model for distributed prefetching of web content at access points to be encountered in the near future. This data transfer strategy allows the vehicular client to better exploit the one hop link to the access point and the resulting throughput is not limited by Internet bottlenecks. Next, we develop BRAVE – an SNR-based rate adaptation algorithm for vehicular WiFi access environments. Because of the highly dynamic nature of the outdoor vehicular WiFi link, BRAVE only considers short history to make rate selection decisions and out-performs other well-known frame-based and SNR-based algorithms. In order to compare the algorithms under repeatable channel conditions, we develop and use a novel emulation methodology where a software radio-based programmable noise generator is used to emulate varying link quality under vehicular mobility.

1.4 Outline

The rest of this dissertation is organized as follows. First, we investigate a hybrid wireless access network design that integrates 3G and WiFi networks under vehicular mobility in chapter 2. Then we focus our attention on improving vehicular WiFi performance. In chapter 3, we develop a new handoff strategy that exploits historical connectivity and mobility information. In chapter 4, we show how predictive prefetching at WiFi access points can help in improving large file download throughputs. In chapter 5, we develop an SNR-based rate adaptation algorithm specifically for vehicular environments and show that it out-performs other well-known algorithms and research prototypes. In chapter 6, we show how a multi-radio multi-vehicle system can improve the perceived coverage and throughput performance of vehicular WiFi clients. Finally, we conclude in chapter 7.

Chapter 2

Integrating 3G and Metro-Scale WiFi

2.1 Introduction

Using WiFi to reduce the load on 3G networks is not new, and has been promoted by both networking [34, 120] and policy researchers [67, 119, 46] as well as 3G carriers. The users appear to prefer switching to WiFi when available, as WiFi offers better bit rate and latency as well as lower energy usage [93]. Some carriers are known to even incentivize such switching (e.g., TMobile @Home) to reduce load on their cellular networks. Some others have also been reluctant on carrying bandwidth intensive applications on smartphone platforms on specific pricing plans (e.g., iPhone/AT&T not allowing applications download on the cellular networks).

There is little publicly available data to help us estimate how much 3G bandwidth is available relative to the projected demand. However, anecdotal evidences abound pointing to limitations and under-provisioning of 3G networks in congested urban areas with the growing use of smartphones with networked apps [117]. There is also a growing feeling in the media that the current flat-rate pricing plans for 3G networks are unsustainable in the long run [75]. Indeed most vendors have capped the monthly data transfer limits to a few GBs in spite of offering a so-called ‘unlimited’ plan on their 3G networks.

Thus, taking the WiFi's reach to outdoors can indeed make perfect business sense [67, 119, 46].

From a more scientific point of view, the question can be posed in another way. The competition here is not particularly between 3G and WiFi, but between licensed and unlicensed spectrum. The latter can only provide short range links due to power limitations (regulatory mandate) to encourage spatial reuse. The former, however, can provide much longer ranges, but likely lower bit rates due to spectrum limitations. While 3G is evolving towards 4G, WiFi is also moving towards higher data rates, with 802.11n becoming common and new frontiers opening up in high data rate services (e.g., 802.11VHT).[1] Thus, moving bits from long range links occupying licensed spectrum to shorter range links on unlicensed spectrum is likely to remain an important question in the foreseeable future.[2]

2.1.1 Goals

To set a context to our work, we can take one of two possible viewpoints. One viewpoint is that of the *consumer* who has a free or low-cost access to a metro-area WiFi, but pays a per-bit cost for 3G for his/her pricing plan. The other viewpoint is that of the *provider* that operate both 3G and metro-area WiFi networks. Such examples are not uncommon. For example, AT&T presently maintains WiFi hotspots in many areas while offering a nationwide 3G service in USA. Such a provider can accommodate more users on their 3G network by moving bits to WiFi as much as possible. Our goal is to design and evaluate a *hybrid access network* that uses both 3G and WiFi for the best possible user experience while reducing load on the 3G link.

When we set aside economic questions, technical challenges abound. We address some of them, *specifically focusing on the vehicular networking scenario* – the most challenging scenario for WiFi. Using WiFi with vehicular

[1] Note that spectral efficiency is also an important issue. As a point of comparison, the spectral efficiencies of 802.11g and EVDO Rev. A or HSDPA are roughly the same. The spectral efficiencies of LTE and 802.11n are roughly the same as well.

[2] One can also relate this to the recent interest in deploying femtocells by 3G carriers. However, in this case all spectrum is licensed, though spectrum pooling by providers have been proposed [23].

mobility has many challenges: (i) The WiFi physical layer does not suit outdoor usage, particularly in long distances. (ii) Transmit power limitations limit coverage and/or bit rate. (iii) Coverage limitations give rise to frequent handoffs in a mobile scenario. (iv) Coverage 'holes' may not be uncommon. Thus, the best way to exploit WiFi would be to exploit 'diversity' – where WiFi is used opportunistically when available, while using 3G as a backup when WiFi is either not available or provides a poor bandwidth.

We make the following contributions.

1. Provide an in-depth evaluation of WiFi access with vehicular mobility using a commercially operated metro-scale WiFi network (Section 2.4). This is in contrast to prior studies that only considered open APs in the wild [35, 24] or a limited WiFi deployment [41, 17, 31].

2. Provide a head-to-head comparison of WiFi and 3G access with vehicular mobility (Section 2.4). This lays down the salient features of the two access networks to indicate how a hybrid access network should be designed.

3. Design and implement the hybrid access network that uses a scheduling technique based on network utility and cost functions (Sections 2.5 and 2.6).

4. Evaluate the performance of the hybrid access network for high-bandwidth video streaming vis-a-vis pure 3G network and demonstrate that the hybrid access provides significantly greater user experience while saving bits on the 3G link (Section 2.7).

2.2 Related Work

2.2.1 Vehicular WiFi Access

Several experimental studies have explored the potential of using intermittently available WiFi connectivity from moving vehicles for data transfers. In the Drive-thru Internet project [87, 88] controlled experiments are done with a single car driving past a single access point to measure range and connectivity in an intermittent network. More recently, the CarTel project [24] has focused on upload performance while using open APs in the wild. In the ViFi project [17] the link layer performance is improved by exploiting macro-diversity (using multiple APs simultaneously), and opportunistic receptions by nearby APs. By the nature of the work, both CarTel and ViFi focus on upload.

Downloads and intermittent connectivity have been studied in a vehicular environment in a fleet of taxis in the Cabernet project [35] with an improved handoff scheme and a new transport protocol. In our prior work in this space [31], we also have focused on downloads. The emphasis there has been predictive methods for improving handoffs and use of prefetching in the APs for better download performance.

Improving handoffs with vehicular mobility has been an important area of research. Other than our work in [31], optimized handoffs techniques have also been developed and evaluated in the Cabernet project [35] and also in [41]. One or more of these strategies can nicely complement our work.

Applications that require maintaining a session face problems in vehicular WiFi access because of intermittent connectivity. Some papers address this issue by creating a transport layer protocol that maintains sessions transparently to changing IP addresses [35, 88].

Several other papers focus on sundry issues on vehicular WiFi access. For example, link and transport layer problems are examined in detail in [48]. Web applications are examined in presence of intermittent connectivity in [18]. Link layer measurements are reported in [74]. Use of directional antenna for better connectivity has been explored in [82].

2.2.2 Characterizing 3G Access

There has been less excitement about characterizing 3G access in mobile or stationary environment. This is likely because the 3G network is a closed system and under tight operator control. Thus, much of the work is measurement-based.

The important works in this space include cross-layer measurement study to evaluate TCP performance over 1xRTT networks in [76] and over EVDO networks in [66], capacity evaluation of UMTS networks in [113]. Mobile experiments are done in EVDO networks characterizing cross-layer aspects with TCP in [70]. Bandwidth predictability is evaluated for HSDPA networks in [121]. A measurement system for city-wide measurement of WiFi and EVDO networks was developed in [86].

2.2.3 Using Multiple Network Interfaces

'Striping' across multiple network interfaces (possibly using different technologies) is a popular method to exploit larger aggregate bandwidth. This can be done in the link layer [104], network layer [90] and transport layer [73, 43, 50] depending on the available facility and application. There have been efforts to use striping over multiple WWAN data links. See, for example, the MAR project in [96] and the PRISM project in [61, 62].

The Stream Control Transmission Protocol or SCTP [110] provides a mechanism to exploit multiple interfaces. We expect that our work will benefit from integration within SCTP. Striping efforts using SCTP have been reported, for example, in [53].

Availability of multiple interfaces also helps exploit diversity for better performance, even if only one of them can be used at a time. A key enabler for this is 'vertical handoff' [109]. Vertical handoff between WLAN and WWAN networks have been studied for many years and many practical systems have been built. See, for example, [22].

2.3 Measurement Setup

2.3.1 Network

For our study we use a metro-scale WiFi deployment in the Long Island area in New York (population roughly 7 million). This service is called 'Optimum WiFi' [5] and is provided by Cablevision, a local cable TV provider and ISP. The WiFi network extends to most of populous areas of Long Island where our study is conducted. It also extends to parts of New York City, Pennsylvania, Connecticut and New Jersey, where Cablevision has service. The entire network has roughly 18,000 APs. WiFi access is provided free to all subscribers of Cablevision's TV or Internet services. We note that there are several hundreds of such metro-scale WiFi deployments in USA alone [81] While we expect that our general observations will be repeatable in another metro area deployment, we do note that more specific quantitative observations could be strongly tied to deployment density, radio characteristics of the APs and any handoff control on the APs.

For 3G access, we use Verizon's EVDO Rev. A service using the USB760 USB-based air interface available from Verizon. EVDO Rev. A service is capable of a maximum bit rate of 3 Mbps. Verizon's website [1] claims a maximum of 1.5 Mbps TCP throughout. In our experience we occasionally got a higher TCP throughput (about 2 Mbps).

2.3.2 Testbed

We use a Dell Latitude laptop running Linux as the client machine to be carried in the car. The original miniPCI WiFi interface from the laptop is removed and replaced by a carrier-grade interface (Ubiquity XR2 [6]) with transmit power set to 25 dBm. The interface uses Atheros chipset supported by the latest madwifi driver that we used. The WiFi card is connected to a high-gain (12 dBi) omni-directional antenna. We experimented with various other antennas (5 and 7 dBi) and transmit power choices (within FCC limits). However, the 12 dBi antenna provided the best connectivity. Note that due to the high gain, the beam-width for this antenna on the vertical plane is relatively small (about 9°). Thus, assuming that the APs are deployed at the height of pole-tops, this antenna is likely to connect better to a distant AP relative to a very close AP. This is because the distant AP is much closer to the horizon relative to the car. The antenna is long – about 4ft. The antenna is carried inside the car with its top part sticking out from the sunroof.

The laptop also carries Verizon's USB-based USB760 EVDO Rev. A Modem as the second network interface. A USB-based Garmin GPS receiver is connected to the logging laptop. At startup the GPS receiver is initialized and the GPS location is logged every second.

A server program runs on a lab machine with a public IP address. The server accepts incoming connections from the WiFi/3G networks and transmits 1500 byte packets continuously to the client over TCP. as selected by the client. This enables download throughput tests to be reported in the next section.

We set the WiFi card's AP selection mode to automatic, i.e. the madwifi driver decides on the best AP to connect to. Auto rate control is disabled and a fixed bit rate of 11 Mbps is used. This is based on the observations in [35]. A few empirical tests indeed verified that this was the best policy in our setup.

DHCP is used to obtain the IP address. The Optimum WiFi network allows clients to retain the IP address between associations; thus DHCP delay is incurred only once at the beginning. This also allows us to retain the same TCP socket across associations. Optimum WiFi uses a browser-based authentication for the initial network access. This is done manually. Again this step is needed only once. Authentication is retained across associations.

Finally, a word about TCP. While using the stock TCP implementation on Linux we faced problems during our initial experiments when the WiFi link loses association for more than a few seconds. This sometimes makes the retransmission timeout really long (the timeout doubles at every step). This is a well-known issue for TCP in a mobile environment [25]. To tackle this problem, we simply set the maximum retransmission timer value on the server side TCP to 1 sec. We do note that this is not a general solution and more sophisticated client-side approaches are possible. However for the scope of this measurement study this solution worked reasonably well.

2.3.3 Driving Scenarios

Two driving scenarios are used for our evaluations.

- *Long Drive:* This is approximately a 500 miles drive as shown in Figure 1. The vehicle speed varied over a wide range depending on the road traffic. Each road is driven in both direction as the APs and their signal strengths visible from one side of the road are likely to be different from

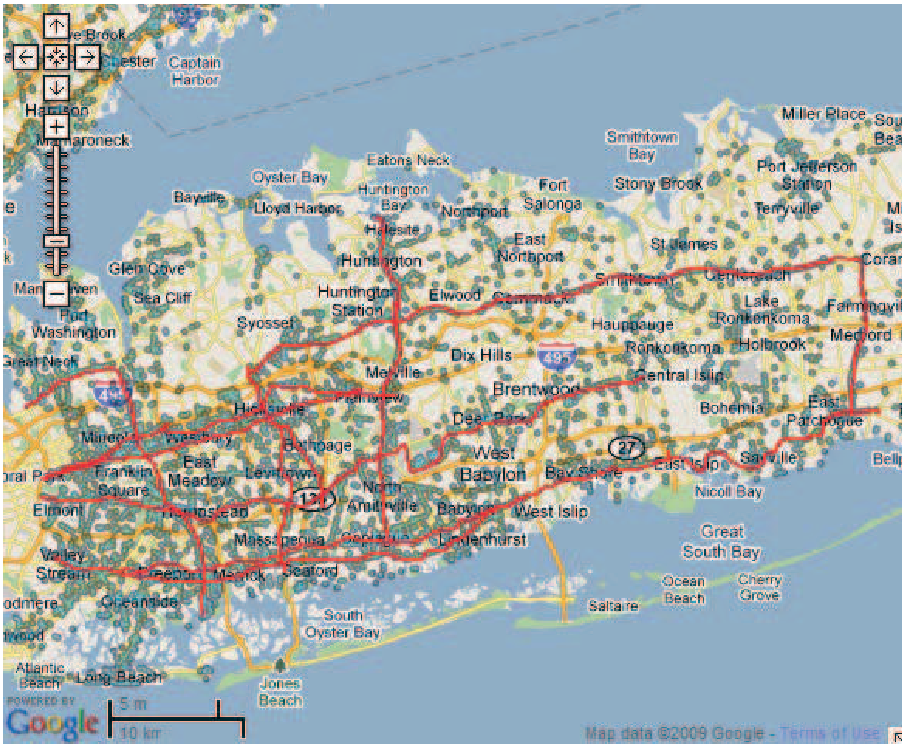


Figure 1: Map of the middle of Long Island – the area used in the long drive experiments. The route is shown in red. Approximate WiFi coverages are shown (from [5]).

the other side of the road. This drive was performed only once. This long drive provides a reasonable sample of the quality of WiFi access from moving vehicles in a metro-scale deployment scenario. The association records for the long drive show over 900 unique APs.

- *Short Repeated Drives:* This is approximately a 9 mile drive on a selected stretch where the quality of the AP coverage is reasonably good. See Figure 2. Once again this drive constitutes a round-trip for the same reason. This drive was repeated 10 times to get a statistical confidence in the experimental results.

The Figures 1 and 2 show the approximate WiFi coverage taken from the provider [5]. Verizon coverage map for the same road stretch shows complete

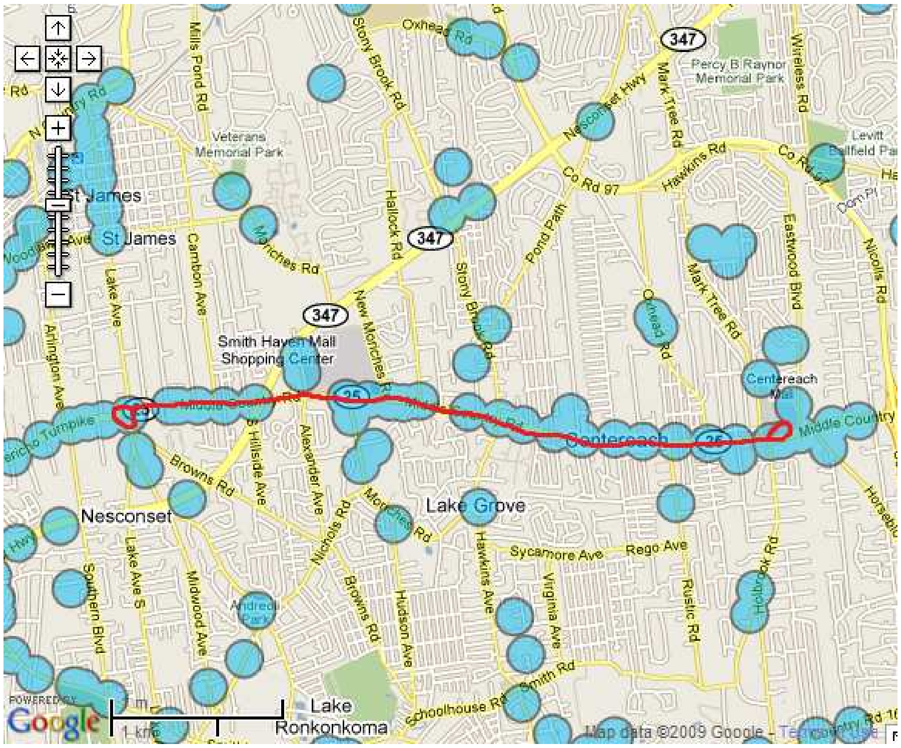


Figure 2: Map of the road stretch used for short repeated driving experiments (part of Route 25), along with the route shown in red. Approximate WiFi coverages are shown (from [5]).

uninterrupted EVDO Rev. A coverage.

2.4 Measurement Results

The laptop logs per second TCP throughputs (called instantaneous throughputs) on both the connections along with GPS location and vehicle speed. The logs are post-processed to develop the following analysis.

2.4.1 Quality of WiFi Coverage

To determine the quality of WiFi coverage we plot the CDF of run lengths (consecutive 1 sec segments) with zero and non-zero throughputs seen on WiFi.

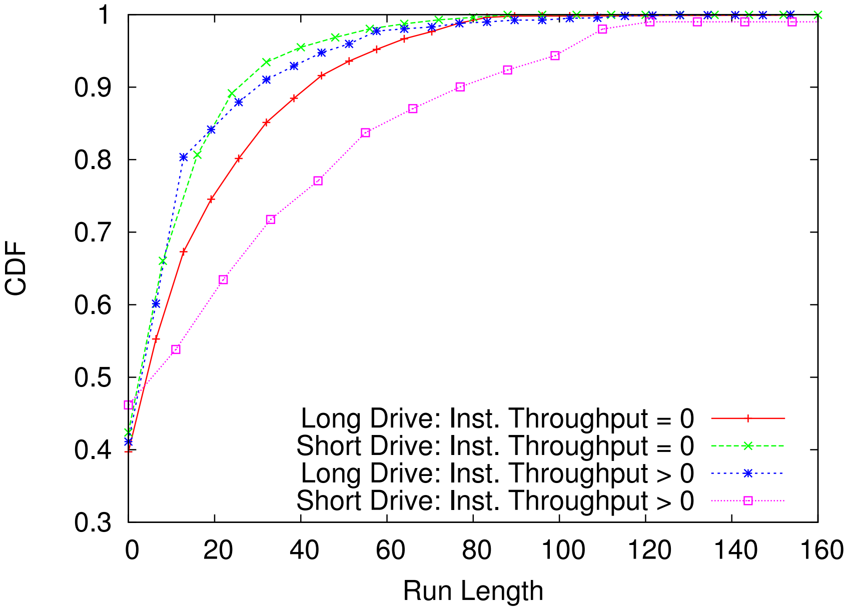


Figure 3: CDF of run lengths (consecutive 1 sec segments) with zero and non-zero throughputs seen on WiFi. Note that the short drive has zero throughput 25% of the times and the long drive 42% of the times.

See Figure 3. Note that the 90-percentile zero throughput run length is 25 sec and 45 sec, for the short and long drives respectively. The same non-zero throughput numbers are 90 sec and 30 sec, respectively. The median numbers for non-zero throughputs are very short, however, just a few seconds. This reinforces the general experience that there are frequent disconnections on WiFi with vehicular mobility. However, we will note in the next subsection that we are nevertheless able to demonstrate impressive performance with respect to 3G.

It is important to note that this evaluation does not reflect true AP visibility; rather it shows whether there is actual connectivity and non-zero throughput on TCP. True AP visibility periods are likely to be longer than non-zero TCP throughput periods as we do not use any optimized handoff technique here. The TCP is also not optimized either except limiting the timeout period. Optimizing both of these will likely produce a much better

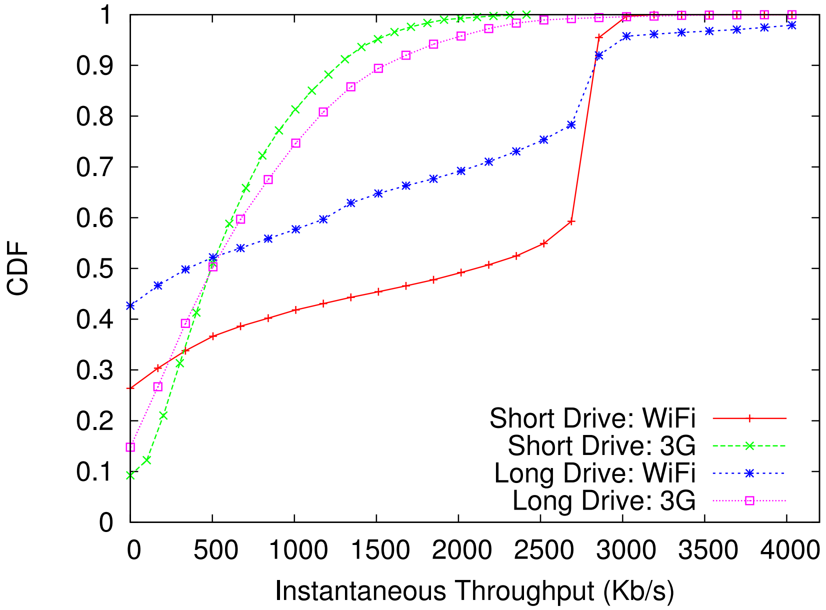


Figure 4: CDF of instantaneous TCP throughputs for WiFi and 3G.

WiFi experience [35]. Thus, our experience could be viewed as a lower bound for the WiFi performance.

2.4.2 Comparing WiFi and 3G Throughputs

In Figure 4 we compare the CDFs of instantaneous TCP throughputs on WiFi and 3G links. The long and short drives are shown separately. Note that WiFi throughputs are generally better in the short drive because of better coverage and slightly slower average driving speed experienced. However, 3G throughputs are very similar. While WiFi provides substantially better median throughput in the short drive (roughly 2400 Kbps vs. 500 Kbps) the median throughputs are similar in the long drive. However, WiFi has zero throughput on more occasions (roughly 25% vs. 10% for the long drive and 42% vs. 15% for the short drive). As noted before, this is expected as no special optimized handoff mechanism has been used. Also, WiFi coverage holes do exist (See Figures 1 and 2).

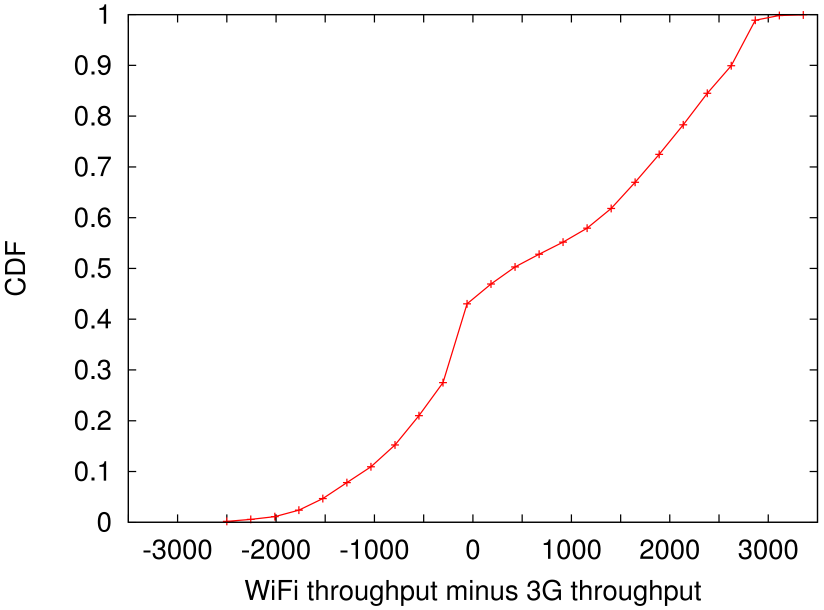


Figure 5: CDF of relative difference of instantaneous throughputs (in Kbps) between WiFi and 3G (i.e., WiFi minus 3G). Plot for the long drive only.

Also, notable is the fact that the *3G throughputs are well distributed in its entire range while WiFi demonstrates approximately tri-modal distribution* – zero or very low throughput, 0-2700 Kbps, higher up-to about 3300 Kbps. Depending on the scenario each of these three modes persists for about 25%-40% of the times. The latter high throughput regions are specifically interesting for WiFi in our context.

Now we turn our attention to relative performance differentials between WiFi and 3G in the same time instants. See Figure 5 that plots the CDF of the algebraic difference between WiFi and 3G throughputs in the same time instants. This plot is done only for the long drive. Note that the fraction of times 3G performs better (45%) is only slightly higher than the times when WiFi has zero throughput (42%). In other words, when WiFi does have network layer connectivity, it is very likely that WiFi demonstrates better throughput.

This again points to optimizing handoff making a serious impact.

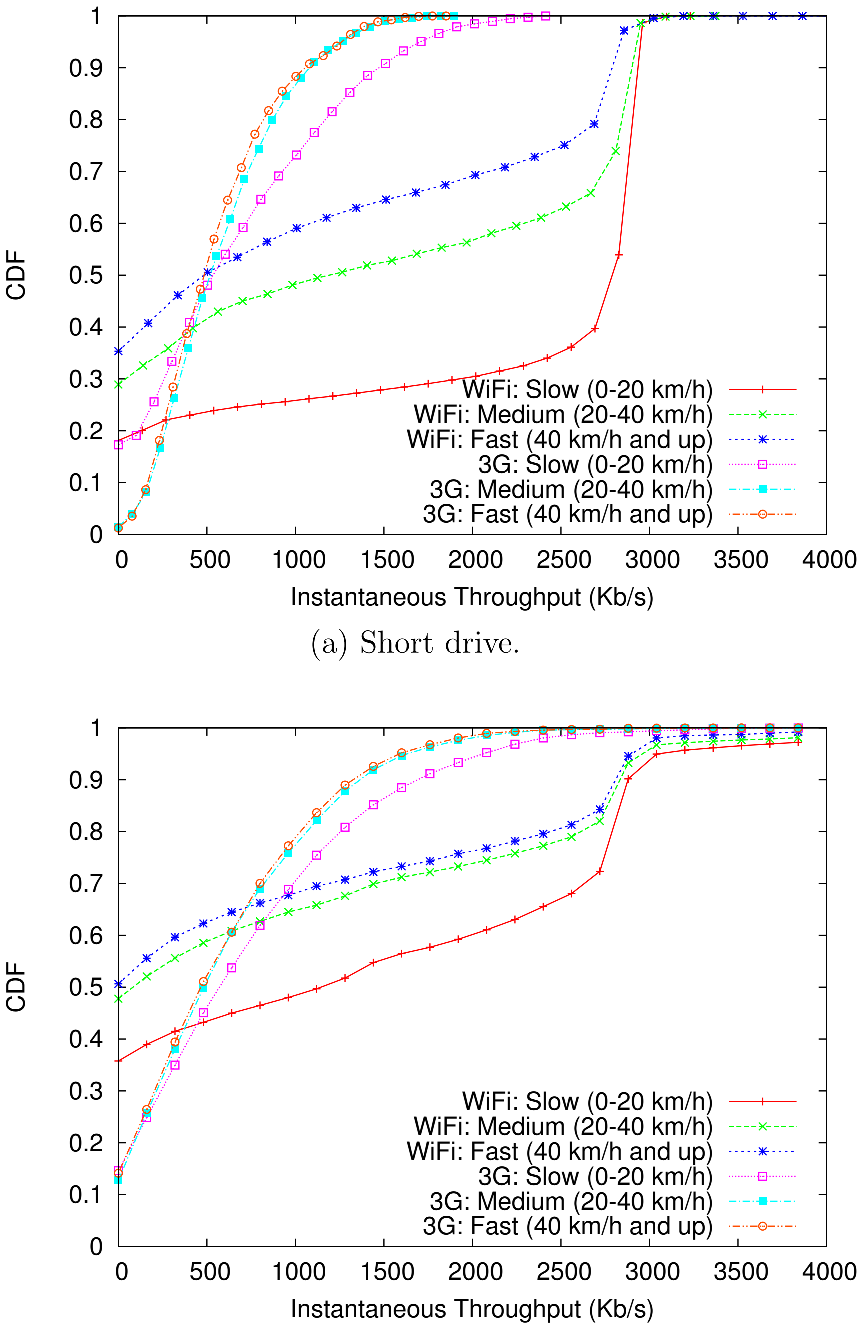
Also, when WiFi outperforms 3G, it does it overwhelmingly so. The median difference is over 1500 Kbps. On the other hand, when 3G outperforms WiFi, the median difference is roughly 500 Kbps.

We have evaluated the correlation between WiFi and 3G throughputs for the same 1 sec intervals. *The correlation is very poor.* For the short drives it is 0.03 and for the long drive it is -0.03. This shows that these two networks can complement each other quite well.

2.4.3 Correlation with Vehicle Speed

It is also interesting to find out whether high throughputs are specifically correlated to slow vehicular speeds. In Figure 6 we present speed-wise throughput statistics by breaking up the throughput data in the previous plot in three categories of speed – slow (0-20 Km/h), medium (20-40 Km/h) and high (40 Km/h and up). Note that the 3G throughput plots at different speeds are roughly similar. They also provide similar median throughputs. But the same cannot be claimed for WiFi, where slow speed has a clear advantage. This is expected as the WiFi physical layer is not built to support vehicular mobility, and the stock WiFi handoff implementations are not optimized for speed. The differences are much more for the short drive. This is likely because a better coverage better exposes the relationship between performance and speed.

Still at the fast speed, the median WiFi throughput is comparable to 3G for short drive. At slow speeds, it roughly doubles (for the long drive) or quadruples (for the short drive). WiFi is indeed impressive at slow speeds – providing over 2500 Kbps for over 60% or 35% of the times for short and long drives respectively. Thus, it is likely that WiFi would be much preferred in urban roadways in rush hour traffic, relative to high speed drives on rural highways. Incidentally, the former scenarios are likely to see metro-scale WiFi deployments.



(a) Short drive.

(b) Long drive.

Figure 6: CDF of Instantaneous TCP throughputs at different speeds for WiFi and 3G.

2.4.4 Correlation with Location

Here, we evaluate how throughput is correlated with location. Previous research has indicated that WiFi signal strength in the same location has a reasonable degree stability and this property has been utilized to develop location-based handoff techniques [31]. However, throughput stability has not been evaluated. In regards to 3G, throughput stability has been evaluated to conclude that the entropy of the throughput distribution reduces significantly when conditioned on location [121]. However, no direct comparison between WiFi and 3G exists in this aspect.

For our analysis here, we use the general approach reported in [121]. The idea is to use the notion of *information entropy* that measures the level of uncertainty associated with a random process. The information entropy of a discrete random variable X is defined as,

$$H(X) = \sum_{x \epsilon X} p(x) \log_2 p(x), \tag{1}$$

where $p(x)$ is the probability mass function, $0 \leq p(x) \leq 1$. The lower the entropy, the lower the uncertainty associated with the process. Hence, lower entropy indicates more predictability. Here, the random variable is the throughput. To address the influence of location on throughput we use the concept of *location entropy* as in [121]. Location entropy $H(X|l_i)$ is the entropy of throughput for a specific location l_i. This is defined as follows,

$$H(X|l_i) = \sum_{x \epsilon X} p(x|l_i) \log_2 p(x|l_i), \tag{2}$$

where $p(x|l_i)$ is the probability mass function of throughput at location l_i. We use a similar technique as in [121] to calculate the entropy of instantaneous throughput measurements from the data corresponding to the short drives. Both total entropy (as in Equation 1) and location entropy (as in Equation 2) are computed.

The entire geographic space containing the drives is checkered with grids of different sizes: 10m×10m, 20m×20m and 30m×30m, with each grid square being indicative of a location. The grids simply discretize the space. The three different sizes are chosen to estimate what level of discretization is appropriate.

	3G		WiFi	
Grid Size	Total H	Loc. H	Total H	Loc. H
10x10	1.56	1.19	2.26	1.44
20x20	1.52	1.16	2.25	1.56
30x30	1.51	1.12	2.23	1.71

Table 1: Comparison of total and location entropies for 3G and WiFi networks.

The average instantaneous throughput per drive per grid square is treated as the realization of single independent random observation. Hence, each drive can produce at the most one observation per grid square. The total entropy is computed over all instantaneous throughput measurements for 3G and WiFi separately. The location entropy is computed similarly for each grid square and then averaged over all squares. The results are presented in Table 1. We can see that the total entropy of 3G is much lower than the total entropy of WiFi showing better predictability of 3G throughputs. Note that the size of grid squares makes little difference. We also see that the location entropy of both 3G and WiFi is lower than their respective total entropies. This result suggests that when conditioned on location, instantaneous throughput becomes more predictable in both networks. But the degree of improvement is somewhat similar.

2.4.5 Temporal Correlation

We also evaluate the temporal correlation of instantaneous throughput values. Figure 7 shows the autocorrelation for the long drive. Note the excellent temporal stability of 3G relative to WiFi. This is somewhat in contrast with measurements in [121], where the authors have found that the autocorrelation in HSDPA throughputs was quite small. Also, interestingly for lower values of lag (k), the autocorrelation in WiFi throughputs is higher than that of 3G, though it decays promptly.

The knowledge of autocorrelation is useful is estimate how well the recent throughput experiences can predict the throughputs to be seen in near future.

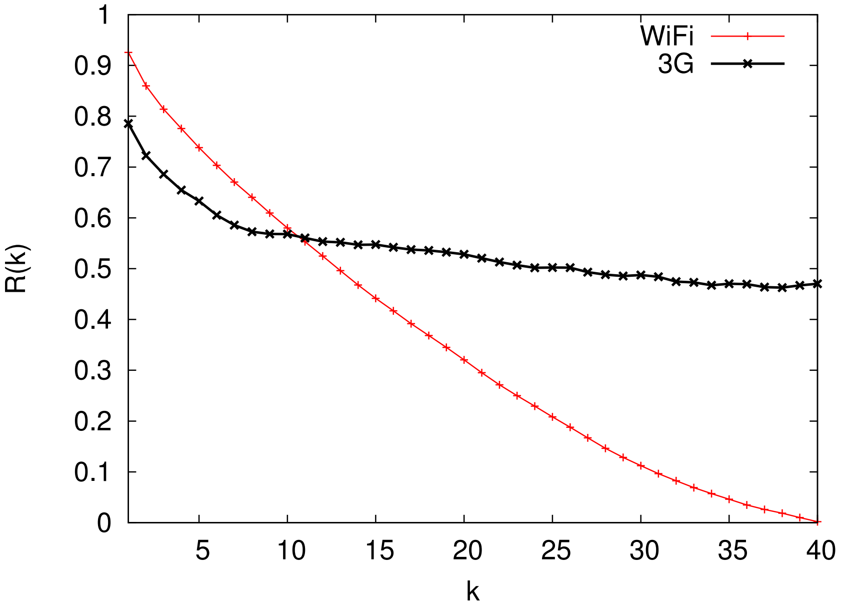


Figure 7: Autocorrelation $R(k)$ of the instantaneous throughputs measured in 1 sec intervals, where k denotes the time lag in seconds.

AR(k) Model	MSE(Y_{est})/VAR(Y)
AR(1)	0.894
AR(2)	0.802
AR(4)	0.824
AR(8)	0.853

Table 2: Prediction accuracy using AR(k) model.

This could be useful in packet scheduling in a hybrid access scheme.

The prediction model can use one of several standard techniques such as autoregressive (AR) model, moving average (MA) model, autoregressive moving average (ARMA) model, etc. In [121] the authors have found that even the simplest of the models (AR) produces a very good estimate of the current throughput based on throughput measurements in the recent past. Thus, we have kept our analysis limited to AR models only. We have evaluated the performance of the AR(k) model for several values of k to see the quality of estimation. Table 2 shows the mean-squared estimation error divided by the sample variance (MSE/s^2). This is based on a standard AR(k) fitting method used in the mathematical software Octave, when applied on the time series WiFi throughput data.[3]

2.5 Augmenting 3G with WiFi

The measurements indicate that compared to using 3G alone, a *hybrid access network* using both 3G and WiFi has several potentials. First, it can improve long term average throughput by at least a factor of 2 and likely more. Second, with a careful design, it can reduce the load on the 3G network by simply choosing to transmit over WiFi as much as possible. However, the 3G network should be used judiciously. The hybrid access should not use simple striping over multiple interfaces, as the cost models for these networks could be very different. On the other hand, avoiding 3G network to the extreme (e.g., using WiFi alone) is not appropriate either, as zero throughput regions on WiFi are not infrequent and cannot be ignored. This indicates that a hybrid design should use the interfaces intelligently – mostly using WiFi except augmenting it with 3G when appropriate and to the extent appropriate.

But what is really appropriate? The answer to this question entirely depends on the application. A download application (elastic traffic) may completely avoid the 3G link and can still provide acceptable user-perceived performance. On the other hand, media streaming with QoS requirements will freeze when WiFi is not available or provides very low bandwidth. This could happen even when playout buffers are used, though buffers certainly soften the impact. Also, the cost model for the links are important input to the design decision. When the per-bit cost of the 3G network is very low, striping across both networks may be appropriate for most applications. When it is very high, 3G must be avoided except when really necessary. In order to develop a

[3] If x_t is the average throughput at the t-th sec, the AR(k) model estimates x_t as $\hat{x_t} = c + \sum_{i=1}^{k} \phi_i x_{t-i} + \epsilon_t$, where c is a constant, ϕ_i's are the parameters of the model, and ϵ_t is white noise.

general design that is not tied to specific application or cost models, we take the help of network utility function.

2.5.1 Modeling User Benefit

Network utility function [100] is a widely accepted mechanism in networking to express user's utility of a network provided service. Utility of a flow $U(x)$ is expressed as a function of throughput x. $U(x)$ describes how much the user values throughput. For elastic flows, such as TCP downloads – one of the most prevalent form of traffic on the current generation Internet, the utility function has a diminishing marginal rate of increase with increasing throughput. This presents a strictly concave function. Modeling $U(x)$ as a logarithmic function has been common in networking literature. We will also take this approach here. However, our general approach does not depend on the exact nature of the utility function so long as it can be specified by the user for the specific service to be used.

Just like the utility function the network service using different types of link defines a cost function $C(x)$ that also depend on the throughput x on that link. The difference of $U(x)$ and $C(x)$ models the user's benefit that we want to maximize.

For our specific problem, assume that x_w and x_g are the throughputs on the WiFi and the cellular 3G link respectively. Assume that these throughputs are controllable by a throttle; however, certain maximum bounds exist, i.e., $x_w \leq X_w$ and $x_g \leq X_g$. Assume that the costs on the two networks are denoted by $C_w(x)$ and $C_g(x)$, respectively. Then, we want to maximize

$$H(x_w, x_g) = U(x_w + x_g) - C_w(x_w) - C_g(x_g).$$

See Figure 8. Assume $c_w(x)$ to be a constant, i.e.,

$$c_w(x) = K_1,$$

and $C_g(x)$ to be a constant plus a linear function of x, i.e.,

$$c_g(x) = K_2 + K_3 x.$$

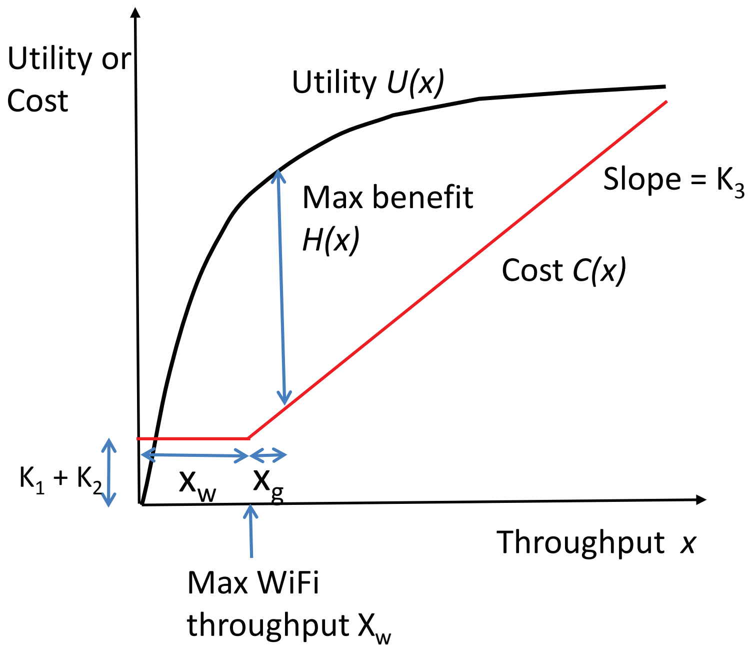


Figure 8: Example plot of utility, cost and benefit functions, demonstrating the benefit maximization approach.

Assume a logarithmic utility function suitable for elastic traffic:

$$U(x) = K_4 \log(x).$$

With a simple exercise of multivariate optimization it can be shown that $H(x_t)$ is maximized when

$$\begin{aligned} x_w &= X_w \\ x_g &= \begin{cases} \min(X_g, \frac{1}{K_3} - X_w), & \text{if } X_w < \frac{1}{K_3} \\ 0, & \text{otherwise} \end{cases} \end{aligned} \tag{3}$$

The above analysis indicates that if the cost and utility functions are as above, then the optimal strategy is as follows (see Figure 8).

Strategy 1

1. Use the maximum possible throughput on WiFi ($x_w = X_w$).

2. If the throughput on WiFi is not sufficient (i.e., $X_w < \frac{1}{K_3}$), use 3G only to fill in the slack (i.e., $x_g = \frac{1}{K_3} - X_w$), but no more. Thus, the 3G link is to be throttled unless the maximum throughput on 3G is already less than $\frac{1}{K_3} - X_w$.

The combined throughput target is related to the maximum possible WiFi throughput (X_w) and the slope of the 3G cost function (K_3). *Our goal is to develop a scheduler that implements the above strategy in a continuous basis.* However, in practice the scheduler takes scheduling decisions *once per scheduling interval.* The design of the scheduler will be discussed in the next section.

While we have used specific formulations, the above technique allows for modeling any type of utility and cost functions. Also, note that we are only concerned here about a single flow's utility from the user's point of view. The technique can be extended when multiple concurrent flows are present.

2.6 Hybrid Network Access

It is possible to implement the scheduling scheme as a part of the transport layer using protocols like SCTP [110]. However, instead of the complexity in dealing with a new transport protocol, we decided to implement the scheme in the application layer. The main components of the implementation are two proxies – one on the client side and the other on the server side. The client proxy resides in the vehicle and has a direct wired or wireless link to the client. (The client proxy could be a part of the client itself.) The server proxy resides anywhere on the Internet. It is possible to have many server proxies

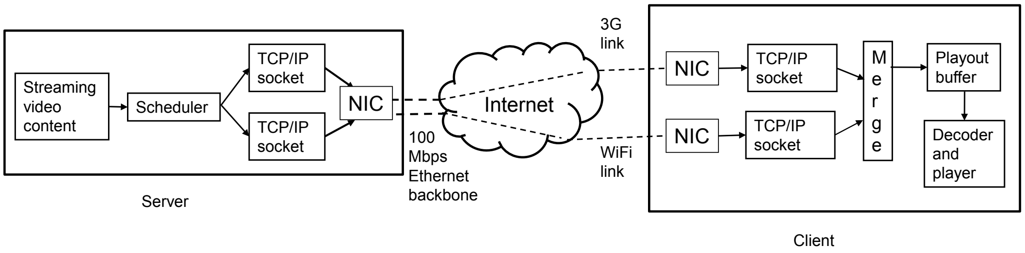


Figure 9: Components of Video Streaming Experimental setup. No separate client/server proxy is used. The server is a machine in the lab. The client laptop is carried in the car.

distributed over the Internet for optimizing routing paths. The proxy nearest to the server will be chosen. Techniques to choose the right proxy to service a download request are outside the scope of this chapter. One can imagine that this could work somewhat similar to a content distribution network (CDN). The reader will note later that in our experiments we have used co-located client and server proxies. Thus any proxy selection overhead or routing path expansion due to use of proxy do not show up in our evaluations.

The proxies maintain two TCP socket connections[4] between – one via the WiFi network and the other via the 3G network. Control messages are transmitted over the 3G link for better reliability. Scheduling decisions are taken once per scheduling interval. The interval is to be chosen appropriately such that it is not too small such that the overheads dominate, nor too large such that the throughput on the WiFi link in the current scheduling interval cannot be predicted well using throughputs of prior scheduling intervals. See Section 2.4.5, where 1 sec intervals produced very good predictability using just the simplest AR model (order 1). Thus, we stick to this interval and use the experience in the prior interval for prediction in our implementation and evaluation.

For each scheduling interval i, the TCP connection using the WiFi network is used to download for the entire duration of the interval. There is no

[4] We did a TCP-based design rather than UDP. This choice is tied to the final evaluation and will be discussed in Section 2.7.1.

throttling and thus the available bandwidth X_w^i can be measured. The TCP connection using the 3G network is used to download only for part of the interval. The 3G download time ($\tau \leq 1$ sec) is calculated as follows. τ is simply the time taken to download x_g^i bits, where x_g^i is determined using Equation 3 (or, Strategy 1). Note that in step 2, X_w^i must now be estimated as this value needs to be known at the start of the interval i. As discussed in Section 2.4.5, AR(1) and measured value of X_w^{i-1} in the prior interval is used to estimate X_w^i.

Since we are interested only in downloads in this work, the scheduling is implemented in the server-side proxy. Control messages are used via the 3G connection to communicate throughput related information (e.g., X_w^{i-1}) from the client to the server proxy.

The two TCP datastreams are then merged into one single stream using application layer information. To implement this the application layer already segments the byte stream into messages at the server side that are meaningful to the application. For example, in our experimental work with videos (next section) we have used video frames as individual messages. The scheduler at the server proxy simply decides which connection to send individual messages to. The application layer messages are treated as atomic in our implementation. They go through one connection or the other and not broken.

We have also implemented a simple error control scheme for the WiFi connection. This has initially been deemed necessary if the WiFi connection is broken for a long duration due to long lasting coverage holes, for example. In that case the TCP connection could break, and would need to be restarted. This might require retransmitting some of the messages sent on the previous connection. We have implemented such a scheme. Control messages to implement this scheme are transmitted over the 3G connection. In our initial experiments, such connection breaks indeed occurred occasionally. But with increasing robustness of our implementation and some careful choice of experimental driving routes that avoids really long coverage holes, such breaks were so rare that they do not make any impact in our evaluations. We believe that this will be a moot issue if we implement the protocol using SCTP, for example.

2.7 Evaluation

2.7.1 Methodology

For evaluation, we assume streaming video as the application. We also assume stored video and not live video, as this is most common form of video streaming on the Internet today. We assume TCP as the transport protocol rather than UDP. There are primarily two reasons for this.

- While both protocols are used for multimedia streaming (using RTP [97], for example), there seem to be more prevalence of TCP as increasingly HTTP/TCP is used for downloading multimedia. See, for example, progressive download, or Apple's new proposal on HTTP live download [92]. The industry's argument in favor of this approach, is that the HTTP-specific port is always open in firewalls thus eliminating protocol-specific configurations.

- Using TCP removes consideration of packet losses from the application side and allows us concentrate on only one aspect – delay arising out of inadequate bandwidth. If packet losses were involved, we needed to evaluate loss of video quality due to such losses. This means that we needed to use perceptual models for video quality assessment. This would strictly be tied to the actual video being played. It would also be unclear whether retransmitting some packets would be in fact beneficial for improving video quality. Overall, this approach would distract us from studying the networking side of the performance issues in isolation.

We assume that the user's media player maintains a playout buffer. See Figure 9. Incoming frames are stored here after re-sequencing. Re-sequencing is needed as there are two TCP links - 3G and WiFi. The media player simply picks the frames to display in sequence. The initial media playout is delayed for a short period until a certain *playback threshold* (in terms of a specific number of frames in buffer) is reached. This constitutes the *start-up delay.* During the media playout, the packets are consumed from the playout buffer by the media player. As long as the playout buffer is non-empty, the media playout is

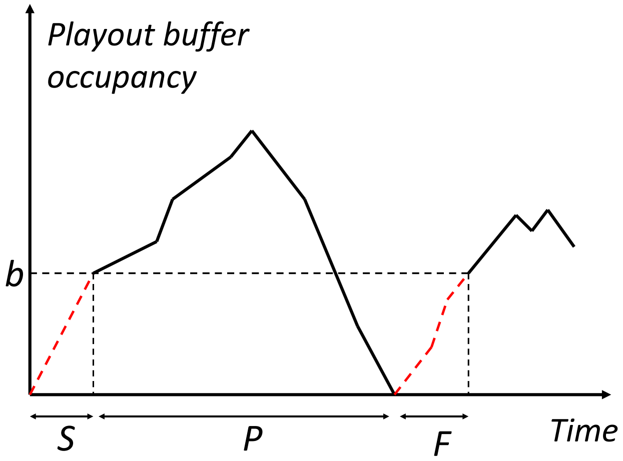
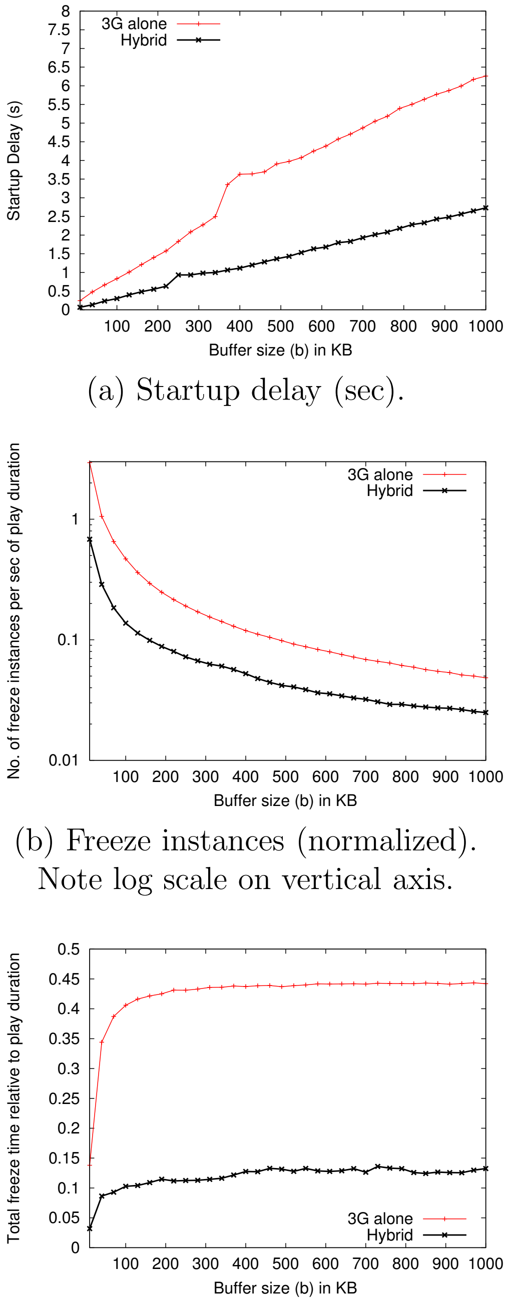


Figure 10: Evolution of the playout buffer during media playback (from [71]). b is the playout threshold, S is the startup delay, P is an actual playback period, F is a frozen period. The playback consists of sequences such as S,P,F,P,F,....

uninterrupted. However, when the buffer becomes empty, the playout is *frozen* until such time the playback threshold is again reached. Most media players follow this general approach. The playback threshold in effect guarantees the user with as 'smooth' video experience as possible. See Figure 10 for a time trace of the evolution of the playout buffer during media playout. We assume that the playback buffer size has no upper bound to prevent packet losses at the buffer. This is acceptable as TCP would be doing a flow control upstream of the buffer.

Clearly, the playback threshold controls the user's viewing experience. Larger threshold delays the startup more, but provides lesser frozen instances. Smaller threshold has the opposite effect. Multimedia research has indeed caught up to how to optimally set such thresholds [71]. However, we do not directly use such results. Instead, we evaluate three relevant performance metrics for a large range of playback thresholds. The metrics are: (i) Startup delay, (ii) Number of playback frozen instances during the playback, and (iii) Aggregate frozen time (startup delay not counted). These metrics provide a measure of the user's viewing experience. Our general goal is to evaluate these



(a) Startup delay (sec).

(b) Freeze instances (normalized).
Note log scale on vertical axis.

(c) Total freeze time (normalized).

Figure 11: User visible performance metrics for the video streaming experiment for the hybrid network vs. 3G alone.

metrics for a large range of playback thresholds and compare these plots for different network setups, i.e., a) '3G alone' as baseline, and b) '3G plus WiFi hybrid access' with a chosen set of utility/cost functions.

The experimental setup is the same as described in Section 2.3.2. The client and server platforms now implement the scheduling scheme described in the previous section except that no separate proxies are used. Individual video frames are treated as messages that are transmitted over the 3G or WiFi link. For evaluation, all received frames at the playout buffer are timestamped and logged at the client laptop. The actual viewing experience is evaluated offline using a time-driven simulator that plays these logged frames – one frame per inter-frame time. The simulator models the playout buffer and evaluated the three metrics. Note that using a simulator here does not introduce any modeling inaccuracy. A real media player would behave exactly as the simulator for the purpose of this evaluation.

For the utility model K_3 is chosen as 0.0005 per Kbps. The other constants are immaterial. This effectively makes an aggregate throughput target of 2000 Kbps. The video is selected from a pool of available video traces for networking research [98]. The specific chosen video (Tokyo Olympics) is MPEG-4 coded, has about 2000 Kbps mean bit rate, 18,000 Kbps peak bit rate and runs at 30 frames/sec. The mean frame size is 8.5 KB.

2.7.2 Performance Results

These experiments use the short drives repeated 3 times. The average of the three performance metrics as a function of the playout threshold b (in KB) is evaluated as detailed above and results are plotted in Figure 11. Note that the hybrid network performs significantly better than the 3G alone for all metrics for any playout threshold. This guarantees a significantly better user experience regardless of any specific parameter choice the media player makes.

Roughly, the startup delay is better in the hybrid network by a factor of 2–3. For the hybrid network the number of freeze instances and total freeze time are both much smaller. For the 3G network, they present a tradeoff. Smaller buffer means more freezes, but also smaller total freeze time. In general, looking at the numbers it is not a pleasing video experience on the 3G link for any buffer size. On the other hand, the hybrid network produces a much better video experience at least for larger buffer sizes. Comparing the total number of bits transmitted on the 3G link, we find that the hybrid network put in 77% less number of bits on 3G relative to 3G alone.

Note that these specific numbers are relative to the assumed cost and utility models and could be somewhat different when the models change. For example, (i) with a higher 3G cost, it is possible that the 3G network will rarely be used in the hybrid design, if at all; or (ii) with flat-rate pricing on both networks, both networks will be used to the maximum extent, providing an even better performance. Choice of a different video will also expose different degrees of performance differential. Nevertheless, the results above demonstrate clearly the potential of the hybrid design for wireless broadband access with vehicular mobility.

Aside from the user-level performance metrics we also evaluate the performance of scheduler itself. We find from the logs that 3G is used 41% of the times. This corresponds well with the previous measurement plots in Figure 4. The distributions of the aggregate throughput on both the links are plotted in Figure 12 for two cases – 3G fired and not fired, as well the overall – for all the scheduling intervals used. Note that for about 23% of the cases when 3G is not fired, the throughput is less than 2000 Kbps. Here the scheduler should have fired 3G, but it did not. Similarly, for about 35% of the cases, 3G is fired while the aggregate throughput is more than 2000 Kbps, indicating that there are more bits on the 3G network than needed. There may be a range of reasons why the performance of the scheduler is not perfect. Other than possible implementation inefficiency, it is likely that 1 sec scheduling interval is too short given that the scheduling is done in the application layer. Due to buffering in TCP, the application layer has little control when the the packets are actually transmitted. We expect that implementing the scheduler inside the transport layer in a protocol like SCTP may reduce this problem.

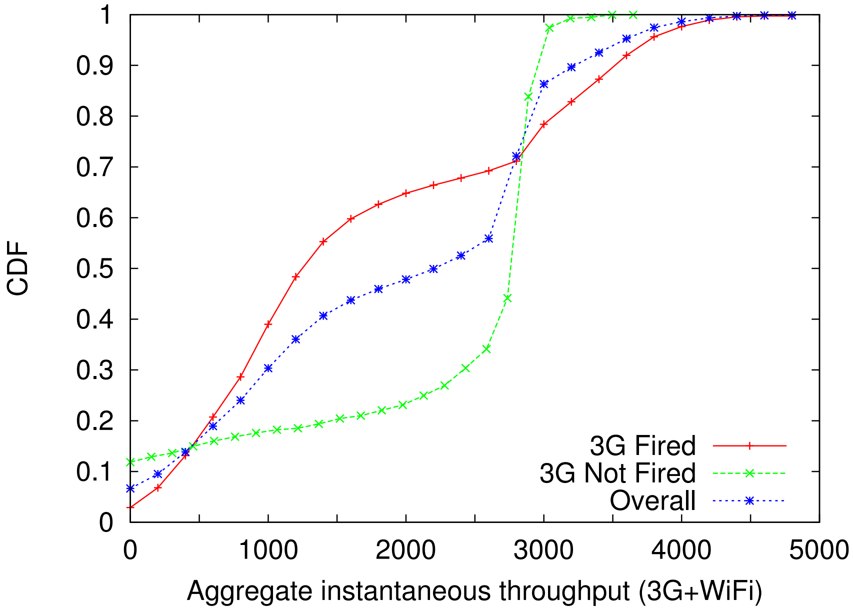


Figure 12: Distribution of the instantaneous throughputs in the hybrid network for two separate cases: (i) when 3G is not used (41% of the times), (ii) when 3G is used (59% of the times). Note that the aggregate throughput target is 2000 Kbps.

2.8 Conclusions

We have shown by experimental measurements that a metro-scale WiFi and 3G networks exhibit very different characteristics. WiFi has frequent disconnections even in a commercially operated, metro-scale deployment (roughly one-third of the times); but when connected indeed delivers high throughout even in a mobile scenario. The 3G network offers lower throughputs (half or lower relative to WiFi on the median), but provides excellent coverage and less throughput variability. The correlation of the throughputs of these two networks is very poor. These characteristics coupled of with the different cost model for network access indicates that a hybrid design that exploits the best properties of the two networks will be very successful.

We have developed a general purpose utility and cost function based formulation and a scheduling system that follows such models. The scheduling system is designed to delivers the optimal benefit to the user based on the chosen utility and cost models. The utility function may be application specific and can be chosen from a library of functions, by associating applications with utility functions. It is also possible to provide the user with an interface to dial up or down the utility dynamically.

Our experiments with stored video streaming show that the hybrid network is able to deliver much better video experience to the user relative to using 3G alone. At the same time, the hybrid system reduces the load on the 3G network by about three-fourth. While our experiments are indeed limited with single choices of video, utility and cost models, the general trend they demonstrate is extremely promising.

We note however that WiFi being on a license-free band can potentially become congested even when the operator has provisioned the network well. This will likely limit its throughput benefits. On the other hand, new technologies such as 802.11n and 802.11r can provide potential for higher throughputs and faster handoffs making WiFi use even more productive than reported here.

Chapter 3

Predictive Handoffs in Vehicular Environments

3.1 Introduction

WiFi access points have a range of a few hundred meters. This introduces frequent handoffs at the vehicular client. Absence of connectivity is a common phenomenon even in very dense WiFi deployments because of inefficient handoff policies employed in mobile clients. In this chapter we investigate the following two challenges and suggest predictive methods to improve vehicular WiFi connectivity.

3.1.1 Challenges

i. The *connection establishment latency* is ordinarily very high due to large probing delays for AP discovery (a typical client probes on all 11 channels before connecting), high loss rates (control packets during connection establishment may get lost and require retransmissions), and delay in acquiring IP address via DHCP. Several methods have been proposed to decrease this delay, such as using selective scanning, reducing timeout periods in case of losses, and using static IPs instead of DHCP [35, 24]. A recent work also proposes improving link layer performance using macrodiversity and opportunistic reception [17].

ii. A good *handoff strategy* is critical. Several APs may be visible at any location along the drive, and it is important to choose the best AP to connect to. Regular WiFi clients initiate handoff only when disconnected and choose the AP with the strongest signal strength. While this works well in home or office scenarios where clients are rarely mobile, this is not a good option for vehicular mobility. One can lose opportunity to connect to a strong AP (both in terms of signal strength and available backhaul bandwidth). A method of active scanning while connected has been proposed to address this issue [41].

3.1.2 Using Predictive Methods

We take a unique approach to address the above challenges by using a set of predictive methods that combines various forms of caching and estimation at link, network and application layers. Many studies have shown that people often drive on familiar routes, i.e., routes they have used before and the routes are highly predictable [64, 38]. In fact, our most frequent drives are only between a few locations (e.g., home, office, school, etc.). Further, it is expected that people have pretty consistent driving habits, e.g., use of lanes and speed. We show that these considerations can greatly improve the performance of vehicular WiFi access. The WiFi connectivity along a familiar route, and the mobility pattern of the vehicle can be predicted with a reasonably high accuracy using historical information. We develop techniques based on these predictions to readdress the challenges above. Our specific contributions are as follows.

Faster connection establishment Driving on familiar routes provides the opportunity to learn and cache the relevant information about APs along a route, which can aid in quick connection establishment. During periods of inactivity, the client listens for beacons and records the channel, network name (ESSID), and MAC address of the APs it can successfully associate with. This information is tagged with GPS locations. In addition, cooperation with the APs and clients and/or use of auto-configuration [47, 28] can eliminate the need for an IP address assignment via DHCP each time the client associates with an AP. All these speed up the connection establishment process significantly.

Scripted handoffs In addition to recording the connectivity parameters for an AP (as above), the client also builds a *radio frequency (RF) fingerprint* for the route when inactive, by recording the signal strength from beacons and tagging them with the GPS location of the car where the beacon is heard. This data, when collected over a period of time, provides a rich estimate of the RF level connectivity of various APs along the route. This connectivity estimate combined with an estimate of the vehicle's mobility are input to an algorithm which computes the locations where the client needs to handoff and to which AP. Thus, the handoffs are *scripted*. This computation is done offline and hence no bandwidth is wasted in scanning for better APs as in other online techniques. Also, the algorithm ensures that the client is always connected to the best estimated AP. This is unlike most stock implementations, where a connection is maintained until it breaks.

3.2 Related Work

3.2.1 Vehicular WiFi Access

Several experimental studies have explored the potential of using intermittently available WiFi connectivity from moving vehicles for data transfers [87, 88, 24]. In the Drive-thru Internet project [87, 88] controlled experiments are done with a single car driving past a single access point to measure range and connectivity in an intermittent network. The key contribution in this work is a session protocol that provides persistent end-to-end communication even in the presence of intermittent connectivity. More recently, the CarTel project [24] has focused on upload performance while using APs in the wild. This paper observes an upload TCP bandwidth of 240 Kbps and median transfer size per contact of 216KB. In the ViFi project [17] the link layer performance is improved by exploiting macrodiversity (using multiple APs simultaneously), and opportunistic receptions by nearby APs. In their opportunistic reception scheme when an AP overhears a packet but not its acknowledgement, the AP probabilistically relays the packet to the intended next hop in order to minimize the wasted transmissions. By the nature of their design, both CarTel and ViFi focus on upload. Downloads and intermittent connectivity have been studied in a vehicular environment in a fleet of taxis in the Cabernet project [35] with an improved handoff scheme and a new transport protocol providing several times better throughput than stock implementations. Our goal is to improve handoff performance even further by using prior RF fingerprinting and scripted handoffs.

3.2.2 Fast WiFi Handoffs

Several techniques have been proposed to improve the handoff mechanism of a mobile client when switching associations in a wireless network. One of the earlier work in this context was done by Shin *et al.* [101]. They proposed a technique of using neighbor graphs in a WLAN to reduce the number of channels to probe, thus reducing the probe latency during association. In [78], Mhatre *et al.* propose a scheme for improving handoff decisions based on long-term and short-term trends in signal strengths observed from beacons from nearby APs. This work however is not in a vehicular environment. More recently, in the Cabernet project [35] an optimized handoff technique is developed for vehicular WiFi access. Here, a probing sequence is determined among channels to reduce the link association delay. They also propose techniques for reducing the DHCP delay.

On the standards track, a recent development is the IEEE 802.11r [8]. Its goal is to permit continuous connectivity to wireless devices in motion, with fast and secure handoffs between APs. This amendment is specifically targeted for vehicular environments where the mobile device moves from one AP to the other in a matter of seconds. 802.11r minimizes the number of transition messages to 4 by piggybacking the security and QoS related messages with the 802.11 authentication and re-association messages, but does not target specifically to reduce scanning/probing delays. Our work does not incur any scanning/probing delay and handoff decisions are precomputed.

The above techniques reduce the handoff delay, but they do not directly address the problem of when to perform a handoff. Giannoulis *et al.* [41] proposed a solution for vehicular clients, where they keep scanning for better APs, even while they are associated. Ramani *et al.* [95] have developed a system called SyncScan which reduces the cost of active scanning with short periods of passive scanning. The short listening periods are synchronized with regular periodic transmissions from each access point. However, this system has not been tested in vehicular environments. In [21], two wireless cards are used so that a client can associate with more than one AP at the same time thus eliminating most of the handoff delay. Again vehicular environment has not been studied. In contrast to these works, our scheme uses historical signal strengths information to make handoff decisions.

Applications that require maintaining a session face problems in vehicular networks. Some papers address this issue by creating a transport layer protocol that maintains sessions transparently to changing IP addresses [35, 88]. The specific prefetching-based download application we develop does not need to maintain session at the transport layer. Sessions are maintained only at the application layer.

3.2.3 Mobility Prediction

Mobility prediction is well-studied in the domain of mobile phone networks. In [11, 12, 20, 69, 89, 106, 124], a central authority tracks the movement of devices to pre-provision network resources. In [63, 123], user mobility models are built from traces of several users, while Breadcrumbs [84], advocates that the mobile devices should retain their own mobility models for reasons of security and fine grained accuracy. In [107], authors compare different Markov based and compression based prediction models. Their study show that a second-order Markov model with fallback to a first-order model when the second-order model fails is the most accurate. Incidentally, Breadcrumbs [84] also implements a second-order Markov model for predicting mobility.

In addition to predicting mobility, it is also important for our system to be able to predict future network connections. War-driving databases could be used to predict networks to be encountered in the future. A system like Virgil [83], can be used to determine the quality of these connections in terms of available bandwidth, network connectivity parameters, availability of access to certain ports, etc. In [48] authors show that it is possible to predict wireless conditions of roadside APs and that this information can be used to improve vehicular network access.

3.3 Predicting Mobility and WiFi Connectivity

In our context, predicting mobility essentially means estimating, at time $t = T$, the location of the vehicle at a future time instant $t = T + \Delta T$. Predicting connectivity means predicting all physical, link and network layer information necessary or useful for establishing connectivity to each AP visible at every location. The connectivity information includes (i) physical layer information, such as the received signal to noise ratio (SNR) for each visible APs, (ii) link layer information, such as the MAC address, identifying name of the wireless network (ESSID), channel, and wireless security information (WEP, WPA, etc.) for each AP, and (iii) network layer information, such as a usable IP address, default gateway, and DNS server information. All the above information are tagged with GPS location. The GPS-tagged SNR data is also referred to as the *RF fingerprint*.

The inherent assumption that guides the mobility prediction is that people's driving habits are highly predictable. Similarly, the assumption that guides the connectivity prediction is that the set of necessary or useful physical, link and network layer information for the APs at a location are typically stable over time. These assumptions can give rise to simple-to-use predictive models based on historical data that can improve handoff performance and make prefetching possible and worthwhile.

3.3.1 Data Collection

To exploit predictability, we propose an architecture where each car maintains an estimate of its mobility and connectivity as defined above. This requires collecting related historical data using a methodology we will outline here. This data is later used in estimations.

The data collection is done when the mobile client is 'idle', i.e., not communicating. The idle periods are chosen simply to avoid impacting actual data communication performance, as we use a single processor and single radio interface to manage all aspects of the system. Use of additional hardware can allow for concurrency, and then data collection can be done at all times, whenever the car is switched on.

The data related to mobility is simply a sequence of timestamped GPS locations collected periodically (every second). The data related to connectivity is collected using a 'sniffing' mechanism to log all beacons heard from any AP. The 802.11 beacon header contains the information about MAC address, ESSID, channel, security info, etc., while the Prism Monitoring Header[1] contains the SNR of the received packet. This information is tagged with the GPS location and timestamp where the beacon is heard. Note this data subsumes the mobility data wherever beacons are heard (default beacon frequency in most 802.11 APs is about 100ms). A careful reader will note that this data collection is somewhat similar to traditional *war-driving* [115]. GPS-coded SNR data helps build the RF fingerprint mentioned before. The client also records whether the client has appropriative credentials to connect to the AP and whether the AP provides backhaul connectivity. Depending on the usage scenario, this may involve a table lookup with a pre-existing table provided by a provider, or an actual association attempt with the AP and ensuring that the AP indeed provides backhaul connectivity. The client IP addresses are to be randomly generated following the guidelines in the IETF Zeroconf working group [47] and RFC 3927 [28]. More is detailed in Section 3.7.

The key aspect of our work is to develop protocols that exploit such historical data about mobility and connectivity to design predictive schemes

[1] Prism Monitoring Header is added by certain 802.11 card drivers and contains information such as rate, SNR, noise, etc. for all received frames.

to improve download performance. In the following subsections, we analyze the stability aspects of such collected data. The experimental data for this analysis was collected using two of our colleagues' cars. There are two data sets from the two cars – data set 1 is for a period of about 6 months and data set 2 is for a period of about 3 months. During this period, the cars were driven normally by their owners, carrying out their normal daily routines. Both cars carried the experimental platform described below.

3.4 Experimental Platform

Our experimental platform is essentially a small form-factor embedded single-board computer (SBC) with a GPS unit and an 802.11 interface. The same platform is used in the protocol evaluations in the following section. Specifically, we use a Soekris 4801 [105] for the embedded SBC, with an 18dBm 802.11b miniPCI card with Atheros chipset with an external 5 dBi rubber duck antenna, and a Garmin USB GPS receiver. A 4GB flash drive is used to collect the logged data. The data is periodically manually uploaded on a server for sanity check and later analysis. The computer runs Linux kernel 2.6.19 and the latest `madwifi` driver [3] for 802.11. The computer draws power from the car battery from the regular 12V automobile socket and remains switched on whenever the car is running.

On startup, the 802.11 card is initialized in monitor mode, the computer system clock is synchronized with the GPS receiver and the computer starts running `tcpdump` to log all the packets received by the 802.11 card. The first 200 bytes of every packet are logged which contain the Prism Monitoring header and the 802.11 header. Beacons are usually small packets, and are captured in their entirety. The computer time at which the packet was received is also recorded for every packet. The card switches between the non-overlapping channels 1,6, and 11 at intervals of 0.5 seconds each. Initial tests showed that a very large fraction of APs are on these three channels [10]. Moreover, scanning just these three channels is sufficient to get a large number of packets in other overlapping channels too. Also on startup, the GPS receiver is initialized and

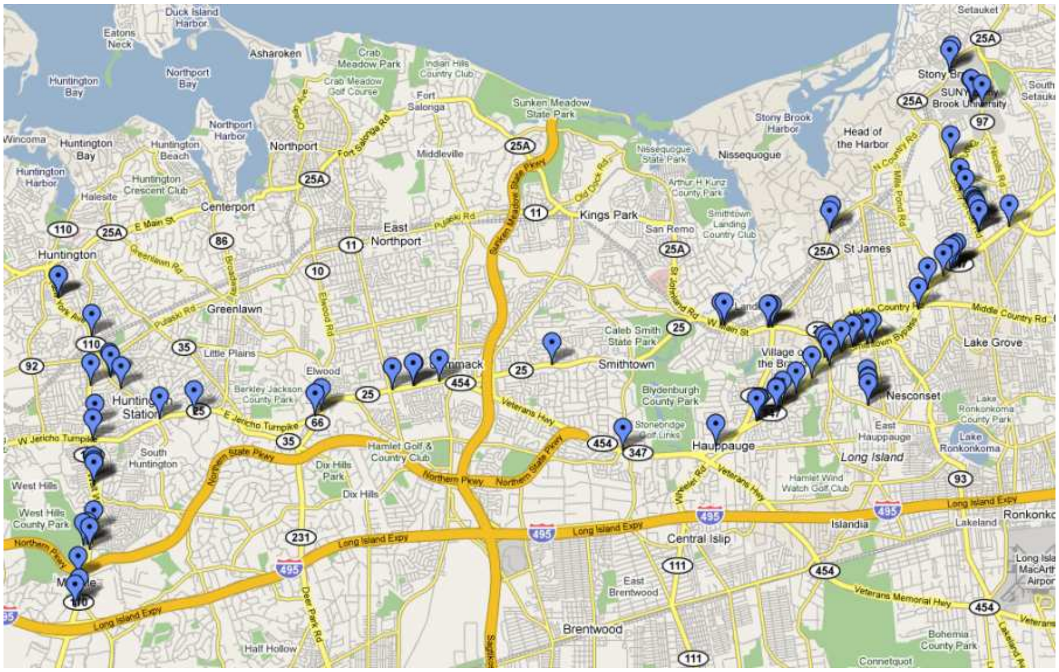


Figure 13: Map of the area driven showing the location of popular APs.

the GPS location and timestamps are logged every second. The packet timestamps in the `tcpdump` trace are later correlated with the GPS timestamps to determine the location at which each packet was received.

3.5 Analysis

As mentioned before, data set 1 has been collected over a period of 6 months while data set 2 has been collected over a period of 3 months. A significant portion of these drives is between home and school. In data set 1, this distance is about 25 miles and consists of a combination of freeways and local roads. In data set 2, on the other hand, it is about 5 miles. It consists entirely of local roads. A few other drives do exist in the data set that are repeated several times. They are usually short and use local roads predominantly. The dataset has 3618 distinct APs. See Figure 13 for a map of locations of APs which are seen with a high frequency.

Figure 14 shows the quality of mobility prediction. We determine the distances traveled from a given start point for different time intervals ΔT. The average of all these distances across all drives on the same route available in the log is used as the predictor for mobility. The quality of the estimation is modeled by the width of the 90% confidence interval. We predict mobility on three different kind of roads found in our data set – (i) an expressway, where the car typically travels at a sustained speed of about 100 km/h, (ii) a signaled highway, where there are large variations in speed due to several signals and also due to traffic variations, the maximum speed being about 90 km/h, and (iii) a village road, where the car travels at a fairly constant speed of about 50 km/h. As shown in Figure 14, the accuracy of the mobility estimates is very high for small values of ΔT, while expectedly, the accuracy decreases for large values. It is interesting to see that the accuracy is low initially for the expressway than for other types of roads. This is due to the high speed on expressways, and any variation is magnified in terms of distance traveled. Also, as expected the mobility estimates become less accurate with time on a signaled highway, but remain excellent for the expressway and the village road.

From our experience, (also in the CarTel data set [24]), the median connectivity duration with an AP is about 13 seconds at regular driving speeds in an urban area. Taking this number as a guide, observe that the uncertainty in distance is very small for this period – about 20 m, much less than a typical range of an AP (about 100 m). Also, to accumulate an uncertainty roughly equivalent to the typical range of an AP, ΔT needs to be very large – about a couple of minutes – almost an order of magnitude more than the typical visibility period of an AP. Thus, one can expect decent prefetch performance with a *prefetch lookahead* of a few APs. By 'prefetch lookahead' we mean the number of APs ahead of the current AP that are asked to prefetch data by the current AP. More will be discussed about this concept in Section 4.3.2.

We now evaluate the accuracy in estimating connectivity by showing that RF fingerprints remain moderately stable over time. For a particular AP, and a particular GPS location, the average SNR of beacons received over several drives from that AP in that location is used as a predictor. 90% confidence

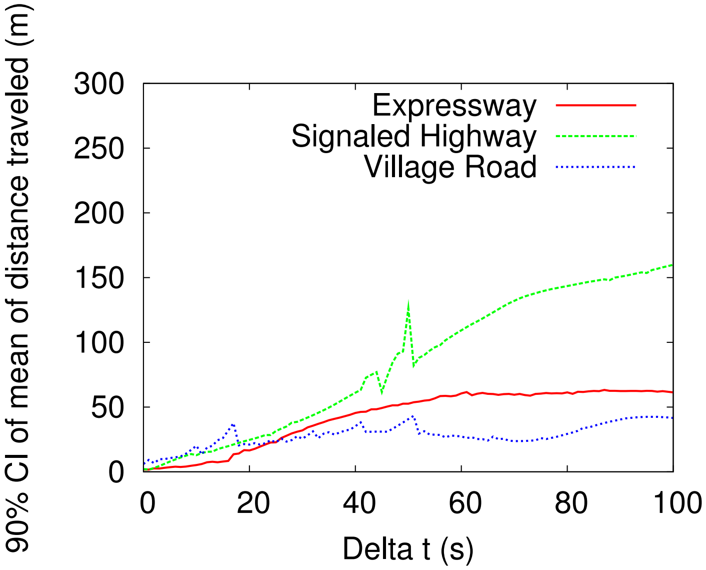
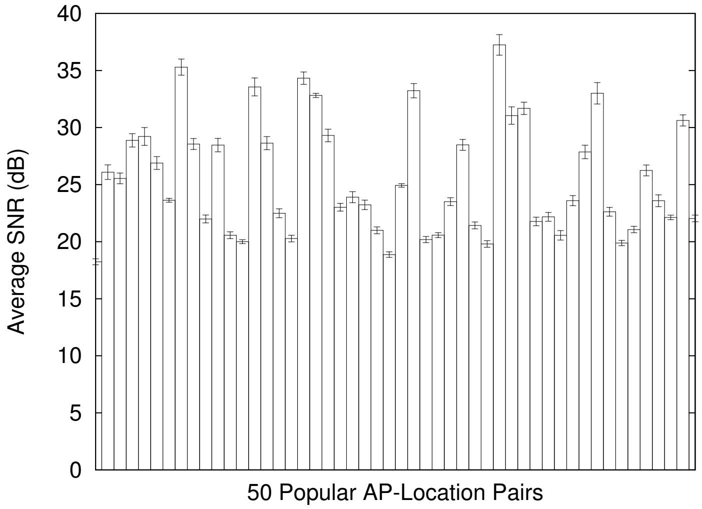
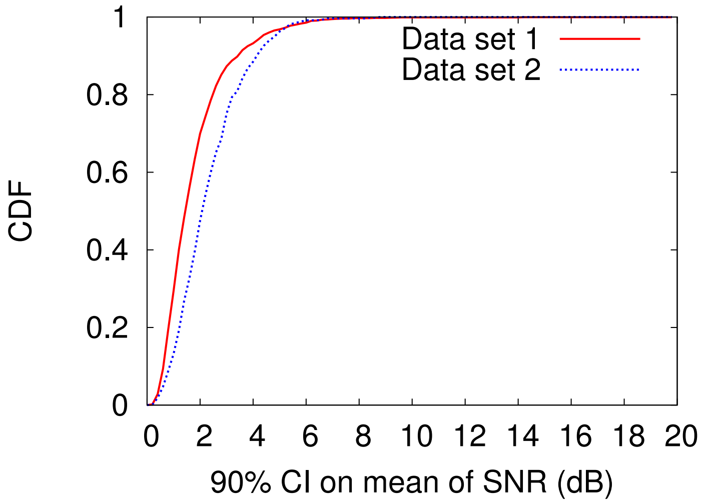


Figure 14: Mobility estimator for the data set.

interval is again used to model the quality of prediction.



(a) Average SNR for the 50 most popular AP-location pairs over the two data sets showing 90% confidence intervals.



(b) CDF on the width the 90% confidence intervals for the SNR of all APs for all locations.

Figure 15: Statistics showing the stability of RF fingerprint data.

We must first discuss the practicality aspects of this evaluation before presenting results. This discussion also lays out how RF fingerprints are used in practice for use in the protocols discussed later. It may not be possible to obtain RF samples at exactly the same locations from different drives. The reason for this is that (i) the car positions could be slightly different on different drives even when driving on the same lane; (ii) the RF samples could be obtained at slightly different points; (iii) GPS is not perfect – typical GPS error is a few meters, as prior measurements with the same GPS unit have shown [111]. To address these issues, we discretize the location data in the RF fingerprint. To do this, we overlay a 10m×10m grid[2] on the geographic area spanned by the data set. All RF samples corresponding to a grid square are mapped to the center of that square. Since we are interested in variations across drives, we assume that for each drive, there is only one RF sample per AP per location (center of a grid square). If there are more than one such sample on the same drive, the SNR values are averaged to produce just one sample. Now, for each location that actually has samples from sufficient number of drives (more than 5), the *average SNR over these multiple drives* is computed.We also compute the 90% confidence interval for this statistic.

The statistics (average and the confidence intervals) of the 50 most popular AP-location pairs and CDF of confidence interval widths for all APs for all locations are shown in Figure 15. Note excellent stability of the SNR statistic in Figure 15(a). Also, note that the median width of the confidence interval is around 2 dB (in Figure 15(b)). This is small when compared with the median overall SNR values logged, which is about 22 dB. In prior work for vehicular WiFi access, 2 dB has been considered as a margin of error in estimating the mean signal strength [41]. In that work, a client scans for better quality APs even when connected, and a handoff is triggered, if an AP is discovered with an average signal strength 2 dB greater than the current AP.

It is indeed true that about 10-15% of the data shows 4 dB or larger confidence intervals that may not be acceptable to make good handoff decisions. However, data stability aspects can be taken into account in making handoff decisions. Thus, it is not a critical issue.

The above analysis uses the complete data set as actually collected. While this presents the summary statistics, it does not directly analyze concerns

[2] We also tried 20m×20m grid with no significant variation of results.

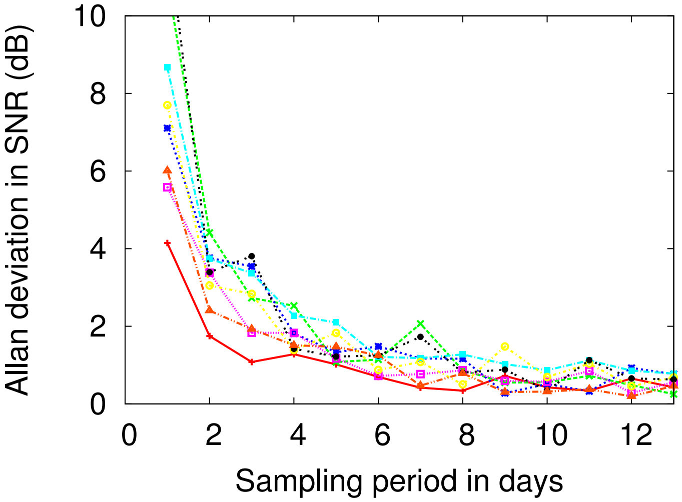


Figure 16: Allan deviation in different time scales for 8 selected AP-location pairs for which we have the most number of samples.

about temporal nature of the data set, for example, whether the SNR is relatively independent in the sequence of drives, or whether there are short term temporal dependencies. In other words, is it possible that it is best to estimate SNR using relatively fewer, but more recent samples, as opposed to long term measurements using the entire available data set? To study this, we use Allan deviation [13] as a metric. It is similar in spirit as standard deviation, but uses difference between subsequent samples rather than differences from the mean. Allan deviation is typically used to quantify the burstiness in the variation of any quantity and has been previously used to study temporal variations in link quality in 802.11 networks [9]. Allan deviation is computed at various time scales from 1 to 12 days for all APs for all locations as above. In Figure 16 we plot the Allan deviation in SNR for 8 ⟨AP, location⟩ combinations for which we had the most number of samples. Note that while there are fluctuations at different time scales, there is indeed a downward trend signifying lack of strong temporal correlation. Summary statistics are plotted in Figure 17, where we plot median, top and bottom quartiles for the Allan deviation across the data sets for different time scales. Again note the strong downward trend

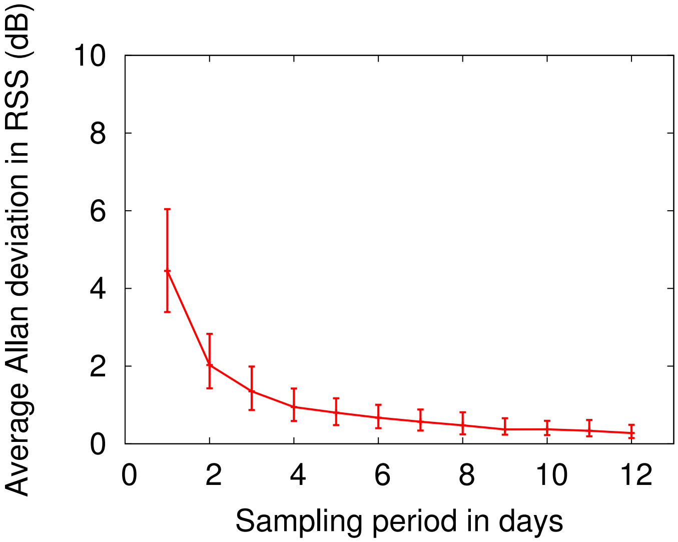


Figure 17: Summary statistics of Allan deviation on different time scales (median, top and bottom quartiles).

and stabilization beyond just 1 week.

The above analysis confirms that average SNR over multiple drives is a good measure for estimating SNR as it shows little variation when averaged over for a week or more. Another concern could be about the staleness of the historical information collected. For example, the average of SNR values for an ⟨AP, location⟩ combination computed over a week may change when measured again after a few weeks. We evaluate the extent of staleness by comparing the difference between weekly SNR averages of various ⟨AP, location⟩ combinations, as the distance between the weeks considered increases. The CDF of various such samples is shown in Figure 18. The graph shows that the median difference between weekly SNR averages remains within 1 dB even if we consider weeks as far as 5 weeks apart. This demonstrates that the SNR data remains stable over several weeks, and data collected even 5 weeks ago should be usable for prediction.

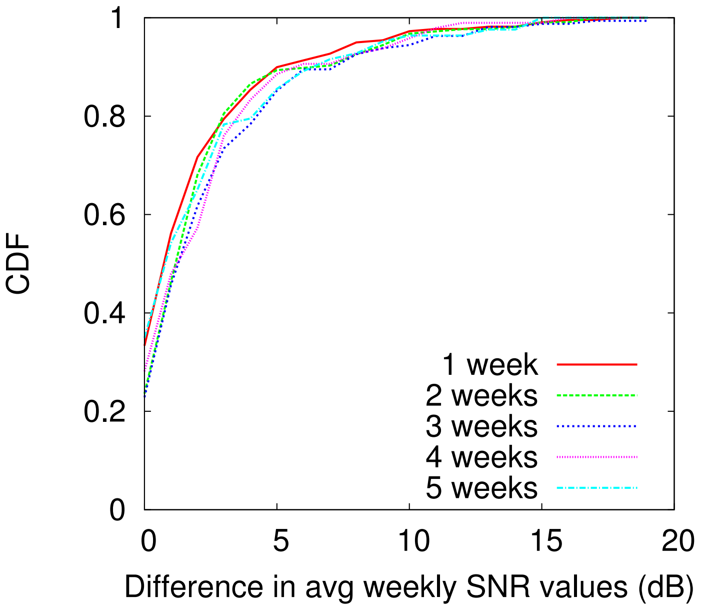


Figure 18: CDF of difference in average weekly SNR values for ⟨AP,location⟩ pairs as the separation between the weeks increases.

3.6 Controlling Handoffs

As mentioned before, stock 802.11 clients typically maintain association with the same AP until the connection breaks, which is determined by the absence of beacons for a certain fixed period of time. Then it searches for another AP to associate with via either active or passive scanning techniques. In active scanning, the client switches between all channels, and for each channel, it broadcasts a probe request message and waits for a finite and configurable timeout period for APs on that channel to reply with a probe response message. This takes about 250ms for 802.11b. In passive scanning, the client just switches between channels, and passively listens for the periodic beacons that are broadcast by APs in each channel.

In our knowledge, the state-of-the-art in determining when to handoff in a mobile scenario is the mechanism proposed in [41]. There the authors propose a method of active scanning even while connected, to maintain a quality score for each AP, and switch association as soon as a better AP is discovered. To counter short-term fluctuations in signal strengths, the authors propose using an exponential moving average of the signal strength. They also propose using hysteresis to prevent handoffs from occurring very frequently. This method is similar to our strategy to connect to the strongest AP always. However, it suffers from the problem that the process of active scanning wastes significant bandwidth for the client when it is connected. This is certainly detrimental to performance when the client is actively communicating. Also, scanning needs to be done very frequently to make an optimum choice in a mobile scenario.

In the technique we propose, historical RF fingerprint data is used to estimate how signal strength is expected to vary along the drive.[3] We already established in the previous section that the data is stable enough that it is feasible to do so. See Figure 19 for an example, using our own experimental data set from the evaluations that we will describe momentarily. The road segments are simply the intersections of the driving path with the grid squares discussed in the previous section. Note that the rises and falls are expected as the car comes near to the AP and then goes away. The brief dips are often due to turns or due to structures causing radio shadows.

A straightforward technique could be as follows. When disconnected, the client establishes connection with the first AP that becomes visible. However, when the client is already connected, the handoff is performed at the intersections of the 'smoothed' SNR vs. distance curves to that AP that will provide the strongest signal for the next segment. See Figure 19. The locations at which each handoff is triggered is computed offline. This could be done sufficiently in advance to the actual handoff. This is the reason behind the name – *scripted handoff*. We present experimental results for this technique in the next section.

Depending on the handoff latency in both the link and network layers, the above simple technique may lead to frequent handoffs that could impact throughput adversely. Stock implementation of connection establishment process between a client and an AP requires several message exchanges for the

[3] The route can be easily estimated using historical data again (see, e.g., [38]) is not discussed here.

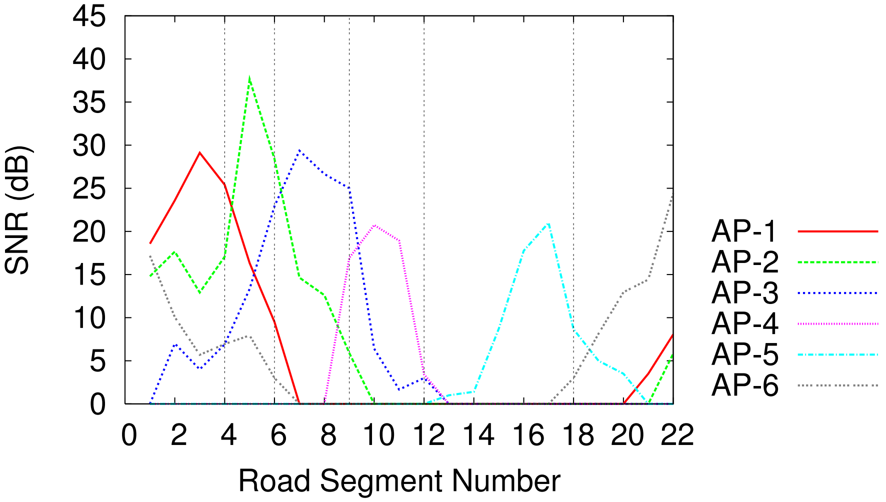


Figure 19: Expected variation of SNR (smoothed) along the drive based on historical RF fingerprint data. Dotted lines show the precomputed locations of handoffs.

link layer association followed by IP address allocation via DHCP. The vehicular environment is often characterized by high loss rates as shown in [35], which leads to significantly high connection establishment time. The mean connection establishment time has been reported as 12.9 seconds in [24].

For large handoff latencies, an optimal handoff strategy thus needs to factor in the handoff latency. This can be done via *scripted handoff* as well. Here, the predicted route and speed can be used to estimate the visibility periods of individual APs along with their signal strengths. This information then can be used to drive an algorithm that pre-computes the optimal handoff points and APs (i.e., <AP, location> tuples) to maximize throughputs by factoring in the handoff latency. The mobile client hands off to the pre-computed AP when the car reaches the specific location.

The scripted handoff is reminiscent of location-based beam steering and AP selection method in our prior work in MobiSteer [33]. However, the focus in MobiSteer was primarily on beam steering and the handoff performance vis-a-vis other fast handoff techniques was not evaluated. In the current work the focus is primarily on AP selection (handoff). In the next section we will

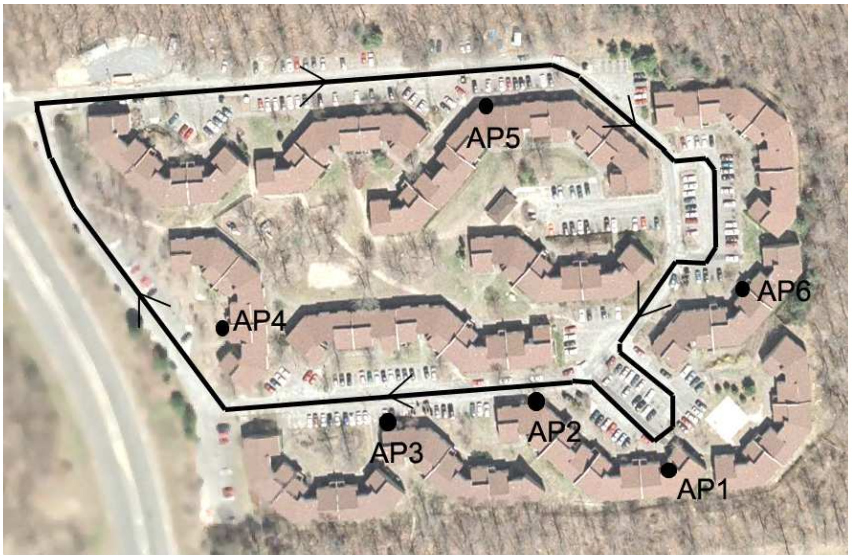


Figure 20: Map of the drive. The AP locations are noted.

evaluate the handoff performance with respect a state-of-the-art method.

3.7 Evaluating Scripted Handoff

We evaluate the simple scripted handoff technique (called SH for easy reference) by comparing them with two existing methods: (i) the naive method used by stock WiFi clients (maintain until broken followed by active scanning or MUB for easy reference), and (ii) the state-of-the-art method (active scanning while connected or ASC for easy reference).

While the scripted handoff (SH) technique relies on historical information, and is thus an offline technique, the other methods, MUB and ASC, are online methods. As an offline method, the SH technique has several advantages over the other online methods. The need for scanning is eliminated as the client already contains historical information about the AP's MAC address and channel, and can thus send an association request message even without scanning.

The need for having to do DHCP for IP address assignments can be eliminated by using a randomly generated link-local IP address in the client per RFC 3927 [28] and then using a NAT at the AP. This approach has been promoted by the IETF Zeroconf working group [47]. The idea is to generate a random IP address from a pre-specified address pool (with 169.254/16 prefix per the RFC [28]). The RFC proposes to detect conflicts by a broadcast ARP probe sent prior to real communication. In our application, the AP can help detect and resolve conflict via a control message and the ARP probe is not needed. Given the large pool of possible addresses, we anticipate that conflicts will be rare. If IPv6 is used, such conflicts can be eliminated as the MAC address can be used to form the host part of the IP address. In our evaluations, however, we have used static IP addresses by means of sufficiently long DHCP leases obtained prior to measurements and then recording the IP addresses as part of the RF fingerprint.

Metric	MUB	ASC	SH
Average throughput (Kbps)	101.6	113.52	232.8
Avg. duration of connectivity per AP (s)	11.04 (2.5)	9.28 (3.1)	12.89 (1.4)
Avg. outage period per AP (s)	36.82 (9.1)	31.2 (9.8)	13.45 (3.9)
Avg. # successful connection establishments per drive	2.5	2.7	4.5
Avg. # attempted connection establishments per drive	10.3	4.9	5
Percentage of successful connection establishments	24 %	53 %	90 %
Avg. scanning latency (s)	0.42 (0.0)	0.32 (0.0)	0 (0.0)
Avg. DHCP latency (s)	3.76 (0.7)	3.88 (0.6)	0 (0)

Table 3: Summary of results for handoff control experiments.

As we want to evaluate the performance of the handoff strategy, it would be unfair to compare the online techniques with their stock implementations, as they would incur a significant handoff latency impacting their performance. We thus optimize the connection establishment process for online techniques as well. The scanning is performed only in three channels – 1, 6 and 11, as all our APs are in one of these three channels. We also optimize the DHCP client as in [35] by reducing the timeout periods for the DHCP DISCOVER and DHCP REQUEST messages. We further describe some implementation details below. We use the same hardware setup as described earlier in Section 3.3.

Maintain until broken (MUB) – This is the stock implementation used by most clients. Once connected, the client maintains the connection until it stops receiving beacons for 5 seconds, after which it disconnects. When disconnected, the client starts the connection establishment phase again.

Active scanning while connected (ASC) – This is based on the method described in [41]. We have described this method briefly in Section 3.6. We use the best-performing parameters for the scan period, hysteresis threshold, and the weight for exponential averaging from the paper. The scan, performed once every second, is done on the three channels – 1, 6 or 11 by sending probe messages and waiting for replies. The signal strength of received probe responses is recorded. For each AP, the value of its signal strength is maintained by taking a weighted average of the current SNR (with a weight of 0.35), and the previous estimate of SNR (weight of 0.65). Whenever an AP is found with the estimate of signal strength greater than a certain threshold (2dB in our case), a handoff is triggered.

3.7.1 Experimental Setup

Scripted handoff is most useful when there is a dense AP deployment with overlapping coverage and no coverage holes in the driving path. We were unable to find such a region with open access APs in our neighborhoods. This is because a significant fraction of APs that we analyzed in Section 3 use some form of security. We perform the experiments in a residential area with several APs that we deployed ourselves inside homes to carry out the experiments. See Figure 20. The APs are all connected to a backhaul providing Internet connectivity. We choose a circular drive of about 650 meters. The driving speed is slow – between 16 Km/h and 24 Km/h. Thus, each round of the 650 meter drive is completed in roughly 2 minutes. At least one AP can be heard from almost all locations along the drive. 10 of these rounds are used to create the RF fingerprint of the area. The RF fingerprint representation according to drive segments is shown in Figure 19. The drive is divided into 22 segments of approximately 30 meters each, and the best AP to connect to in each segment is computed in advance. The five handoffs that need to be performed during this drive are also shown in the figure.

We perform 10 rounds of drive each for the 4 handoff strategies. In each drive, we connect to the APs using the specified handoff strategy. Whenever connected, we send a stream of CBR traffic at 560 Kbps consisting of 1400 byte UDP packets (thus, 50 packets per second) to a server on the Internet, which records the timestamp at which each packet is received. The MAC address of the associated AP is included the packet; so the server can detect change in association. The server calculates the "throughput" in Kbps from the mobile client to the server. The server also computes the "duration of connectivity per AP" by subtracting the timestamps of the last and the first packet received via an AP, and "outage period", by subtracting the timestamp of the first packet received via a new AP and the timestamp of the last packet received via a previous AP. The mobile client records the latencies for the three handoff steps – "scanning latency", "link association latency", and "DHCP latency" for each successful connection establishment. It also keeps a record of the "number of attempted connection establishments", and "number of successful connection establishments."

3.7.2 Results

The results for the three mechanisms are summarized in Table 3. The numbers in the brackets in the table are 90% confidence intervals of the mean value presented. The results clearly show that our handoff strategy based on the offline, scripted handoff method outperforms the other "online" methods by a factor of more than 2. The SH protocol provides an average throughput of 232.8 Kbps which is more than twice the average throughput of other protocols. The average outage is less than half of the other methods. The average duration of connectivity per AP is about 20% higher for SH than other methods as the connection establishment latency is very low. The method of active scanning (AS) performs only slightly better than the maintain-until-broken method (MUB). This is because significant bandwidth is wasted in scanning once every second. SH has no scanning or DHCP latencies, that present high overheads for the other two methods.[4]

[4] We did not separately quantify how much improvements are obtained individually due to the absence of scanning or DHCP latencies. However, lab experiments show saturated UDP throughput drops by 37% when scanning every 2 seconds using the ASC implementation.

We do not show the link association latency because it is very low, and similar for all the three techniques. Due to the high connection establishment latencies for MUB and AS techniques, they have a poor percentage of successful connection establishment.

3.8 Conclusions

WiFi was not developed for outdoor use, let alone using it from moving cars. Every aspect of WiFi communication, from the physical layer performance (due to the change in the fading environment), to handoff performance (due to use of slow and suboptimal handoff strategies) suffers when used from a moving car accessing APs in the wild. While the basic physical layer performance cannot be helped without a change in the standard (these issues are already in the works in the 802.11p working group [54]), our work demonstrates that handoffs can be implemented very efficiently by using predictive methods at the link layer and up. While these techniques depend on use of historical information, such data collection is straightforward and can be done directly on-board the client device. Use of scripted handoff based on advance RF fingerprinting reduces failed handoffs, improves outages and improves overall throughput by a factor of 2, as our experiments demonstrate.

Chapter 4

Predictive Prefetching in Vehicular Environments

4.1 Introduction

In this chapter, we are specifically interested in studying a content delivery network for moving vehicles in urban areas [35, 88]. Such a network can be used for downloading (e.g., songs, movies, podcasts, maps, traffic data, etc.) as well as for uploading (e.g., sensor data, etc.). The APs in such a system may consist of subscription based APs installed by a service provider, municipal WiFi APs, or APs with an architecture like FON [2] or WiFi.com [7] built on the concept of sharing WiFi connectivity. The FON or WiFi.com models are particularly relevant, as they rely on a community model where AP owners share their Internet connection with others in return of a similar service. These models can potentially enable deployment of custom protocols on the APs where APs can 'co-operate' among themselves and with the mobile client for better performance.

For such a content delivery network, a suitable *data transfer strategy* is required, as vehicular WiFi access is characterized by frequent disconnections and lossy links. All these are known to degrade TCP performance, and impact applications requiring session maintenance (e.g., FTP, HTTP, etc.). UDP-like connectionless transport protocols have been proposed to alleviate this issue

[35, 88].

Many studies have shown that people often drive on familiar routes, i.e., routes they have used before and the routes are highly predictable [64, 38]. In fact, our most frequent drives are only between a few locations (e.g., home, office, school, etc.). Further, it is expected that people have pretty consistent driving habits, e.g., use of lanes and speed. In the predictive prefetching scheme addressed in this chapter, mobility estimates are used to predict the periods of connectivity to various APs in future. This can provide further performance gains in download applications by having such AP *prefetch* part of the content to be downloaded. Essentially, the APs now collectively form a distributed cache for the mobile client, and in cooperation with the mobile client prefetch pre-determined portions of the content. This helps mask the large Internet delay on the WAN side of the AP. Inaccurate estimation of mobility causes ‘cache misses,’ hurting download performance. On the other hand, duplicate prefetches (i.e., more than one AP prefetching the same bytes of the content) increases load on the WAN and the content server. These must be optimized carefully.

4.2 Related Work

Prefetching has been used in widely different contexts in computer systems to speed-up downloads by hiding latency. As expected it also has been used heavily in mobile environments to prefetch data, but often in the context of prefetching related objects or content that are expected to be used in near future. See, for example, [29, 55, 56, 84]. In the context of vehicular networks as in our work, [122] has considered a combination of info-stations (with high bandwidth links) and base-stations (with low bandwidth links) and developed prefetching ideas based on the mobile client’s location, direction and speed. In [18, 15] aggressive prefetching is used to make the results from web search queries available to mobile clients in buses using a combination of ad hoc and infrastructure networks. In [52] a technique has been described to use the cellular network to send data prefetch requests and the data is prefetched at the prefetch agents located at hot-spotted networks. To the best of our knowledge no work prior has considered prefetching parts of the same large object at different locations to improve download performance in a highly mobile environment.

4.3 Prefetching

So far, we have used predictions to improve handoff (fast handoff and handoff to an 'appropriate' AP). The techniques have been thus limited to the link and network layers. Now we turn our attention to the application layer and show how prefetching at the APs can improve download performance. Since a vehicle's mobility and connectivity can be predicted quite well on familiar routes (Section 3.3), it is possible to create a protocol for prefetching portions of a large object to be downloaded on the APs on the route of a vehicle. When the vehicle approaches and connects with an AP, the prefetched data can be directly downloaded from the AP instead of connecting to a server on the Internet.

Such prefetching has several advantages. First, the multi-hop Internet path is replaced by a one hop path when a client downloads prefetched data, and thus a higher throughput can be obtained. Recent measurement studies in [32] found that median bandwidth of Internet paths leading to residential broadband hosts is about 5 Mbps. But this paper noted that the downstream bandwidth allocated by DSL and cable modem providers varied widely. Median numbers for different providers studied varied roughly between 1-3 Mbps for DSL and 2-6 Mbps for cable modem. Contrast these numbers with a recent vehicular networking study in the Cabernet project [35] that promotes the use of 11 Mbps data rate for the 802.11 AP-to-client link. This is likely to improve with the use of 802.11g and/or beam steering technologies such as MobiSteer [82]. Also, round-trip times for multi-hop Internet paths are likely to be much larger than the one hop wireless link, affecting TCP throughput significantly than just what the bandwidth numbers show.

Second, the client can use a specialized transport protocol for downloading data from the AP which is not as sensitive to packet losses as TCP. This is not possible if the downloading client is only likely to run legacy protocols.

An important challenge in designing a prefetching protocol is that inaccurate predictions of mobility and/or connectivity can make make impacts in two different ways. When the estimation is ‘noisy’ (i.e., estimation error is relatively high), prefetching could be done ‘conservatively.’ This means that the ranges of data to be prefetched in subsequent APs could be allowed to overlap, likely to a significant extent. The extent of overlap depends on estimation errors. (An extreme view of this would be to prefetch all data on all APs, when no estimation is available). This, however, wastes backhaul bandwidth due to redundant prefetching. It wastes cache storage in APs as well. However, we do not anticipate storage to be a significant issue with the current advances in flash memory technology.

While lesser or zero overlaps save on backhaul bandwidth, they increase the possibility of ‘cache misses,’ slowing download performance. A thorough evaluation of estimation and prefetching strategies would be interesting directions of research. But, our focus in this chapter is more towards systems rather than performance evaluation. We propose the protocol architecture, build a proof-of-concept system, and experimentally demonstrate the performance potential of the prefetching protocol. We start with the protocol description in the following.

4.3.1 Protocol

We assume that APs are cooperative and can communicate with one another directly on the WAN side. This is possible if APs have a public IP address on the Internet, and they are part of the same peer or community group, or are managed by a service provider. These APs could also be nodes in a mesh network.

In our design we have used *HTTP range requests* (partial GETs [37]) for downloads (from client or AP to the server on the Internet). This makes our evaluation reflective of a standards-based protocol that is widely supported on HTTP servers. Similar HTTP range requests are also used for downloading prefetched content from the APs to the client as well. To facilitate this, the APs run a custom HTTP server to serve the prefetched data. The client either makes a HTTP range request to the AP, or to the original server on the Internet (or, to both sequentially). The byte ranges to be used in such requests depend on the byte ranges the AP might have prefetched and client might have downloaded previously.

We now describe the step-by-step operation of the prefetching protocol. The steps are also shown in the Figure 21. To simplify the description, the URL of the object to download is not explicitly mentioned in the below description.

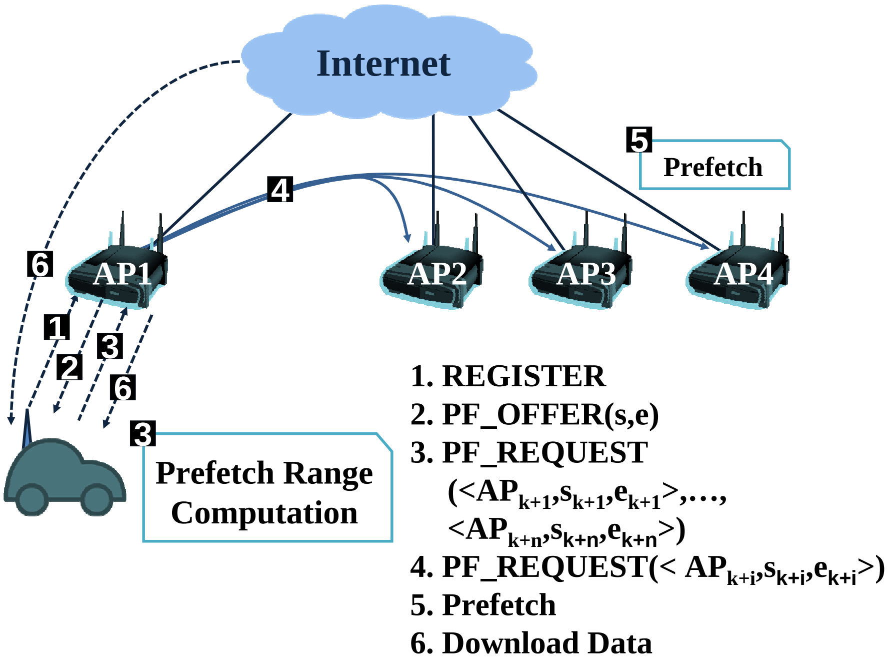

Internet
5
Prefetch
4
6
AP1
AP2
AP3
AP4
1
2
3
6
3
Prefetch Range Computation
1. REGISTER
2. PF_OFFER(s,e)
3. PF_REQUEST
(<AP_{k+1},s_{k+1},e_{k+1}>,…,
<AP_{k+n},s_{k+n},e_{k+n}>)
4. PF_REQUEST(< AP_{k+i},s_{k+i},e_{k+i}>)
5. Prefetch
6. Download Data

Figure 21: Architecture of prefetching protocol.

1. When a client connects with an AP (say AP_k), it sends a *REGISTER* message to the AP indicating the URL of the file it wishes to download, and requesting the range of bytes of the file that has already been prefetched at the AP (if at all).

2. The AP replies with the prefetched byte range to the client using the

message *PF_OFFER*(s, e), where s and e denote the start and end offset of the prefetched byte range respectively. If the AP does not have any prefetched data, it responds with an empty range, i.e., $(s, e = 0)$.

3. The client then computes the byte range that has to be prefetched at the next n APs (n is the lookahead we discussed in Section 3.5) and constructs a message of the format,

   $$PF\_REQUEST \quad (\langle AP_{k+1}, s_{k+1}, e_{k+1}\rangle, \cdots, \langle AP_{k+n}, s_{k+n}, e_{k+n}\rangle),$$

   where each tuple indicates the byte range a particular AP needs to prefetch. We will explain this computation momentarily in Section 4.3.2. The client then sends the above constructed message to the AP it is connected to.

4. The AP on receiving the above message splits it in n *PF_REQUEST*$(\langle AP_{k+i}, s_{k+i}, e_{k+i}\rangle)$ messages, one for every tuple, and sends them to the corresponding APs. Note our assumption that the APs can communicate directly to each other (over the Internet, for example).

5. Upon receiving a *PF_REQUEST* $(\langle AP_{k+i}, s_{k+i}, e_{k+i}\rangle)$ message, the AP AP_{k+i} prefetches the bytes ranging (s_{k+i}, e_{k+i}) from the server. If part or all of this range has already been prefetched, the AP prefetches only the difference.

6. Right after step 3, the client starts the download process. By observing the range (s, e) in the *PF_OFFER* message and the byte ranges the client has already downloaded before, the client determines the byte range of the download request for this AP, which could be a subset of the range (s, e). After the download from the AP is complete (or, if the client already had downloaded the entire byte range (s, e) prior to entering the current AP), the client makes a download request to directly to the server.

The client can download broadly in two ways:

(i) *Sequential download* – the client only fetches bytes sequentially. If the next sequential bytes have not been prefetched in the current AP, the client downloads directly from the server. This technique is appropriate for downloading media that could start playing even before the entire download is complete.

(ii) *Non-sequential download* – the client downloads opportunistically. It first downloads the bytes that have been prefetched in the current AP (excluding duplicates) even if such bytes are not sequential. Then the client downloads from the server byte ranges for which (a) no prefetch instruction has been given yet, and/or (b) prefetched byte ranges in prior APs that could not be downloaded because of insufficient contact time with the APs. The latter can happen when an AP prefetches successfully, but the mobile client has insufficient contact with the AP because of fast movement or poor link quality, for example.

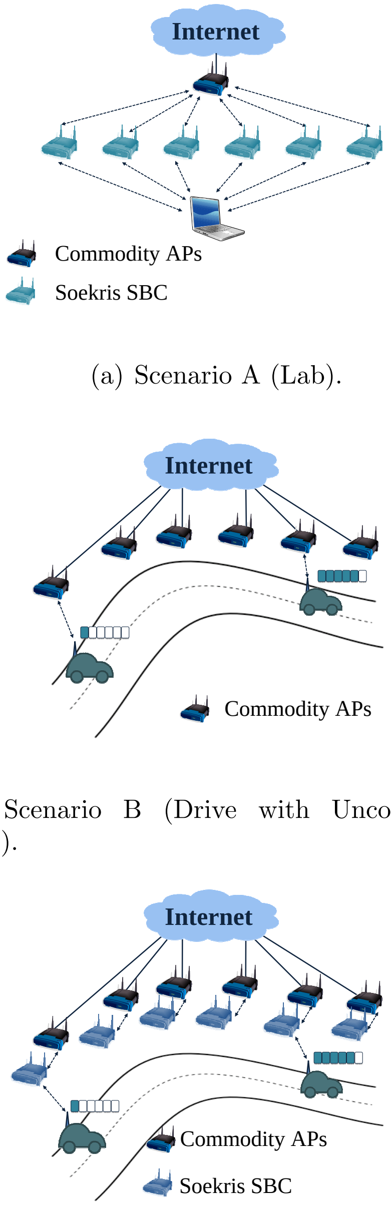
Evidently, non-sequential download can provide better throughput.

4.3.2 Prefetch Range Computation

The scripted handoff technique developed earlier already computes the handoff locations a priori. The analysis in Section 3.5 shows that given a time interval ΔT, the location of the car can be predicted quite well for small ΔT (few minutes). Thus, it is possible to estimate the time instances in the future when the client will connect with the APs ahead in the route. These estimates, together with the knowledge of the current location, the byte ranges already downloaded, and an estimate of AP-to-client bandwidth,[1] are used to compute the byte ranges of the file that should be prefetched on the APs the client is going to connect with in future. This constitutes the prefetch range computation.

As shown in Section 3.5, such estimates could be quite accurate for the near future, but losing accuracy further in the future. Thus, the challenge is determining the right 'lookahead' (n) value to use. This requires analysis

[1] This is the measured average bandwidth across various association sessions. More interesting, or online methods to estimate this are not evaluated in this chapter.



(a) Scenario A (Lab).

(b) Scenario B (Drive with Uncontrolled APs).

(c) Scenario C (Drive with Controlled APs).

Figure 22: Three experimental scenarios are shown. The commodity APs are already deployed in the residences. The Soekris SBC has two cards, one connecting the client and the other connecting to the commodity APs on two orthogonal channels.

with significant amount of urban traffic data. Lookahead will also depend on backhaul bandwidth as it influences the time needed to perform the prefetching. Data for backhaul bandwidth can be collected at the APs and be used for lookahead computations. Such analysis is beyond the scope of the current chapter. We do note that our protocol architecture is flexible, and can accommodate a wide range of estimation methods based on available information. The protocol allows for the *PF_REQUEST* instruction be received by the same AP multiple times, possibly with more refined or corrected estimates. However, this might incur some unnecessary or redundant downloads. Also, the estimates themselves can always be made conservative. For example, APs could be instructed to prefetch overlapping byte ranges. This simply shifts the burden to the backhaul to favor better pre-fetch performance.

Finally, note that in our design the onus is on the client to do such computation. The key reason for this design choice is that data used for estimation reveals driving habits and this can stay private with client-side computation. The downside of this design choice is that the client can potentially ask for excessively redundant prefetches overwhelming the AP backhauls. It is possible to address this issue using some sort of pricing mechanism, or limiting the total amount a client can ask to prefetch etc.

4.4 Evaluation of Prefetching

We use the metric *throughput gain* to measure the benefit of prefetching. This metric is the ratio of the throughputs with and without prefetching in an otherwise identical experimental condition.

The effectiveness of prefetching depends on several factors:

- Difference between backhaul bandwidth and client-to-AP link bandwidth: If the AP-to-client hop is a bottleneck, prefetching may not provide any tangible benefit. On the other hand, prefetch will be significantly beneficial for poor backhaul bandwidths.

- Use of overlapped ranges: As discussed before, the prefetch instructions could be conservative when estimation is poor. This allows for byte ranges prefetched on subsequent APs to overlap to different extents. In experiments we use *padding* to evaluate its impact. We assume that the byte range computation is non-overlapping, but then $x\%$ extra bytes ($x\%$ padding) are prefetched on both sides of the byte range.

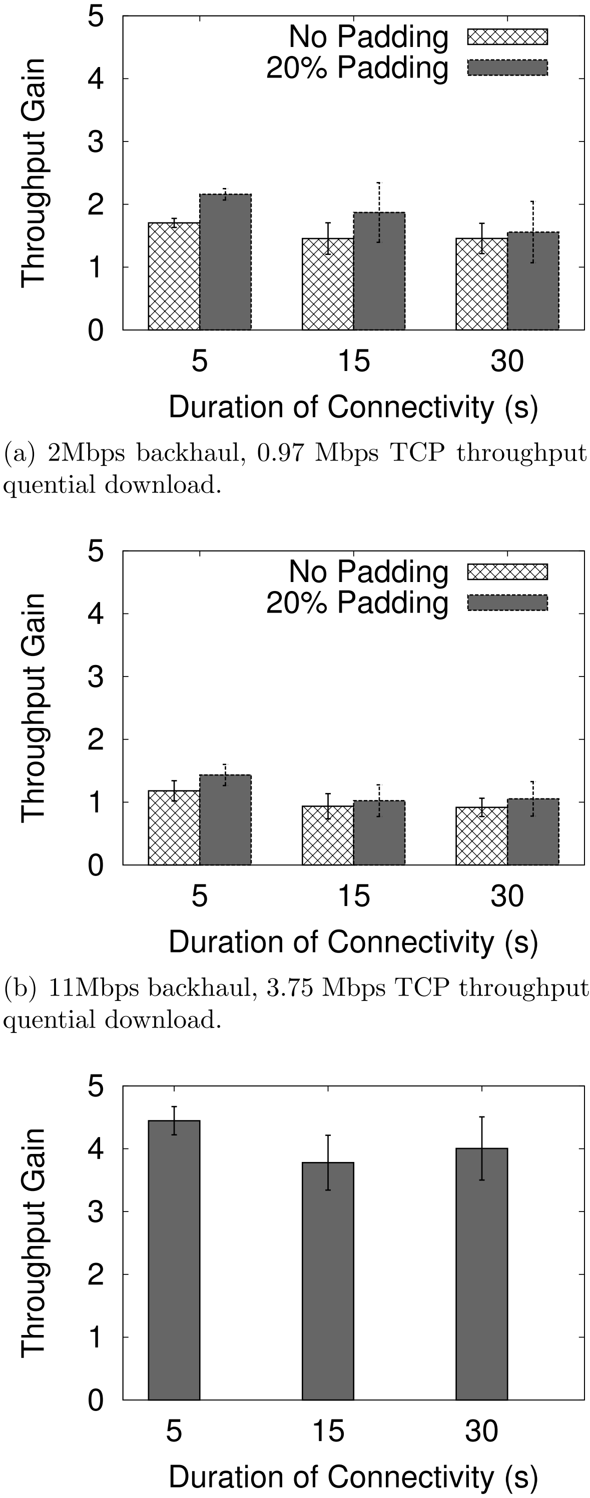
- Duration of connectivity: The throughput gain when prefetching also depends on the duration for which clients associate with the APs. Higher durations of connectivity should aid prefetching as it allows the client's auto-rate control to stabilize. Also, small scale variations of the vehicle speed have now less impact on prefetching performance.
- Download method: As discussed in the previous section, non-sequential download may improve throughput by allowing for opportunistic downloads.

We use three different scenarios to evaluate the impact of the above factors on the throughput gains achieved by prefetching. We first perform a comprehensive set of evaluation experiments in a controlled lab environment. Then, we supplement this evaluation with real driving experiments.

4.4.1 Controlled Lab Experiments

These experiments are in a controlled lab setting to quantify the benefits of the various factors discussed above. We set up six APs in an indoor lab environment in close vicinity, and have a client which simulates a drive through these APs by switching associations between them via a script. The client is in close vicinity with all APs and thus has a good link quality. The APs connect wirelessly to a wireless gateway, which in turn connects to the Internet. We call this as Scenario A as shown in Figure 22(a).

The APs used in this experiment are Soekris embedded SBCs as used in Section 3.4, with the difference that they consist of two 802.11 cards, one to provide access to the client, and the other to connect with the gateway. The interface which runs as an AP also runs a DHCP server. We use NAT (Network Address Translation) to switch packets between the two interfaces. The prefetching protocol described in the previous section is implemented on



(a) 2Mbps backhaul, 0.97 Mbps TCP throughput, sequential download.

(b) 11Mbps backhaul, 3.75 Mbps TCP throughput, sequential download.

(c) 11Mbps backhaul, 3.75 Mbps TCP throughput, non-sequential download, no padding.

Figure 23: Results from Scenario A (Lab) experiments.

this AP. We use these customized APs later in Scenario C as well, to be described momentarily. These APs provide the advantage that they can act as a bridge between the mobile client and an actual AP deployed by a provider where we cannot install any software.

As an indoor lab setup, this scenario differs from a real driving experiment in two aspects. First, the fluctuations in link qualities experienced while driving are not experienced here. Second, as association changes are driven via a script, the mobility estimate is almost perfect. Of course, noise in these parameters can be simulated, but we do not do it here, as we will also describe actual driving experiments using the same hardware in Scenarios B and C.

For the above described scenario, we perform 'simulated' driving experiments, where the client associates sequentially with the six APs, and downloads a very large file from an Internet server 10 hops away. To evaluate the various factors which impact prefetching, we make the following choices. The APs prefetch the estimated byte range with 0% or 20% padding. The duration of connectivity is chosen as 5, 15, or 30 seconds (same for all APs), which are within the typical range found in our driving measurements. We vary the backhaul link badwidth between 2 Mbps and 11 Mbps, while the client connects with the AP using auto-rate algorithm, which generally uses 11 Mbps due to the excellent link qualities. We also experiment with sequential and non-sequential downloads. We perform 15 runs for each experiment. The results for the average throughput gain along with the 90% confidence interval are shown in Figure 23.

Though the use of padding appears to improve throughput for sequential download, the improvement is actually in the noise. Similarly, the difference in performance due to different connectivity duration also appears to be in the noise. However, roughly 50% or more throughput gain is observed when sequential download is used and the backhaul link to the gateway is poor. As expected, the improvement is very marginal, if at all, when the backhaul link has higher capacity similar to the AP-to-client link. However, when non-sequential download is used, because of the use of opportunism the performance improvement is significant, roughly by a factor of 4, EVen with the higher capacity backhaul.

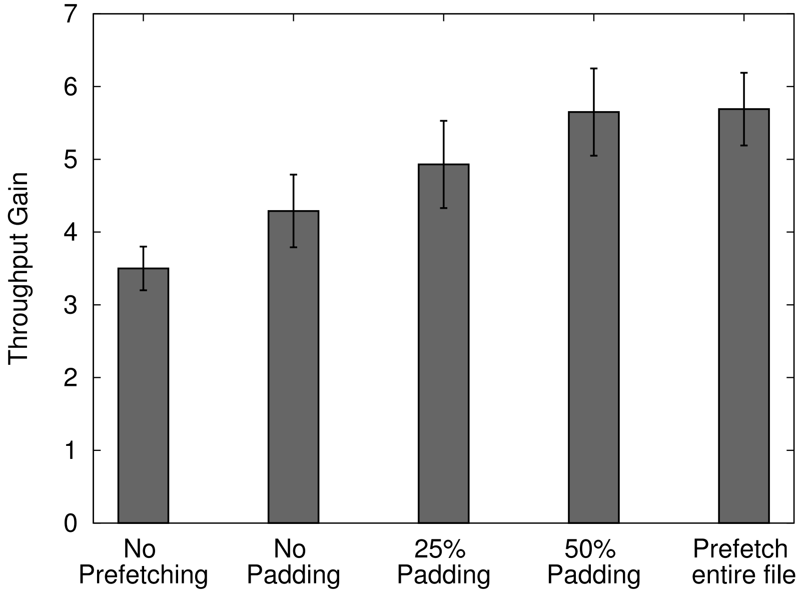


Figure 24: Results for Scenario B (driving experiments with uncontrolled APs) with sequential downloads.

4.4.2 Driving Experiments with Uncontrolled APs

We now evaluate prefetching in realistic driving scenarios. Here, we cannot control the backhaul bandwidth and the duration of connectivity, as they now come from historical estimates. We show the throughput gains when using prefetching, and using different padding levels and sequential downloads.

We use the same location where we evaluated handoff strategies as shown in Figure 20. This experimental setup is shown as Scenario B in Figure 22(b). Since the APs at the location are not in our control[2] we cannot install any software on them to do prefetching. Here, we 'emulate' prefetching using exchange of large ping packets instead of actual file download. Prefetched data downloaded from an AP to a client is emulated as a one-hop ping on the wireless link between them. Data being downloaded from the server is emulated as an end-to-end ping between the server and the client. The client generates

[2] In fact, they are various commodity APs that are closed boxes belonging to the residents in the community.

1400 byte ping request packets and waits for a reply for 1 second. On receiving a reply, or after a 1 second timeout, whichever comes earlier, another ping request is generated. This simulates a simple, stop-and-go transport protocol. The choice of transport protocol is not important since we choose the same transport protocol for prefetching and for downloading from the server on the Internet. We choose to use ping because it is the only packet that commercial APs respond to. Based on the successful pings received, the client maintains an offset of the last byte downloaded. The client maintains the prefetch ranges that are computed and acts as if the ranges are always successfully prefetched at the APs. Thus, there is no messages such as *PF_OFFER* or *PF_REQUEST*, etc.

In Figure 24 we present the averaged results (along with the 90% confidence intervals) in the form of the total data in bytes that are downloaded per drive. There are 10 drives for each different prefetch strategy as outlined in the figure labels. Each drive is approximately 120 s long. For handoffs we have used the scripted handoff (SH) technique described before. Note approximately 20-60% improvement in download performance by using prefetch with various padding levels. With 50% padding, there is an 100% overhead as each byte is prefetched twice on different APs; however, the download performance is about 60% higher.

4.4.3 Driving Experiments with Controlled APs

These experiments are the closest to a real deployment that we have envisioned in this chapter. See Figure 22(c). This is same as the Scenario B except that APs with the implementation of the prefetch protocol as used in Scenario A are now also deployed in the residential area. These APs connect to the same 6 neighborhood APs in Scenario B that now act as gateways.

For this deployment we use higher power 802.11b/g cards 25 dBm on the Soekris SBCs for better wireless link quality. Similar cards are also used on the client side. This is done to make the experiments more similar to Scenario B before. In our experience, the commodity APs in the residences are already using wireless interfaces in a similar power range. In general, however,

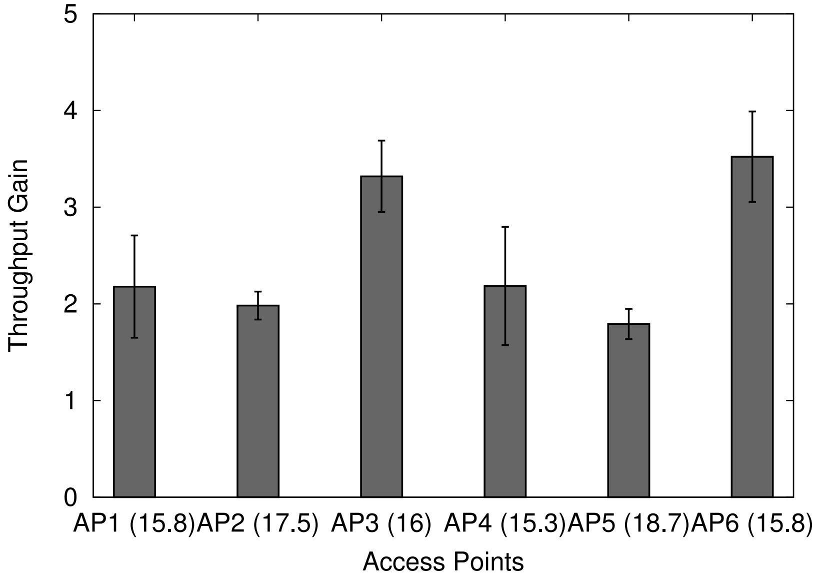


Figure 25: Results for Scenario C (driving experiments with controlled APs) with non-sequential downloads and no padding. The numbers within bracket are the connectivity durations in sec for the APs.

since the scenarios A,B,C are using slightly different hardware, the data are not comparable across scenarios. However, relative comparisons within each scenario are still quite valuable.

In these experiments we perform 10 drives. Non-sequential download is used as it provides the maximum potential performance benefit. No padding is used. Note the significant performance gain over no prefetching scheme with prefetch now shown in a per-AP fashion (Figure 25). No strong correlation is observed for the connectivity duration which are all somewhat similar. The performance differences are likely due to variations on the backhaul bandwidth and the quality of the AP-to-client links. The overall aggregate throughput gain is about 2.5.

4.5 Conclusions

The use of prefetching improves the performance by a factor of another 2.5, if non-sequential download is chosen. Overall, our experiments show that more than 200 Kbps application layer throughput is possible with scripted handoff and over 500 Kbps with the use of pre-fetching, rivaling cellular data speeds on many available technologies. Note also that the prefetching technique is link-layer independent and should be useful in cellular data networks as well.

Chapter 5

BRAVE: Bit-rate Adaptation in Vehicular Environments

5.1 Introduction

Choosing an appropriate physical (PHY) layer rate is important when a WiFi network is accessed with vehicular mobility. The 802.11 protocol specification allows for multiple transmission rates at the PHY layer each having a different modulation and coding scheme. For example, the 802.11b offers 4 different bit rates ranging from 1 to 11 Mbps and 802.11g offers 8 different bit rates ranging from 6 to 54 Mbps. 802.11b/g interfaces are quite common and thus typically there is a choice of as many as 12 different bit rates. Generally speaking, higher bit rates result in higher throughputs if the link quality is good but suffer from higher bit error rate (BER) when the link quality deteriorates. Conversely, lower bit rates provide lower throughput but are more resilient to lower quality links. 802.11 interfaces typically deploy a Rate Adaptation Algorithm (RAA) to optimize the chosen rate so as to maximize the overall throughput.

While RAAs have been widely studied, most of them [79, 4, 116] are targeted for traditional indoor wireless networks and are incapable of quickly adapting to the rapidly changing link quality experienced by the mobile vehicular client [57]. They tend to either under-estimate or over-estimate the real-time link quality and hence end up choosing incorrect bit rates. The unsuitability of industry-standard RAAs towards vehicular WiFi access is also highlighted in our past measurements in [30]. We also noticed that throughputs dropped with increasing vehicular speed since the link quality changed more rapidly at higher speeds adversely affecting the performance of the RAA.

All RAAs try to predict the real-time link quality by examining the past history – often over several seconds or longer. Generally speaking, RAAs belong to one of two classes. *Frame-based* algorithms maintain an estimation window for the link layer metrics such as frame errors, while the *SNR-based* algorithms maintain an estimation window for physical layer metrics such as received signal strength indicator (RSSI) that indicate SNR. In our knowledge the literature does not provide a clear guidance as to what type of RAA should be used in what scenario. Also, the performance of these RAAs in vehicular environments has not been studied. All evaluations are limited to indoor environments only. Indoor evaluations hardly indicate performance under vehicular mobility as the link quality fluctuates rapidly in the latter case and often on a per-packet basis. Thus, the RAA must adapt very quickly.

In this chapter we develop a new SNR-based RAA called BRAVE (Bit Rate Adaptation in Vehicular Environment), particularly targeting vehicular mobility. BRAVE estimates upcoming SNRs to be experienced on the link based on a short history of recent SNRs and exploits its knowledge of BER performances for different SNRs. We employ a coarse-grained training approach to estimate the SNR thresholds for rate selection as opposed to previous approaches that train on a per environment [26] or a per AP basis [58]. We evaluate the performance of BRAVE relative to several prominent RAAs. Given the difficulty of doing a head-to-head comparison under vehicular mobility (vehicular experiments are hard to repeat), we also use a novel emulation methodology. Here, a software radio-based programmable noise generator is used to emulate varying signal quality under vehicular mobility. We show that BRAVE performs significantly better than four other popular RAAs.

5.2 Testbed

For our study, we use a metro-scale WiFi deployment in the Long Island area in New York (population roughly 7 million). This service is called ‘Optimum WiFi’ [5] and is provided by Cablevision, a local cable TV provider and ISP. The WiFi network extends to the more populous areas of Long Island where our study is conducted. It also extends to parts of New York City, Pennsylvania, Connecticut and New Jersey, where Cablevision has service. The entire network has roughly 18,000 APs. WiFi access is provided free to all subscribers of Cablevision’s TV or Internet services. We note that there are several hundreds of such metro-scale WiFi deployments in USA alone [81].

We use two Dell Latitude laptops running Linux to be carried in the car. One laptop acts as the vehicular client and the other acts as a sniffer. The original miniPCI WiFi interfaces from the laptops are removed and replaced by carrier-grade interfaces (Ubiquity XR2 [6]) with transmit power set to 25 dBm. The interfaces use Atheros chipset supported by the latest Madwifi driver (version 0.9.4) which we used. The WiFi cards are connected to 8 dBi omnidirectional antennas using two 2 meter long co-axial cables. The antennas are carried at the back of the car as shown in Figure 27.

The client and the sniffer machines are connected using ethernet. A program on the client machine continuously monitors the current channel and notifies the sniffer machine in case of a channel change. Thus the sniffer always remains in the same channel as the client machine.

On the client, we set the WiFi card’s AP selection mode to automatic, i.e. the Madwifi driver decides on the best AP to connect to. DHCP is used to obtain the IP address. The Optimum WiFi network allows clients to retain the IP address between associations; thus DHCP delay is incurred only once at the beginning. Optimum WiFi uses a browser-based authentication for the initial network access. This is done manually. Again this step is needed only once. Authentication is retained across associations.

The client generates upload traffic using the well known network performance evaluation utility `iperf` (version 2.0.5) . The iperf client is configured to send out 1400 byte UDP packets in a back-to-back manner. The sniffer logs

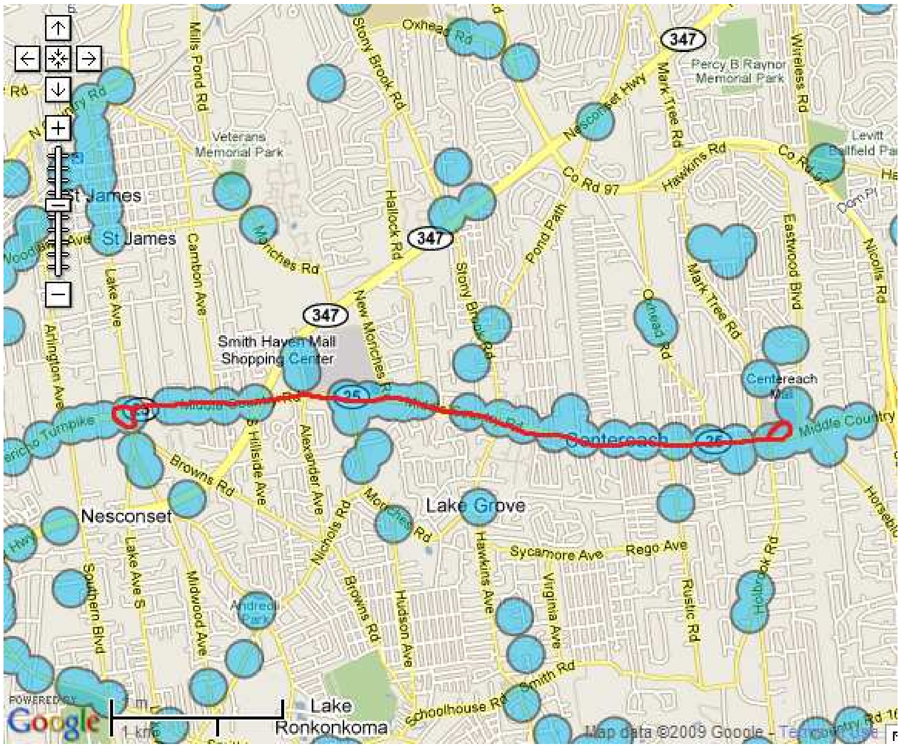


Figure 26: Map of the stretch of the road used for the driving experiments, along with the route shown in red. Approximate WiFi coverages are shown (from [5]).

all traffic on the current channel.

Figure 26 shows the road stretch used in the experiments with approximate locations of WiFi APs. This is a 9 mile round-trip drive and is used to conduct all our experiments.

5.3 Understanding Dynamic Channels

To design an RAA in vehicular scenario it is important to understand the channel dynamics that the algorithm must adapt to. It is also important to use a good measure of channel conditions. Following the general motivations behind the design of a major SNR-based RAA, CHARM [58], we use SNR

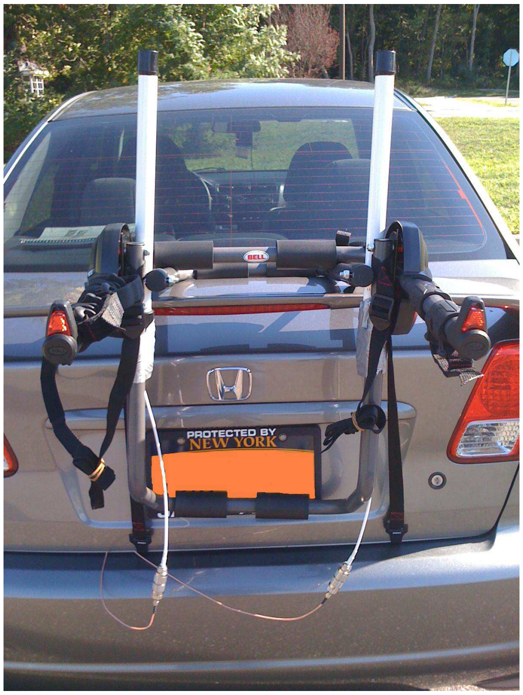


Figure 27: Two 8 dBi omni-directional antennas are mounted at the back of the car using a bicycle rack.

as an indicator for channel conditions. It is well known that SNR determines the bit error rate (BER) for a wireless channel for a given PHY layer rate. Thus, direct measurement of SNR is expected to provide a simple mechanism to estimate the most effective rate to be used. This measurement can be done on a per-packet basis providing a method for quick adaptation. This is in contrast to frame-based methods that require several packet transmissions to build up a reliable statistics for the error rate.

All 802.11 interfaces report RSSI for each received frames. Depending on the interface and driver, the RSSI can be the measured SNR or the measured received signal strength. In the latter case, SNR can be calculated from a separately reported noise floor. In our reported experiments, RSSI is the same as SNR reported directly by the driver.

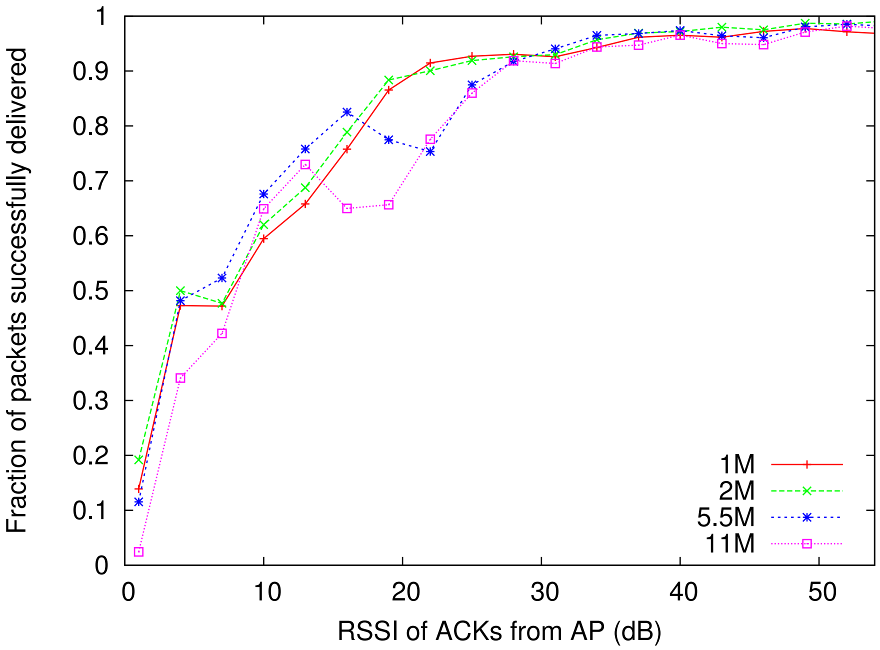
In the following we carry out a set of measurements to understand the error performance of 802.11b/g interface under vehicular mobility. This will form the basis of the BRAVE algorithm that we will describe subsequently.

5.3.1 The CycleRate Algorithm

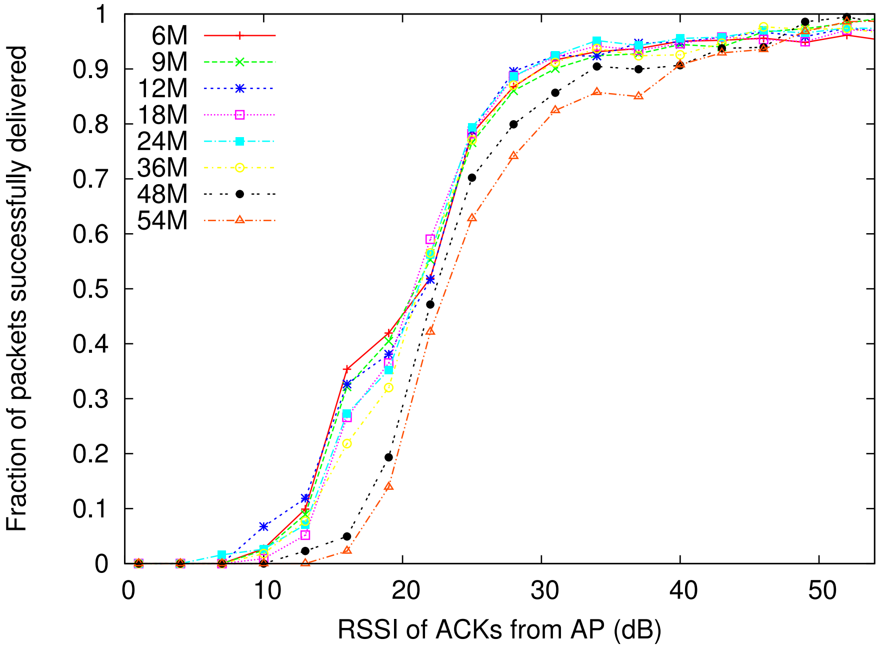
To understand the error performance of all 802.11b/g bit rates under different channel conditions we create a rate selection algorithm that simply loops through all the available bit rates on a per packet basis. We disable retries so that every packet is transmitted exactly once. Thus, there is no exponentially increasing backoff delays between transmissions. We call this the CycleRate algorithm. On average, the duration of one complete cycle is approximately 27 ms. However, this is much larger than the channel coherence time in outdoor environments. In [26] the authors have determined the channel coherence time to be as small as 300 μs in realistic vehicular WiFi access settings. Thus, we do not expect the different PHY layer rates to be directly comparable for error performance within a single cycle of the CycleRate algorithm but on average we believe our measurement methodology provides a good first order approximation of their error performances at different SNR levels.

We conduct a driving experiment using the CycleRate algorithm using our experimental setup described in Section 5.2. UDP upload from the vehicular client to a server on the Internet is used as the traffic load. A saturated load is used – UDP packets are transmitted as fast as possible. The chosen packet size is 1400 bytes. We choose to use upload experiments instead of downloads as the WiFi APs are not under our experimental control. The assumption, however, is that the wireless channels are symmetric and the lessons learnt in these experiments are equally applicable for download scenarios.

In Figure 28 we plot the fraction of packets successfully delivered at different RSSI values. The RSSI is measured off the last received ACK packet. We see that 802.11b and 802.11g have distinctly different packet loss characteristics. The DSSS modulation scheme used for 802.11b rates tends to be



(a) Success fraction for 802.11b rates.



(b) Success fraction for 802.11g rates.

Figure 28: Fraction of packets successfully delivered at varying RSSI levels for various 802.11b/g rates.

AR(k) Model	MSE(Y$_{est}$)/VAR(Y)
AR(1)	0.932
AR(2)	0.928
AR(4)	0.924
AR(8)	0.922
AR(16)	0.920

Table 4: Prediction accuracy using AR(k) model.

much more resilient to packet losses than the OFDM modulation scheme used for 802.11g rates. Within each group, the lower rates show further resilience to packet loss because bit error rate (BER) increases with bits/symbol and lower rates are encoded with smaller bits/symbol. This is more clearly seen in case of 802.11g rates than in case of 802.11b rates. Note that the theoretical proportionality of BER to bits/symbol holds only when the channel remains coherent across the measurement duration. It is practically impossible to attain large channel coherence times in our measurement scenario. This reflects in our analysis in Figure 28 where in several cases the theoretical proportionality of BER to bits/symbol is not held. The 802.11b rates seem to be robust even at low RSSI levels but 802.11g rates start becoming useful when the RSSI is above 20 dB. It is clear that an RAA should use OFDM rates at higher RSSI levels and DSSS rates at lower RSSI levels.

5.3.2 Predicting RSSI

Predicting the channel quality is critical in determining the best bit rate. The study of temporal correlation between RSSI values under vehicular mobility is useful in this regard. To study this, we use the autocorrelation in the time series of RSSIs to estimate how well the recent RSSI values can predict the RSSI values to be seen in the near future.

The prediction model can use one of several standard techniques such as autoregressive (AR) model, moving average (MA) model, autoregressive moving average (ARMA) model, etc. In [30] we have used the AR model successfully to predict the future throughputs based on throughput measurements in the past in a similar vehicular environment. Thus, we limit our analysis to the AR model.

For this analysis we consider a time series of average RSSI values over 500 ms time slots. We evaluate the performance of the AR(k) model for several values of k to see the quality of estimation. Table 4 shows the mean-squared estimation error divided by the sample variance (MSE/s^2). This is based on a standard AR(k) fitting method used in the mathematical software `GNU Octave, version 3.0.5`, when applied on the average RSSI per time slot time series as described above.[1] Note that there is very little difference with different values of k. This implies that the simplest AR(1) model can provide us with a good prediction of the average RSSI in the next 500 ms time slot.

It is interesting to study whether a higher variability in the RSSI values within a time slot leads to better prediction of the RSSI values in the next time slot. In Figure 29 we plot the standard deviation of the RSSI values within a time slot versus the error between the observed and the predicted RSSI of the next time slot. For this analysis we consider only those time slots where we have at least 10 RSSI values in order to obtain statistically meaningful results. The predicted RSSI in this case is determined using the AR(1) model described above. We see that as the standard deviation increases prediction error also increases. We consider an error of 3 dB or higher as significant. A lower difference is not likely to influence the bit error rate appreciably as per Figure 28. Note that the error is 3 dB when the standard deviation is approximately 3 dB.

Figure 30 shows the CDF of the standard deviation of RSSI values within each 500 ms time slot. Again, only those time slots are considered where we have at least 10 RSSI values. Notice that in approximately 90% of the cases the standard deviation is within 3 dB indicating that the prediction error will be

[1] If x_t is the average RSSI at the t-th time slot, the AR(k) model estimates x_t as $\hat{x_t} = c + \sum_{i=1}^{k} \phi_i x_{t-i} + \epsilon_t$, where c is a constant, ϕ_i's are the parameters of the model, and ϵ_t is white noise. Each time slot is considered to be 500 ms.

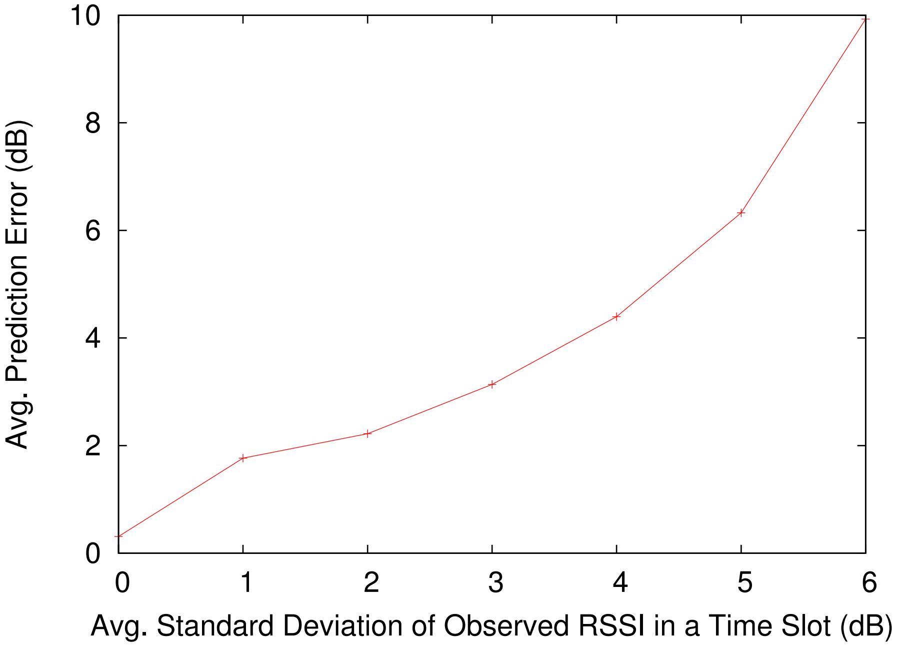


Figure 29: Prediction error increases with increasing standard deviation of the RSSI values within a time slot.

within 3 dB. From this we conclude that in 90% of the cases the predictability of RSSI using the AR(1) model is acceptable for the purpose of a rate selection mechanism.

5.4 The BRAVE Protocol

The BRAVE (Bit-Rate Adaptation in Vehicular Environments) protocol makes use of the popular Madwifi device driver [3]. The Madwifi Hardware Abstraction Layer (HAL) provides the capability of multi-rate retry (MRR). The MRR array can be populated with up to four different rates and the number of retry attempts for each rate. These rates are to be attempted sequentially for the given number of times until ACK is received or the total number of retries is exhausted. In the latter case the frame is dropped.

BRAVE selects bit rates for outgoing frames based on the RSSI values of

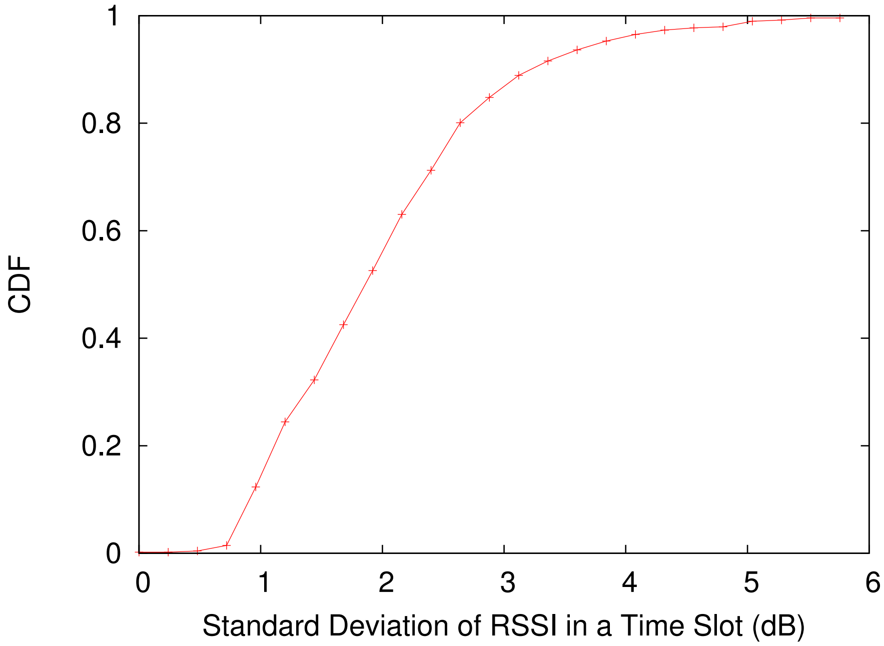


Figure 30: CDF of standard deviation computed over RSSI values in 500 ms time intervals.

the previously received ACKs. It makes use of the MRR array by appropriately populating the rates that should be attempted. We now describe the operation of BRAVE.

The BRAVE protocol operates in two modes – *AGGRO* (aggressive) and $SAFE$ (conservative). The *AGGRO* mode is used when the predictability of future RSSI values is good and the $SAFE$ mode is used when the predictability is poor. Based on the analysis in the previous section, the standard deviation of prior RSSI values is used as an indicator of predictability.

From Figure 28 one can deduce that at RSSI levels below 20 dB the 802.11b (DSSS) rates provide the highest throughputs and above 20 dB the 802.11g (OFDM) rates provide the highest throughputs. Table 5 shows the rates used by BRAVE in both the *AGGRO* and $SAFE$ modes at different RSSI levels. For the *AGGRO* mode, the three rates that provide the highest throughput at every RSSI level make up the first three elements of the MRR array. The fourth element is always chosen to be the base rate of 1 Mbps as

RSSI Range	*AGGRO* MRR Chain	*SAFE* MRR Chain
Under 20dB	11, 5.5, 2, 1	11, 1, 1, 1
20 – 28 dB	48, 36, 11, 1	48, 11, 5.5, 1
Over 28 dB	54, 48, 36, 1	54, 11, 5.5, 1

Table 5: Rates in the Multi-Rate Retry array for BRAVE. Each rate is attempted only once. *SAFE* mode is used in case the standard deviation of RSSI values in the previous time slot is more than 3 dB.

this rate has the lowest error rate at any RSSI level. For the *SAFE* mode, BRAVE sets the first MRR array element as the rate that provides the best throughput from the entire rate set and the remaining elements are chosen from the 802.11b rate set. This is because 802.11b rates show higher resilience to frame loss. Overall, the BRAVE protocol works as follow.

1. BRAVE makes rate adaptation decisions every time slot (500 ms).

2. At startup, BRAVE uses the *SAFE* mode and assumes the average RSSI over the previous time slot to be 0.

3. If at least 10 ACKs are received in the previous time slot of 500 ms and the standard deviation of the RSSI values in the last slot is less than 3, BRAVE switches to the *AGGRO* mode. Otherwise the mode remains as *SAFE*.

4. Depending on the average RSSI in the previous time slot and the mode of operation, the MRR array is populated with values from Table 5. Each rate is tried only once.

In its current implementation BRAVE uses static RSSI threshold levels (as in Table 5) based on prior analysis with data collected using the CycleRate algorithm described in Section 5.3.1. Our expectation is that a training similar to the above needs to be done on a per Metro-area WiFi network basis to create a base line. This is feasible as the APs belonging to a network deployment tend to be identically configured both in hardware and software. Such a training component can be potentially incorporated within the BRAVE protocol itself.

	SampleRate				*AMRR*				*CHARM*			
Bit-Rate	% Used		% ACKed		% Used		% ACKed		% Used		% ACKed	
(Mb/s)	GL	PL	GL	PL	GL	PL	GL	PL	GL	PL	GL	PL
1	1.61	24.0	59.8	5.80	3.61	43.9	59.3	16.2	0.03	0.83	5.97	0.44
2	1.89	6.73	74.6	20.9	0.75	1.52	66.7	21.7	0.00	0.16	0	0.78
5.5	18.0	12.3	86.0	18.4	1.12	1.04	79.6	50.1	0.02	0.51	0	0.36
6	7.25	0.23	90.6	1.40	1.67	3.51	55.9	0.42	0	0	0	0
9	0.02	2.77	3.37	0	1.87	1.33	35.6	0	0	0	0	0
11	26.7	11.6	79.1	14.7	4.89	1.52	84.0	23.7	0.03	1.05	0	0.08
12	7.36	2.19	77.1	0.15	5.09	7.58	63.8	0.04	0	0	0	0
18	3.85	5.60	77.3	0	3.19	10.1	60.8	0	0.05	4.72	0	0
24	9.53	7.05	83.2	0	5.01	10.5	63.3	0	0.07	4.73	0.74	0
36	14.8	11.3	87.9	0	7.31	18.8	60.7	0	0.09	3.99	0	0
48	5.22	7.35	82.2	0	10.8	0	71.7	0	0.00	6.24	0	0
54	3.54	8.72	73.8	0.03	54.5	0	92.2	0	99.6	77.7	49.7	0.52

	RapidSample				*BRAVE*			
Bit-Rate	% Used		% ACKed		% Used		% ACKed	
(Mb/s)	GL	PL	GL	PL	GL	PL	GL	PL
1	8.56	34.8	85.9	63.4	0	0.04	0	70
2	6.04	3.81	96.6	83.7	0.05	2.20	100	99.6
5.5	7.65	3.84	98.1	83.5	0.71	19.6	93.0	88.7
6	7.06	3.72	56.1	0.65	0	0	0	0
9	7.78	3.26	54.3	0.51	0	0	0	0
11	8.39	3.79	97.8	75.9	5.50	77.2	86.9	74.5
12	7.08	3.15	60.8	0.83	0	0	0	0
18	6.90	3.94	61.5	0.71	0	0	0	0
24	7.67	3.88	61.5	0.53	0	0	0	0
36	7.75	6.69	59.4	0.25	1.66	0.28	80.7	22.1
48	9.37	10.2	54.2	0.62	9.25	0.33	82.1	16.0
54	15.6	18.7	50.2	0.35	82.8	0.35	88.8	4.70

Table 6: Physical layer bit rates used for different RAAs. When the RSSI is below 20 dB the link is classified as being a 'Poor link' (PL). Otherwise, it is classified as a 'Good link' (GL). The percentage of transmissions using specific bit rates and percentage of transmissions ACKed are shown.

5.5 Evaluation

In this section we compare BRAVE with four other RAAs. They will be described momentarily. All experiments are done on the 9 mile stretch of road shown in Figure 26 over multiple drives, each drive using one specific RAA. The AP density in this stretch of the road is quite high. The experimental setup is the same as described in Section 5.2. Unfortunately, the vehicular experiments are not repeatable, as it is virtually impossible to ensure that the vehicle is driven exactly at the same speed at every location on the road on the repeated drives. Thus, the wireless link quality and AP associations vary differently across time for different drives. It is possible to 'ride out' the impact of this variability by using a very large sample size for the RAAs (i.e., many drives). However, this would be an enormous experimental effort given that several RAAs are to be evaluated. Thus, a different evaluation method is used.

Table 6 shows the performance of each of the algorithms under conditions of good and poor link quality. Link quality is considered to be good if the RSSI is 20 dB or higher and considered to be poor otherwise. We show the usage percentage for each rate and the percentage of times each rate is successful. This RSSI-based grouping allows us to evaluate how an algorithm performs when faced with different channel conditions.

In the subsections below we describe and evaluate each of the algorithms used in this analysis.

5.5.1 SampleRate

SampleRate [79] is a frame-based RAA. It is the default RAA in Madwifi. *SampleRate* adapts rates based on transmission statistics over a sliding window. The transmission time for a frame is defined as the time to send the frame successfully. This includes time for retransmissions and backoffs. The bit rate chosen is the one that achieved the smallest *average transmission time* in the last sampling period of 10 seconds. This way, *SampleRate* tries to achieve the best average throughput in the long term. *SampleRate* starts transmitting at the highest rate and decreases the rate immediately if it experiences four consecutive transmission failures. To explore other rates that could provide better transmission times, *SampleRate* randomly selects the rate from the set of rates whose average transmission time is less than the average lossless transmission time of the rate in use, as the rate for every tenth frame. Also those rates that have experienced 4 consecutive transmission failures are excluded from this sampling.

From the above description it can be easily deduced that *SampleRate* is not the ideal choice for highly dynamic channel conditions. The sampling period of 10 seconds is too large given the highly dynamic nature of the vehicular wireless channel. Also, a random loss of a probe packet which is non-reflective of the channel conditions proves to be costly to *SampleRate* since the statistics for the probed bit-rates are built slowly – alternate rates are probed at every tenth packet. This makes *SampleRate* prone to under-selecting rates.

From Table 6 we can see that even when the link quality is good, *SampleRate* relies heavily on the 11 Mbps and 5.5 Mbps bit rates. When it does choose higher OFDM bit rates (24 Mbps and 36 Mbps), the link quality tends to be excellent. This is highlighted by the fact that 83.2% and 87.9% of packets sent at these bit rates receive link layer ACKs. *SampleRate* is also quite slow in reacting to a drop in link quality. It sends out about 40% of its packets at rates above 11 Mbps even when very few packets are actually acknowledged. Thus, we conclude that *SampleRate* is slow in reacting to changing channel conditions, and tends to under-select bit rates when the link quality is good and over-select bit rates when the link quality is poor.

5.5.2 AMRR

The *Adaptive Multirate Retry (AMRR)* algorithm [65] is also a frame-based algorithm whose implementation is a part of Madwifi. It also adapts its rate based on transmission statistics but the period over which rate selection decisions are made is shorter. *AMRR* increases the current bit rate by one if in each of the last 10 successive 500 ms time-slots at least 10 packets were transmitted and less than 10% of packet transmissions failed. The bit rate is decreased by one if more than 33% of the packets fail in a 500 ms time slot. In *AMRR* the selected rate and two rates below it constitute the first 3 elements of the MRR chain. The fourth rate is set to 1 Mbps. One transmission attempt is made per rate in the MRR chain.

Referring to Table 6 *AMRR* tends to be very aggressive in rate selection when the link is good. The three highest rates are selected over 72% of times.

However, when the link quality is poor, the same aggressiveness tends to pull it away from the more stable DSSS rates to higher OFDM rates that are not as successful. Close to half of the packets are sent at rates higher than 11 Mbps and almost none of these packets are successfully delivered. Hence overall, *AMRR* is guilty of over-selecting rates.

5.5.3 CHARM

Channel-aware Rate Adaptation Algorithm (CHARM) [58] is an SNR based rate adaptation algorithm. *CHARM* uses channel reciprocity to obtain channel information. The technique is based on predicting the path loss to a receiver in the near future by passively overhearing messages sent by the receiver. *CHARM* automatically adapts the signal thresholds that are used by the transmitter to select the transmission rate. This adaptation is however very slow – in the order of a few seconds.

CHARM is known to outperform popular frame based RAAs like *SampleRate* in indoor environments [58]. However in outdoor environments at vehicular speeds *CHARM* performs very poorly. Our experiments show that *CHARM* continues to predict the link quality as being excellent even when the link quality is very poor in reality. Referring to Table 6, in case the link quality is good, *CHARM* almost exclusively uses the highest available bit rate. However, *CHARM* is very slow in reacting to deteriorating link quality and continues to use the highest bit rate over three quarters of the time even when the link quality deteriorates.

5.5.4 RapidSample

RapidSample [103] is a frame-based algorithm that has been developed specifically for mobile environments. The algorithm starts with the fastest bit rate. If a packet fails to get a link layer ACK it records the time of failure and switches to the next lower bit rate. After success at a particular bit rate for more than 5 ms the sender attempts to sample the highest bit rate that has not failed in the last 10 ms and there is no lower bit rate that has failed in the last 10 ms. If the faster rate fails, it reverts to the original bit rate; otherwise it adopts the new bit rate.

We have implemented *RapidSample* as a Madwifi rate adaptation module. It is important to note that *RapidSample* does not rely on the MRR mechanism and hence there are no retries at the hardware level. Every packet loss is reported to the algorithm and a different rate selection is made. We reflect this in our implementation by disabling multi-rate retry in Madwifi. This makes *RapidSample* very susceptible to packet losses that happen frequently in outdoor mobile environments. Also, RapidSample is tuned for indoor channel coherence times (approximately 10 ms) and walking speeds. Outdoor channel coherence time is much shorter.

From Table 6, we see that *RapidSample* uses all bit rates approximately equally when the link quality is good. This is unlike other RAAs which tend to converge at one stable rate. This highlights the problem of being overly sensitive to transmission failures which is compounded by the absence of multi-rate retry capability. During poor link conditions *RapidSample*, tends to converge at the lowest and the highest bit rates. Approximately half of transmissions happen at these two rates. The convergence at 1 Mbps bit rate can be explained by the increased number of packet losses due to poor link quality. The convergence at 54 Mbps bit rate can be explained by the tendency of the algorithm to jump to the highest bit rate that has not failed in the last 10 ms.

5.5.5 BRAVE

We implement the proposed *BRAVE* protocol as a rate adaptation module in the Madwifi driver (version 0.9.4). The operation of the *BRAVE* protocol has been previously described in Section 5.4. *BRAVE* correctly estimates the link quality and appropriately selects bit rates for outgoing packets.

From Table 6 note that *BRAVE* selects the highest 3 bit rates almost 93% of the times when the link quality is good. Also the packets sent at these rates are ACKed with a very high probability. In case of poor link quality, BRAVE selects the loss resilient 802.11b bit rates around 99% of the times. The link quality is overestimated less than 1

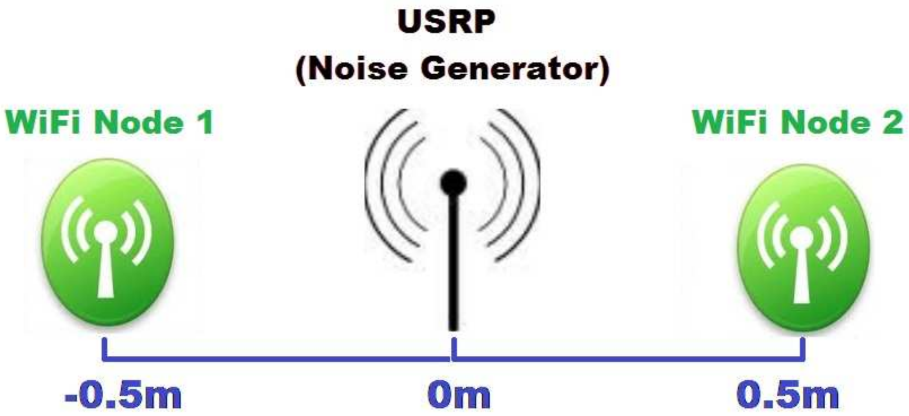


Figure 31: Emulation setup showing two WiFi nodes 1 meter apart with the USRP noise generator exactly in the middle.

5.6 Vehicular Link Emulation

We mentioned previously the non-repeatability of vehicular experiments that prevented us from doing a real head-to-head comparison of RAAs. In this section we describe an indoor emulation environment that can vary wireless link qualities in a repeatable fashion. Though realistic outdoor channels cannot be emulated, variability in the link qualities in a repeatable fashion still is quite useful for a first-order comparison of RAAs. The essential idea is to use a software radio as a programmable noise generator.

5.6.1 Emulation Setup

The emulation setup as shown in Figure 31 consists of two 802.11 devices in the 'ad hoc' mode placed 1 meter apart. This ensures that the nominal link quality is excellent between the two nodes. We use a Universal Software Radio

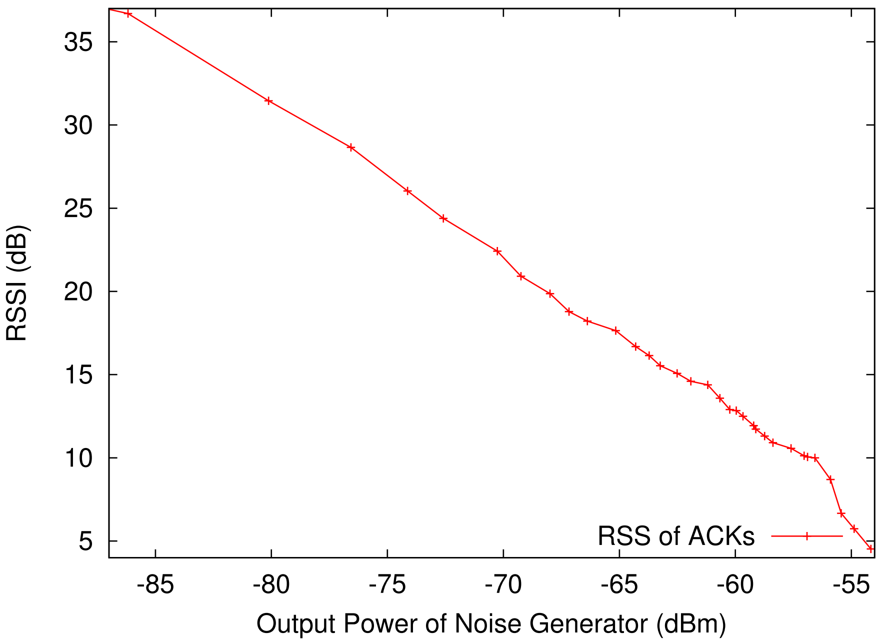


Figure 32: RSSI of received ACKs with changing output power of noise generator.

Peripheral (USRP) [36] device in between the two WiFi nodes acting as a noise generator. The USRP has a 2.4GHz radio transceiver and is programmed to generate Gaussian white noise over a 4 MHz bandwidth centered around the center frequency of the chosen 802.11 channel. Even though the channel width used by the USRP is small – only 4 MHz compared to 20 MHz for 802.11 – the noise floor at the 802.11 receiver can still be raised enough by increasing the transmit power of the noise generator such that the link quality is impacted adversely.

The output power from the noise generator can be increased by programmatically increasing the amplitude of the noise. This progressively worsens the link quality between the two WiFi nodes. Figure 32 shows the effect of increasing the output power from the noise generator on RSSI of the received ACKs at the transmitter validating that the generated noise has an intended effect. Notice that the RSSI decreases linearly with increasing output power from the noise generator.

5.6.2 Emulating Drives

We use the emulation setup described in the previous section to emulate a 50 km drive for each RAA. We assume that 50 APs are placed at regular intervals along the drive. Our goal is to vary the noise generated by the USRP such that the link quality between the two WiFi nodes will be similar to what it would be for an actual drive at various points of time.

Choosing an appropriate propagation model is important for the emulation set up. We note that since the vehicular antenna height is short, we need to use near-ground propagation models that take into account effects due to the lack of Fresnel zone clearance. We choose the model developed and validated for 802.11 signal propagation in the 2.4 GHz band as reported in [44]. The model is as follows:

$$P_{loss} = 40Log_{10}D + 20Log_{10}F - 20Log_{10}h_t h_r,$$

where: P_{loss} is the path loss in dB, D is the distance in kilometers, F is the frequency in MHz, and h_t and h_r are the heights of the transmitting and receiving antennas in meters.

This model estimates the path loss for 802.11 WLAN line-of-sight links with antenna between 1 and 2.5 meters in height. In our emulation we assume the APs to be at a height of 2.5 m and the vehicular client to be at a height of 1 m.

The output power from the noise generator is varied using a script in order to emulate the changing distance from the APs at different driving speeds. Three different driving speeds are emulated in this fashion – 5 m/s (or 18 km/hr), 10 m/s (or 36 km/hr) and 15 m/s (or 54 km/hr). The speed remains constant during the emulation. The 'ad hoc' mode is used specifically to ensure that there is no issue with AP-client association and there is no handoff. This ensures that there is no influence from any handoff protocol on the overall throughput performance. Note that use of 'infrastructure' mode instead of 'ad hoc' mode may reduce the repeatability of the emulation. The AP-client association messages are broadcast in nature and thus are highly susceptible to losses. Hence the handoff delays can vary widely during repeat experiments. The same saturated UDP traffic is used as before and the received throughput is plotted for each RAA for different speeds (see Figure 33). Note that this emulation perhaps over-estimates the absolute performance as the experiments are stationary and indoors, and no handoff delay is modeled. Its sole use is relative comparison between different RAAs. We see that *BRAVE* still performs the best. The rest of the RAAs perform much worse, specifically at higher vehicular speeds.

At slow speeds *AMRR* seems to be better than SampleRate because of its aggressive nature. See the relevant discussion in Section V. This over-aggressiveness hurts *AMRR* at higher speeds since now it over-selects bit rates. *SampleRate*'s nature of under-selecting bit rates helps it in competition with *AMRR* at higher speeds. *CHARM* over-selects bit rates very aggressively and mostly relies on the 54 Mbps bit rate even when the link quality deteriorates. With the rapidly changing link quality experienced under vehicular mobility, *CHARM* incurs heavy retransmissions and packet losses hurting its performance significantly.

5.7 Related Work

5.7.1 Frame Based Rate Adaptation

Frame based rate adaptation algorithms rely on link layer frame delivery statistics.

Auto Rate Fallback (ARF) [59] is a simple rate adaptation algorithm proposed in 1996 that drops the transmission rate on successive packet losses and increases the rate on successive successful packet transmissions. Since then many other rate adaptation algorithms have been proposed. Adaptive Auto Rate Fallback (AARF) [65] improves upon ARF by dynamically choosing the threshold for increasing and decreasing the bit-rate instead of assuming a fixed value. This adds a measure of stability to the protocol. ONOE [4] is another simple but popular algorithm whose implementation is available in the Madwifi driver code. It decreases the bit-rate when packets need at least

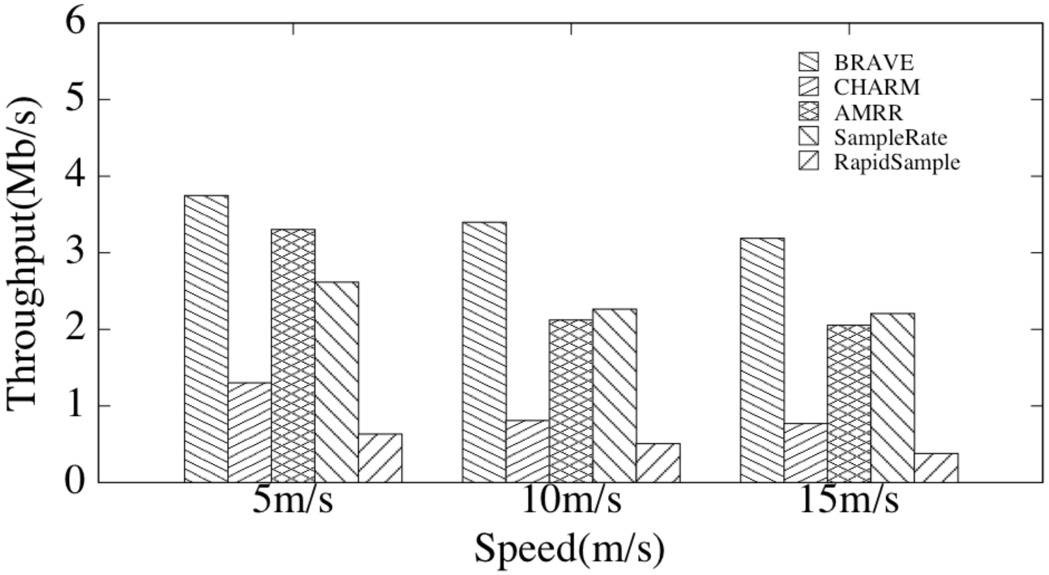


Figure 33: Throughput performance of rate adaptation algorithms in the 50 km emulated drive for three different speeds.

1 retry on average and increases it if less than 10% of the packets require retransmissions.

SampleRate [79] and Adaptive Multi-Rate Retry (AMRR) [65] are the other two popular algorithms whose implemenations are available in the Madwifi driver. We have previously described their operation in Sections 5.5.1 and 5.5.2.

All the above algorithms are designed for indoor WiFi access scenarios where channel conditions are relatively stable. In out-door vehicular WiFi access scenarios the packet loss rates and variance in the channel conditions are much higher and the frame based algorithms tend to either under-estimate or over-estimate the link quality depending on the sizes of their estimation windows and rate adaptation procedures.

5.7.2 SNR Based Rate Adaptation

In case of signal-to-noise ratio (SNR) based rate adaptation algorithms the bit-rate adaptation is dependent on PHY layer statistics like RSSI.

Receiver Based AutoRate (RBAR) [49] is an SNR based scheme that relies on the RTS/CTS mechanism and a pre-computed wireless channel model to choose an appropriate bit-rate. In this scheme, the transmitter sends an RTS frame before every data frame and the receiver measures the SNR and compares it with SNR thresholds from the channel model to select an appropriate bit-rate. This bit-rate is conveyed to the transmitter as part of the CTS frame. The obvious downside of this scheme is the overhead introduced by the RTS/CTS frames.

CHARM [58] is another SNR based algorithm which uses channel reciprocity to obtain channel information while avoiding the RTS/CTS overhead. We have already described its operation in Section 5.5.3.

5.7.3 Rate Adaptation in Mobile Environments

A recent paper [103] has proposed a hint-aware rate selection algorithm which explores the possibility of using mobility hints for bit-rate selection. They propose switching between the RapidSample algorithm (described in Section 5.5.4) and the SampleRate algorithm (described in Section 5.5.1) based on whether the client is mobile or not, respectively. Our analysis in Section 5.5.4 shows that in outdoor vehicular environments RapidSample under-performs in comparison to all other algorithms.

The CARS algorithm proposed in [99] also accepts sensor hints to make rate selection decisions. Unlike the hint-aware rate selection algorithm in [103], CARS has been designed for out-door vehicular WiFi access scenarios. In the CARS scheme, the communicating nodes exchange GPS locations in order to track distance and relative speeds between the two nodes. This forms the current "context". Each node maintains a context to link quality mapping and fast rate adaptation is performed based on the current context. However, this scheme has the drawback of requiring a lot of training data to be collected to account for every possible context.

Camp and Knightly [26] have proposed a cross-layer framework for rate adaptation in vehicular networks. The study concludes that both frame-based and SNR-based algorithms suffer from the smaller channel coherence time in out-door vehicular environments and training SNR-based protocols according to the environment's coherence time can improve throughput performance.

SoftRate [114] is a bit rate adaptation algorithm that uses physical layer hints to make rate adaptation decisions. SoftRate is known to out-perform most RAAs in slow and very fast fading scenarios, but tends to suffer under intermediate mobility where the channel changes every few packets.

The Strider [45] system is designed to achieve optimal rate adaptation by the use of a novel code which is rateless and collision-resilient. The current implementation of this system is on a `GNU Radio` platform and only uses 6.25 MHz width instead of the full 20 MHz width required for WiFi.

5.8 Conclusions

Optimizing WiFi access from moving vehicles is an important issue. Physical layer rate selection is a large component of this optimization. However, all rate adaptation algorithms for WiFi networks have been studied only indoors with fairly static scenarios or with very limited mobility (walking speeds). These algorithms often uses a significant amount of history (over several seconds) to make a rate selection decision, rendering their use fairly limited in very dynamic scenarios under vehicular mobility. We have addressed this problem by developing *BRAVE*, a rate adaptation algorithm that uses some amount of training to understand SNR versus error rate performance and then dynamically uses only a brief (sub-second) history to determine the rate to be used. *BRAVE* has been compared with four other prominent rate adaptation algorithms in a metro-scale WiFi deployment exposing the advantages of using *BRAVE* vis-a-vis the other protocols. A head-to-head comparison in an emulation environment finally establishes its superiority, specifically at higher vehicular speeds.

Chapter 6

Improved Vehicular WiFi Access using a Multi-radio multi-vehicle System

6.1 Introduction

The central problem in vehicular WiFi access is that of frequent handoffs. Since the cell sizes of WiFi access points (APs) are of the order of a few hundred meters, the vehicular client experiences handoffs every few seconds. Also, despite the fact that most outdoor spaces are over-provisioned, handoffs can be very time consuming. This creates an illusion of coverage holes for the vehicular client. We call these *perceived coverage holes.* Indeed, there are some outdoor spaces that have no or very poor coverage. We call these spaces *real coverage holes.*

In this chapter, we address the problem of overcoming coverage holes – both perceived and real by using a multi-radio multi-vehicle (MRMV) system. The MRMV system comprises of two or more MRMV clients (each in a separate vehicle) which can communicate with each other and function as relays for each other's traffic. Each vehicular client is further equipped with multiple radios. The idea is to exploit the spatial diversity of road-side APs by providing each client multiple paths to the network.

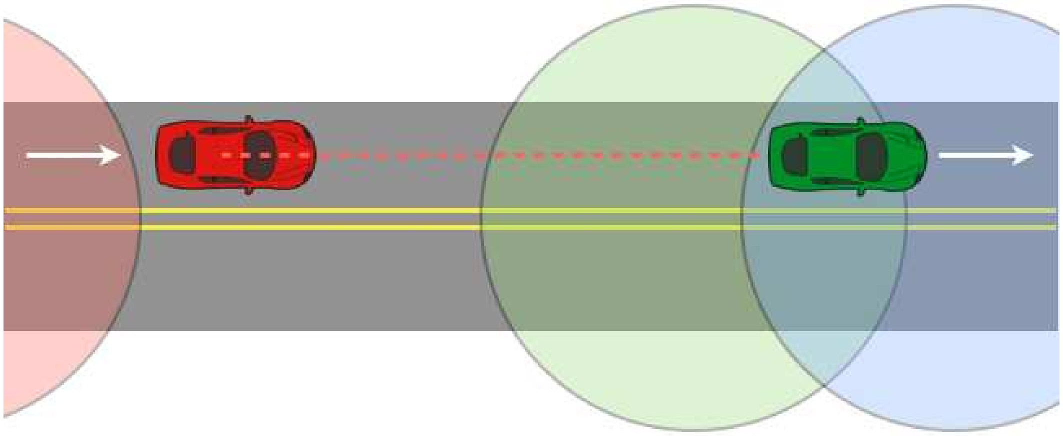


Figure 34: Vehicular WiFi access model to exploit spatial diversity of WiFi APs and vehicular clients.

A schematic of our MRMV system in operation is shown in Figure 34. In the schematic, the green vehicle is exiting the space covered by the green AP and entering the space covered by the blue one; and the red vehicle has exited the space covered by the red AP and is now in a *real coverage hole*. If a single-radio system was in place in the green vehicle, the client would only try to associate with the blue AP once it's connectivity with the green AP is completely lost. This is not a quick process. The client will experience loss of connectivity even when the space is well covered by the WiFi network. However, in a multi-radio system, the second radio can associate with the blue AP even before the client looses connectivity with the green AP. In this way, a multi-radio client can achieve nearly 100% connectivity and eliminate *perceived coverage holes.*

If the red vehicle, which has entered a *real coverage hole* can communicate with the green vehicle then the green vehicle can be used as a relay for the red vehicle's data. Such a relay system has the potential of providing connectivity to cars in *real coverage holes* hence reducing their impact. We show with experiments that a relay network of two MRMV clients can reduce *real* *coverage holes* to about half. In addition, by appropriately choosing an AP for data transfer the MRMV system increases overall throughput and also reduces throughput variability which is common in vehicular WiFi access.

We make the following contributions:

1. Provide an in-depth evaluation of the coverage characteristics of a commercially operated metro-scale WiFi network (Section 6.3). This is in contrast to vehicular connectivity characteristics provided for the same network in [30] and other studies related to APs in the wild [35, 24] or a limited WiFi deployment [41, 17, 31].

2. Develop a novel AP filtering technique which limits the number of APs that are visible to each radio of an MRMV client. We show that our AP filtering technique, along with the use of a multi-radio client system helps in overcoming *perceived coverage holes* (Section 6.4).

3. Show how a network of multi-radio vehicular clients that can relay each other's traffic can help in masking *real coverage holes* to a large extent (Section 6.5).

4. Design and implement a network with two MRMV clients, conduct real driving experiments and demonstrate the benefits of the MRMV system over the traditional single-radio-single-vehicle model (Section 6.6).

6.2 Related Work

6.2.1 Exploiting Diversity

Diversity has been a central theme in many of the prior works on mobile networks. Wireless network diversity can take the form of technology diversity (using multiple underlying technologies like 3G, CDMA, WiFi), network diversity (like AP diversity, provider diversity, channel diversity).

MAR [96] proposes a commuter access router infrastructure for mobile clients that tries to exploit technology and network diversity by using multiple different cellular technologies and network providers at the same time. The authors in [16] design a system, Wiffler, that augments 3G capacity by offloading data to WiFi whenever possible. In UCAN [72], a mobile client has both 3G cellular link and 802.11 peer-to-peer links. The 3G base station forwards packets for destination clients with poor channel quality to proxy clients with better channel quality. A similar work in iCar [118] proposes an integrated cellular and ad hoc relaying system that addresses the congestion problem due to unbalanced traffic in a cellular system.

ViFi [17] exploits macro diversity i.e. using simultaneously multiple APs and opportunistic receptions by near-by APs to reduce disruptions for mobile clients. R2D2 [94] uses multi-lobe beam pattern switching on a smart antenna to use multiple access points while using directional antennas. Both [17, 94] require coordination among the APs. In Spider [108] a node associates with multiple APs to avoid association and dhcp delays. They perform scheduling to decide when to switch to an AP and how much time to spend on each AP. But the scheme suffers because of inherent channel switching delays in commodity hardwares, making them conclude that the scheme truly benefits only if nodes connect to APs on the same channel. The work in [21] uses two wireless cards where one card is dedicatedly used to scan and pre-associate to an alternate AP and avoid handoff delays but the context is not vehicular. The works [27, 42, 60, 85] use multiple access points to aggregate bandwidth but the solutions are designed for static networks.

6.2.2 Vehicular WiFi Handoff

Several techniques have been proposed to improve the handoff mechanism of a mobile client when switching associations in a wireless network. The solutions address two issues related to regular handoff mechanism: reducing the handoff latency and improving handoff decisions. Handoff latency is improved by reducing the number of channels to probe [101], reducing the number of control messages exchanged [35, 8] and using a second radio to pre-associate to an alternate AP [21]. Handoff decisions are improved either by using historical information [82, 31] or by scanning actively [41, 78, 95].

However only [82, 31, 41] propose handoff mechanisms that are designed for vehicular environment. The authors in [31] propose a scripted handoff using historical information of signal strengths to decide which AP to associate to and also eliminate scanning or probing delays. Mobisteer [82] uses steerable beam directional antenna and past historical information to improve AP selection process. The protocol in [41] actively scans even when a client is associated to select the best AP.

6.2.3 Relaying Via Mobile Ad-Hoc Networks

The idea of relaying via a mobile ad-hoc network has been studied previously [72, 125]. The work in UCAN focuses on finding a routing path from a mobile device to other mobile devices with better cellular network coverage. However this work is not in a vehicular context, where vehicle mobility can cause frequent interruptions of the relay link. The work in [125] proposes a vehicle-vehicle relay scheme to improve coverage but only performs simulations and some controlled experiments using a single AP.

6.2.4 Vehicle-to-Vehicle Communication

There have been some experimental studies in the context of V2V communication. In [102], the authors measure the performance of 802.11b-based V2V communication in different environments. They show that the performance of V2V links are greatly affected by the nature of the environment. Authors in [80] report experiences with static 1 and 3-hop scenarios and a mobile 3-hop scenario. TCP and UDP performance results are presented in a 2-hop vehicular network in [51]. The work in [112] uses directional antennas to improve the transmission range in the context of V2V communication.

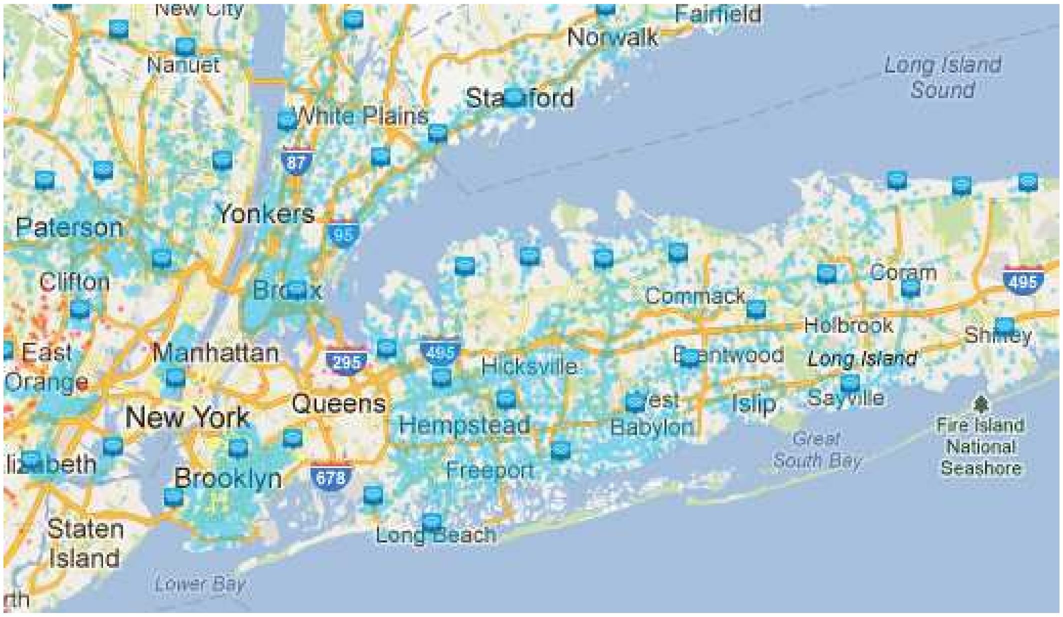
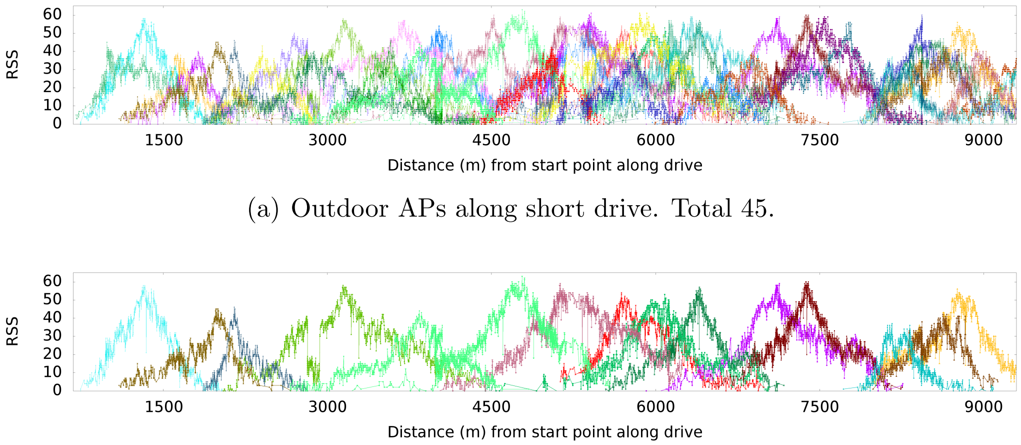


Figure 35: Optimumwifi deployment in NYC, Long Island and parts of Connecticut. Approximate WiFi coverages are shown (from [5]).

6.3 Quality of Coverage

6.3.1 Network

For our study we use a metro-scale WiFi deployment in the Long Island area in New York (population roughly 7 million). This service is called 'Optimumwifi' [5] and is provided by Cablevision, a local cable TV provider and ISP. The WiFi network extends to most of populous areas of Long Island where our study is conducted. It also extends to parts of New York City, Pennsylvania, Connecticut and New Jersey, where Cablevision has service. This can be seen in Figure 35. The deployment consists of both indoor and outdoor WiFi hotspots. Optimumwifi primarily targets urban commercial centers. Regions closer to NYC are more densely covered than those farther away. In the Long Island area, the outdoor coverage along signaled highways is excellent. Villages are less densely covered and expressways are mostly not covered. WiFi access is provided free to all subscribers of Cablevision's TV or Internet services. We note that there are several hundreds of such metro-scale WiFi deployments in USA alone [81] While we expect that our general observations will be repeatable in another metro area deployment, we do note that more specific quantitative observations could be strongly tied to deployment density, radio characteristics of the APs and any handoff control on the APs.



(a) Outdoor APs along short drive. Total 45.

(b) 15 APs retained after filtering.

Figure 36: Result of wardriving exercise on short drive (6 mile road stretch).

6.3.2 Wardriving Setup

Wardriving is the act of searching and locating WiFi APs. Typically this is done by putting the radio interface of a computing device in monitor mode and passively sniffing and geo-tagging the recorded beacons. We used a `Dell OPTIPLEX 780` computer to perform the wardriving. Three Ubiquiti XR2 cards were attached to the computer using a miniPCI-to-PCI converter. The three interfaces were configured to operate in monitor mode and were statically assigned the orthogonal channels 1, 6 and 11. Our experience shows that all APs of the Optimumwifi network operate in only these three channels and the above static assignment is sufficient. `tcpdump` was used to log the captured beacons. A `Garmin 18x USB` GPS device [39] was used to log the GPS coordinates. The pcap files generated by `tcpdump` were then merged with the GPS log as a post-processing step.

For our experiments we consider two driving scenarios; a short drive of 12 miles and a long drive of nearly 100 miles. The short drive is a round trip on a 6 mile stretch of road.

6.3.3 Coverage Characteristics of Short Drive

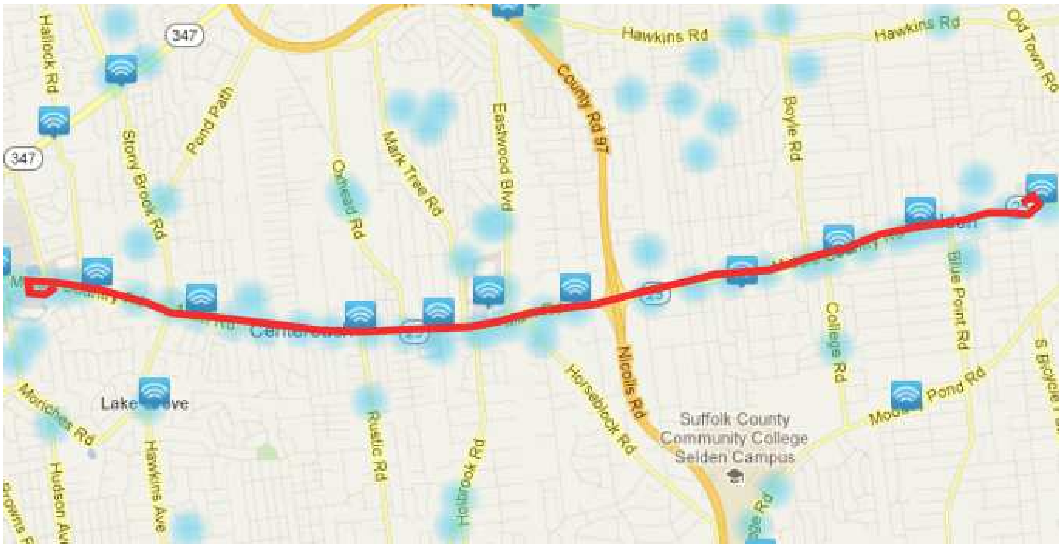


Figure 37: Map of the road stretch used for short driving experiments (part of Route 25), along with the route shown in red. Approximate WiFi coverages are shown (from [5]).

The 6 mile stretch of road that we use to perform short driving experiments on is in a very busy urban area and has large malls and other commercial enterprises on both sides of it. Figure 37 shows the road stretch the driven route shown in red. Optimumwifi provides both indoor and outdoor APs along this road and our wardriving uncovered nearly 130 of them. 45 of these APs were visible across distances of 500 m or more. We classify these 45 APs as being outdoor APs and the rest as indoor APs. Figure 36(a) shows the RSS in dB of the received beacons from these 45 outdoor APs along the 6 mile drive. Each color represents a different AP. By performing a simple form of localization i.e. assuming the location of an AP to be the mid-point over the range that it is observed, we derive the average inter-AP distance to be only 212 m. Also, as seen from Figure 36(a), at every location along the drive, the RSS of at least one AP is at least 20 dB above the noise floor. Also, at every location along the drive, at least three APs are visible to the vehicular client. The number of APs using channels 1, 6 and 11 are 18, 12 and 15 respectively.

Figure 36(b) shows that even if we artificially remove 30 outdoor APs the entire stretch of road can be covered. Even in this case, the observed RSS at each location from at least one AP is almost always above 20 dB above noise. There are no *real coverage holes* along this drive however a single radio vehicular client will experience several *perceived coverage holes*.

6.3.4 Coverage Characteristics of Long drive

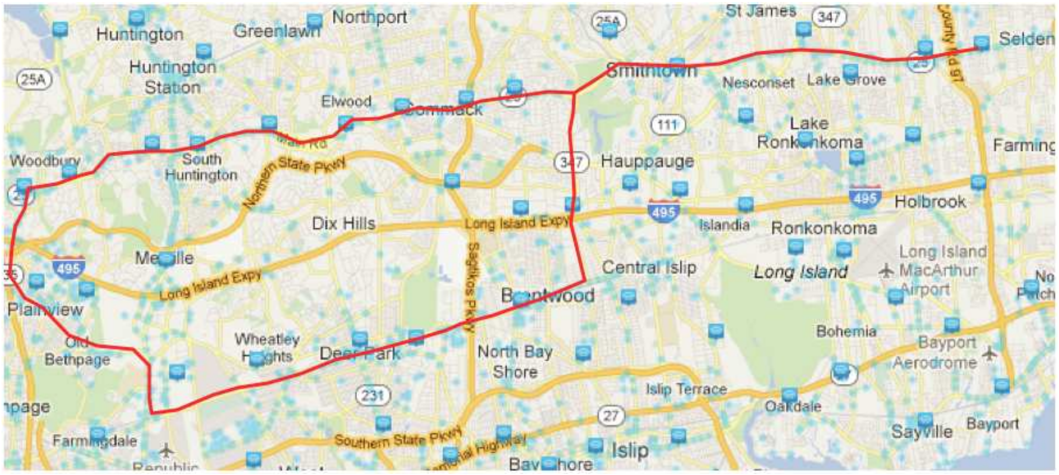


Figure 38: Map of the middle of Long Island – the area used in the long drive experiments. The route is shown in red. Approximate WiFi coverages are shown (from [5]).

The long drive shown in Figure 38 is nearly 100 miles long. This drive

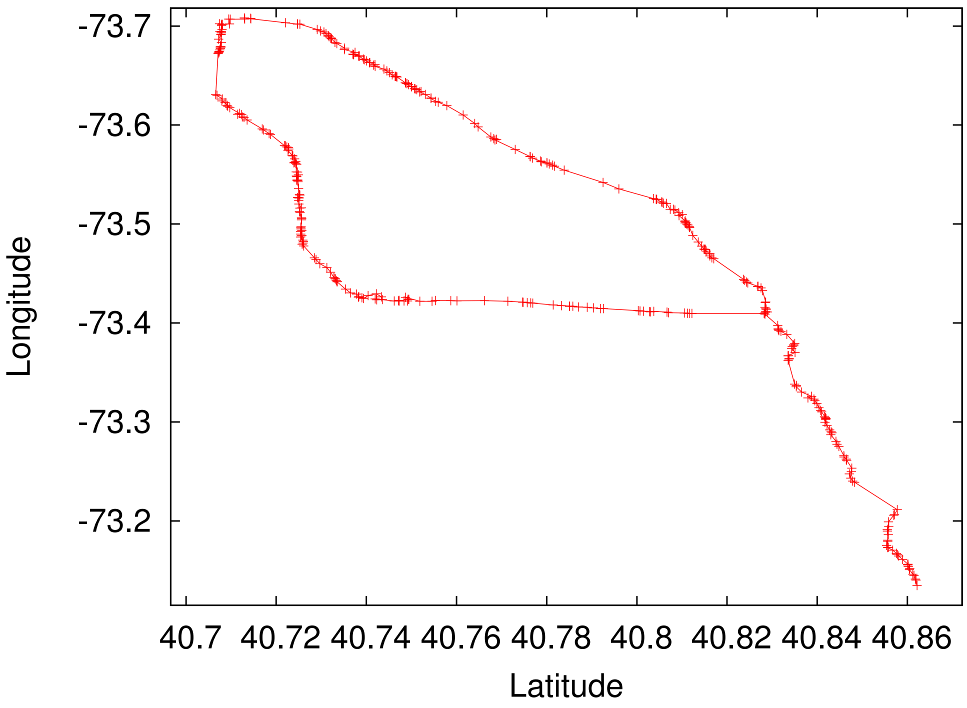


Figure 39: Locations of outdoor APs along the 100 mile long drive.

also runs through urban parts of central Long Island and there are several malls and commercial enterprises on both sides for most of the drive. Our wardriving effort revealed over 1050 unique APs along this drive. We classify 352 of these as being outdoor APs using the same heuristic as above. Figure 39 shows GPS locations of the outdoor APs in decimal latitudes and longitudes. Again, the AP location is assumed to be the mid-point of the range over which the AP is observed.

As can be seen from Figure 39, the APs are located in clusters and there are fair number of *real coverage holes* along this drive. We use a simple heuristic to cluster the APs along the drive. If a pair of successively encountered APs are 1000 m apart then they belong to different clusters. Using this technique we obtain 9 clusters. The average number of APs in each cluster varies

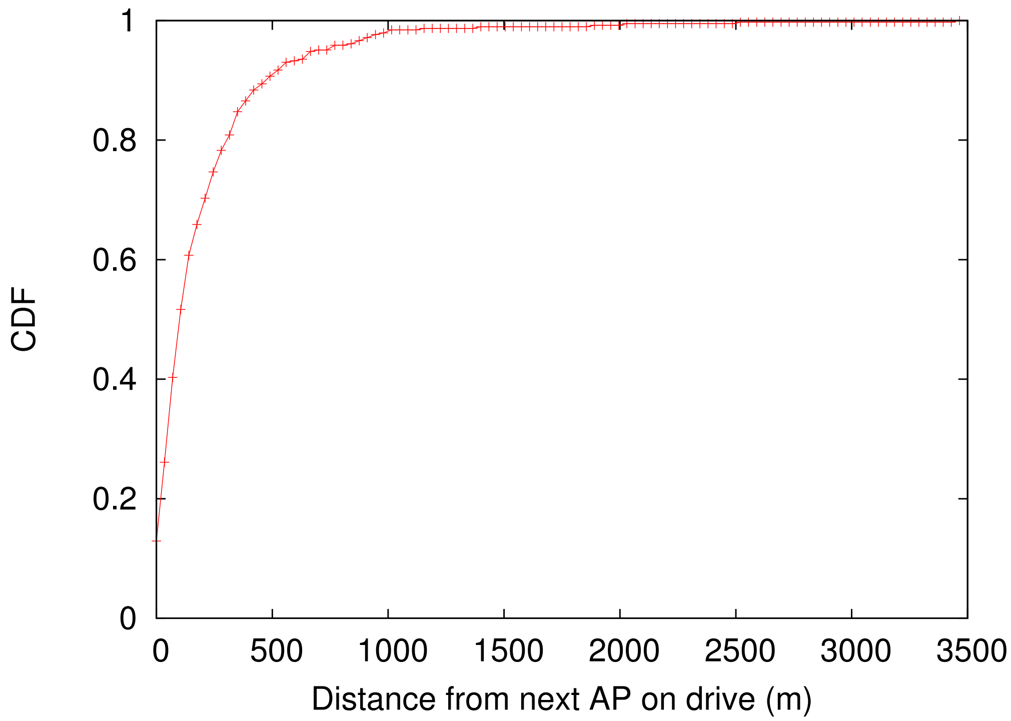


Figure 40: CDF of inter-AP distance along the 100 mile long drive

between 16 and 110. The average number of APs per cluster is 39. The inter-cluster distance varries between approximately 1000 m and 3500 m and the average distance is 1740 m.

Figure 40 shows the CDF of the inter-AP distances. Notice that about 90% of the times the inter-AP distance is less than 500 m and only about 2% of the times the inter-AP distance is larger than 1000 m.

6.4 Eliminating Perceived Coverage Holes

In this section, we describe how a multi-radio vehicular client can maximally exploit AP diversity to improve its own perceived coverage and also potentially improve its throughput.

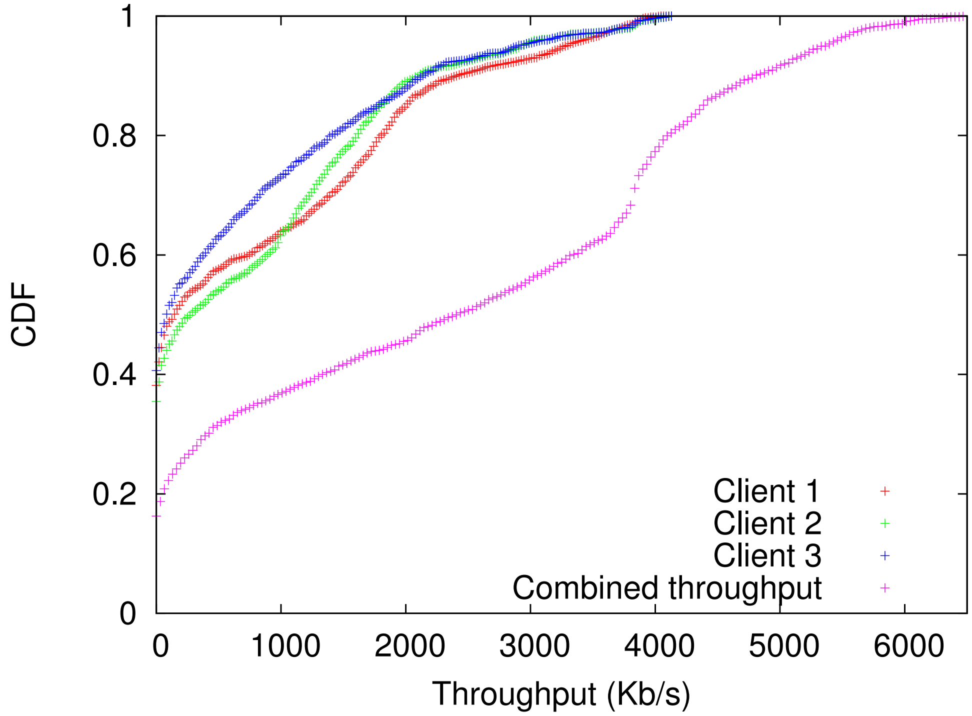


Figure 41: CDF of individual and combined throughputs when each client uses the default handoff strategy. Bit-rate is fixed at 11 Mbps.

6.4.1 Default Handoff in Madwifi

When a client using `Madwifi version 9.0.4` is in roaming mode it tries to handoff from one AP to another when its link quality with the current AP drops below 10 dB above the noise floor. At this time the client will associate with the AP with the strongest signal from among those in its cache which is populated by background scans. Under vehicular mobility, the client will always handoff to the strongest AP. This results in an excessive number of handoffs since the client does not associate with an AP in its entry phase but rather in the production phase [48].

6.4.2 Exploiting AP diversity with Multiple Radios

We first study whether three independent single board computers (SBCs) functioning independently can exploit AP diversity and reduce *perceived coverage holes*. The system is set up so that each board uses the default Madwifi driver and performs handoffs when it deems fit. Each client is configured to statically use only 11 Mb/s bit-rate. The client runs an iperf client and sends upload traffic to an iperf server which is run on a lab machine with a public IP. We conduct this driving experiment on the short drive seen in Figure 37 where there are no *real coverage holes* (see Figure 36(a)). We plot the CDF of the throughput seen on the server in Figure 41. As we can see, each client is unable to communicate with the server about 40% of the time. The median overall throughput received at the sever is however close to 2100 Kb/s and the overall loss of connectivity as seen by the server is about 20%. By simply using a multiple radio system we can reduce the *perceived coverage holes* by half.

6.4.3 Improving Coverage using AP Filtering

In this section we describe how AP filtering techniques can improve the vehicular client's perceived coverage. To begin with we note that it is best for the client to only associate with outdoor APs. These provide a longer duration of connectivity and hence fewer handoffs. Also, it is good if the three Avilla boards associate with mutually exclusive set of APs. In this way we can better exploit the spatial diversity of the APs. With these goals in mind we modify the Madwifi driver so that each board can only process the beacons from a small set of APs. The rest of the beacons are simply dropped. To this end, the outdoor APs encountered along the drive are assumed to be located at the mid-point of their range. In this way we form a serial order of APs to be encountered along the drive. The APs to be considered are then assigned to each SBC in round-robin manner following the serial order.

Upon assigning the 45 outdoor APs to the three clients in the above manner (we call this 15-15-15 filtering), the overall coverage improves slightly. This can be seen in Figure 42 which shows the CDF of the instantaneous

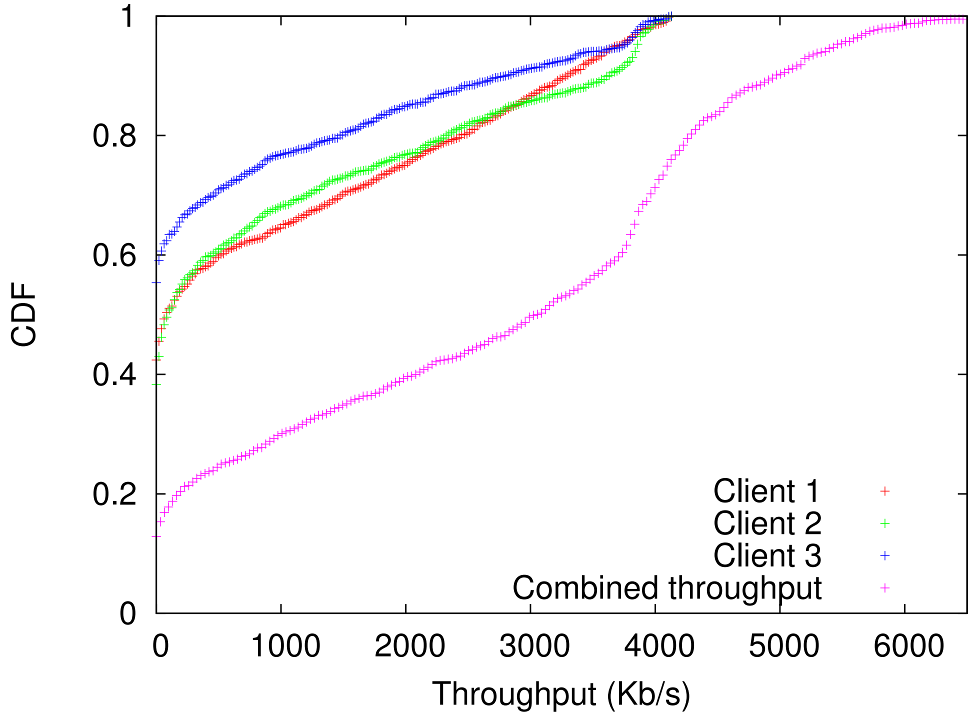


Figure 42: CDF of individual and combined throughputs with 15-15-15 filtering. Bit-rate is fixed at 11 Mbps.

throughputs seen at the server. Notice that in terms of coverage, each client performs slightly worse than when using the default Madfiwi driver but the overall coverage is slightly improved. The median combined throughput improves significantly from 2100 Kb/s to around 3000 Kb/s.

We also studied the impact on coverage and throughputs by filtering out even more outdoor APs. For each stage of filtering we drop every M/Nth AP to be encountered in the serial order; where M is the total number of outdoor APs and N is the number of APs to be filtered out. The remaining APs are distributed among the three boards in round-robin manner as above. As seen in Figure 43 increased filtering improves median combined throughputs and overall coverage up-to the 5-5-5 level of filtering. For 5-5-5 filtering, the median combined throughput at the server is approximately 3800 Kb/s and

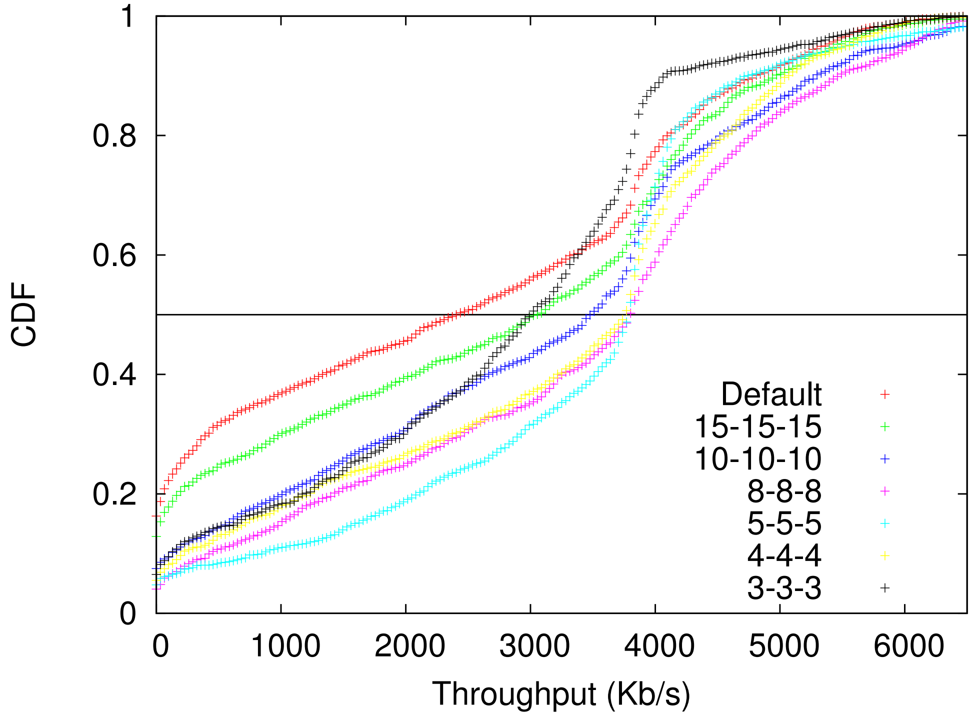


Figure 43: CDF of combined throughput with various filtering levels. Bit-rate is fixed at 11 Mbps.

the overall coverage is improved to 96%. Beyond this the median combined throughput starts dropping and the coverage reduces. Figure 44 shows the result for the same experiment albeit when the SampleRate [79] bit-rate adaptation algorithm is used. Notice that using SampleRate improves the median combined throughput for the best case (5-5-5 filtering) to about 4500 Kbps and the coverage is further improved to about 98% because of the use of lower bit-rates (5.5 Mb/s and 2 Mb/s) which are more resilient to packet loss than 11 Mb/s. Also, notice that with SampleRate and best filtering, the client experiences throughputs greater than 2000 Kb/s more than 90% of times.

It is interesting to observe that this increase in median throughput and coverage happens despite the fact that the overall link qualities become progressively worse with increased AP filtering. This can be seen in Figure 45

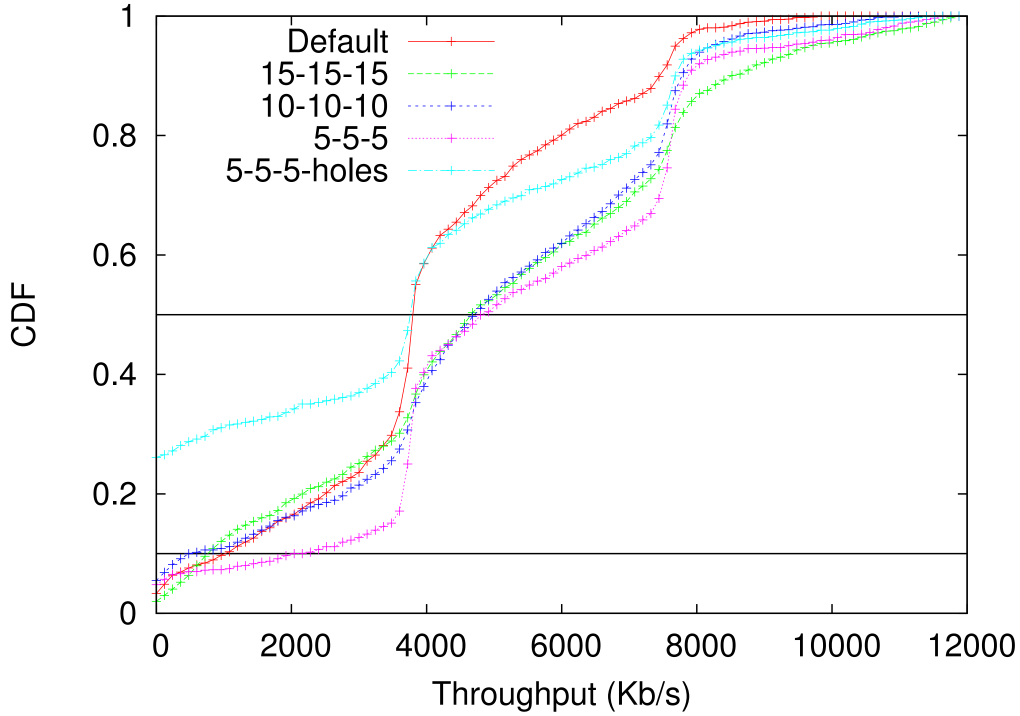


Figure 44: CDF of combined throughput with various filtering levels. SampleRate rate adaptation mechanism is used.

which shows the CDF of RSS values of acknowledgements received at the boards. Notice that for 5-5-5 filtering (best case) the median RSS is about half that of the default handoff case.

Table 7 captures some key statistics when using different levels of filtering. As can be seen from the table, the average time spent while being associated with a single AP increases with increasing number of APs being left out by filtering. This translates to an increase in distance in meters over which a single association lasts. Notice that the maximum range of association is around 800 m and 5-5-5 filtering achieves this range. Further filtering only minimally impacts the association range but has more impact on the distance travelled between associations. Also, notice that staying associated with a single AP for longer time entails associating with it much earlier than when

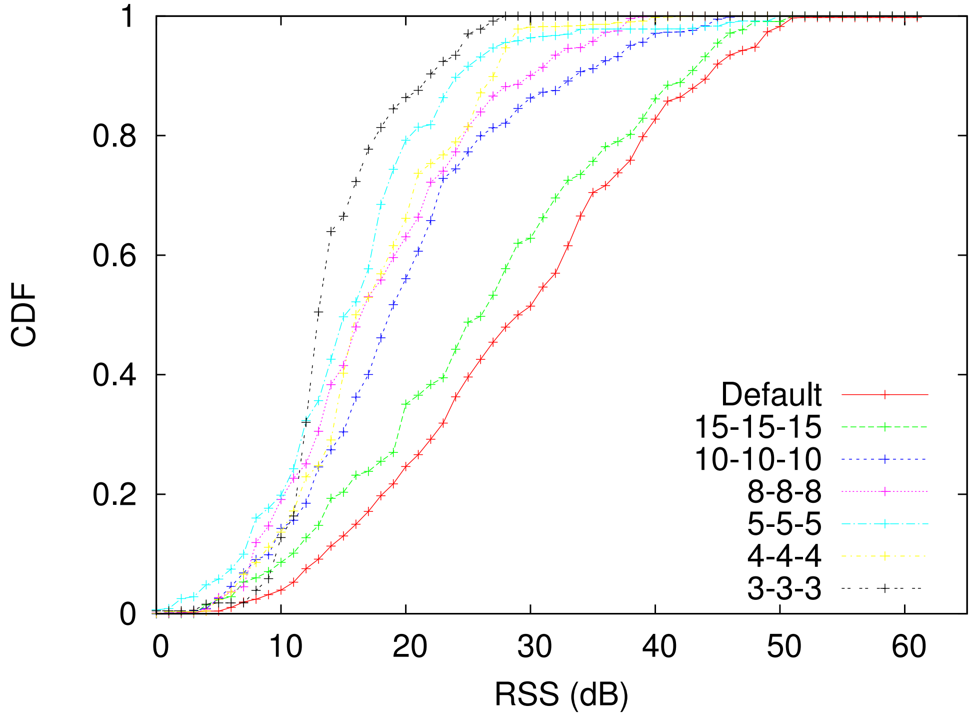


Figure 45: CDF of RSS of received ACKs at different filtering levels.

	Default	15,15,15	10,10,10	8,8,8	5,5,5	4,4,4	3,3,3
Avg. time per association (s)	61	59	62	81	93	110	112
Avg. range of association (m)	466	497	524	736	802	806	814
Avg. time between associations (s)	17	17	25	27	56	76	87
Avg. dist. between associations (m)	163	158	225	285	537	731	742
Avg. RSS (dB) when associating	31	27	19	20	12	11	10
Overall coverage	84%	88%	93%	97%	96%	94%	92%

Table 7: Key observations with various levels of filtering.

the client is associated with it for shorter times. This can be seen from the last row of Table 7. The RSS in dB when associating with a new AP progressively decreases with increased filtering. In the default case (unmodified Madwifi), a new association happens when the link quality is quite high (RSS over 30 dB). In contrast, this number drops down to nearly 10 dB with progressive filtering. We conclude that the default Madwifi driver misses out on a significant part of the production phase (RSS greater than 15 dB [48]) which can be better exploited if the vehicular client associates with an AP earlier like in the case of 5-5-5 filtering. As a result of better exploitation of the production phase, the resulting median throughput is higher for 5-5-5 filtering. Even though for 4-4-4 filtering and 3-3-3 filtering, the production phase of an AP is fully exploited the average distance travelled between associations is large enough to adversely impact the overall coverage and throughput.

6.5 Reducing Real Coverage Holes

As we have seen in Section 6.3.4, *real coverage holes* do exist. In order to overcome these, we propose the use of a V2V relay system so that vehicles that are outside the *real coverage holes* can forward the traffic of other vehicles that are inside. For the V2V link we prefer to operate in the 900 MHz frequency. Using 900 MHz allows longer V2V links than when operating in 2.4 GHz. Also, the link remains interference free from V2I communication in the 2.4 GHz band. To determine the feasibility of using a 900 MHz V2V link, we conducted a driving experiment on the long drive. Over nearly 100 miles, the receiver car followed the transmitter car. The transmitter car ran an `iperf` client and sent 1400 byte UDP packet to the receiver at 12 Mb/s. The receiver ran an `iperf` server. As can be seen from Figure 46, the average throughput drops gracefully with respect to distance up a distance of 1750 m. After this, we barely see any connectivity.

We show the CDF of the distance between the two cars in Figure 47. About 90% of times the two cars were within 1750 m of each other. Correlating with Figure 46, about 90% of times the vehicles could communicate at 3000 Kb/s or higher.

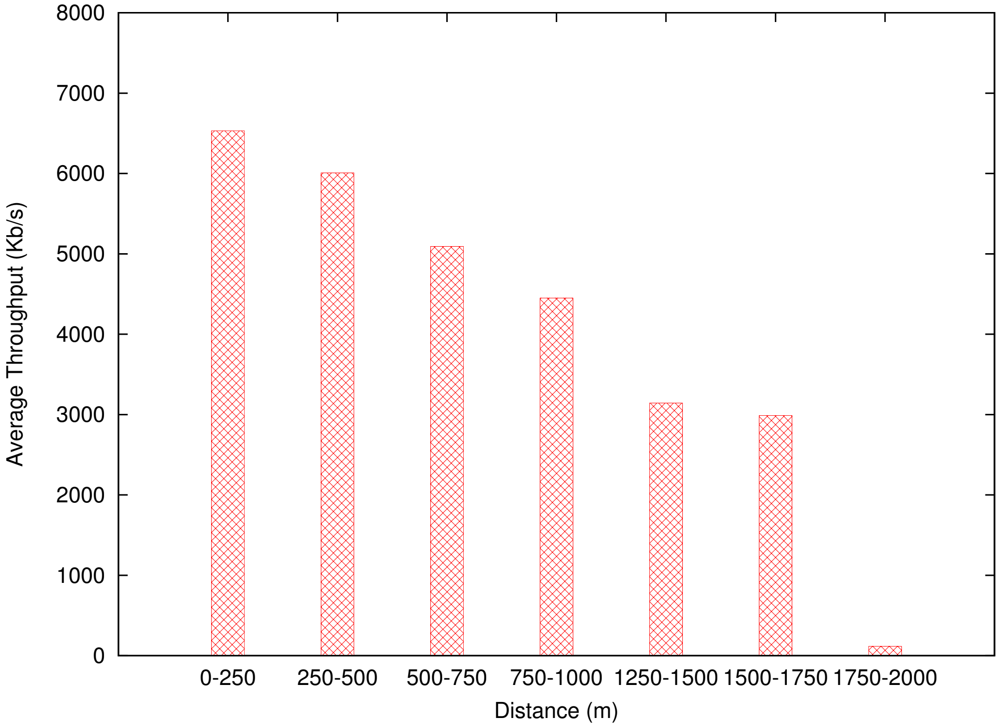


Figure 46: Average throughputs at various distances for the vehicle-to-vehicle 900 MHz link.

6.6 The MRMV System

6.6.1 Implementation

Figure 48 shows the various components of the MRMV client. Each MRMV client comprises of four Avila GW2348-2 boards [14]. Each board is fitted with a single carrier-grade radio interface. Three boards are fitted with Ubiquiti XR2 [6] interfaces and one is fitted with Ubiquiti XR9 [6] interface. The trasnmit powers of all interfaces is set to 25 dBm. The interfaces use Atheros chipset supported by the Ubiquiti driver that we used. The radio interfaces are connected to our vehicular antenna setup shown mounted on a car in Figure 49 using RF cables. The antenna setup comprises of three 2.4 GHz antennas and one 900 MHz antenna mounted on top of the car using a mounting rack. The

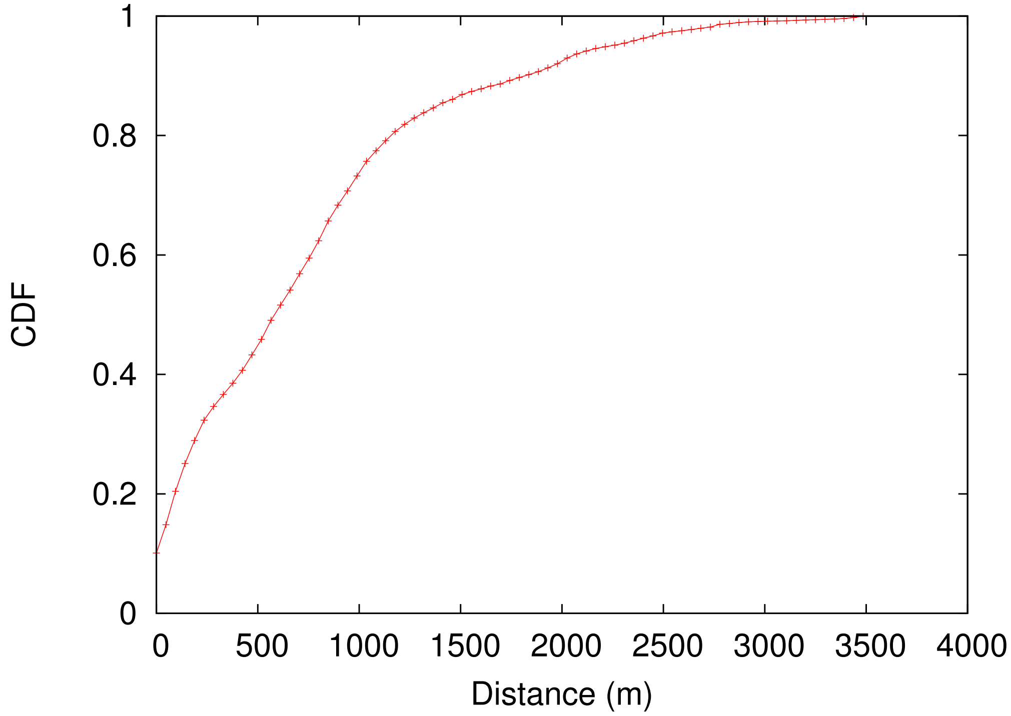


Figure 47: CDF of distance between 2 cars during long drive.

2.4 GHz antennas are used for vehicle-to-infrastructure (V2I) communication and the 900 MHz antenna is used for vehicle-to-vehicle (V2V) communication. The antenna gains for the 2.4 GHz and the 900 MHz antennas are 8 dB and 6 dB respectively. Each vehicle also has a laptop which is connected to the four board using a switch. The laptop is the actual data transmitting source. It runs `iperf` in client mode and sends saturated upload UDP traffic to a `iperf` program running in server mode on a lab machine with a public IP address.

The laptop decides which of the three SBCs it should route the `iperf` traffic to. It does this by monitoring the connectivity status and the link qualities experienced by the three SBCs and changing its own routing table to route data through the SBC that is connected and has best link quality. The connectivity status is determined by running a per second ping from each of the SBCs to the DNS server. The ping serves as a heartbeat message. If none

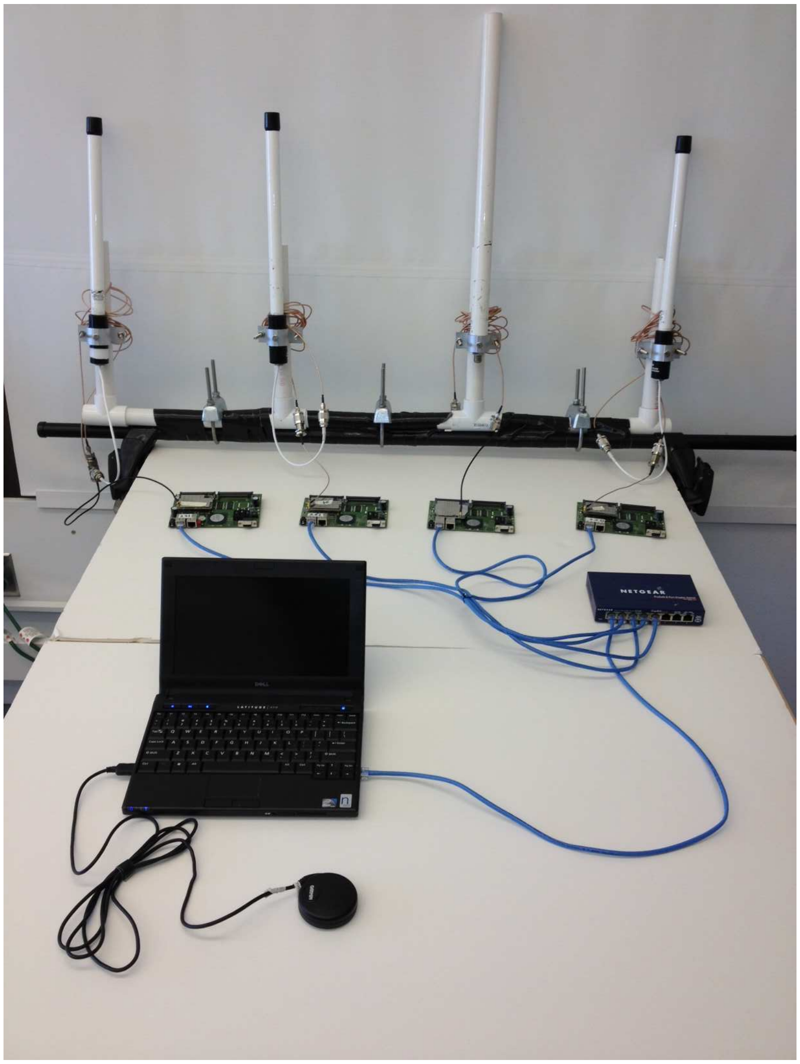


Figure 48: Components of an MRMV client.

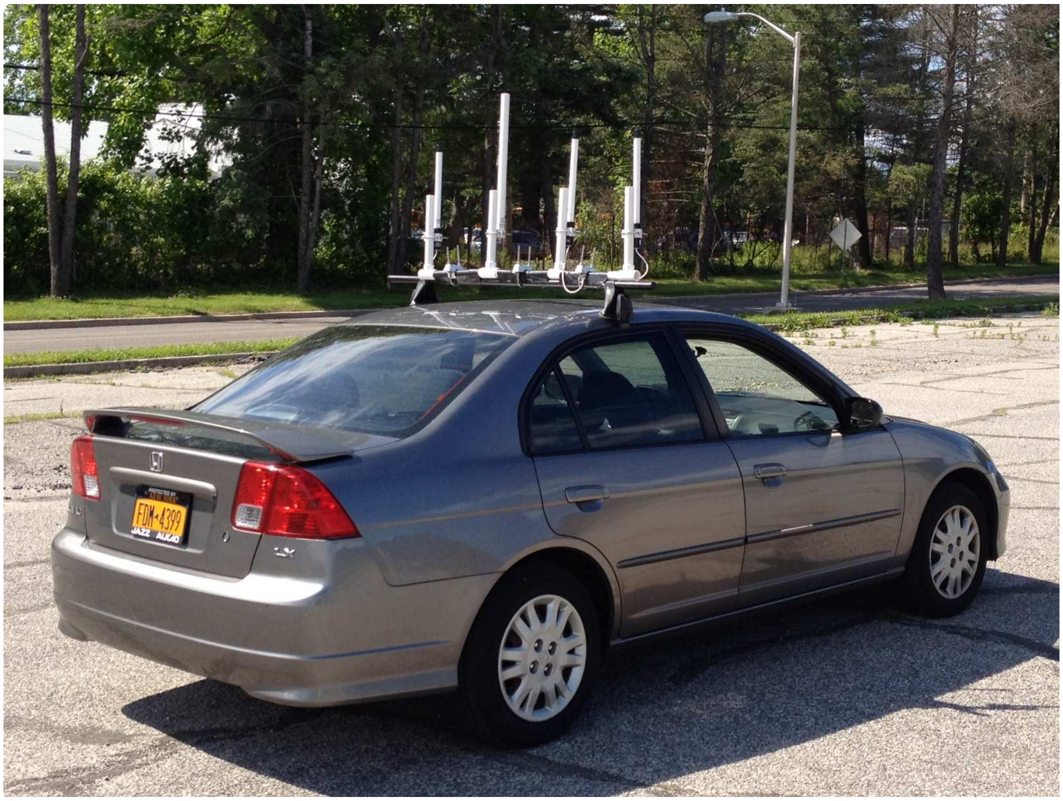


Figure 49: Picture of vehicular antenna setup used for the driving experiments.

of the SBCs have connectivity the 900 MHz relay link can be used to forward the data to a neighboring MRMV vehicular client which can forward the data on the primary vehicles behalf.

6.6.2 Results

6.6.2.1 Single MRMV client

To test the performance of a single MRMV client, we performed driving experiments on both the short and the long drives. As described above the MRMV client runs an `iperf` client and sends saturated traffic to a lab server with a public IP address. The MRMV client chooses the best SBC to route its traffic to as discussed above. Figure 50 shows the CDF of the instantaneous

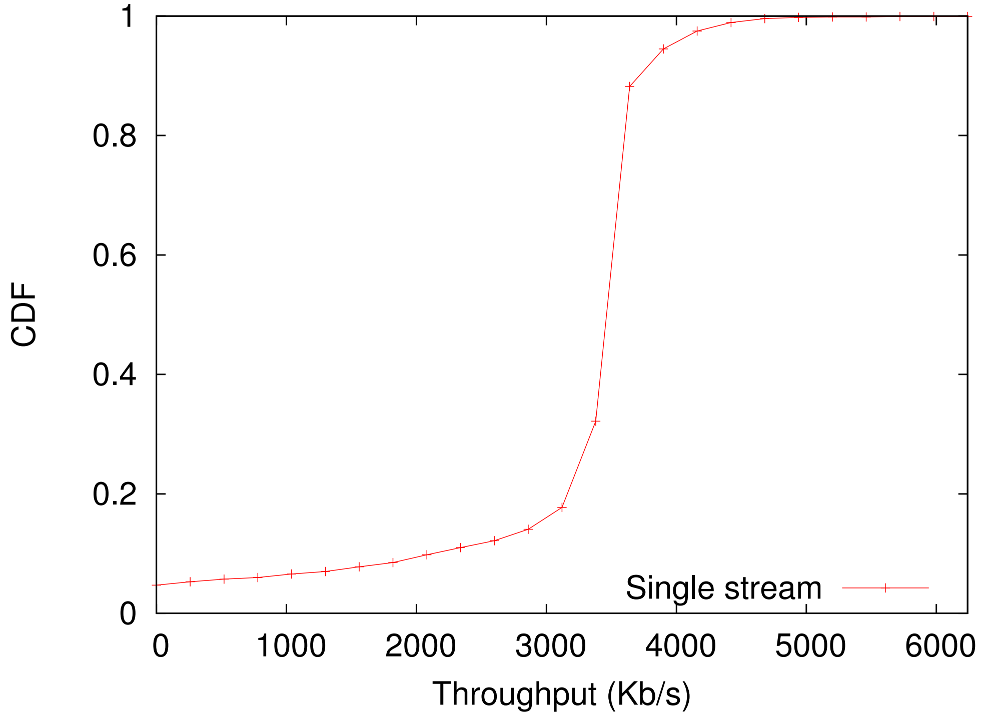


Figure 50: CDF of single stream with 5-5-5 filtering.

throughputs as seen at the server for both the short and the long drive. Notice that because of better coverage and the absence of any real coverage holes, the performance seen for the short drive is much higher than that for the long drive. The perceived connectivity for the short drive is 95% and the median throughput is around 3500 Kb/s. Also, 80% of the times the throughput is between 3300 Kb/s and and 4200 Kb/s. This shows a promise of reduced variance in throughputs as compared to the single radio case. Correspondingly, a single MRMV client provides 70% coverage on the long drive with a median throughput of around 2000 Kb/s.

6.6.2.2 Network of 2 MRMV clients

We also experimentally test the performance of a network of two MRMV clients. Each vehicle is setup to function as an MRMV client as above. In

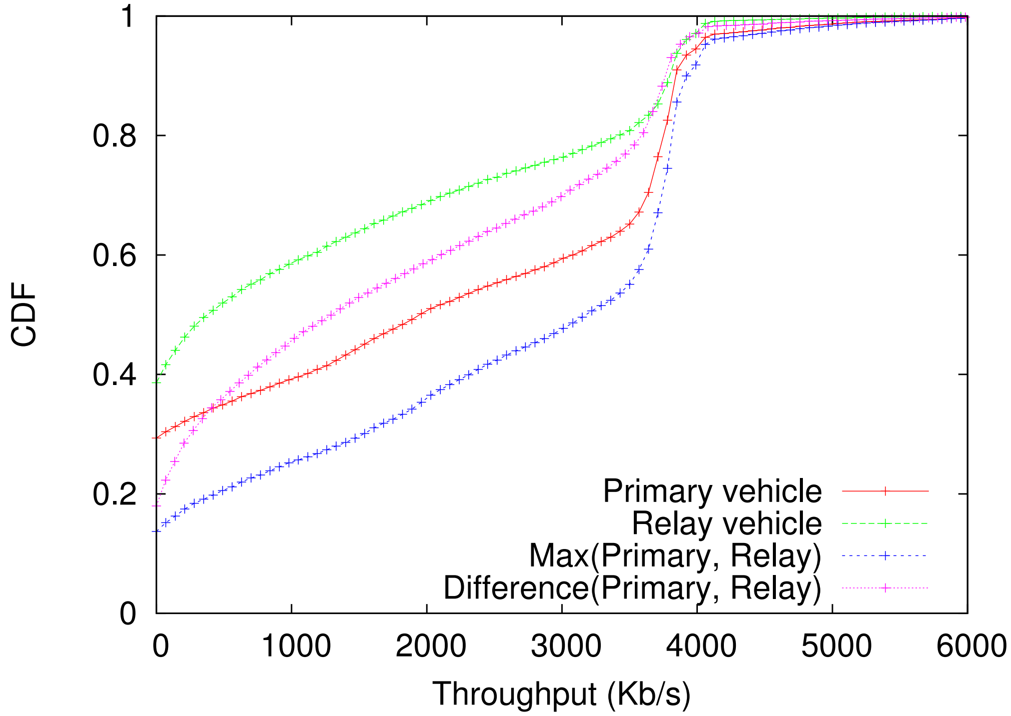


Figure 51: CDF of instantaneous throughputs for long drive.

addition there is also a 900 MHz V2V link between them. The MRMV client in the primary vehicle runs an `iperf` client which continuously sends a saturated traffic of 1400 byte UDP packets to the `iperf` server running on the MRMV client in the relay vehicle. The relay vehicle only forwards the traffic that it receives on the 900 MHz link and does not generate any traffic of its own. The two vehicles are driven along the long route with the primary vehicle in the lead and the relay vehicle following behind.

When the primary vehicle was in a coverage hole, i.e., when it did not have any V2I connectivity, about 85% of the times it could communicate with the relay vehicle over the 900 MHz link with an average throughput of nearly 4000 Kb/s. Overall, when the primary vehicle was in a coverage hole, 48% of times data originating from the primary vehicle could reach the lab server via the relay vehicle.

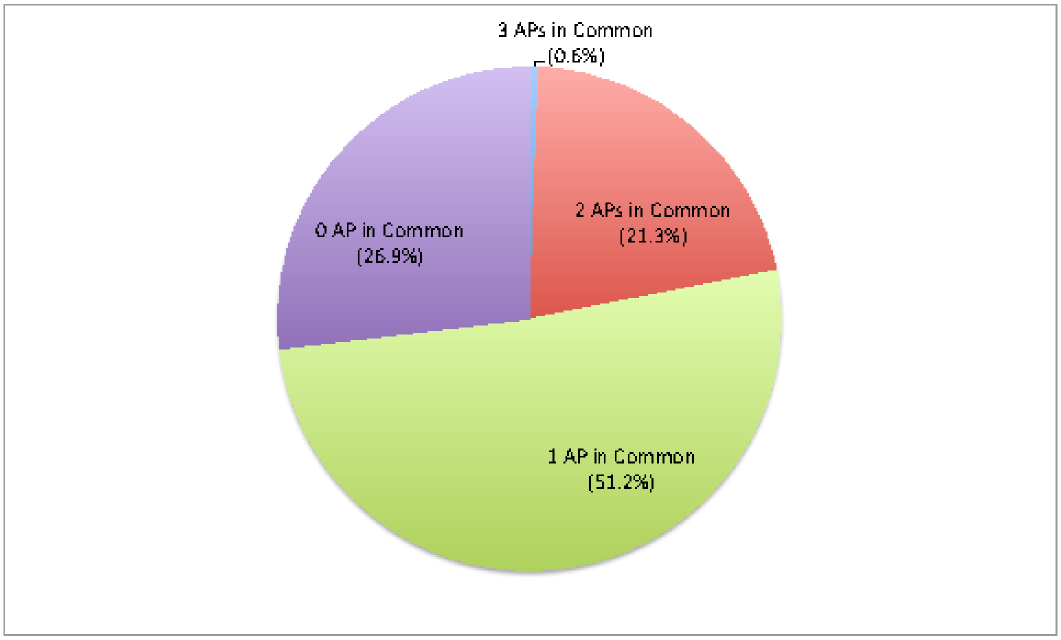


Figure 52: AP diversity experienced by the two vehicles on long drive.

Figure 52, shows the AP diversity experienced by the two vehicles. About one-fourth of the times the two vehicles have no APs in common. Around half the time, the two vehicles have only one AP in common and about one-fifth of the times the two vehicles share two APs. This points at the potential of exploiting AP diversity to improve throughput performance beyond simply using the relay vehicle for improving coverage.

CDF of instantaneous throughputs as seen at the server are shown in Figure 51. The primary vehicle experienced an overall coverage of 70% with median throughputs around 2000 Kb/s. 48% of times, the V2V2I throughput was higher than the V2I throughput. Assuming perfect scheduling of the data stream, the overall coverage improves to about 84% and the median throughput increases to about 3100 Kb/s. The difference in the throughputs experienced by the two vehicles is also substantial. The median difference in throughput is around 1500 Kb/s.

6.7 Conclusions

In this chapter, we have experimentally showed the benefits a multi-radio multi-vehicle system can have in terms of improving vehicular WiFi performance. Using the MRMV system, and a novel AP filtering method we were able to achieve almost 100% coverage when the underlying AP deployment did not have any coverage holes. We showed that even though metro-scale WiFi deployments are very dense, real coverage holes do exist. We showed that using a network of two MRMV equipped vehicles these coverage holes could be reduced to about half. In addition to improving vehicular WiFi coverage, the MRMV system also increases the overall throughput of the system and reduces throughput variance.

Chapter 7

Conclusions

In this dissertation, we have proposed several methods to improve vehicular network access. We have relied on experimental analysis rather than using analytical or theoretical methods, which makes our results realistic. Specifically, we have made the following contributions –

- First, we have shown by experimental measurements that a metro-scale WiFi and 3G networks exhibit very different throughput and coverage characteristics. These characteristics coupled with different cost models for network access indicates that a hybrid design that exploits the best properties of the two networks will be very successful. We have developed a general purpose utility and cost function based formulation and a scheduling system that follows such models. Our experiments with stored video streaming show that the hybrid network is able to deliver much better video experience to the user relative to using 3G alone while reducing loan on the 3G network by about three-fourth.

- Then, we have shown that handoffs can be implemented very efficiently by using predictive methods at the link layer and up. While these techniques depend on use of historical information, such data collection is straightforward and can be done directly on-board the client device. Our experiments indicate that use of scripted handoff based on advance RF fingerprinting reduces failed handoffs, improves outages and improves overall throughput by a factor of 2.

- We have experimentally determined that use of prefetching can improve throughput performance by a factor of up-to 2.5. Our prefetching technique is link-layer independent and should be useful in cellular data networks as well.

- We designed BRAVE – an SNR-based rate adaptation algorithm specifically for vehicular WiFi access environments. Unlike well-known algorithms which arrive at an estimate of the current channel conditions by considering history over several seconds, BRAVE considers only sub-second history to determine the rate to be used. BRAVE has been compared with four other major rate adaptation algorithms in a metro-scale WiFi deployment exposing the advantages of using BRAVE vis-a-vis the other protocols. We finally established BRAVE's superiority by performing a head-to-head comparison in an emulation environment.

- Finally, we experimentally showed the benefits a multi-radio multi-vehicle system can have in terms of improving vehicular WiFi performance. Using the MRMV system, and a novel AP filtering method we were able to achieve almost 100% coverage when the underlying AP deployment did not have any coverage holes. We showed that using a network of two MRMV equipped vehicles these coverage holes could be reduced to about half. In addition to improving vehicular WiFi coverage, the MRMV system also increases the overall throughput of the system and reduces throughput variance.

Bibliography

[1] Coverage Locator – Verizon Wireless. http://www.verizonwireless.com/b2c/CoverageLocatorController.

[2] Fon. http://www.fon.com/.

[3] Multiband Atheros driver for WiFi (MADWIFI). http://sourceforge.net/projects/madwifi/.

[4] Onoe Rate Control. http://madwifi.org/browser/trunk/ath_rate/onoe.

[5] Optimum WiFi. http://www.optimum.net/MyServices/WiFi/.

[6] Ubiquity Networks, Inc. http://www.ubnt.com.

[7] Wifi.com. http://www.wifi.com/.

[8] IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks specific requirements 802.11r Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2: Fast Basic Service Set (BSS) Transition , July, 2008.

[9] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris. Link-level measurements from an 802.11b mesh network. *ACM SIGCOMM Computer Communications Review*, 2004.

[10] A. Akella, G. Judd, S. Seshan, and P. Steenkiste. Self-management in chaotic wireless deployments. *ACM MobiCom*, 2005.

[11] I. F. Akyildiz and W. Wang. The predictive user mobility profile framework for wireless multimedia networks. *IEEE/ACM Transactions on Networking*, 2004.

[12] A. Aljadhai and T. F. Znati. Predictive mobility support for QoS provisioning in mobile wireless environments. *IEEE Journal on Selected Areas in Communication (JSAC)*, 2001.

[13] D. W. Allan. Time and frequency (time domain) characterization, estimation and prediction of precision clocks and oscillators. *IEEE Transactions on UFFC*, 1987.

[14] Avila GW2348-2. http://www.gateworks.com/products/avila/gw2348-2.php.

[15] A. Balasubramanian, B. Levine, and A. Venkataramani. Enhancing interactive web applications in hybrid networks. *ACM MobiCom*, 2008.

[16] A. Balasubramanian, R. Mahajan, and A. Venkataramani. Augmenting mobile 3G using WiFi. *ACM MobiSys*, 2010.

[17] A. Balasubramanian, R. Mahajan, A. Venkataramani, B. N. Levine, and J. Zahorjan. Interactive WiFi connectivity for moving vehicles. *SIGCOMM Computer Communication Review*, 2008.

[18] A. Balasubramanian, Y. Zhou, W. B. Croft, B. N. Levine, and A. Venkataramani. Web search from a bus. *ACM CHANTS Workshop*, 2007.

[19] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz. Sumo - simulation of urban mobility: An overview. *SIMUL 2011, The Third International Conference on Advances in System Simulation*, 2011.

[20] A. Bhattacharya and S. K. Das. LeZi-Update: an information-theoretic framework for personal mobility tracking in PCS networks. *Wireless Networks*, 2002.

[21] V. Brik, A. Mishra, and S. Banerjee. Eliminating handoff latencies in 802.11 WLANs using multiple radios: Applications, experience, and evaluation. *IEEE/ACM Internet Measurement Conference*, 2005.

[22] M. Buddhikot, G. Chandranmenon, S. Han, Y. Lee, S. Miller, and L. Salgarelli. Integration of 802.11 and third-generation wireless data networks. *IEEE Infocom*, 2003.

[23] M. Buddhikot, I. Kennedy, F. Mullany, and H. Viswanathan. Ultra-broadband femtocells via opportunistic reuse of multi-operator and multi-service spectrum. *Bell Labs Technical Journal (Special Issue on 4G Networks)*, Feb. 2009.

[24] V. Bychkovsky, B. Hull, A. Miu, H. Balakrishnan, and S. Madden. A measurement study of vehicular Internet access using in situ Wi-Fi networks. *Proc. ACM MobiCom Conference*, 2006.

[25] R. Caceres and L. Iftode. Improving the performance of reliable transport protocols in mobile computing environments. *IEEE Journal on Selected Areas in Communications (JSAC)*, 1995.

[26] J. Camp and E. Knightly. Modulation rate adaptation in urban and vehicular environments: cross-layer implementation and experimental evaluation. *IEEE/ACM Transactions on Networking*, 2010.

[27] R. Chandra, P. Bahl, and P. Bahl. MultiNet: Connecting to multiple IEEE 802.11 networks using a single wireless card. *IEEE Infocom*, 2004.

[28] S. Cheshire, B. Aboba, and E. Guttman. Dynamic configuration of IPv4 link-local addresses, May 2005. RFC 3927.

[29] G. Cho. Using predictive prefetching to improve location awareness of mobile information service. *Springer Lecture Notes in Computer Science*, 2002.

[30] P. Deshpande, X. Hou, and S. Das. Perfomance comparison of 3G and metro-scale WiFi for vehicular network access. *Internet Measurement Conference*, 2010.

[31] P. Deshpande, A. Kashyap, C. Sung, and S. R. Das. Predictive methods for improved vehicular WiFi access. *ACM Mobisys*, 2009.

[32] M. Dischinger, A. Haeberlen, K. P. Gummadi, and S. Saroiu. Characterizing residential broadband networks. *Internet Measurement Conference*, 2007.

[33] S. Dornbush and A. Joshi. Streetsmart traffic: Discovering and disseminating automobile congestion using vanets. *IEEE Vehicular Technology Conference*, 2007.

[34] A. Doufexi, E. Tameh, A. Nix, and A. Molina. Hotspot wireless LANs to enhance the performance of 3G and beyond cellular networks. *IEEE Communications Magazine*, 2003.

[35] J. Eriksson, H. Balakrishnan, and S. Madden. Cabernet: A WiFi-Based Vehicular Content Delivery Network. *ACM MobiCom Conference*, 2008.

[36] Ettus Research LLC. http://www.ettus.com/.

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

[38] J. Froehlich and J. Krumm. Route prediction from trip observations. *Society of Automotive Engineers (SAE) World Congress*, 2008.

[39] Garmin. http://www.garmin.com/.

[40] A. Ghazy and T. Ozkul. Design and simulation of an artificially intelligent vanet for solving traffic congestion. *International Symposium on Mechatronics and its Applications*, 2009.

[41] A. Giannoulis, M. Fiore, and E. W. Knightly. Supporting vehicular mobility in urban multi-hop wireless networks. *ACM MobiSys*, 2008.

[42] D. Giustiniano, E. Goma, A. Lopez, and P. Rodriguez. WiSwitcher: an efficient client for managing multiple aps. *ACM SIGCOMM Workshop on Programmable Routers for Extensible Services of Tomorrow*, 2009.

[43] T. Goff and D. S. Phatak. Unified transport layer support for data striping and host mobility. *IEEE Jorunal on Selected Areas on Communication (JSAC)*, May 2004.

[44] D. Green and A. Obaidat. An accurate line of sight propagation performance model for ad-hoc 802.11 wireless lan (wlan) devices. *IEEE International Conference on Communications*, 2002.

[45] A. Gudipati and S. Katti. Strider: Automatic Rate Adaptation and Collision Handling. *ACM SIGCOMM*, 2011.

[46] V. Gunasekaran and F. Harmantzis. Towards a WiFi ecosystem: Technology integration and emerging service models. *Telecommunications Policy*, 2008.

[47] E. Guttman. Autoconfiguration for IP networking: Enabling local communication. *IEEE Internet Computing*, 2001.

[48] D. Hadaller, S. Keshav, T. Brecht, and S. Agarwal. Vehicular opportunistic communication under the microscope. *ACM MobiSys*, 2007.

[49] G. Holland, N. Vaidya, and P. Bahl. A rate-adaptive MAC protocol for multi-hop wireless networks. *ACM MobiCom*, 2001.

[50] H. Hsieh and R. Sivakumar. A transport layer approach for achieving aggregate bandwidths on multihomed mobile hosts. 2002.

[51] F. Hui and P. Mohapatra. Experimental characterization of multi-hop communications in vehicular ad hoc network. *ACM International Workshop on Vehicular ad hoc networks*, 2005.

[52] N. Imai, H. Morikawa, and T. Aoyama. Prefetching architecture for hot-spotted networks. *International Conference on Communications*, 2001.

[53] J. R. Iyengar, P. D. Amer, and R. Stewart. Concurrent multipath transfer using SCTP multihoming over independent end-to-end paths. *IEEE Transactions on Networking*, Oct. 2006.

[54] D. Jiang and L. Delgrossi. IEEE 802.11p: Towards an international standard for wireless access in vehicular environments. *Proc. IEEE Vehicular Technology Conference (VTC) Spring 2008*, pages 2036–2040, 2008.

[55] Z. Jiang and L. Kleinrock. Web prefetching in a mobile environment. *IEEE Personal Communications*, 1998.

[56] A. Joseph, J. Tauber, and M. Kaashoek. Mobile computing with the Rover toolkit. *IEEE Transactions on Computers*, 1997.

[57] A. U. Joshi and P. Kulkarni. Vehicular wifi access and rate adaptation. *ACM SIGCOMM*, 2010.

[58] G. Judd, X. Wang, and P. Steenkiste. Efficient channel-aware rate adaptation in dynamic environments. *ACM MobiSys*, 2008.

[59] A. Kamerman and L. Monteban. WaveLAN®-II: a high-performance wireless LAN for the unlicensed band. *Bell Labs technical journal*, 1997.

[60] S. Kandula, K. C.-J. Lin, T. Badirkhanli, and D. Katabi. FatVAP: aggregating ap backhaul capacity to maximize throughput. *USENIX NSDI*, 2008.

[61] K.-H. Kim and K. G. Shin. Improving TCP performance over wireless networks with collaborative multi-homed mobile hosts. *ACM MobiSys*, 2005.

[62] K.-H. Kim and K. G. Shin. PRISM: improving the performance of inverse-multiplexed TCP in wireless networks. *IEEE Transactions on Mobile Computing*, Dec. 2007.

[63] M. Kim, D. Kotz, and S. Kim. Extracting a mobility model from real user traces. *IEEE Infocom*, 2006.

[64] J. Krumm. A Markov model for driver turn prediction. *Society of Automotive Engineers (SAE) World Congress*, April 2008.

[65] M. Lacage, M. Manshaei, and T. Turletti. IEEE 802.11 rate adaptation: a practical approach. *ACM International symposium on Modeling, analysis and simulation of wireless and mobile systems*, 2004.

[66] Y. Lee. Measured TCP performance in CDMA 1xEV-DO network. *PAM*, 2006.

[67] W. Lehr and L. McKnight. Wireless Internet access: 3G vs. WiFi? *Telecommunications Policy*, 2003.

[68] F. Li and Y. Wang. Routing in vehicular ad hoc networks: A survey. *IEEE Vehicular Technology Magazine*, 2007.

[69] B. Liang and Z. J. Haas. Predictive distance-based mobility management for multidimensional PCS networks. *IEEE/ACM Transactions on Networking*, October 2003.

[70] X. Liu, A. Sridharan, S. Machiraju, M. Seshadri, and H. Zang. Experiences in a 3G network: interplay between the wireless channel and applications. *ACM MobiCom*, 2008.

[71] H. Luan, L. Cai, and X. Shen. Impact of network dynamics on user's video quality: Analytical framework and QoS provision. *IEEE Transactions on Multimedia*, Nov. 2009.

[72] H. Luo, R. Ramjee, P. Sinha, L. E. Li, and S. Lu. Ucan: a unified cellular and ad-hoc network architecture. *ACM MobiCom*, 2003.

[73] L. Magalhaes and R. Kravets. Transport level mechanisms for bandwidth aggregation on mobile hosts. November 2001.

[74] R. Mahajan, J. Zahorjan, and B. Zill. Understanding WiFi-based connectivity from moving vehicles. *Internet Measurement Conference*, 2007.

[75] Marketman. AT&T will lose money with the iPhone data plan. http://www.geldpress.com/2008/07/att-lose-money-iphone/, 2008.

[76] K. Mattar, A. Sridharan, H. Zang, I. Matta, and A. Bestavros. TCP over CDMA2000 networks: A cross-layer measurement study. *Passive and Active Measurements*, 2007.

[77] S. McCanne and S. Floyd. ns network simulator. http://www.isi.edu/nsnam/ns/.

[78] V. Mhatre and K. Papagiannaki. Using smart triggers for improved user performance in 802.11 wireless networks. *ACM MobiSys*, 2006.

[79] R. T. Morris and J. C. Bicket. Bit-rate selection in wireless networks. Technical report, Masters thesis, MIT, 2005.

[80] M. Moske, H. Fussler, H. Hartenstein, and W. Franz. Performance measurements of a vehicular ad hoc network. *Vehicular Technology Conference*, May 2004.

[81] Muniwireless.com. List of cities and counties with large WiFi networks. http://www.muniwireless.com/reports/Mar-28-2009-list-of-cities.pdf.

[82] V. Navda, A. P. Subramanian, K. Dhanasekaran, A. Timm-Giel, and S. R. Das. MobiSteer: using steerable beam directional antenna for vehicular network access. *ACM MobiSys*, 2007.

[83] A. J. Nicholson, Y. Chawathe, M. Y. Chen, B. D. Noble, and D. Wetherall. Improved access point selection. *Proc. ACM MobiSys*, pages 233–245, 2006.

[84] A. J. Nicholson and B. D. Noble. BreadCrumbs: forecasting mobile connectivity. *ACM MobiCom*, 2008.

[85] A. J. Nicholson, S. Wolchok, and B. D. Noble. Juggler: Virtual networks for fun and profit. *IEEE Transactions on Mobile Computing*, Jan. 2010.

[86] J. Ormont, J. Walker, S. Banerjee, A. Sridharan, M. Seshadri, and S. Machiraju. A city-wide vehicular infrastructure for wide-area wireless experimentation. *ACM WinTech Workshop*, 2008.

[87] J. Ott and D. Kutscher. Drive-thru internet: IEEE 802.11b for automobile users. *Proc. IEEE Infocom*, 2004.

[88] J. Ott and D. Kutscher. A disconnection-tolerant transport for drive-thru internet environments. *Proc. IEEE Infocom Conference*, 2005.

[89] P. N. Pathirana, A. V. Savkin, and S. Jha. Mobility modelling and trajectory prediction for cellular networks with mobile base stations. *ACM MobiHoc*, 2003.

[90] D. Phatak and T. Goff. A novel mechanism for data streaming across multiple IP links for improving throughput and reliability in mobile environments. *Infocom*, 2002.

[91] M. Piórkowski, M. Raya, A. L. Lugo, P. Papadimitratos, M. Grossglauser, and J.-P. Hubaux. Trans: realistic joint traffic and network simulator for vanets. *SIGMOBILE Mobile Computing Communications Review*, 2008.

[92] R. Pantos, Ed. HTTP live streaming. Internet Draft, Informational, http://tools.ietf.org/html/draft-pantos-http-live-streaming-02.

[93] A. Rahmati and L. Zhong. Context-for-wireless: context-sensitive energy-efficient wireless data transfer. *ACM MobiSys*, 2007.

[94] K. Ramachandran, R. Kokku, K. Sundaresan, M. Gruteser, and S. Rangarajan. R2D2: Regulating beam shape and rate as directionality meets diversity. *Proc. ACM MobiSys Conference*, 2009.

[95] I. Ramani and S. Savage. SyncScan: practical fast handoff for 802.11 infrastructure networks. *IEEE Infocom*, 2005.

[96] P. Rodriguez, R. Chakravorty, J. Chesterfield, I. Pratt, and S. Banerjee. MAR: a commuter router infrastructure for the mobile internet. *Proc. ACM MobiSys Conference*, 2004.

[97] H. Schulzrinne. RTP: About RTP and the Audio-Video Transport Working Group. http://www.cs.columbia.edu/ hgs/rtp/.

[98] P. Seeling, M. Reisslein, and B. Kulapala. Network performance evaluation using frame size and quality traces of single-layer and two-layer video: A tutorial. *IEEE Communications Surveys and Tutorials*, 2004.

[99] P. Shankar, T. Nadeem, J. Rosca, and L. Iftode. Cars: Context-aware rate selection for vehicular networks. *IEEE ICNP*, 2008.

[100] S. Shenker. Fundamental design issues for the future Internet. *IEEE Journal on Selected Areas in Communications*, 1995.

[101] M. Shin, A. Mishra, and W. A. Arbaugh. Improving the latency of 802.11 hand-offs using neighbor graphs. *ACM MobiSys*, 2004.

[102] J. Singh, N. Bambos, B. Srinivasan, and D. Clawin. Wireless lan performance under varied stress conditions in vehicular traffic scenarios. *Vehicular Technology Conference*, 2002.

[103] L. Sivalingam, C. Newport, H. Balakrishnan, and S. Madden. Improving wireless network performance using sensor hints. *USENIX NSDI*, 2011.

[104] K. Sklower, B. Lloyd, G. McGregor, D. Carr, and T. Coradetti. The PPP Multilink Protocol (MP). *RFC*, August 1996.

[105] Soekris Engineering. http://www.soekris.com.

[106] L. Song, U. Deshpande, U. C. Kozat, D. Kotz, and R. Jain. Predictability of wlan mobility and its effects on bandwidth provisioning. *IEEE Infocom*, 2006.

[107] L. Song, D. Kotz, R. Jain, X. He, D. Coll, and N. Hanover. Evaluating location predictors with extensive WiFi mobility data. *IEEE Infocom*, 2004.

[108] H. Soroush, P. Gilbert, N. Banerjee, B. N. Levine, M. Corner, and L. Cox. Concurrent Wi-Fi for mobile users: analysis and measurements. *CONEXT*, 2011.

[109] M. Stemm and R. Katz. Vertical handoffs in wireless overlay networks. *Mobile Networks and applications*, 1998.

[110] R. Stewart. Stream control transmission protocol. RFC 4960.

[111] A. Subramanian, P. Deshpande, J. Gao, and S. Das. Drive-by localization of roadside WiFi networks. *IEEE Infocom*, 2008.

[112] A. P. Subramanian, V. Navda, P. Deshpande, and S. R. Das. A measurement study of inter-vehicular communication using steerable beam directional antenna. *ACM International workshop on VehiculAr InterNETworking*, 2008.

[113] W. L. Tan, F. Lam, and W. C. Lau. An empirical study on 3G network capacity and performance. *IEEE INFOCOM*, 2007.

[114] M. Vutukuru, H. Balakrishnan, and K. Jamieson. Cross-Layer Wireless Bit Rate Adaptation. *ACM SIGCOMM*, 2009.

[115] War-driving. http://www.wardriving.com.

[116] S. H. Y. Wong, H. Yang, S. Lu, and V. Bharghavan. Robust rate adaptation for 802.11 wireless networks. *ACM Mobicom*, 2006.

[117] J. Wortham. Customers angered as iphones overload at&t. http://www.nytimes.com/2009/09/03/technology/companies/03att.html.

[118] H. Wu, C. Qiao, S. De, and O. Tonguz. Integrated cellular and ad hoc relaying systems: icar. *IEEE Journal on Selected Areas in Communications*, 2001.

[119] S. Yaipairoj, F. Harmantzis, and V. Gunasekaran. A Pricing Model of GPRS Networks with Wi-Fi integration for Heavy Data Users. *ICETE*, 2005.

[120] P. Yang, H. Deng, and Y. Ma. Seamless integration of 3G and 802.11 wireless network. *MobiWac*, 2007.

[121] J. Yao, S. S. Kanhere, and M. Hassan. An empirical study of bandwidth predictability in mobile computing. *ACM WiNTECH*, 2008.

[122] T. Ye, H. Jacobsen, and R. Katz. Mobile awareness in a wide area wireless network of info-stations. *ACM MobiCom*, 1998.

[123] J. Yoon, B. D. Noble, M. Liu, and M. Kim. Building realistic mobility models from coarse-grained traces. *ACM MobiSys*, 2006.

[124] F. Yu and V. Leung. Mobility-based predictive call admission control and bandwidth reservation in wireless cellular networks. *Elsevier Computer Networks*, April 2002.

[125] J. Zhao, T. Arnold, Y. Zhang, and G. Cao. Extending drive-thru data access by vehicle-to-vehicle relay. *ACM International workshop on VehiculAr Inter-NETworking*, 2008.