# ENHANCING NETWORK FUNCTIONALITIES FOR EMERGING MOBILE NETWORKS THROUGH LEARNING

by

**TINGTING SUN**

A Dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Profs. Yanyong Zhang and Wade Trappe

and approved by

New Brunswick, New Jersey

May, 2014

## ABSTRACT OF THE DISSERTATION

# Enhancing Network Functionalities for Emerging Mobile Networks through Learning

By Tingting Sun

Dissertation Director:
Profs. Yanyong Zhang and Wade Trappe

With rapid evolution of technology and growing use of wireless devices in our daily lives, mobile network is becoming one of the most promising platforms for many brand new applications. Several distinguished features make mobile network different from traditional computer networks, such as high mobility and unpredictable mobility patterns. With the networks becoming increasingly diverse and complex, it's more and more difficult to know the properties of a network a-priori. Therefore, the need for "learning" important network characteristics in such a dynamic knowledge setting becomes crucial. In this thesis, we show our research efforts to explore methods to learn important network properties, and improve the following three aspects of mobile networks: data management, load management, and identification services.

The data management issue is most critical when there is no central infrastructure available or when the mobile-to-infrastructure communication bandwidth is limited. Since blindly uploading every piece of sensor data to a remote server is inefficient, local data aggregation is required to reduce the communication cost and improve efficiency. We propose the Geocache concept and the Boomerang anchoring protocol to address this issue, and further introduce adaptive learning methods to better deliver time-sensitive data.

Our efforts in load management are focused on adaptive load-balancing schemes for wireless

LANs where multiple access points are present. We propose a distributed access point selection scheme by which nodes select an appropriate access point to associate with, based on each individual devices channel utilization. This approach effectively reduces unnecessary reassociations and improves upper layer performance such as throughput and packet delivery delay. We further enhance the association protocol by using reinforcement learning to dynamically schedule the probing of neighboring access points (APs). By learning from past experience, we ultimately bring down the probing overhead.

Lastly, we focus on the security aspect of the network by improving the identification process. We examine the problem of identifying different association protocols based on probing patterns, such as probing frequency and probing frame types. We apply learning methods to identify several association protocols and propose an approach which combines k-means clustering and Gaussian fitting to classify the association protocols.

# Acknowledgements

First, I would like to thank my advisor, Prof. Yanyong Zhang, for the inspiring guidance, strong support and great help that she has given me throughout my PhD studies. She offered me invaluable advice on my research, and has provided me many precious professional opportunities. This thesis would not have been possible without her support.

I am especially grateful to my co-advisor Prof. Wade Trappe, who has been a constant and important source of guidance, support and encouragement. I have benefited tremendously from his extensive knowledge and technical insight, especially in his primary research area of wireless security and security protocols.

I also wish to thank Prof. Marco Grutesor, who has significantly helped my thesis work. I'm fortunate to have the opportunity to collaborate with him and benefit from his expert knowledge in Mobile/Vehicular networks. I am grateful to Prof. Yingying Chen, for her warm support and constructive feedback on my thesis.

I would like to express my gratitude to all WINLAB graduate students, especially to Prof. Zhang's and Prof. Trappe's research groups. Beyond my immediate group, I really appreciate the helpful technical or nontechnical conversations with Bin Zan, Ben Firner, Yao Li, Chenren Xu, Zhuo Chen, Feixiong Zhang and all my WINLAB colleagues.

Finally, I would like to thank my family. My parents and my husband, who have been a constant source of emotional support. And my daughter, who greatly inspires me to overcome difficulties and pursue happiness in life.

# Dedication

To my family.

# Table of Contents

# List of Figures

# Chapter 1

# Introduction

## 1.1 Motivation and Problem Overview

Recent advancements across a variety of communication and computing technologies, ranging from wireless communication to localization techniques, are driving mobile networks to become one of the most promising platforms for many brand new applications. As the wireless systems offer more convenience, they are inevitably facing more challenges. With networks becoming increasingly diverse and complex, it's more difficult to know properties about a network a-priori. They also expose inefficiencies and vulnerabilities in performing the very applications they aim to support.

In mobile networks, mobile sensing takes place anytime, anywhere. The generated data is in huge amount, in terms of both type and volume. To take advantage of the anytime-anywhere mobile sensing, traditional data management solutions usually employ well-known data servers [1] to provide bridges between data producers and consumers. However, due to the large volume of data, it is far from efficient to store all the data on servers. Therefore, local data aggregation is required to reduce the communication cost and improve the transmission efficiency. However, the question of how and when the system should aggregate the data is not trivial. The design of a communication protocol which maintains high data delivery rate in an infrastructure-less model is even more challenging.

In wireless LANs, the explosion in demand for connectivity requires that efficient and effective methods be used to optimally manage connectivity for nearby wireless networks. The widespread WiFi availability presents many choices to a wireless user in terms of which AP it can associate to [2][3][4][5]. Current association schemes are mostly based on the measurements of received signal strength indicator (RSSI) or consecutive beacons lost. They perform poorly in many situations because they overlook the load and bandwidth of the AP, which leads to degraded services. Alternative proposals to address these shortcomings are discussed in [6, 7][8][9][10]. However, most of these models either have a centralized infrastructure or need special features from the APs, such as they use a designated server or the APs to collect and analyze the bandwidth information, and distribute the association decisions throughout the network. As a result, these methods relies heavily on the existing infrastructure, often making them inflexible or unavailable for those without infrastructure support.

In addition, many network services relies heavily on the use of administration information, which can be obtained either a-priori or on the fly. The information may either be public, or the client may obtain the information regarding a particular service from the provider as needed. This approach, however, has been noted to be easily affected by a malicious entity providing fake administration information to clients and perform attacks once the client is connected. Association attack takes advantage of the security risks of the association protocol used by the clients. Various follow-up attacks can be easily performed by passing false administration frames once a client has been tricked to switch its association to a compromised gateway. Therefore, it becomes increasingly important that the information utilized by these services is trustworthy. Notably, before an entity gains access to a certain gateway, it is essential that the administration information be verifiable.

## 1.2 Our Contribution

To address the challenges introduced by mobile sensing, we advocate building directories around location of interests by having nearby mobiles carry the data (or the metadata of these data) generated around these locations. We refer to the directory information as Geocache of the location, and the location of interest as anchor location. By always storing Geocache on nodes that are close to the anchor location, we keep the data around the location where they were generated, thus easily facilitate location-based queries by directing them to the corresponding anchor location. In this thesis, we study protocols that retain Geocache around the anchor location through inter-vehicle communication. We identify two major challenges : (i) returning the Geocache to the anchor location with high probability, even if the carrier of the Geocache becomes temporarily disconnected; (ii) minimizing the communication overhead for retaining the Geocache near an anchor location. The boomerang protocol addresses these challenges by using a trajectory-based approach. It increases the successful return rate of the Geocache even in temporary disconnected scenarios. While the boomerang protocol is inspired by delay-tolerant geographic routing, it is unique in being able to efficiently record a nodes trajectory as the node is moving away from the anchor location and using this trajectory as a guide to carry back the Geocache. Further, to reduce communication overhead, instead of having each node send out the Geocache as soon as they receive it, we have the node keep the Geocache until it diverges from the original trajectory. Thus, it exploits an important characteristic of vehicular networks, which is: vehicles move on well-defined and usually bidirectional paths. In connected networks, the return probability is significantly improved along with reduced communication overhead, by allowing a node to briefly carry the Geocache away from the anchor location before returning it, instead of constantly keeping the Geocache at the anchor location. We further look at anchoring which has specific time constraints. For those where the Geocache is required to return to the anchor location within a specific time frame, we implemented the learning-based Boomerang protocol to adaptively adjust packet handoff time based on the received network feedback.

In Wirelss LANs, we notice that most Internet applications have certain bandwidth requirement, but it is often overlooked when making association decisions. The most commonly used strategy to choose an AP is to look at the signal strength. We propose that an access service would perform better if also consider the bandwidth requirement when choosing an AP. If at any time, the available bandwidth of the current AP drops below the required bandwidth level, the node may start looking for a better AP. The core component of this scheme is an distributed available bandwidth estimation method. This method can give accurate estimations in different stages of the data transmission, e.g. when the wireless node is transmitting packets, receiving packets, and when the wireless node is probing a channel. We also identify two potential problems for the bandwidth-based association scheme. One is over-probing: a node may probe other APs too frequently when no better AP is available. Over-probing can adversely affect the performance since the node cannot transmit packets during the probing delay. We address this problem by enhancing our scheme with reinforced learning techniques so that a node only probes when there is adequate gain. Another problem is overswitching: a set of nodes may decide to switch to the same AP around the same time, which leads to the thrashing effect. We propose to address this problem by using delayed switching and probabilistic switching. Simulation results show that the above mentioned enhancements can effectively alleviate these problems.

In the third part of the thesis, we investigate the identification methods for association services. We argue that being able to identify the association protocols used by nearby clients may provide a mobile client great advantage in choosing its own association approach, based on its need or the network condition. In this thesis, we examine the problem of identifying different association protocols. We combined k-means clustering and Gaussian fitting to classify the association protocols based on probing patterns, and tested the designed scheme on traffic traces for a variety of network scenarios. We also designed a method to quantify the likelihood of the identification using confidence intervals. Further interpretation of the results also reveals import information about the metrics of the clients chosen association protocol.

## 1.3 Thesis Roadmap

The rest of thesis is organized as follows. We first present our location-based Boomerang protocol in Chapter 2. Next, we examine the problem of load-balancing in wireless LAN through channel utilization and adaptive probing in Chapter 3. Our work on association protocol identification is presented in Chapter 4. Finally, we conclude the thesis and discuss opportunities for future works in Chapter 5.

# Chapter 2

# Tying Data to Geographic Locations in Mobile Disconnected Networks

## 2.1 Introduction

As daily mobile devices such as smart phones, PDAs and digital cameras are more and more used as sensing devices [11, 12, 13], mobile sensing is becoming a social event instead of a high-tech phenomenon. Compared to today's special-purpose sensing applications such as automotive traffic congestion monitoring [14] and pothole detection [15], mobile sensing takes place anytime, anywhere, and will have far more diverse meanings. A direct consequence of this trend is the production of a vast amount of data, in terms of both type and volume. Example data types include pictures, videos, audios, and plain text-based sensor readings. These data can potentially bring great convenience to the society as they can serve as traces of our lives and logs of the physical world.

In order to take full advantage of the truly anytime-anywhere mobile sensing, directories [1] are necessary to provide bridges between data producers and consumers. Traditional data management solutions usually employ well-known data servers to facilitate effective data communication. Having data servers, however, is far from suitable in our case. Due to the large volume of data (imagine the number of pictures generated if every smart phone takes one picture per day), it is unrealistic to rely on servers to store all the data. Additionally, it is highly wasteful as most of the data will not be used at all. Finally, privacy will be a serious concern in such a solution as well.

To address this challenge, we take inspiration from real life solutions. Suppose if we lost/found an item, a common practice is to post a note around the area where it was lost/found, and later we refer back to the same location to check for further updates. Similarly, in the anytime-anywhere mobile sensing era, information is commonly tagged with location, thus encouraging location-based queries. To facilitate such queries, we advocate building "directories" around locations of interest by having nearby mobiles carry the data (or the metadata of these data) generated around these locations. We refer to the directory information as *Geocache* of the location[1], and the location of interest as *anchor location*. By always having the node close to the anchor location carry the Geocache, we can tie the data around the location where they were generated, thus easily facilitate location-based queries by directing them to the corresponding anchor locations. Once the Geocache for an anchor location reaches a certain size, we have the options of compressing the data, or applying the "chaining" technique, which retains only the latest Geocache entries around the anchor location while saving a link to the storage of older entries. Finally, we may also delete outdated or trivial entries.

In this thesis, we study protocols that retain Geocache around the anchor location through inter-vehicle communication. Specifically, we address two major challenges: (i) returning the Geocache to the anchor location with high probability if the carrier of the Geocache becomes temporarily disconnected; (ii) minimizing the communication overhead for retaining the Geocache near an anchor location.

The boomerang protocol addresses these challenges by using a trajectory-based approach. It increases the successful return probability of the Geocache even in temporary disconnected scenarios. While the boomerang protocol is inspired by delay-tolerant geographic routing, it is unique in recording a node's trajectory as the node is moving away from the anchor location and using this trajectory as a guidance to carry back the Geocache. Further, to reduce communication overhead, instead of each node sending the Geocache over the wireless link as soon as it was received, we have the node keep the Geocache until it drives off the original trajectory. Thus, it exploits an important characteristic of vehicular networks, which is: vehicles move on well-defined and usually bidirectional paths. We will show through analysis and simulations how this characteristic impacts the performance. In connected networks, the increased return probability allows significantly reduced communication overhead by purposefully allowing a node to briefly carry the information

---

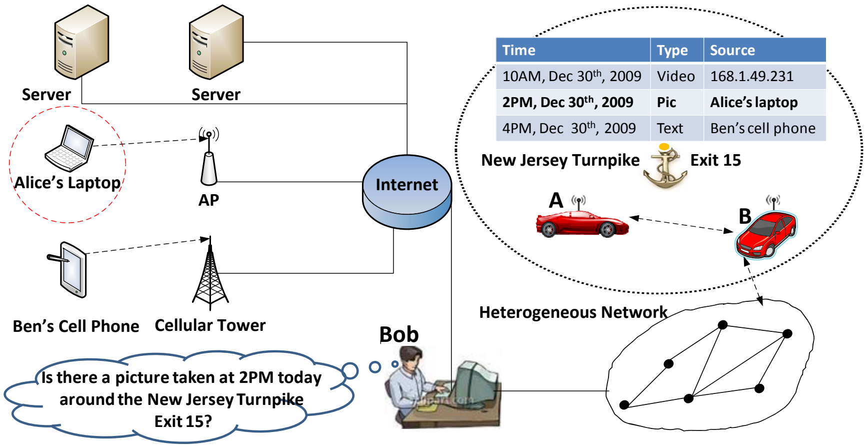
[1] Inspired by physical Geocaches that store information and items at specific locations. Finding them with GPS receivers has become a popular pastime (http://en.wikipedia.org/wiki/Geocaching).

away from the anchor location before returning it, instead of constantly keeping the Geocache at the anchor location.

In summary, the salient contributions of our work are:

- Outlining the Geocache concept, which can be used to make sensed data available at anchor locations, to support mobile sensing applications over a distributed network of mobile nodes.
- Designing the boomerang protocol which periodically return the Geocache to its anchor location. The protocol employs two alternative heuristics in selecting Geocache carriers, with the baseline approach based on a node's distance to the anchor location, and the improved approach based on a node's location relative to the Geocache's reverse trajectory.
- Efficient trajectory construction and divergence detection algorithms for the trajectory-based boomerang protocol.
- Showing through simulations using the traffic trace for a southern New Jersey area, that the use of trajectory in the boomerang protocol increases the Geocache return probability by an average of 70% compared to the baseline shortest distance routing approach.
- Introducing the parameter $\rho$ that defines the connectivity of a road map, and showing through analysis that the trajectory-based boomerang protocol outperforms the baseline distance-based boomerang protocol under realistic $\rho$ values.
- Designing protocols that return Geocache within a specified time constraint through Q-learning.

The remainder of the chapter is organized as following: Section 2.2 briefly discusses the platform assumption and system model. Section 2.3 describes in detail the boomerang protocol and Section IV discusses its implementation, especially the techniques used for detecting divergence from a recorded trajectory based on GPS traces. Section 2.5 analytically evaluates the boomerang protocol's performance on a Manhattan grid. Section 2.6 compares the performance of the two boomerang protocol variants using real world road maps and traffic patterns. Section 2.7 extends the protocol to support applications with return time constraints. Section 2.8 discusses the related work, and Section 2.9 concludes the chapter.



| Time | Type | Source |
|---|---|---|
| 10AM, Dec 30th, 2009 | Video | 168.1.49.231 |
| **2PM, Dec 30th, 2009** | **Pic** | **Alice's laptop** |
| 4PM, Dec 30th, 2009 | Text | Ben's cell phone |

Figure 2.1: Motivating scenario of Geocache.

## 2.2 System Model and Applications

We consider a scenario where nodes move along constrained two-way paths. Nodes can communicate intermittently via high-bandwidth short-range radios (e.g., 802.11) with other nearby nodes or through a continuous low-bandwidth wide-area network (e.g., cellular network). We assume that nodes have high storage capacity and are aware of their geographic positions (e.g., via GPS), but the communication and localization systems do not rely on proprietary road maps.

**Mobile Data Management Through Geocache:** As mobile devices start to produce large volumes of data, efficient management of such data can bring great convenience to our daily life. Let us look at the motivating scenario illustrated in Fig. 2.1.

Alice took a picture of a car accident using her cell phone when she drove by the accident scene. Bob, the victim in the accident, was eagerly seeking such pictures as evidence to support his claims in front of the judge or his insurance company. A traditional solution for exchanging the information would likely involve Alice uploading her picture to a server where Bob can download from (after consulting popular search engines such as Google). However, as mentioned earlier, this solution does not scale well with the data volume we may expect from anytime-anywhere mobile sensing.

Instead, we propose a highly distributed approach in which mobile devices keep the data locally, but leave a log around the geographic location where the data was generated. A typical log may include the time and location at which the data was generated, the data type (video, audio, picture, etc), and an (encrypted) ID of the mobile device. In this approach, logs generated around the same location from different mobile devices will form a location-based log file, which we call *Geocache*, and it will be retained around the anchor location by passing by mobiles ($A$ and $B$ in Fig. 2.1). The Geocache serves the same purpose as the bulletin board in our daily lives. Queries such as "Is there a picture taken at 2PM today around the New Jersey Turnpike Exit 15?" will be routed to the corresponding anchor location to look up the Geocache and find out whether related information exists. If matching traces are found, the data requester may directly contact the data producers, and possibly paying a fee for the retrieved data. Based on the same idea, range queries can also be enabled in which queries concerning wider areas will be sent to multiple anchor locations.

A lot of issues need to be carefully evaluated in implementing this architecture. To name just a few, a Geocache may quickly become too large for a popular anchor location, a Geocache may get lost for a remote anchor location that few mobiles access, a Geocache may receive so many requests that it incurs a large delay for an urgent request, etc. In this thesis, we set out to attack the most important challenge: *how can a Geocache be retained around its anchor location by passing by mobile nodes in an efficient way?*

In order to retain the Geocache around the anchor location, we have the mobile node that is close to the anchor location carry the Geocache. Before it moves too far away, the node transmits the Geocache to one or more nodes that are closer to the anchor location. This procedure is rather straightforward when enough nodes pass the anchor location all the time, but its execution becomes challenging in a more sparse setting. When few nodes pass the anchor location, mobile nodes are often isolated and temporarily unable to communicate with each other over the short-range communication link. The challenge here is to design a protocol that can maintain the Geocache around the anchor location with high probability of success. While in dense settings, the same protocol can be used to reduce the communication cost.

## 2.3 Geocache Anchoring Protocols

The goal of the Geocache anchoring protocols is to retain Geocache data around the corresponding anchor location while minimizing communication overhead.

Intuitively, we envision the following anchoring process: the mobile node that currently carries the Geocache (referred to as the *carrier*) is moving away from the anchor location. To avoid taking the Geocache away, it hands off the data to other nodes, preferably those traveling towards the anchor location. After receiving the data, the new carrier node will periodically examine whether another handoff is needed. This process repeats until the data returns to the anchor location, and we call this protocol a *boomerang* protocol because the data eventually returns to its origin like a boomerang.

To motivate how this boomerang approach can reduce communication overhead, let us consider a brief gedanken experiment. One could retain information at the anchor location by simply handing off the Geocache whenever the anchor location moves out of the radio range. In an idealized model with constant radio range $r$, vehicular velocity $v$ and high vehicle density, retaining the Geocache for a duration $t$ would require $m = \frac{tv}{r}$ handoffs. However, under ideal settings, the boomerang approach can reduce the number of transmissions to $m = 2$ (one to the new carrier heading back and one to the anchor location). Thus, the boomerang approach has the potential to significantly reduce transmission overhead when the Geocache content is only needed periodically. Under more realistic settings, the number of transmissions in the boomerang protocol may be larger because the chosen carrier may diverge at intersections from the original path and not return to the anchor location. The vehicles may also need to send periodic broadcast messages to identify Geocache for the same anchor location on other nodes and enable operations such as aggregation and update.

### 2.3.1 Protocol Description

The main challenge for implementing boomerang protocol lies in the choice of a new carrier node at each handoff, especially if the first handoff occurs somewhere far away from the anchor location.

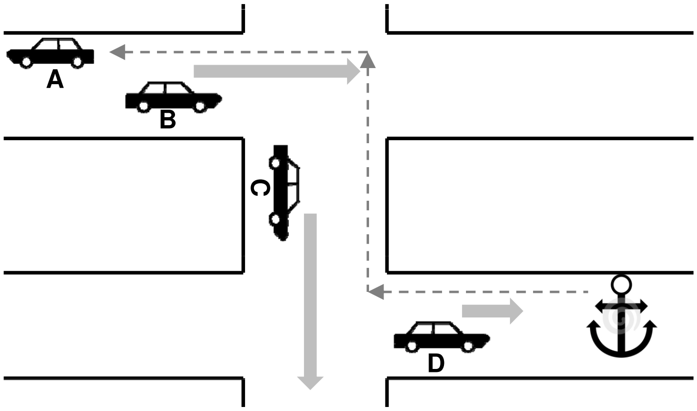


Figure 2.2: Sometimes, a single carrier node is insufficient to anchor the data. In this example, after $A$ hands off the data, we need $B$, $C$, $D$ to return the data to its anchor.

The data may have traveled along a rather complicated route before the current carrier looks for a new carrier, as illustrated in Fig. 2.2. In this case, a single carrier node may not be sufficient to bring back the data; instead, nodes $B$, $C$, and $D$ all needed to be involved in this returning process. Efficiently choosing a set of suitable carriers is thus the key to the success of the boomerang protocol. A set of poorly-selected carriers may incur a long delay in bringing back the data (note that the data may lose its value after a long delay). The task of choosing appropriate carrier nodes is particularly daunting because at each handoff, neither the current carrier nor the nodes within the hand off range have knowledge beyond their current velocity and location, and the traversed trajectory. In this thesis, we propose a trajectory-based carrier selection approach and compare it to a baseline shortest-distance-based selection scheme.

**Shortest-Distance-Based Selection:** Heuristics that fall in this category choose the node, among all those within the handoff range, that is closest to and moving towards the anchor location. They share the same rationale as many Geo-routing algorithms[16], and we consider such an algorithm which we refer to as *MaxProgress*. A simple example is given in Fig. 2.3. $A$ is the anchor location, and $B$ is the carrier node. At handoff, $E$ will be chosen as the next carrier node because its distance from $A$ is the shortest among all other nodes within radio range and still decreasing.

Next, let us look at the detailed handoff procedure for MaxProgress. After traveling away from

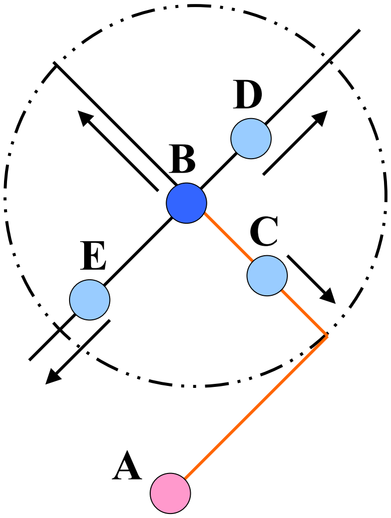


Figure 2.3: An example handoff situation. In this case, $B$ is the current carrier. MaxProgress will choose node $E$ as the new carrier (because its distance from A is decreasing and it is currently the closest to A among all the nodes), while RevTraj will choose $C$ as the new carrier.

the anchor location for a certain amount of time, the carrier initiates a handoff by first broadcasting the Geocache along with the anchor location. Every node within the handoff range responds by checking its distance from the anchor location, and will become a candidate if it's moving towards the anchor location. Next, each candidate node will calculate their individual backoff time before sending the acknowledgement (ACK). The backoff time $T_{backoff}$ is defined by:

$$T_{backoff} = \frac{(d - d_0 + r)}{2r} T_{max}, \tag{2.1}$$

where $T_{max}$ is the maximum backoff time set by the system; $d$ is the distance between the candidate node and the anchor location; $d_0$ is the distance between the carrier node and the anchor location; $r$ is the radio range. Using this equation, we can distribute the ACK backoff times between $[0, T_{max}]$. More importantly, the node with the shortest distance to the anchor location will have the smallest backoff time, and send out ACK the earliest, thus becoming the next carrier.

The new carrier will carry the Geocache until it finds out its distance to the anchor location starts increasing. It will then initiate another handoff to look for new carrier nodes using the same procedure.

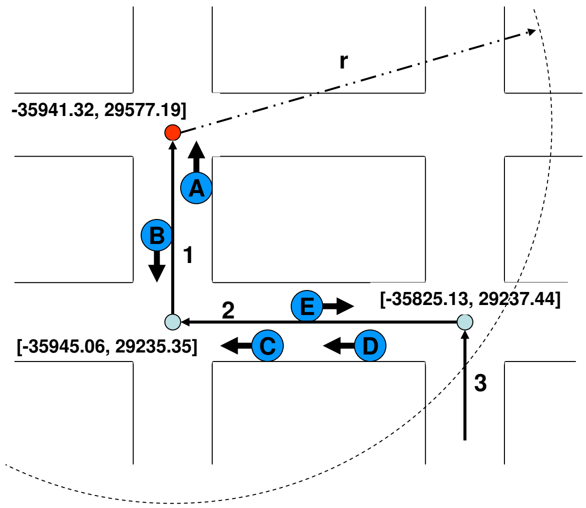


Figure 2.4: Illustration of segments and trajectory-based handoff procedure.

**Trajectory-Based Selection:** While the distance-based approach works well for geographic routing in ad-hoc networks, it may not be suitable for vehicular networks because it ignores the fact that vehicles only move along fixed roadways. Therefore, progress in euclidian distances does not always yield a feasible path that returns to the anchor location. For instance, node $E$ in Fig. 2.3 is on a path (dead end) that never lead to $A$.

The above concerns lead us to the trajectory-based selection approaches. These approaches select new carrier nodes from those that are traveling in the opposite direction of the same trajectory passed by the Geocache. The rationale is that the trajectory describes a general feasible return path (with the exception of one-way paths scenarios). The heuristic we consider in this study is thus called *RevTraj* (**Rev**erse **Traj**ectory). Under this scheme, in the same example given in Fig. 2.3, node $C$ which is in the opposite direction of $B$'s trajectory will be chosen as the next carrier node.

The key component of RevTraj is trajectory recording: the aggregated path the pervious carriers have traveled so far. The trajectory grows when a carrier is moving away from the anchor location, and shrinks when it's moving towards the anchor location. Depending on the storage and processing power available on the mobile units, we can use either raw GPS traces or "segmented" trajectory which only consists of the critical points on the path.

Next, let us look at the detailed handoff process in RevTraj. In the discussion below, we assume segmented trajectory is used instead of continuous trace. As illustrated in Fig. 2.4, a segment is represented by the coordinates of the two end points, e.g., $seg_1$: $[(-35941.32, 29577.19), (-35945.06, 29235.35)]$. The trajectory is implemented as a stack of end points, so that the latest segment is always on top of the stack. Below is the summary of the handoff procedure used in RevTraj:

1. *Handoff Initiation.* The current carrier broadcasts the Geocache along with the trajectory.

2. *Candidate Identification.* Every node within the radio range pops out the latest segments from the trajectory stack. We use a parameter, *lookahead distance* (**LD**), to limit how many recent segments we examine. These lookahead segments can be numbered as $seg_1, seg_2, ...seg_{LD}$, with $seg_1$ being the latest segment. If the node finds itself on one of these lookahead segments, it becomes a candidate node and proceeds to the next step.

3. *Candidate Prioritization.* All the candidates are prioritized according to the following rules: (1) nodes traveling on higher-numbered segments are granted higher priority than those on lower-numbered segments; (2) for nodes traveling on the same segment, we give higher priority to those closer to the anchor location.

   The prioritization rules can be easily implemented if each candidate node calculates its ACK backoff time using the following equation:

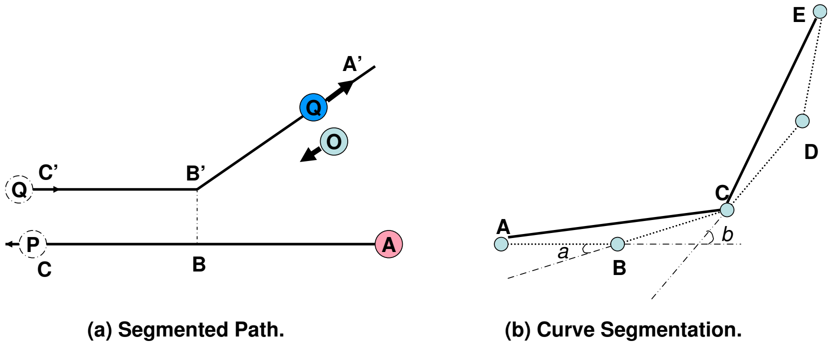
   $$T_{backoff} = \frac{LD - i + \frac{d-d_0+r}{2r}}{LD} T_{max}, \tag{2.2}$$

   where the definitions for $d$, $d_0$, $T_{max}$ are the same as before, and $i$ is the segment number.

4. *Carrier Selection.* The node with the smallest ACK backoff will send out the ACK earliest among all the candidates. To avoid hidden or exposed terminal problem, we suggest ACK be sent using higher transmission power to cover a wider range. Upon receiving the ACK, the old carrier as well as other candidates delete their own copy of Geocache.

5. *No Acknowledgement.* If the current carrier does not receive any ACK, it keeps the Geocache and initiates another handoff after a short interval.

We further distinguish between prioritized RevTraj as described above, and non-prioritized RevTraj, where all candidate nodes pick a random backoff value from $[0, T_{max}]$, and the node with the smallest backoff becomes the new carrier.



(a) Segmented Path.

(b) Curve Segmentation.

Figure 2.5: Segmented path and curve segmentation.

## 2.4 Trajectory construction in RevTraj

The primary challenge in implementing the trajectory-based boomerang protocol lies in the construction of the Geocache trajectory, and the detection of divergence from a given trajectory. To illustrate these challenges, let's consider the example in Fig. 2.5, where the Geocache was handed off to $Q$ who was traveling in the opposite direction of $P$'s trajectory. After traversing the path until $B'$, $Q$ diverges from $P$'s trajectory and heads for $A'$. In this case, $Q$ must be able to detect the divergence and start extending the path history from $B'$, so that at its next handoff, a modified path $A \rightarrow B(B') \rightarrow A'$ will be passed to the new carrier node $O$. The challenge in this implementation lies in developing robust trajectory update and divergence detection algorithms that are feasible across a variety of road networks.

### 2.4.1 Data Pre-processing and Trajectory Recording:

In RevTraj, we need to construct trajectories from location (latitude and longitude) recordings reported by the GPS. First, we aggregate consecutive samples with little spatial distance in between (20m in our experiments), to reduce sample noise. The effect of this pre-processing is illustrated in Fig. 2.6.

Next, we segmentize the path, retaining only critical turning points by comparing the heading difference between the node's driving direction and the direction of the current segment. If the heading difference exceeds a threshold, we decide we are heading at a new direction and add the

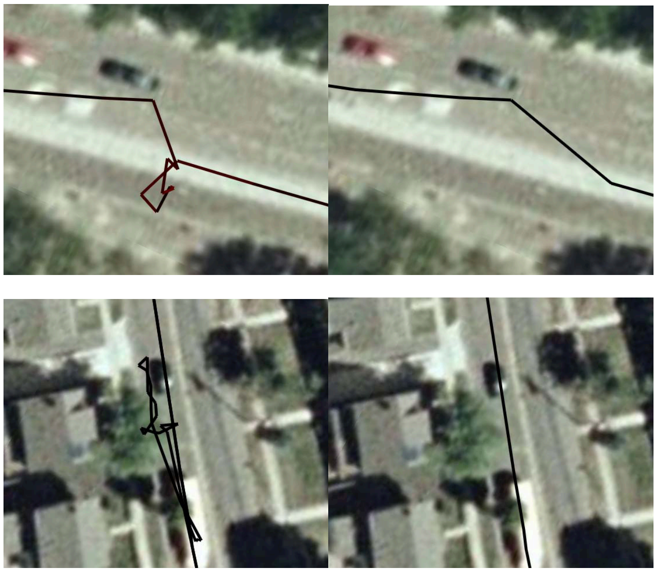


Figure 2.6: Traces before (two pictures on the left) and after (two pictures on the right) data preprocessing.

turning point to the trajectory to mark the start of a new segment. Consider the consecutive GPS recordings $A, B, C, D, E$ as illustrated in Fig. 2.5(b). Initially the heading is the direction of $AB$. When $C$ is present, we check the angle $\alpha$ between $BC$ and $AB$, and determines $\alpha$ is small enough to consider $C$ still on the same segment as $AB$. Next when $D$ is recorded, we check the angle $b$ between $CD$ and $AB$. At this time, $b$ is above the threshold, therefore we identify a new segment $CD$. At the end of this example, we have two segments: $AC$ and $CE$.

In the implementation, we also defined a threshold for the minimum segment length, to deal with large curve scenarios with consecutive small angle differences.

### 2.4.2 Divergence Detection:

When on the return path to the anchor location, a node shrinks the saved trajectory by removing segments it has passed. Meanwhile, it also needs to continuously check if it has diverged from the remaining trajectory.

Intuitively, a divergence from the trajectory will result in a noticeable change in the heading direction, as well as a distance increase from the trajectory. However, using one factor alone to determine divergence could be erroneous. Lane shift, the individual's driving behavior and many other factors may all lead to a sudden direction change without actual divergence. Further, the variance in road widths (e.g., 15 to 60 ft for city roads[2]) makes the selection of a single distance threshold difficult.

In our divergence detection algorithm, we monitor the following conditions when a new GPS data is generated: (1) if the distance $d$ between the current location and the trajectory has exceeded the distance threshold $d_0$, and the heading change has exceeded the heading threshold $h_0$, (2) if $d$ has exceeded the maximum road width $d_{max}$. Divergence is declared if either condition is met for $k$ consecutive GPS readings. Therefore, the rule for divergence detection is defined as:

$$divergence = \begin{cases} 1 & d > d_{max}, \text{ or} \\ & d > d_0 \text{ and } h > h_0; \\ 0 & \text{otherwise.} \end{cases} \tag{2.3}$$

Here, $d_{max}$ is the maximum road width, which can be obtained from road design manuals. $d_0$ and $h_0$ are the thresholds for distance and heading difference. Next, we will discuss how these threshold values are determined based on analysis with a real-time traffic trace collected from the southern New Jersey area.

### 2.4.3 Divergence Detection Model based on Real-world Traces

Given the divergence detection method in Eq. 2.3, the key to the solution is thus to set suitable values for the two thresholds: $d_0$ and $h_0$. In this study, we use the GPS samples taken from a field study to determine appropriate threshold values.

We collected 2 hours of GPS traces in the New Brunswick and North Brunswick area in New Jersey. We drove on both highways and local streets, covering about 55 miles with an average speed of 35 Mph. Fig. 2.7 depicts out route on the local map. In the experiment, we covered the loop shown in Fig. 2.7 twice. In the first pass we strictly stayed on the main loop, while in the second pass, we constantly drove into detour and side streets to emulate divergence from the main loop.

---

[2] http://www.greensboro-nc.gov/visitors/

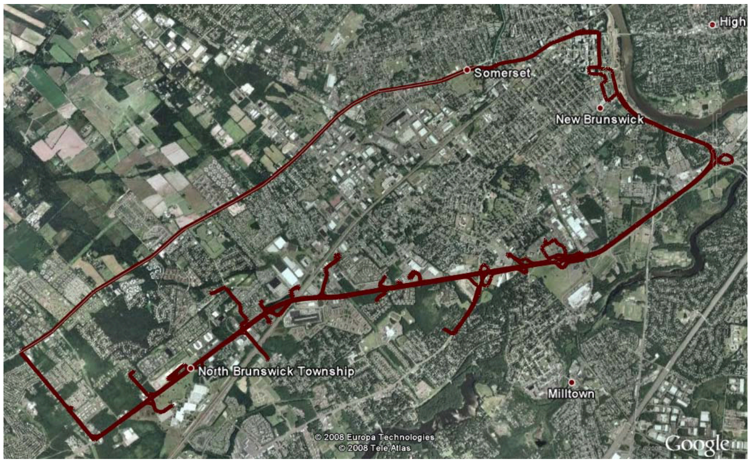


Figure 2.7: The routes we traversed in the experiment is colored in red.

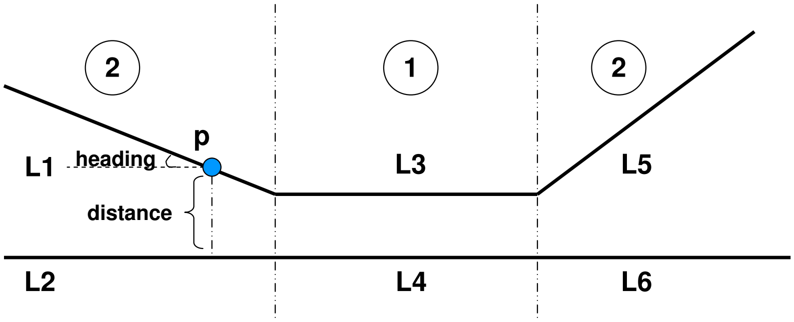


Figure 2.8: Divide route pairs into 2 groups.

Next, we overlay the traces from the two passes and manually divide them into segment pairs. As shown in Fig. 2.8, in each segment pair the two paths either diverge or remain parallel. The segment pairs are then manually labeled into 2 groups: Group 1 for parallel pairs (e.g. the $\{L3, L4\}$ pair in Fig. 2.8) or Group 2 for diverging pairs (e.g. the $\{L1, L2\}$ pair in Fig. 2.8).

We then process the segment pairs in each group as the following. Taking the $\{L1, L2\}$ pair for example, for each location record $p$ on $L1$, we create a new record ($d_p$, $h_p$), where $d_p$ is the distance from $p$ to $L2$, and $h_p$ is the heading difference between $p$ and $L2$. Fig. 2.9 plots the distance and heading difference values of all records from both groups ($5444$ records in total). Normally, a simple machine learning algorithm such as neural network or support vector machine (SVM) can be

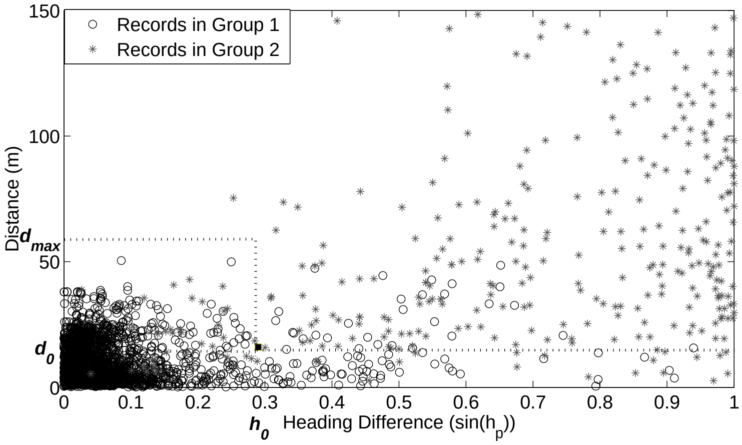


Figure 2.9: The records of the two groups.

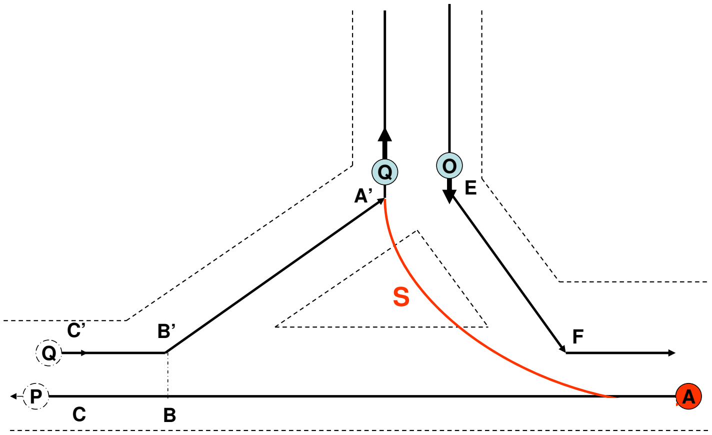


Figure 2.10: Illustration of corner smoother.

applied to easily classify the two groups of records. But in our case, the classification pattern in the raw data is already so obvious that even a heuristic based method can achieve satisfying divergence detection rate. With $d_0 = 16$m and $h_0 = 0.29$, we are able to achieve a detection rate of $98.7\%$. The average detection delay is $2.26$(sample), meaning the divergence will be correctly detected at the 3rd sample from the start of the divergence. $d_{max}$ is set to 60m according to the street design manual for the city of New Brunswick, NJ.

### 2.4.4 Optimization Techniques

We also implemented the following techniques to further optimize the performance.

1. *Corner smoother.* On major highways, cars use different ramps to enter and exit the highways. An example is given in Fig. 2.10, in which $B'A'$ is the exit ramp and $EF$ is the entry ramp. This will cause confusion to a carrier node, e.g. $O$, that receives the Geocache from previous carrier $Q$, who exited the highway thus diverged from the trajectory $ABC$. Even though $O$ is entering the highway and approaching the trajectory, it may not think so because the recorded trajectory it received from $Q$ is $A'B'BA$, and due to the mismatch between the exit and entrance ramps, it is traveling along $EFA$. To avoid this inaccuracy, after a node exits from the high way, besides recording its exact trace, the node also transforms the trace into an approximation which is denoted by the curve S in Fig. 2.10. This is achieved by removing the sharp corner in a trace and replace it by a more smoothed, realistic corner, and we call it *corner smoother*. We tested the corner smoother on 16 typical highway divergence scenarios with varying ramp shapes, and are able to get accurate approximation on 14 of them.

2. *Look ahead.* This is an alternative approach to solve the above problem. If a node notices a possible divergence from the most recent segment, instead of declaring divergence immediately, it looks ahead $n$ segments in the record and checks whether it is approaching one of these segments. A drawback in this method, however, is that it may pose additional computational overhead.

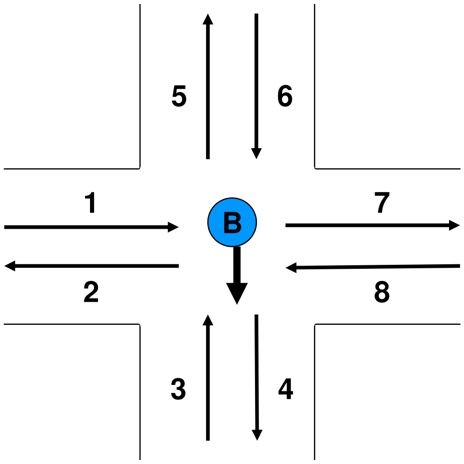


Figure 2.11: 8 possible directions for a car at a crossing.

## 2.5 Return Probability Analysis in Manhattan Grid

In this section, we analyze the performance of Geocache anchoring protocols in terms of return probability using a Manhattan grid topology. The return probability represents the likelihood for the protocol to return the Geocache back to the anchor location.

To simplify the analysis, we make the following assumptions: (i) the nodes are uniformly distributed on the roadways, (ii) grid blocks are of the same unit-size length, (iii) the radio range for all nodes are the same, which is also the unit-size, so that at any time only one intersection is under the radio range's coverage, (iv) the probability for a node to turn left, right, or move straight at an intersection is equally set to $p_t = \frac{1}{3}$ (assuming no U-turns), and (v) in system implementation, if the first handoff is not successful (due to low node density, etc), we allow further handoff attempts after a certain time interval. However, to simplify the analysis, for both protocols we only consider a single attempt for each handoff.

We note that we use the above assumptions to simplify the analysis, and we can apply the results derived from the analysis to other situations as well.

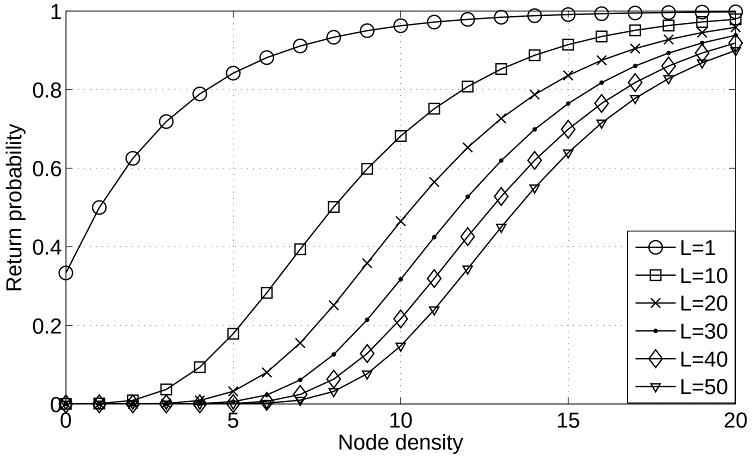


Figure 2.12: Return probability for prioritized RevTraj with varying distance and node density.

### 2.5.1 Return Probability Analysis for RevTraj

In this subsection, we discuss the Geocache return probability if carriers are chosen based on a recorded trajectory such as in RevTraj.

First, we calculate the probability that a suitable carrier node is available (within the transmission range) when handoff occurs. Due to assumption (iv), the probability for one node at a four-way intersection to follow a fixed trajectory is $\frac{1}{4}$ (as shown in Fig. 2.11, the probability that the node is already on the trajectory with the correct direction, which is $\frac{1}{8}$, plus the probability that the node will turn to the trajectory, which is $\frac{3}{8} \times \frac{1}{3}$). Therefore, the probability to find at least one node on the trajectory given $d$ nodes available in radio range is $p_h = 1 - (1 - \frac{1}{4})^d$.

Next, we calculate the return probability for a complete trajectory with length $L$. By length $L$, we mean there are $L$ remaining segments on the trajectory other than the segment the node is currently on. Therefore, for a path with length $L$, there are $L$ remaining intersections to the anchor location. At each intersection, the node either follows the trajectory with probability $p_t$, or needs to hand off with probability 1-$p_t$. Therefore, in RevTraj, the return probability $P_{return}$ with path length $L$ is defined as:

$$P_{return} = [p_t + (1 - p_t)p_h]^L. \tag{2.4}$$

Fig. 2.12 plots the return probability for prioritized RevTraj. We vary the distance $L$ from

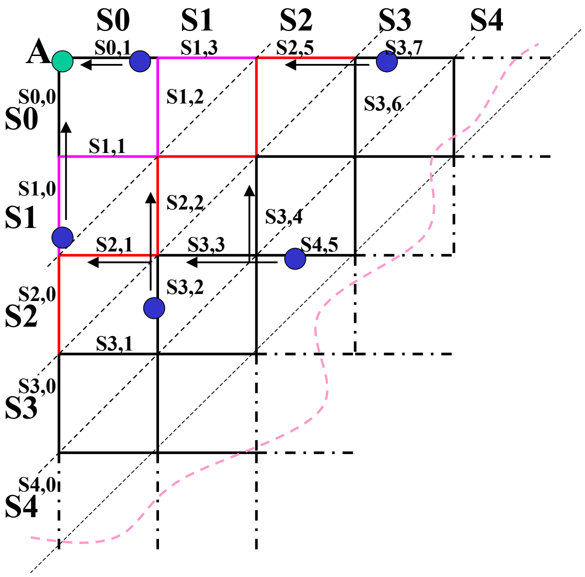


Figure 2.13: Dividing grid into groups of segments.

$1$ to $50$, and the results show that the return probably increases when the node density increases, and gradually approaches $1$ as the node density reaches a certain level.

### 2.5.2 Return Probability Analysis for MaxProgress

In this subsection, we discuss the Geocache return probability if carriers are chosen according to its distance to the anchor location such as in MaxProgress.

We start by labeling the road segments on a grid. Fig. 2.13 depicts a fully-connected grid with $A$ as the anchor location. First, we divide the segments into different sets $S_i$ based on their distance to the anchor location. In Fig. 2.13, we are showing 5 sets, $S_0$ to $S_4$. The $i$th set $S_i$ contains $2i + 2$ segments. We label each segment as $s_{i,j}$, where $i$ is its set number, and $j$ ($0 \leq j \leq 2i + 1$) is the segment's index number within set $S_i$. In Fig. 2.13, the segments within the same set are numbered from lower left to upper right.

Next, we compute the return probability $P_{i,j}$ for segment $s_{i,j}$, which is the Geocache's return probability when it's currently located on segment $s_{i,j}$. We distinguish four classes of segments: (1) segments adjacent to $A$ (in $S_0$), (2) segments on or adjacent to the left vertical edges where $j = 0$ or $j = 1$, (3) segments on or adjacent to the upper horizontal edges where $j = 2i$ or $j = 2i + 1$, and

(4) all remaining segments.

According to the protocol, handoff will occur as long as the distance between the carrier and the anchor location starts to increase, suggesting the carrier diverges from the shortest path to the anchor location. As shown in Fig. 2.13, for the two segments connecting the anchor location in case (1), their return probability is $1$. In cases (2) and (3), there is only one choice to remain on the shortest path (e.g., for $s_{2,0}$, the immediate next segment along the shortest path to $A$ is via $s_{1,0}$). The node may turn into the shortest paths itself, or find a node that is or will be on the shortest path with probability $k_1 = 1 - (\frac{3}{4})^d$. In case (4), for each segment, there are two choices of shortest path, so similarly, the probability $k_2$ for the Geocache to remain on the shortest path after the intersection is $k_2 = 1 - (\frac{1}{2})^d$. Therefore, we give our recursive equation for the return probability in Eq. 2.5. $P_{i,j}$ equals to the probability for the Geocache to get onto the shortest path(s), times the return probability of the chosen shortest path(s).

### 2.5.3 Return Probability Analysis in Partially-connected Grids

After analyzing the return probability in a fully-connected grid, we next look at the return probability for MaxProgress in partially-connected situations, which are more common in real life.

First, we define a *successful path* as a shortest path that can lead to the anchor location from the hand off location in a partially-connected grid (e.g., the path$\{s_{2,3}, s_{1,1}, s_{0,0}\}$ from $B$ to $A$ in Fig.2.14). We can always generate a partially-connected grid with smaller $\rho$ by removing segments from a fully-connected grid. For instance, the incomplete grid shown in Fig. 2.14 is generated by removing segments ($s_{0,1}$ and $s_{1,3}$) from a full grid, and the value of $\rho$ according to Eq. 2.6 is $\frac{1}{3}$.

$$P_{i,j} = \begin{cases} 1 & \text{for } i = 0; \\ (\frac{1}{3} + \frac{2}{3}k_1)P_{i-1,0} & \text{for } j \in [0,1]; \\ (\frac{1}{3} + \frac{2}{3}k_1)P_{i-1,2i-1} & \text{for } j \in [2i, 2i+1]; \\ (\frac{1}{3} + \frac{1}{6}k_2)(P_{i-1,j-2} + P_{i-1,j-1}) & \text{otherwise,} \\ & \text{when } j\text{'s odd;} \\ (\frac{1}{3} + \frac{1}{6}k_2)(P_{i-1,j} + P_{i-1,j-1}) & \text{otherwise,} \\ & \text{when } j\text{'s even.} \end{cases} \tag{2.5}$$

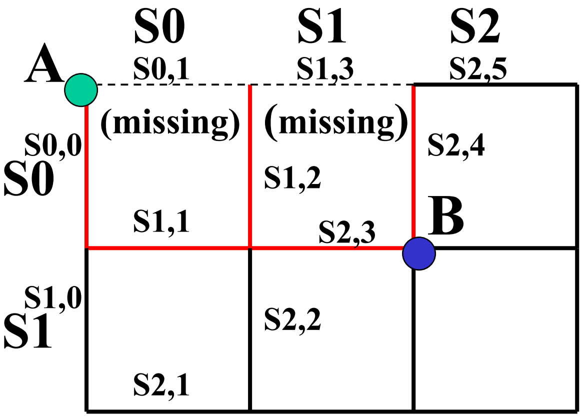


Figure 2.14: An example of partially-connected Grid.

Next, we introduce a parameter

$$\rho = \frac{N_{success}}{N_{total}}, \tag{2.6}$$

where $N_{success}$ is the number of successful paths in the partially-connected grid, and $N_{total}$ is the total number of successful paths in the otherwise fully-connected grid. For a fully connected grid, like the one in Fig. 2.13, we have $\rho = 1$.

To apply Eq. 2.5 on the example shown in Fig. 2.14, we assign 0 to $P_{0,1}$ and $P_{1,3}$ to invalidate the two missing segments. Therefore, assuming $P_t = \frac{1}{3}$, with $A$ as anchor location and $B$ as handoff location, we have the return probability in the partially-connected grid as:

$$P = (\frac{1}{3} + \frac{1}{6}k_2)(\frac{1}{3} + \frac{1}{6}k_2)(\frac{1}{3} + \frac{2}{3}k_1), \tag{2.7}$$

while the return probability from B to A in an otherwise fully-connected grid is:

$$P_{complete} = (\frac{1}{3} + \frac{1}{6}k_2)(\frac{1}{3} + \frac{2}{3}k_1)(1 + \frac{2}{3}k_1 + \frac{1}{3}k_2). \tag{2.8}$$

Therefore we have

$$\frac{P}{P_{complete}} = \frac{\frac{1}{3} + \frac{1}{6}k_2}{1 + \frac{2}{3}k_1 + \frac{1}{3}k2}. \tag{2.9}$$

The equation $\frac{P}{P_{complete}}$ achieves its maximum $\frac{1}{3}$ when both $k_1$ and $k_2$ are equal to 0. That is, $\frac{P}{P_{complete}} \leq \frac{1}{3}$. Note that for this particular topology, $\rho$ also equals to $\frac{1}{3}$. In fact, we have examined numerous grid-based road topologies and we are always able to observe $\frac{P}{P_{complete}} \leq \rho$ for the cases we have studied.

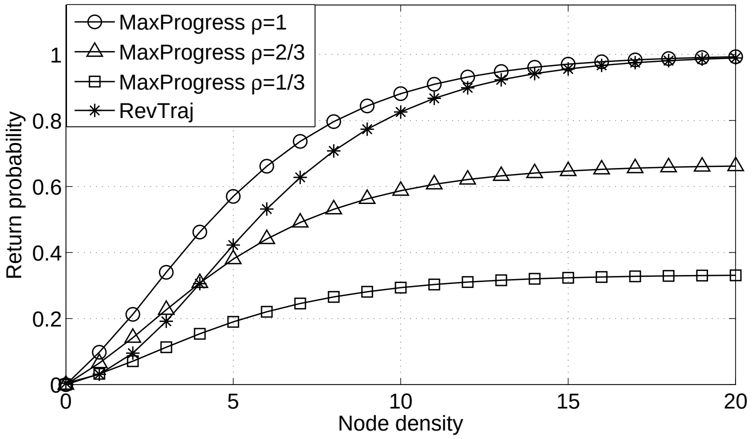


Figure 2.15: Comparing MaxProgress and RevPath in grids with different connectivity level.

This is a powerful observation, and it also confirms our hypothesis about distance-based approaches such as MaxProgress: the return probability will degrade when the connectivity of the road network decreases. However, in the trajectory-based approach, the return probability actually increases when removing segments from the fully-connected grid according to Eq.2.4. In the real world, the road topology approximating a fully-connected grid is very rare, but rather most of the road maps have medium to low $\rho$ values, which may severely degrade the return probability using MaxProgress.

Fig. 2.15 compares the return probability of RevPath with MaxProgress using Eq. 2.4 and Eq. 2.5. $L$ is fixed to $L = 5$. For MaxProgress, we show the Geocache return probability with varying $\rho$ value. For RevPath, we shown its minimum return probability value when $\rho = 1$. We observe that for MaxProgress, the return probability degrades significantly when $\rho$ decreases, while the RevPath outperforms MaxProgress with medium and low $\rho$ values, which corresponds to typical road topologies in the real world.

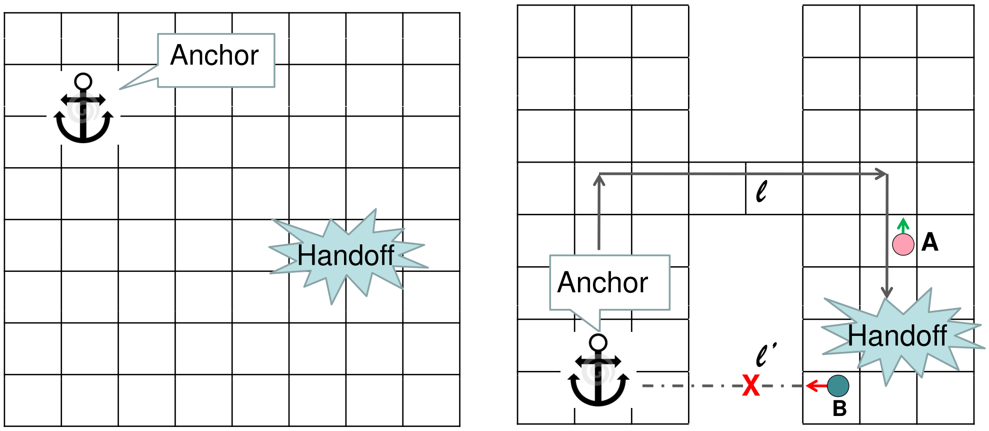
## 2.6 Performance Evaluation

In this section, we study the performance of Geocache anchoring protocol through simulation. We measure the return probability of the Geocache when varying the vehicular density and the connectivity of the road map.

### 2.6.1 Effect of Road map Connectivity

As we have discussed in the previous section, intuitively, the distance-based MaxProgress works better under fully or mostly connected road map topologies. We capture the connectivity characteristic using parameter $\rho$ give in Eq. 2.6. A well-connected city road map such as Manhattan's is a good example for road topologies with large $\rho$ value. Most other areas, however, do not have this property. In partially-connected areas, we expect MaxProgress to exhibit suboptimal performance. An example is given in Fig. 2.16(b). The Geocache was handed off after traveling along a trajectory $l$. When choosing the next carrier, MaxProgress always favors those that are physically closer to the anchor location, which is $B$ in this case. But the nature of the road map makes this choice a bad decision, because the seemingly shortest path $l'$ does not exist in the actual road map due to low connectivity, whereas the longer alternative $l$ is the only feasible path leading back to the anchor location. On the other hand, according to RevTraj, node $A$ who's following the trajectory $l$ will be chosen as the next carrier. By following the trajectory, no matter how poorly connected the road map is, we are always confident that the trajectory can lead to the anchor location. A trade-off is the probability for RevTraj to find a carrier is lower than that of MaxProgress. Especially with low node density, RevTraj may suffer from not being able to find a carrier that's exactly on the trajectory.

To verify the above hypothesis, we simulated the performance for the two protocols on the two road maps depicted in Fig. 2.16(a) and (b). Fig. 2.16(a) represents a well-connected city road map with $\rho = 1$, while Fig. 2.16(b) shows a road map with a low $\rho$ value, representing two cities being connected by a major highway. The simulator of choice is *NS-2*, with 802.11 as MAC and PHY layer protocol. The radio range is 250m. At each intersection, the probability for a car to turn left,



(a). Fully-connected grid.

(b). Two-city grid.

Figure 2.16: Two road maps with different $\rho$ values. (a). A fully-connected grid with $\rho = 1$ representing the Manhattan city road map. (b). A partially-connected twin-city grid with low $\rho$ value, representing two cities being connected by a major highway.

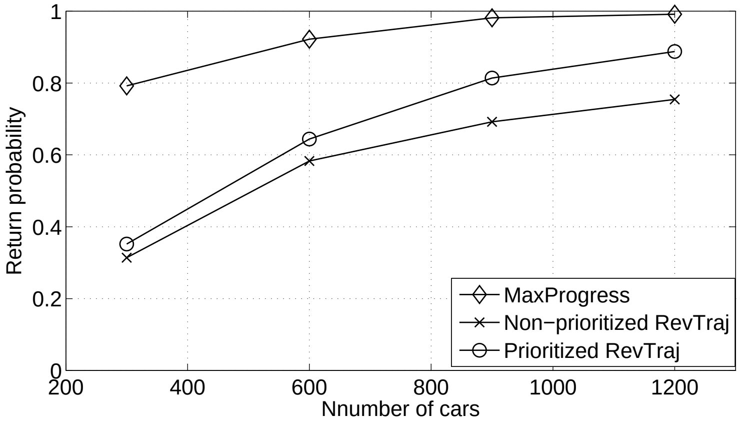
right and go straight is equally $\frac{1}{3}$. The length of each segment is 300m. The vehicular has constant speed of 30m/s. In RevTraj, the number of look ahead segments is 3, meaning we look at the most recent 3 segments when comparing the trajectory. Fig. 2.16 also indicates the approximate anchor location and handoff location for the two topologies.

Fig. 2.17(a) and (b) present the return probability of the two protocols. As expected, in the fully-connected grid topology, MaxProgress outperforms RevTraj. But in the partially-connected grid topology, MaxProgress's return probability degrades severely, while RevTraj shows much better performance. These results strongly confirm our hypothesis and analysis in Section 2.5.

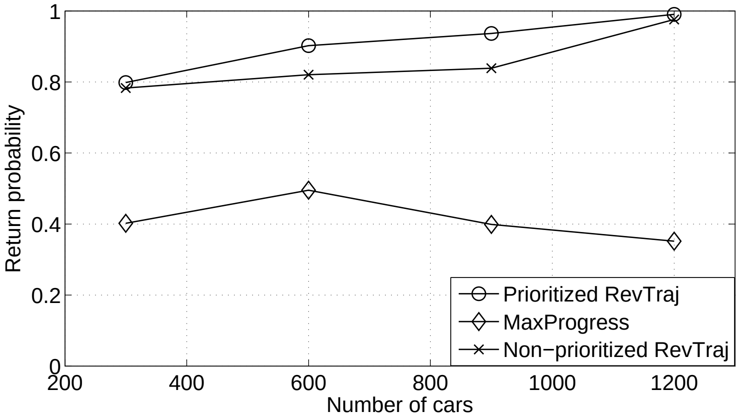
Between prioritized-RevTraj and non-prioritized-RevTraj, the former performs better than the latter. Finally, we also point out that the performance of RevTraj will significantly improve with the increase of the node density because the probability of finding the next carrier on the recorded trajectory will be higher with a larger node density.

### 2.6.2 Evaluating Anchoring Protocols Using Real Road Traces

After studying the example scenarios, we next investigate the protocols' performances using a realistic traffic trace collected from the PARAMICS model based on the southern New Jersey traffic



(a). Geocache return probability in fully-connected grid.



(b). Geocache return probability in twin-city grid.

Figure 2.17: Comparing RevTraj with MaxProgress using two road map settings. MaxProgress fares better with $\rho = 1$, but is significantly outperformed by RevTraj under a low $\rho$ value.

topology, as shown in Figure 2.19. The trace contains $984,445$ records collected by $5000$ cars for a duration of $3395$ seconds during the 6am-7am off-peak traffic period. In our simulator, free-space propagation is used as communication model.

Every result we show in this section is averaged over $5000$ simulation runs. In each run, we select a random car's location at a random time as the anchor location, and let that car drive for a period of $T_h$ before handing off the Geocache. We end a simulation run either when a successful return is made or after $T_{end}$ elapses but still no successful return. After completing $5000$ simulation runs, the return probability is then calculated as the ratio of the number of successful returns over $5000$.

The return probability results are shown in Fig. 2.18. Since the node density is fixed in the trace,

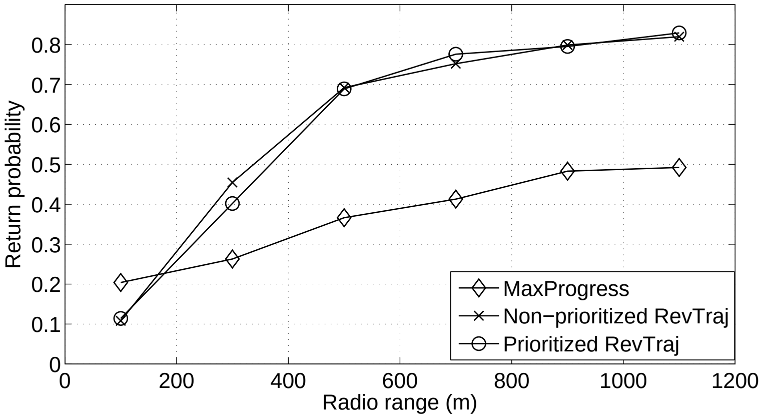


Figure 2.18: The comparison of three anchoring schemes when increasing the radio range. $T_h = 750$ seconds.

we vary the radio range in the experiments to change the number of cars covered in the handoff range, thereby simulating varying node densities. $T_h$ is set as $750$ seconds. We find that except for extremely short radio ranges (100m), corresponding to extremely low node density in our case, RevTraj significantly outperforms MaxProgress, with an improvement of about $70\%$. This suggests that many real-world road maps are of small $\rho$. Finally, we notice that all three schemes benefit from a larger radio range, which in our case, suggests that the anchoring protocols benefit from larger vehicular densities.

### 2.6.3 Adaptive Anchoring Scheme

Through the previous simulation results, the two anchoring protocols exhibit their individual advantages under different circumstances. Therefore, it is natural to develop a protocol which combines the two and take advantage of both their strengths. In our enhanced scheme, we still keep the trajectory to ensure a guaranteed return path to the anchor location. When we can not find suitable carriers via RevTraj, we extend our search by including nodes found using MaxProgress. At all time we keep the trajectory, hoping that later carrier nodes may return back to the trajectory, or at least follow the general direction.

In Fig. 2.20(a), we set $T_h = 1000$ seconds and vary the radio range of the mobile nodes. Results show that across all radio ranges we picked, the adaptive protocol always returns Geocache with

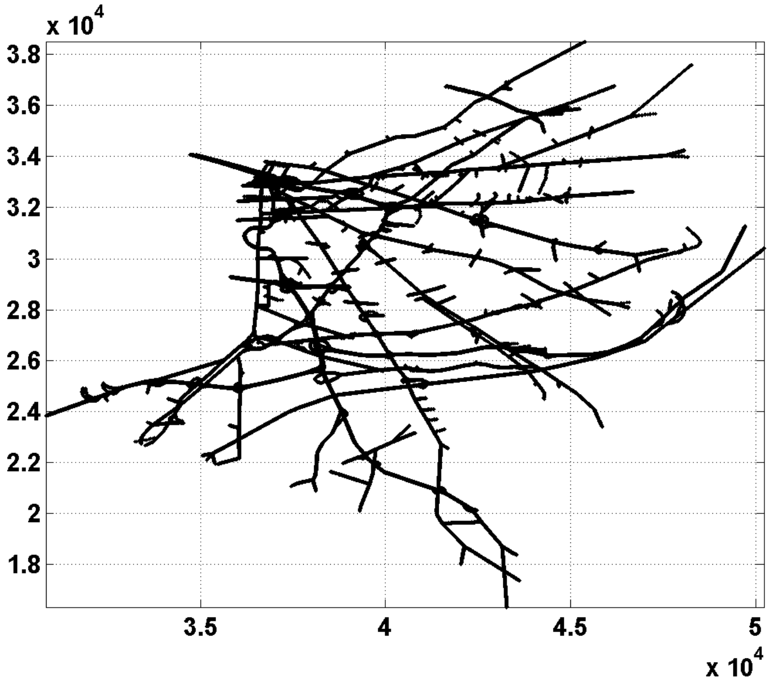
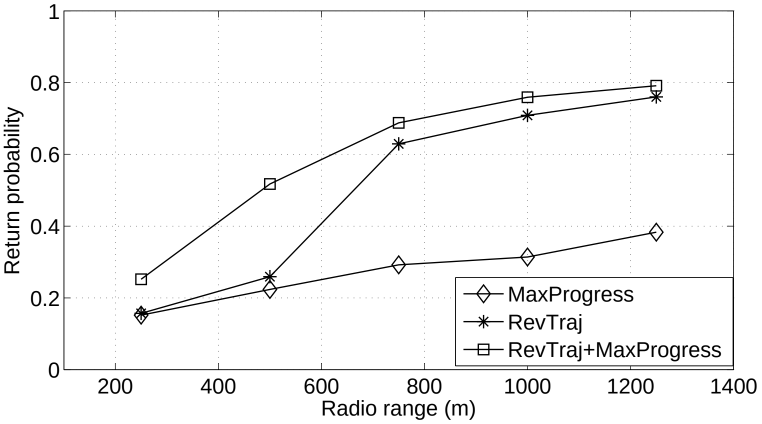
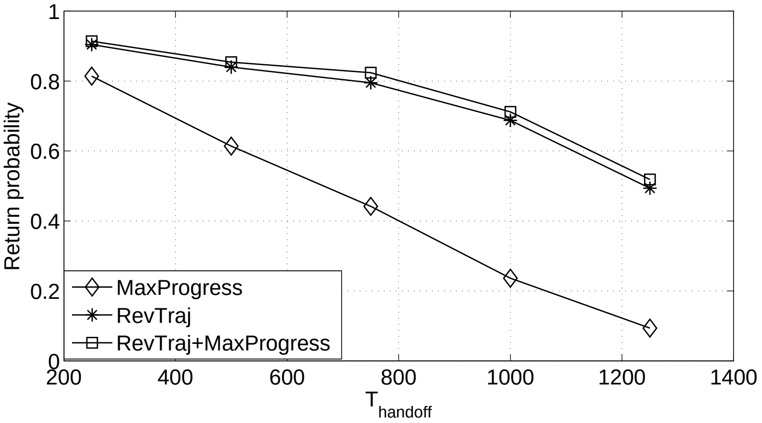


Figure 2.19: We also evaluated the anchoring protocols using real traffic data from south New Jersey. This is the road map for the traffic data.

the highest probability. RevPath exhibits lower return probability when node density is low since it misses many opportunities to conduct a successful return if MaxProgress was used. MaxProgress shows the worst performance among the three. In Fig. 2.20(b), we compared the three protocols with constant radio range of 750m while varying $T_h$. We observe the adaptive approach still achieves the highest return probability. And for all the three protocols, longer hand off time leads to lower return probabilities.



(a). Return probability when varying radio range values with $T_h$ of 1000 seconds.



(b). Return probability when varying $T_h$ with radio range of 500 meters.

Figure 2.20: Performance evaluation for the adaptive anchoring algorithm.

## 2.7 Adaptive Handoff for In-time Anchoring

After studying how to return Geocache to the anchor location in the general setting, we next look at a specific anchoring requirement, *in-time anchoring*. Here, the Geocache is required to return to the anchor location within a specific time interval. A wide range of mobile applications have such requirements. For example, a mobile user may have a query “what is the average car speed around my location now?” He only accepts real-time responses within the next few minutes, and after that, he will lose interest and leave.

The challenge here is two-fold. First, we would like to have in-time returns. Second, we would like the return time to be as close to the expected return time as possible. If the Geocache is returned too early, we need to either keep the Geocache exactly at the anchor location until the deadline, which incurs high communication overhead, or continue to boomerang the Geocache, with the risk of late returns.

To meet this challenge, we need to carefully control the initial handoff time $T_h$ to ensure the Geocache’s timely return. In this section, we discuss two such control algorithms.

### 2.7.1 Addictive Adjust Multiplicative Decrease Handoff Policy

The first protocol is called **A**ddictive **A**djust **M**ultiplicative **D**ecrease (**AAMD**). Basically, we control $T_h$ based on the observed Geocache return time $T_r$. Intuitively, $T_h$ should be increased when the previous $T_r$ is well below the expected return time $T_{ERT}$, and should be decreased when the previous $T_r$ approaches or even exceeds $T_{ERT}$. We introduce a threshold rate $\beta (0 < \beta < 1)$, and we increase $T_h$ by $\Delta$ if $T_r < \beta T_{ERT}$, and decrease $T_h$ by $\Delta$ if $T_r \geq \beta T_{ERT}$. Here $\Delta$ is usually a small value compared to $T_{ERT}$, e.g., $\Delta = \frac{T_{ERT}}{20}$.

When we have a late return ($T_r > T_{ERT}$), $T_h$ is reduced by half in order to quickly get back to in-time returns. Therefore, the complete policy for AAMD is:

$$T_h = \begin{cases} T_h + \Delta, & \text{if } 0 < T_r < \beta T_{ERT} \\ T_h - \Delta, & \text{if } \beta T_{ERT} \le T_r \le T_{ERT} \\ \frac{T_h}{2}, & \text{if } T_r > T_{ERT} \end{cases} \tag{2.10}$$

### 2.7.2 Q-Handoff: Q-Learning-based Handoff Policy

AAMD has two problems. First, it is very conservative by cutting $T_h$ by half when late return occurs, which will lead to a slow start for $T_h$ and therefore excessive communication overhead. Second, it does not take history information into consideration. We address these shortcomings in our second policy. Here, we first build an MDP (Markov decision process) model for the handoff adjustment scenario, then train the policy using a reinforcement learning approach: Q-learning. We also introduce an adaptive algorithm to adjust the *setPoint* which is used to reset $T_h$ in case of late returns. We call this handoff policy *Q-Handoff.*

#### MDP Model for Q-Handoff

Q-learning learns the expected utility of taking a given action in a given state, whose quality is represented by the function: $Q(s, a)$. Formally, the basic reinforcement learning model, as applied to MDPs, consists of:

- a set of environmental states $S$;
- a set of possible actions $A$;
- a set of scalar rewards $R$.

Therefore, we give the MDP model according to our problem scope:

- $S = (s_0, s_1, s_2, s_3, s_4, s_5)$;

  These states represent different Geocache return situations. $s_0$ represents late returns with $T_r > T_{ERT}$, while states $s_i$ (i≠0) represent different in-time return situations where $\frac{(i-1)T_{ERT}}{5} < T_r \le \frac{iT_{ERT}}{5}$. For example, if we have $T_r = 0.35T_{ERT}$, then the state is $s_2$.

- $A = (a_0, a_1, a_2, a_3)$;

  These actions define how to adjust the value of $T_h$. Specifically,

  - $a_0$: reset $T_h$;
  - $a_1$: increase $T_h$ by $\Delta$;
  - $a_2$: keep $T_h$ unchanged;
  - $a_3$: decrease $T_h$ by $\Delta$.

Next we define the reward function $r$. Between in-time returns and late returns, the reward function would favor the former. Further, among all the in-time returns, those that are closer to the expected return time are preferred. Therefore, we define the following reward function:

$$r = \mathbb{1}_{T_r < T_{ERT}} - \frac{|T_r - T_{ERT}|}{T_{ERT}}, \tag{2.11}$$

let $\mathbb{1}_{T_r < T_{ERT}}$ be the indicator that takes value 1 if $T_r < T_{ERT}$ and 0 otherwise.

We use the quality function $Q$

$$Q : S \times A \rightarrow \mathbb{R}$$

to calculate the score of an action in a certain state. At the beginning, $Q(s_i, a_j)$ is initialized with random values, and then gets updated iteratively by using:

$$Q(s_i, a_j) = (1 - \alpha)Q(s_i, a_j) + \alpha[r + \gamma \max_{a \in A} Q(s', a)], \tag{2.12}$$

where $r$ is the immediate reward defined by Eq. 2.11, and $s'$ is the next state. The first term evaluates the importance of the history information, where the learning rate $\alpha (0 < \alpha \leq 1)$ determines to what extent the recent information will override the old information. The second term in the equation evaluates the immediate reward and the expected future reward, where $\gamma (0 < \gamma \leq 1)$ is the discount factor.

Therefore, we have our $\epsilon$-greedy Q-Handoff policy:

**Policy:** At each state $s_i$, the agent picks the action $a = \text{argmax}_a\, Q(s_i, a)$ with probability 1-$\epsilon$, and picks a random action with probability $\epsilon$.

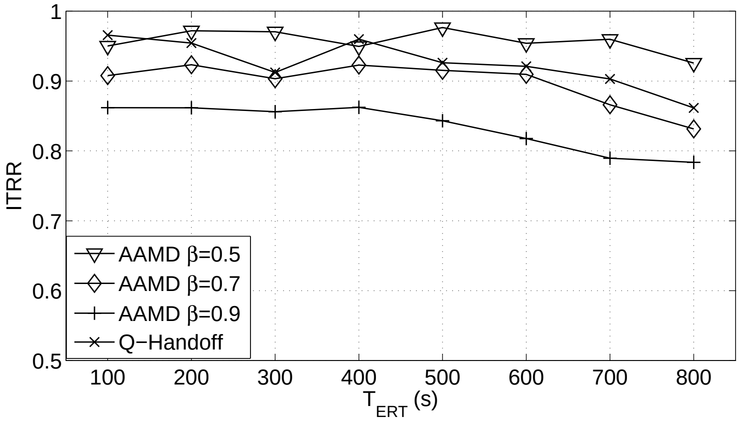


Figure 2.21: In-time return probability for AAMD with various $\beta$ values and Q-Handoff Policy.

Using this policy, we guarantee that most of the time we act greedily, selecting actions that lead to the greatest reward. Occasionally, we select random actions to explore unknown space which may potentially lead to even higher rewards. In machine learning, $\epsilon$-greedy policies are commonly used to deal with the exploit-explore dilemma.

#### SetPoint Adjustment

In Q-Handoff, when a late return occurs, $T_h$ is reset to a *setPoint* value (action $a_0$). A ideal *setPoint* should have "good record", meaning previous $T_h$ values around the *setPoint* have resulted in a large number of successful in-time returns. A practical approach here is to save all the successful $T_h$ values but give more weight to recent ones, since recent $T_h$, especially those a few iterations before a reset usually yield in-time returns closer to the expected return time. Therefore, we use an Exponentially Weighted Moving Average (EWMA) to update the *setPoint* as the following:

$$setPoint = \theta T_h + (1 - \theta) setPoint. \tag{2.13}$$

### 2.7.3 Performance Evaluation

We evaluate the two policies using NS-2, with 802.11 as MAC and PHY layer protocol. We use the same southern New Jersey traffic trace as in Section 2.6, with $T_{ERT}$ ranging from 100s to 800s.

For each $T_{ERT}$, we run the simulation for $\frac{T_{ERT}}{10}$ times. For example, with $T_{ERT} = 100$s, we run the trace for 10 times. This is because for the constant-sized trace file (which covers approximately 5000s), large $T_{ERT}$ values will lead to fewer simulation results (number of results $\approx \frac{5000}{T_{ERT}}$). By varying the number of runs for different $T_{ERT}$, we can guarantee sufficient simulation results for each $T_{ERT}$ value.

In AAMD, we choose $\beta$ from $(0.5, 0.7, 0.9)$. In the $\epsilon$-greedy Q-Handoff policy, we set the learning rate $\alpha$ as $0.1$, the smoothing factor $\theta$ for EWMA as $0.1$, and $\epsilon$ as $0.01$. For both protocols, the addictive adjustment amount is $\Delta = \frac{T_{ERT}}{20}$.

The first metric we look at is the in-time return rate (ITRR). Fig. 2.21 shows the ITRR for Q-Handoff and AAMD with different $\beta$ values. For AAMD, smaller $\beta$ values yield earlier returns, thus higher ITRR, and the average ITRR ranges from $83.4\%(\beta = 0.9)$ to $95.73\%(\beta = 0.5)$. Using Q-Handoff, we can achieve an average ITRR of $92.5\%$.

Next we compare the average return time of the different policies. The metric we use here is the ratio of the average return time to the expected return time. It is an important metric because it reflects an algorithm's ability to adjust the handoff time to meet certain return time constraints. Further, small return time usually indicates frequent handoff, and thus higher communication overhead. Fig. 2.22 shows that Q-Handoff yields closer-to-expectation returns. Its average ratio is $0.60$, which is $20\%$ better than the best ratio of AAMD when $\beta = 0.7$.

To further investigate the adaptive feature of the handoff policies, we look at a third metric: $|T_r - T_{ERT}|$, which measures the average distance of $T_r$ from $T_{ERT}$ for both in-time and late return scenarios. Intuitively, smaller distance indicates the average return time is closer to $T_{ERT}$. This metric is especially useful if the system has certain level of delay tolerance, so that slightly late returns are still acceptable. In Fig. 2.23, the y-axis is the normalized difference $\frac{|T_r - T_{ERT}|}{T_{ERT}}$. Using Q-Handoff policy, the average normalized difference is $0.407$, which is a $15\%$ improvement over the best result using AAMD ($\beta = 0.7$).

In summary, compared with the AAMD handoff policies, Q-Handoff is able to achieve high in-time return probability, as well as average return time closer to expectation. Further, due to its

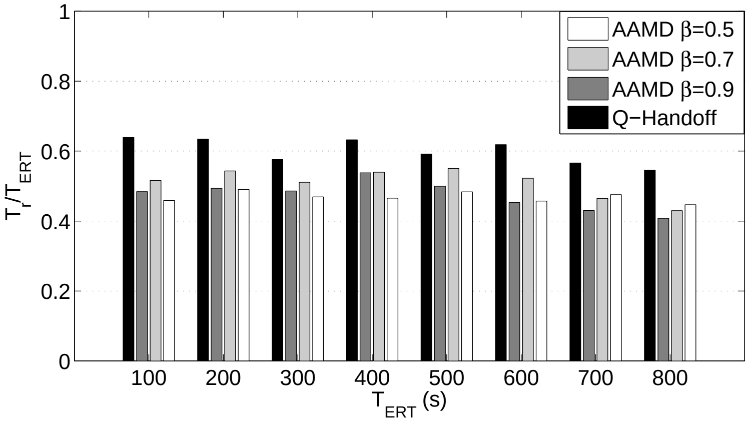


Figure 2.22: Return time ratio for AAMD with various $\beta$ values and Q-Handoff Policy.

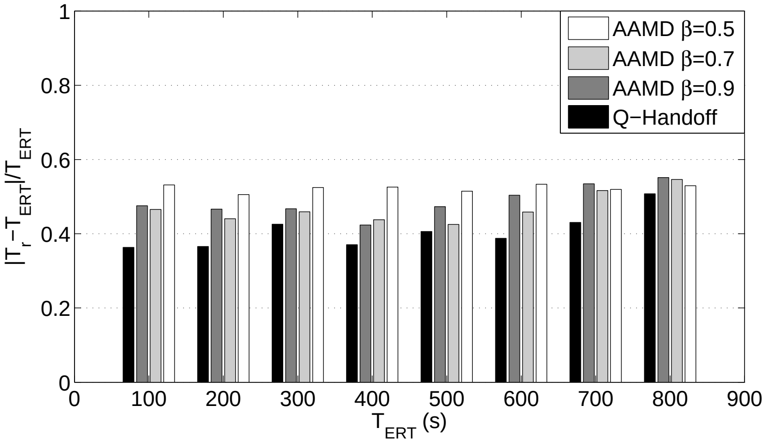


Figure 2.23: Normalized distance from expectation for AAMD with various $\beta$ values and Q-Handoff Policy.

on-line training feature, Q-Handoff policy can better adapt to the environment.

## 2.8 Related Work

This work spans the fields of mobile sensor networks and vehicular networks. Perhaps closest in spirit to the Geocache programming abstraction are geographic hash tables [17] and Tags[18], which provide a programming interface for data-centric storage in stationary sensor networks. Spatialviews [19] provides location-oriented programming language abstractions for mobile ad hoc networks, to ease application development and maintenance. This work does not address distribution of information at the protocol level, which is a key focus of this chapter.

### Mobile sensor networks

Recent works in mobile sensor networks exploit mobility when it is not feasible to build a dense network of fixed sensors. Notably, Zebranet [11] places sensors on zebras to collect valuable zoology data. In under water sensor network [20], mobile nodes are robots that collect data from regions of interest. Several projects target specifically at vehicular sensing. CarTel [12], for example, is a comprehensive distributed mobile computing system used to collect, process and visualize data from sensors located on mobile units. It aims at exploring in-network computing on individual mobile units, as we do, but it does not use inter-vehicle communication, which in our project, is a main focus to enable distributed aggregation of sensor readings from multiple cars. Another vehicular sensor network: MobEyes [13, 21], introduces MDHP (MobEyes Diffusion/Harvesting Processor), a protocol used to spread information within wireless sensor networks and build low-cost index of mobile storage. Although our projects bear similarities in that we also aim to develop low-cost yet efficient inter-vehicle communication protocol, MobEyes relies largely on an opportunistically broadcast approach, possibly with the emphasis of simplified protocol, while we aim at minimizing traffic overhead by the cost of more sophisticated schemes. In VEDAS [22], the authors concentrate on mobile and distributed data stream mining system that allows real time vehicle-health monitoring and driver characterization, instead of addressing inter-vehicle communication. TrafficView [23] exploits inter-vehicle communication. It presents a specific application dedicated to monitoring automotive traffic congestion, while in our project, we aim to design an application-independent platform to enable collaborative sensor-driven applications on vehicles. Our project differs in the way data is collected: we do not deploy nodes to send data to a centralized server or a dedicated mobile node, but keep the data to where they are generated using nodes that are passing the location.

### Inter-vehicle, geographic, and delay-tolerant communication

Many projects have addressed scalable communication in mobile ad hoc networks (e.g., [24]), in sparse or disconnected mobile ad hoc networks (e.g., [25, 26, 27, 28, 29, 30]), or through Infostations. In [31], the authors introduce Infostations to deliver data to mobile nodes. In [32, 33], the authors aim at providing location-specific information to mobile devices, in which they developed schemes for detecting and transferring information of interest. All of these techniques adopt a server-client approach, but in our case, the information is provided by mobiles that have passed the location. The MaxProp [26] routing protocol is used to ensure effective routing of DTN (disruption-tolerant networks) messages via intermittently connected nodes. These protocols are based on different communication workloads, such as unicast between randomly chosen nodes, or multicast to random node sets. These techniques focus on delivering messages to certain nodes, while our protocols try to keep information around a certain location. In [25], designated mobile nodes (message ferries) store and carry messages. Our project differs in that virtually all nodes are “peers”. In [27], the authors aim to guarantee message transmission in minimal time, at the expense of additional messaging overhead. Instead, our applications are more delay tolerant, and the main goal is to reduce communication overhead. In [34], the authors propose the concept of ad-hoc peer-to-peer (p2p) network based on grouping. It also discusses the communication among vehicles in opposite directions, but again our work is based on different application scenarios.

Geocast protocols [35, 36, 37] transmit messages to a predefined geographical region. They are suitable for location-based services such as position-based advertising and publish-and-subscribe. Repeated geocasts or time stable geocasts [38][39] could also be used to maintain Geocache in a certain area and bear similarities to our baseline scheme. It is different in concept though in that it requires the definition of a geographic region, which is not needed in our case. Most geocast schemes concentrate on routing messages to the areas of interest, or distributing messages to all nodes [35, 37], while Geocache is established close to the anchor location and needs only be known to very few nodes. Further, time-stable geocasts continuously remain in the region of interest, while Geocache can travel away from the anchor location.

In [40], it mentions some trajectory concepts, but it fails to take into account the peculiarities of vehicular networks and still only forwards data to a node that is physically closer to the destination. Geopps [41, 42] are maybe the most similar works to ours, however, it requires each mobile node to have full topology information which is not feasible in realistic scenario.

## 2.9 Conclusions

We presented the trajectory-based boomerang protocol to periodically make available data at certain geographic locations in a highly mobile vehicular network. The boomerang protocol returns the Geocache through nodes traveling toward the anchor location. To increase the probability of successful return, it records a node's trajectory while moving away from the anchor location then select nodes to return the Geocache based on the trajectory (RevTraj). We compared this scheme with a shortest-distance georouting scheme MaxProgress, and demonstrated that our scheme significantly outperforms its counterpart in realistic traffic simulation, with a return probability improvement of up to $70\%$. We also extend the boomerang protocol to satisfy more stringent anchoring requirements, such as returning the Geocache within specified time limits. This is achieved through adapting the initial handoff time based on the return time history.

# Chapter 3

# Improving Access Point Association Protocols Through Channel Utilization and Adaptive Probing

## 3.1 Introduction

The high speed, low cost, and wide availability of WiFi is making it an important player in providing Internet access to places like airports, hotels, sports venues and college campuses. With an ever-increasing number of “hotspots” being WiFi-enabled, a wide range of new applications and wireless usage patterns are looming the surface, ranging from mobile video and VoIP [43][44], to location services, social networking and pervasive applications. This explosion in demand for connectivity will require that efficient and effective methods be used to optimally manage connectivity for nearby wireless networks.

The widespread WiFi availability presents many choices to a wireless user in terms of which AP it can associate to [2][3][4][5]. Current commercial association schemes are mostly based on received signal strength indicator (RSSI) measurements or consecutive beacons lost, and perform poorly in many situations because they overlook the load and bandwidth of the AP. For example, suppose a mobile is running an on-line video application. Even though the signal quality is good (e.g. the mobile is physically close to the AP), the AP may be heavily loaded, and cannot satisfy the bandwidth requirement of the application. According to the signal-based approach, the mobile will stay with this AP, receiving a much degraded service.

There have been alternative proposals to address these shortcomings, such as the schemes in [6, 7][8][9][10]. However, most of these methods are either centralized or need special features from the APs. For example, they use a designated server or the APs to collect and analyze the bandwidth utilization, and distribute the association decisions throughout the network. As a result, these methods demand significant modifications to the existing infrastructure, therefore making it difficult and unrealistic to apply them.

In this thesis, we propose a distributed association scheme based on the available bandwidth. Most Internet applications have a certain level of bandwidth requirement, but the bandwidth requirement is often overlooked when making association decisions. Instead of choosing the AP with the best signal strength (as in [45]) or the AP with the lightest load (as in [46]), we argue that a wireless node should choose an AP that can provide sufficient bandwidth. If at any time, the available bandwidth of the current AP drops below the required bandwidth level, the node should probe for a better AP and switch its association. The core of this scheme is an efficient and accurate available bandwidth estimation method. The estimation method can give accurate estimates in different scenarios, e.g. when the wireless node is transmitting packets, when the wireless node is receiving packets, and when the wireless node is probing a channel. Our estimation method relies on existing protocol framework, and does not introduce any additional overhead. The simulation results show that the bandwidth-based association scheme can improve the average per-node throughput by 38.4% compared to the signal-strength based scheme, and a factor of 29.1% by the load-based scheme.

We also identify the two potential problems for our bandwidth-based association scheme (the problem can also occur to any association scheme). One is over-probing: a node may decide to probe other APs when no better AP is available. Over-probing can adversely affect the performance since the node cannot transmit packets during the probing delay. We propose to address this problem by enhancing our scheme with reinforced learning so that a node only probes when the expected gain is reasonable. The other problem is over-switching: a set of nodes may decide to switch to the same AP at the same time. Over-switching may lead to the *thrashing effect*, and we propose to address this problem by delayed switching or probabilistic switching. The simulations show that these enhancements can effectively alleviate the problems.

Our contribution is summarized as the following:

- We develop multiple schemes for estimation bandwidth in the different stages of the protocol, which will be utilized by the clients to make association decisions for better load balancing.
- We propose strategies for the clients to make decisions incorporating bandwidth requirements information, which improves the association efficiency.
- We design two probabilistic switching schemes to alleviate the ”thrashing” problem and improve the communication efficiency.

The rest of the thesis is organized as follows. Section 3.2 discusses the background of the work, and sets up the stage for our association scheme. After presenting the bandwidth estimation method in Section 3.3, we explain our association scheme in section 3.4. We compare the performance of three association schemes and present the simulation results in section 3.5. In section 4.1, we discuss our enhancement through reinforced learning to prevent the over-probing problem, and demonstrate its effectiveness through simulations. We summarize the related work in section 3.7 and the concluding remarks in section 3.8.

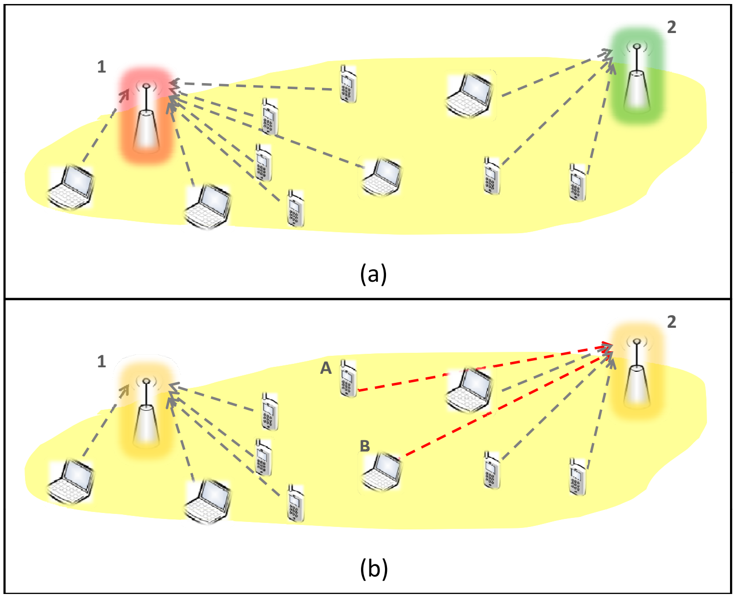


Figure 3.1: An example scenario with 2 APs and 10 mobiles. In (a), we show an unbalanced situation caused by legacy 802.11 association protocols, and in (b), we show a more balanced association situation.

## 3.2 Overview of the Bandwidth-based Association Scheme

### 3.2.1 Example Scenario

In Fig. 3.1 is the scenario picture with 2 APs and several mobile devices. Both APs can be reached by all the mobile nodes. According to the legacy association protocol, nodes make their association decisions only based on the received signal strength from the APs. Therefore, as indicated in Fig. 3.1(a), 7 out of 10 nodes are associated with AP 1, causing load imbalance between the 2 APs. A better approach would be for the devices to take into consideration the load information of the APs. As shown in Fig. 3.1(b), if nodes A and B switch to AP2, the load on the 2 APs will become more balanced. Several schemes have been proposed to achieve this purpose. For example, in [6], admission control and load-balancing algorithms are implemented in the AP to collect the state information of the network, such as available capacity in each cell, number of users per cell, etc. In [7], a network operation center (NOC) is employed to make and distribute association decisions, as well as balance load across all the APs. Both of the above methods, however, need

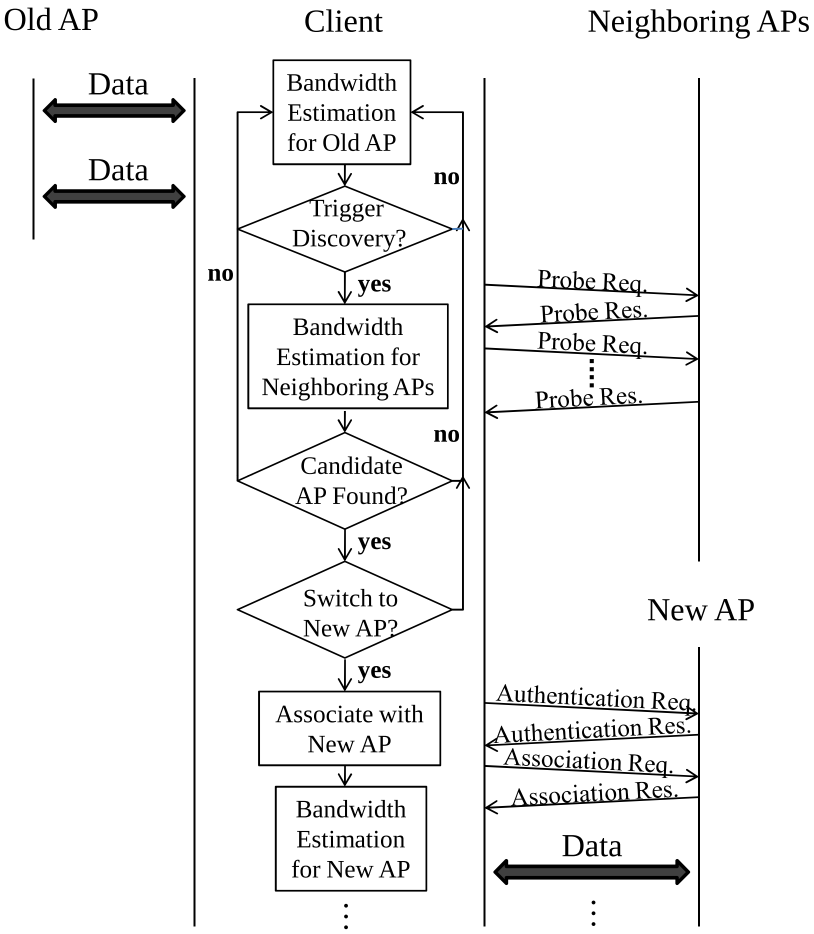


Figure 3.2: The complete association protocol frame work.

significant modifications to the existing infrastructure by either adding extra components into the network or revising the communication protocol on the APs. In this thesis, we design our distributed association management protocol to achieve better load balancing implicitly, by having the clients independently evaluate bandwidth estimation.

### 3.2.2 Protocol framework

We design an AP association scheme based on channel utilization. Improvements are made to all the major stages of the AP association process: First, we discuss the bandwidth estimation methods used to trigger probing and handoff decisions. We then discuss the AP probing protocol used to effectively probe and estimate the bandwidth of available APs. Further, since one concern with any association protocol is the risk of thrashing, we address this by making appropriate modifications to our association protocol.

Fig. 3.2 shows the protocol framework of our association scheme. The design philosophy is to incorporate available bandwidth estimation in different stages of the protocol, and use such information to make association decisions. The protocol starts from when the node is actively exchanging data frames with its current AP. While exchanging data frames, the client constantly monitors the available bandwidth of the AP. If the bandwidth drops below the required level, the client will start probing neighboring APs. The probing protocol already defined in the 802.11 MAC merely uses the received signal strength from APs. In this thesis, we design a bandwidth estimation scheme for the probing phase, to take into consideration available bandwidth at each AP. By the end of probing, the client will choose a candidate AP based on its available bandwidth, load, and signal strength. Then the client will decide whether to switch to that candidate AP according to its policy. If the client decides to switch, it then initiates the association process by exchanging management frames with the new AP and resumes normal data transmissions.

### 3.2.3 Rationale for channel utilization based trigger

Internet applications have widely varying bandwidth requirements. On one end, applications with stringent QoS requirements such as video conferences, video streaming, VOIP and radio broadcasting have high bandwidth requirements, while on the other end, best-effort applications such as email or file transfer have very low bandwidth demands. Bandwidth requirements, however, are not generally taken into consideration when link layer decisions are made. As discussed in the previous section, the triggering schemes used in 802.11 are mostly beacon-based or signal-based, which do not reflect the application's bandwidth requirement, and thus perform poorly in many situations. Further, these schemes may lead to unnecessary reassociation and inflict bad performance on the upper layers as shown below.

To elaborate further, let us consider an example scenario in which a mobile is running the email service and experiencing poor signal quality at the moment. According to the signal-based approach, the mobile should switch its AP. However, considering the application's best-effort nature and its light bandwidth requirement, it may be more beneficial to stay with the same AP as long as the application can function at the current signal level. Also, it may well be the case that the signal quality is only briefly degraded, and thus a switch might actually do more harm (e.g. additional overhead) than good.

Therefore, we believe that it is highly beneficial to consider the application's bandwidth requirement and the available bandwidth provided an AP when making association decisions. Since the bandwidth requirement is already known to the application, we next examine methods for estimating the available bandwidth for an AP.

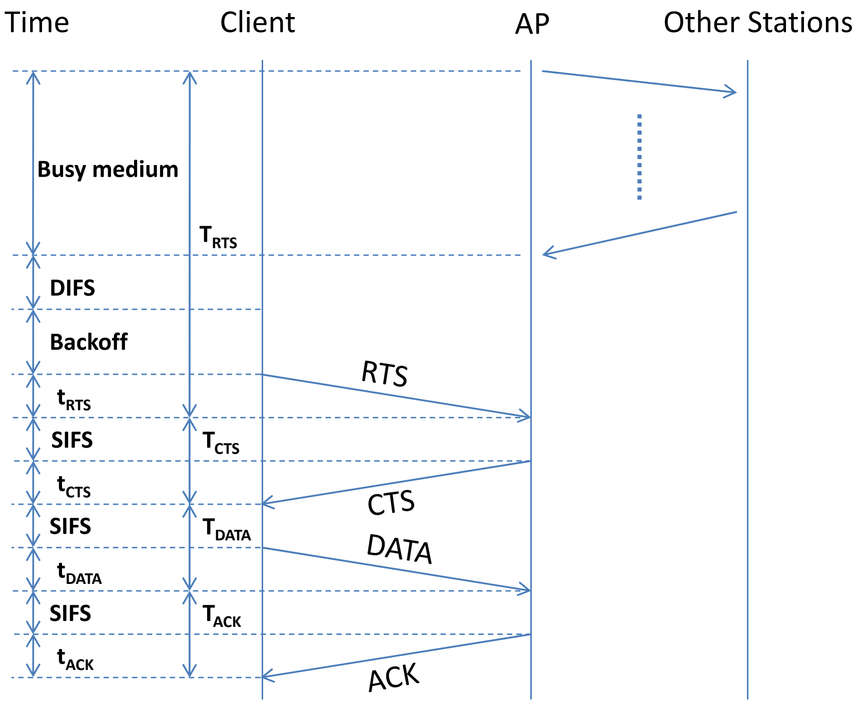


Figure 3.3: Frame Exchange Sequence in 802.11 DCF.

## 3.3 Available Bandwidth Estimation

In this section, we discuss in depth how a node estimates the available bandwidth on a channel in several situations: when the node is transmitting packets, when the node is receiving packets, and when the node is probing a channel. Experiments proved that our method can give accurate estimates in all these situations without introducing any additional overhead.

### 3.3.1 Frame Exchange Sequence in 802.11 DCF

In this thesis, we assume the underlying MAC protocol is 802.11 DCF (distributed coordinated function). 802.11 defines two channel access schemes: DCF (distributed coordinated function) and PCF (point coordination function). As a centrally controlled mechanism, PCF performs poorly in the presence of other nodes using DCF[47], and is not supported by most wireless vendors. Therefore, we mainly focus on evaluating the available bandwidth with DCF. Fig.3.3 illustrates the frame exchange sequence in 802.11 DCF using an uplink example. Nodes that have data packets to transmit must first sense the channel until the channel is idle. Then after a short interval DIFS (Distributed Inter-Frame Sequence), each competing node chooses a random number from [0, $CW$] as its backoff timer, where $CW$ is their individual contention window size. Once the backoff timer reaches zero, the node will transmit the Request-to-Send (RTS) frame first, and after an interval SIFS (Short Inter-Frame Sequence) the receiver responds with a Clear-to-Send (CTS) frame. Other nodes that hear either the RTS or the CTS frame will delay their transmissions until the end of the current frame exchange. Upon receiving the CTS frame, the sender waits for a duration of SIFS and sends its data frame. Finally, the receiver responds with an ACK frame after duration of SIFS. The absence of either a CTS or ACK frame causes timeout and retransmission[1].

We use $T_{RTS}$ to represent the RTS interval, defined as the interval between the time when the RTS frame is placed in the sending buffer and the time when the RTS frame successfully reaches the receiver. The interval consists of: the waiting time when the medium is busy ($t_{busy}$), a DIFS duration, the backoff duration ($t_{backoff}$), and the transmission delay of the RTS frame ($t_{RTS}$). Thus, $T_{RTS}$ is calculated as:

$$T_{RTS} = t_{busy} + DIFS + t_{Backoff} + t_{RTS}.$$

Similarly, we denote the CTS interval $T_{CTS}$, DATA interval $T_{DATA}$ and ACK interval $T_{ACK}$ as:

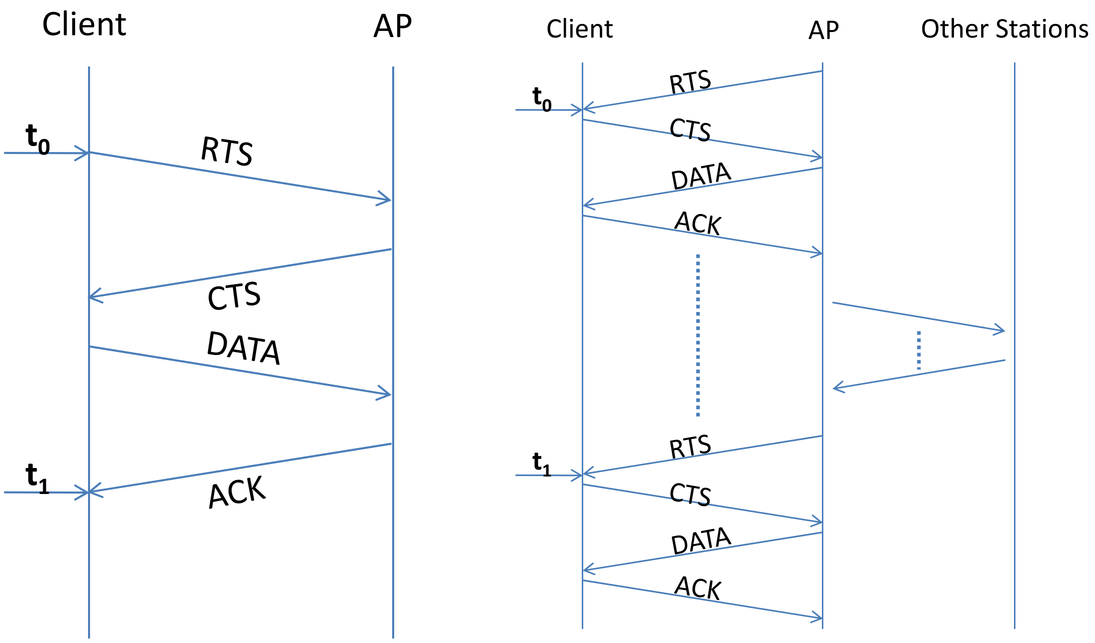
$$\begin{aligned} T_{CTS} &= t_{CTS} + SIFS \\ T_{DATA} &= t_{DATA} + SIFS \\ T_{ACK} &= t_{ACK} + SIFS. \end{aligned}$$

The sum of these four intervals defines the duration of the message exchange sequence for a data packet.

We next look at bandwidth estimation schemes for different scenarios. Fig.3.4(a) shows the uplink data transmission sequence from a client to an AP, while Fig.3.4(b) shows the downlink data

---

[1] In this work, we consider RTS/CTS-enabled DCF MAC in the bandwidth estimation. However, the estimation method can be easily extended to other cases.



(a) Uplink frame exchange

(b) Downlink frame exchange

Figure 3.4: Uplink and Downlink Frame Exchange Sequence.

transmission sequence from an AP to a client. We will discuss the two scenarios separately in the following subsections.

### 3.3.2 Estimating Available Bandwidth Via Uplink Traffic

We first look at bandwidth estimation for uplink traffic (the sequence shown in Fig. 3.4(a)). We define $ABE_{uplink}$ (available bandwidth estimation for uplink traffic) as the following ($S_{DATA}$ is the data size):

$$ABE_{uplink} = \frac{S_{DATA}}{T_{RTS} + T_{CTS} + T_{DATA} + T_{ACK}} \tag{3.1}$$

$$= \frac{S_{DATA}}{t_1 - t_0}. \tag{3.2}$$

By dividing the size of the actual data packet by the total time used to compete for channel and transmit the whole frame sequence, $ABE_{uplink}$ can be used as an indicator of the effective bandwidth that a client can achieve from the current association. As illustrated in Fig.3.4(a), we have:

$$ABE_{uplink} = \frac{S_{DATA}}{t_1 - t_0}, \tag{3.3}$$

In Eq.(3.3), $t_0$ and $t_1$ are the start and end time of the frame transmission sequence (Fig. 3.4(a)). This can be conveniently implemented in the client code by recording the time when RTS is placed in the sending buffer ($t_0$) and the time when ACK is received ($t_1$). Here, we define the overhead $T_{OH}$ as

$$T_{OH} = T_{RTS} + T_{CTS} + T_{ACK}.$$

Therefore, Eq.(3.1) can be further derived as:

$$\begin{aligned} ABE_{uplink} &= \frac{S_{DATA}}{T_{DATA} + T_{OH}} \\ &= \frac{1}{\frac{T_{DATA}}{S_{DATA}} + \frac{T_{OH}}{S_{DATA}}} \\ &= \frac{1}{\frac{1}{r_{DATA}} + \frac{T_{OH}}{S_{DATA}}}, \end{aligned} \tag{3.4}$$

where $r_{DATA}$ is the PHY transmission rate for data frames. As shown in Eq.(3.4), $ABE_{uplink}$ is related to $r_{DATA}$, $T_{OH}$ and the frame size $S_{DATA}$. The equation shows that larger data frames can lead to higher bandwidth estimation. As a result, the estimated bandwidth reaches its maximum by using frames of maximum size. Hence, we redefine $ABE_{uplink}$ as following:

$$\begin{aligned} ABE_{uplink} &= \frac{S_{DATA\_MAX}}{T_{OVERHEAD} + T_{DATA\_MAX}} \\ &= \frac{S_{DATA\_MAX}}{T_{OH} + \frac{S_{DATA\_MAX}}{r_{DATA}}}. \end{aligned} \tag{3.5}$$

By using Eq.(3.5), we can eliminate the inconsistency in bandwidth estimation when using different frame sizes. when calculating $ABE_{uplink}$ of different frame sizes. However, we are not suggesting using large frames will definitely improve bandwidth utilization. As the frame size becomes larger, the frame error rate also increases, which may introduce further overhead to retransmit the frame. We only use Eq.(3.5) as a indicator of bandwidth utilization, not a guideline to arrange frame transmission.

### 3.3.3 Simulation Validation

Several simulations are carried out to validate the correctness of Eq.(3.5). The simulations are conducted in the Qualnet 4.5 simulator with one AP and two nodes, *A* and *B*. A CBR client on node

*A* sends packets to the CBR server on node *B*. The PHY layer transmission rate is 2Mbps, and the traffic is CBR. We first configure the simulations to achieve the maximum bandwidth. As discussed before, the bandwidth reaches its maximum with the largest allowable frame size. Since Qualnet 4.5 does not implement MAC layer fragmentation, the maximum frame size is determined by IP layer fragmentation. The IP fragment threshold is 2048 bytes in the simulator. The part of the original packet is divided into fragments, each, except the last one, being an integer multiple of 8 octets long. Thus, the maximum fragment size using IPv4 protocol is $\lfloor \frac{2048-20(\textit{IPv4 header})}{8} \rfloor * 8 = 2024$. Therefore, in order to generate such a fragmentation, the data packet should be no less than 2024-8(UDP header)=2016 bytes, which gives the size of MAC layer frames 2016+8(UDP header)+20(IP header)+28(MAC header)=2072 bytes. In our simulations, the average time spent sending one frame is 2.5ms. Hence, we let the CBR client transmit 500 packets per second in order to saturate the channel. The simulation runs for 60 seconds, and the CBR throughput shown by Qualnet is 1.667Mbps. Taking into account of the overhead from adding UDP and IP headers, we get the MAC layer throughput of $\frac{1.667}{2016} * 2072 = .1.706Mbps$ (saturation throughput).

Next, we estimate $ABE_{uplink}$ on a per-frame basis using Eq.(3.5). When sending frames to the AP, node $A$ records the time when the frame is placed in the sending buffer ($t_0$) and the time when the ACK is received ($t_1$). The bandwidth can then be estimated as

$$ABE_{uplink} = \frac{S_{DATA\_MAX}}{t_1 - t_0}. \tag{3.6}$$

The results calculated using Eq.(3.6) are presented in Fig.3.5, showing the $ABE_{uplink}$ values during the 6 second simulation period. The horizontal dash line represents the simulated bandwidth 1.706Mbps, and the blue line plots the $ABE_{uplink}$ values calculated on a per-frame basis using Eq.(3.6). Though the estimated values are close to the average throughput, there are a lot of noises and jitters. Since we will be making critical decisions based on the $ABE$ values, these noises may severely undermine the reliability of the protocol. In order to reduce the noises, we average the $ABE$ values over k samples. Fig.3.5 shows the resulting estimation values (represented by the redline) using $k = 10$ can greatly reduce jitters compared to original values.

The above formula gives the per-packet bandwidth estimation. In reality, we usually take the

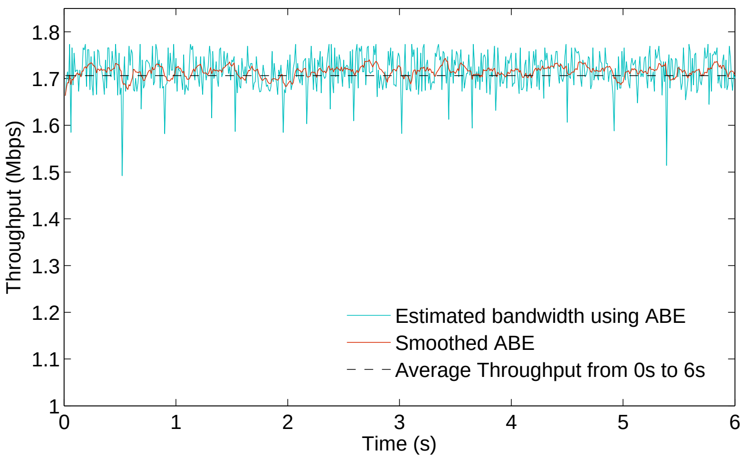


Figure 3.5: ABE and smoothed ABE compared with the actual effective bandwidth when using maximum frame size.

average of estimation calculated from multiple consecutive packets, or apply smoothing factor (e.g. EWMA), to smooth the estimated bandwidth. $y_t = \alpha y_{t-1} + (1-\alpha)x_t$

Next, under the same setting, we run another set of simulations with varying packet sizes $S_{DATA}$. Since the $ABE$ calculation uses the maximum frame size, we use the following equation to normalize the estimation when using varying packet sizes:

$$\begin{aligned} ABE_{uplink} &= \frac{S_{DATA\_MAX}}{T_{OH} + T_{DATA\_MAX}} \\ &= \frac{S_{DATA\_MAX}}{(t_1 - t_0 - \frac{S_{DATA}}{r_{DATA}}) + \frac{S_{DATA\_MAX}}{r_{DATA}}}. \end{aligned} \tag{3.7}$$

Fig. 3.6 presents the $ABE_{uplink}$ values using variable packet sizes from 200 bytes to 2000 bytes. We can see the $ABE_{uplink}$ values calculated using Eq.(3.7) remain almost unchanged with varying packet sizes. This confirms our conjecture because we consider $ABE_{uplink}$ as a metric to reflect the channel environment, which is mainly affected by signal quality or contention level, irrelevant to other factors such as packet sizes.

### 3.3.4 Estimating Available Bandwidth Via Downlink Traffic

The downlink bandwidth estimation is different from the uplink case. Since it is the AP who initiates transmissions, the client would not know when the AP places the RTS into the sending buffer($t_0$). To

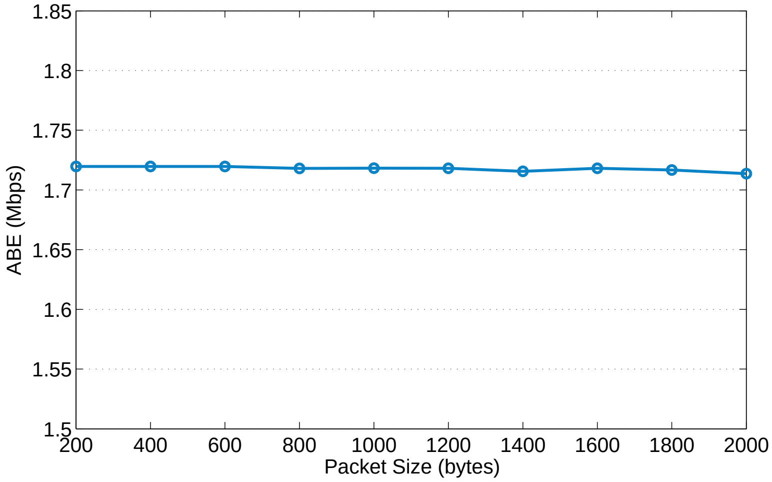


Figure 3.6: ABE with various frame size

address this issue, we mark the time when RTS is received as $t_0$, and the time when the following RTS is received by the client as $t_1$, as illustrated in Fig.3.4(b). Therefore, $ABE_{downlink}$ can be written in the same form as $ABE_{uplink}$ in Eq.(3.7), and we use $ABE_{DATA}$ to unify the above two estimations in the following equation:

$$ABE_{DATA} = \frac{S_{DATA\_MAX}}{(t_1 - t_0 - \frac{S_{DATA}}{r_{DATA}}) + \frac{S_{DATA\_MAX}}{r_{DATA}}}. \tag{3.8}$$

We note that for downlink estimation to work properly, there must be immediate following frames on the AP dedicated to the same client. According to our definition, $T_{RTS}$ is the sum of $T_{\textit{busy medium}}$, $DIFS$, $BACKOFF$ and $t_{RTS}$, and therefore, any gap between the two frame sequences due to the lack of traffic should be eliminated. This can be achived by only looking at frames that are fragmented from the same packet, usually with ”1” as the MF (more fragments) bit in the IP header. An improved approach is to have the client examine the size of the current frame. If the frame size is $S_{DATA\_MAX}$, very likely the packet has been fragmented, and the client can use successive fragments to estimate bandwidth until it receives a fragment smaller than $S_{DATA\_MAX}$. This may avoid the delay of passing the frame to the IP layer to check its MF bit.

Similarly, the per-packet estimation can be written in the same form as in equation **??**:

$$ABE_{downlink} = \frac{DATA}{t_1 - t_0} \tag{3.9}$$

Averaging and smoothing can be used to achieve better accuracy.

### 3.3.5 Comparison with Other Bandwidth Estimation Methods

Our bandwidth estimation methods collect statistics from the frame exchange sequence to estimate the available bandwidth. In addition to data frames, there are management frames in the system, which can also be utilized to estimate bandwidth. There are mainly two types of measurement schemes based on management frames: passive scanning and active probing. Passive scanning methods[48] are non-intrusive, as they only passively listen to the beacons periodically sent by the AP. However, a drawback of such methods is that they can only measure the downlink channel environment, and do not reflect the uplink channel environment given the link asymmetry in 802.11 networks[45]. Active probing methods, on the other hand, can reflect channel condition in both ways by having the client send probing requests and the AP respond to the probes. Normally, active probing introduces overheads because of probing requests/responses. We, however, argue that active probing methods are suitable to our association probing phase, in which we have to exchange probing requests/responses to discover new APs anyway. Thus we can calculate bandwidth estimation without any additional overhead. Our final bandwidth estimation scheme works as follows:

- During normal data transmissions, we use Data-based methods are used during normal data transmission, with uplink and downlink variations discussed before, to evaluate the bandwidth utilization for the current association. (It can also be enhanced by active-probing during the less active data exchange session)
- Active-probing-based methods are used in the probing phase when nodes need to search for better association options from other APs.

We will discuss the implementation details of the probing-based method in the next section.

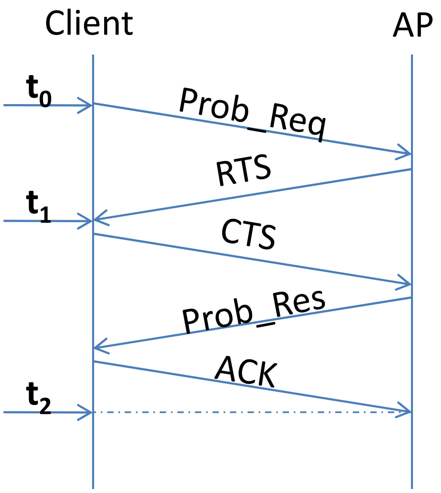


Figure 3.7: Frame Exchange Sequence during Probing

### 3.3.6 Estimating Available Bandwidth While Probing

A node not only needs to estimate the available bandwidth for its current AP, as discussed in the previous two subsections, but also needs to do so for other APs. Specifically, when the current association cannot satisfy its bandwidth requirement, a node needs to examine the available bandwidth of other APs and may change its association to an AP that offers more bandwidth.

Fig. 3.7 shows the frame exchange sequence during probing. First, after tuning to a specific channel, a client will send a broadcast $Probe\_Request$ frame. The AP that receives the frame will go through a complete frame exchange sequence: RTS $\rightarrow$ CTS $\rightarrow$ $Probe\_Response$ $\rightarrow$ ACK. Similar to the previous bandwidth estimation methods, clients need to know the start time and the end time of the sequence to calculate the available bandwidth estimation (ABE). Here, the client can mark the time when ACK is transmitted ($t_2$) as the end time. The start time of the sequence is not available to the client, but we can record the time when the Probe_Requst frame is placed in the client's sending buffer as $t_0$ and the time when the client receives RTS as $t_1$. Note that the transmission of a Prob_Reqest frame is the same as RTS in that they both need to contend for channel, so we assume the interval $t_1 - t_0$ the same as the period for transmitting two RTS frames, thus $T_{RTS}$ as $\frac{t_1 - t_0}{2}$. In this way, although during probing, the client only exchanges data with each AP once, we still manage to average over two RTS's for a better accuracy of $ABE_{Probing}$ estimation. Using

$t_2 - t_1$ to replace $T_{CTS} + T_{DATA} + T_{ACK}$ in Eq.(3.1), we can calculate $ABE_{Probing}$ as:

$$\begin{aligned} ABE_{Probing} &= \frac{S_{Probe\_Response}}{\frac{t_1 - t_0}{2} + (t_2 - t_1)} \\ &= \frac{S_{Probe\_Response}}{t_2 - \frac{t_1}{2} - \frac{t_0}{2}}. \end{aligned} \tag{3.10}$$

After normalization using the maximum frame size $S_{DATA\_MAX}$, and using $t'$ to substitute $t_2 - \frac{t_1}{2} - \frac{t_0}{2}$, Eq.(3.10) becomes:

$$ABE_{Probing} = \frac{S_{DATA\_MAX}}{t' - \frac{S_{Probe\_Response}}{r_{DATA}} + \frac{S_{DATA\_MAX}}{r_{DATA}}}. \tag{3.11}$$

To validation this equation, we use the same simulation setup as before, and apply Eq.(3.11) to the probing phase. The average $ABE_{Probing}$ is 1.69Mbps. We note that this value is in agreement with the ABE value in the transmission phase (1.706Mbps). We thus conclude our scheme can give accurate estimates of available bandwidth during the probing phase[2].

---

[2] Here, we only consider the scenario where each channel has one incumbent AP. Investigation of multiple incumbent APs will be in future work.

## 3.4 Access Point Association Policy using Channel Utilization

In this section, we present our access point association policy using channel utilization. Specifically, we discuss the policies of when to start probing other APs, which AP to choose as the next candidate, and when to switch association.

### 3.4.1 Selecting a new AP candidate

We take the viewpoint that a wireless node determines which AP to associate based on its own bandwidth requirement and the available bandwidth on the AP. A node's bandwidth requirement can change with time. For example, a client may need 200Kbps from 2:00pm to 3:00pm for VOIP calls, and 1Mbps from 3:00pm to 4:00pm for on-line video.

Based on this, we define Channel Utilization (CU) as the following:

$$CU = \frac{\mathit{Bandwidth\ Requirement}}{\mathit{Available\ bandwidth}}. \tag{3.12}$$

Here is an example to illustrate the use of CU. Say during 2:00pm and 4:00pm, the available bandwidth for the client's current association is 800Kbps. Then in the VOIP session, the CU is 200Kbps/800Kbps=25%, but during the video session, the CU becomes 1Mbps/800Kbps=125%, indicating the current association will not meet the client's bandwidth requirement, and a new association is preferred.

We set a threshold values: $CU_{probing}$, $T_{threshold}$ and $RSSI_{threshold}$. Probing will be triggered if **either** of the following criteria is met:

1. $CU \geq CU_{probing}$
2. $rssi \leq RSSI_{threshold}$

The first criteria is mainly used to guarantee the client bandwidth requirement, while the second criteria is inherited from the legacy protocol to guarantee the signal quality.

After probing all the channels and collecting statistics from each AP (e.g., signal strength, available bandwidth), the candidate AP is chosen based on **either** of the following conditions. We use

$rssi_{currnet}$ and $CU_{current}$ for the current association, and $rssi_{candidate}$ and $CU_{candidate}$ for the candidate AP,

1. $rssi \leq R$, and $rssi' - rssi \geq \delta_R$;
2. $CU \geq CU_{probing}$, and $CU' \leq CU_{probing}$, and $rssi' > R$,

where $rssi$ and $CU$ are statistics of the current channel, while $rssi'$ and $CU'$ are statistics for the candidate channel. $R, \delta_R$, and $CU_{probing}$ are threshold values.

### 3.4.2 Implementation Details

Each client saves its most recent $k$ ABE records in a list. Each ABE record includes the size of the data frame, and the two time stamps $t_0$ and $t_1$. Whenever a new ABE record is generated, the client first checks the list for any outdated record based on the time stamp (e.g., older than 5 seconds). Outdated records will be discarded to make room for the new record. If all $k$ records are valid, the oldest record will be discarded. EWMA is used to smooth the estimated bandwidth.

$y_t = \alpha y_{t-1} + (1 - \alpha)x_t$

The decision of which AP to use is made based upon their near-term or long-term future bandwidth predictions. For example, we use the trend algorithm described as the following: Trend to the current time t is calculated by:

$$\Delta = \frac{y_t - y_{t-L}}{L}$$

And prediction for future time t' is:

$$y_{t'} = y_t + (t' - t) * \Delta$$

Based on the predications from the available APs as well as the current AP, the client either choose a candidate whose predicted further bandwidth can satisfy its need, or stay with current AP if nothing better is found.

1. The trend algorithm. Using trend and term to make prediction. Trend to the current time t is calculated by:

$\Delta = \frac{y_t - y_{t-L}}{L}$,

and prediction for future time t' is: $y_{t'} = y_t + (t' - t) * \Delta$

2. Beyesian curve fitting

   $p(t|x, \mathbf{x}, \mathbf{t}) = \mathcal{N}(t|m(x), s^2(x))$

   $m(x) = \beta\phi(x)^T \mathbf{S} \sum\limits_{n=1}^{N} \phi(x_n) t_n$

   $\mathbf{S}^{-1} = \alpha\mathbf{I} + \beta \sum\limits_{n=1}^{N} \phi(x_n)\phi(x)^T$

   where the predicted value is $m(x)$. $x_n$ represents time and $t_n$ is the according previous measurements.

### 3.4.3 AP Switching Policy

To better understand the proposed switching policy, let us consider an example where there are 2 APs A and B, with several clients associated with A. Suppose a new client with a heavy load joins A's network, which will seriously degrade the performance of the existing clients. If the clients all start to look for a better candidate, and switch to B at the same time (though independently), they will soon saturate B, and need to switch again. The worst case would be for the clients to bounce back and forth between A and B, leading to a phenomenon known as "thrashing". In fact, we observe several trashing situations in our simulations. In some extreme cases, multiple clients switched between the same source and destination APs within less than 1 second. Some other clients only stayed briefly with the current AP before switching to another one.

We thus have the following three switching alternatives:

- **Immediate Switch.** This is the baseline scheme in which once a candidate AP is chosen, a node immediately switches its association.
- **Delayed Switch**. In this scheme, after selecting the candidate AP, the client postpones the switching by $T_{delay}$, which is a random value from [0, $Delay_{max}$]. During $T_{delay}$, normal transmission continues and the node keeps monitoring the bandwidth of the current AP. After $T_{delay}$, the client switches if the CU level remains above the threshold.
- **Probabilistic Switch**. In probabilistic switch, the node switches to the candidate AP with a probability $p$.

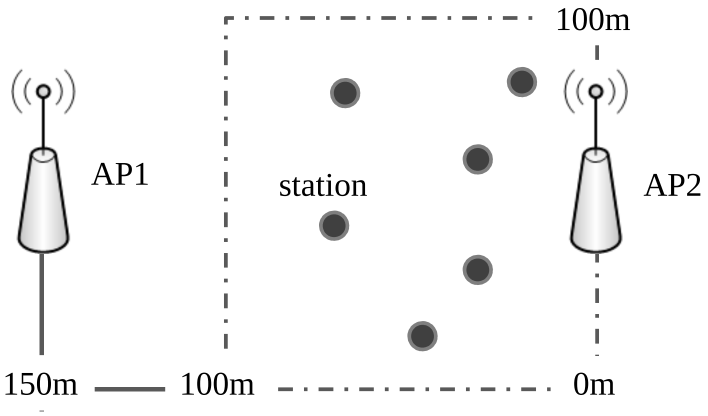


Figure 3.8: The Topology of Simulation

## 3.5 Performance Evaluation

In this section, we present the comparative performance of different AP association and switching policies.

We conduct simulation-based studies using the Qualnet 4.5 simulator, with 802.11b as the PHY module at a 2Mbps transmission rate. The transmission power is 20 dBm power. The considered topology is shown in Fig. 3.8, which consists of two APs and multiple wireless stations. The two APs are separated by $150m$ from each other, and the stations are randomly placed in an area of $100 \times 100m^2$, closer to AP2. In this topology, we expect more stations to associate with AP2 in the legacy 802.11 protocol or any signal strength based association protocol. By using our CU metric, despite how the nodes are located, the load on the two APs will be more balanced. Different channel frequencies are assigned to each AP to avoid inter-channel interference. The carrier sensing range is set to 200m so that both APs can be detected by all the clients. The combined load of the stations is set to the total capacity of the 2 APs, which is approximately 3.4 Mbps as discussed in Section 3.3.

We first compare our CU-based protocol with two baseline protocols, Signal Strength First (SSF) and Least Load First (LLF). In SSF, a node chooses the AP with the best signal-noise ratio (SNR). Whenever the SNF of the current association is below a threshold (e.g., -83dBm), the node will switch to a AP with an SNR greater than the current SNR by at least a small margin (e.g.,

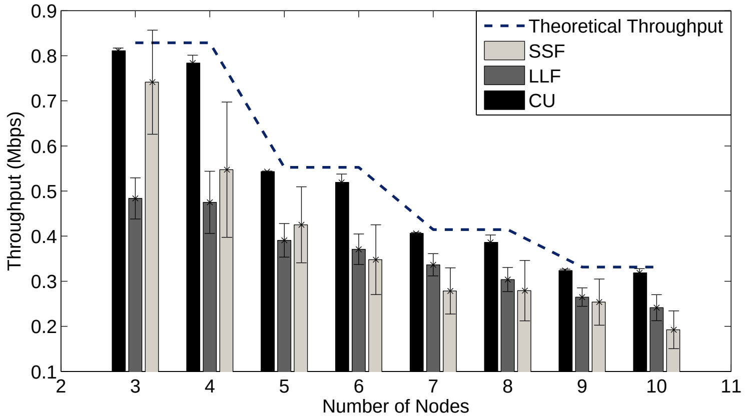


Figure 3.9: Average Per-Station Throughput

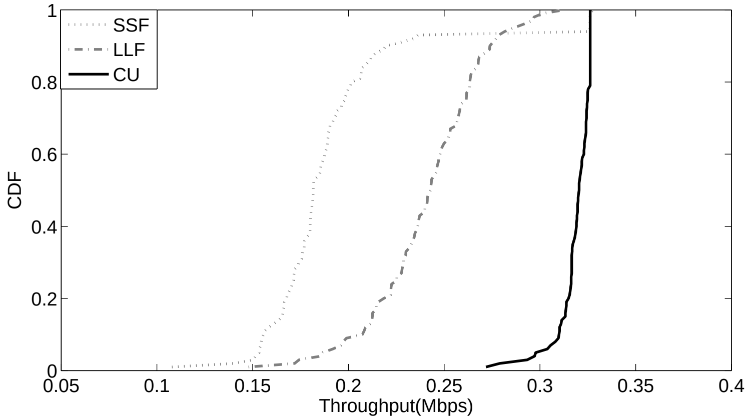


Figure 3.10: CDF of achieved throughput of each contending stations when the number of stations is 10

2dBm). LLF is a load-based protocol in which a node probes the neighboring APs periodically and associates with the AP with the least load ([46]). We note that our CU protocol differs from LLF in that CU only switches to a new AP when the current association cannot satisfy the node's bandwidth requirement while LLF always looks for a least-loaded AP no matter whether the current association is sufficient. In LLF, the node probes every 0.1 second as used in [46].

In our results, each data point is the average over 10 simulation runs with random node placements. In the following subsections, we will first compare our protocol (CU) with the two baseline protocols (SSF and LLF) in the perspective of per-node throughput, system-wide throughput, delay and jitter. Next, we will evaluate the performance of CU protocol with and without AP switching policy, and investigate its effect to thrashing.

### Throughput.

Fig. 3.9 shows per-station throughput for the three association schemes when the number of wireless stations increases from 3 to 10. We observe that the proposed CU protocol shows a significant improvement over the other two protocols. Compared with SSF, CU achieves a 38.4% gain on average, and a 65.7% improvement in certain scenarios (when we have 10 nodes). More importantly, the CU throughput is very close to the best throughput in theory shown in the dotted line.

We also show the standard deviation of the throughput measurements in Fig. 3.9. Since each station has the same load, the standard deviation in throughput can reflect the extent to which these stations are treated equally, further indicating whether the load on each AP is balanced. The legacy SSF protocol shows the largest variance for all cases. LLF shows an improved variance on throughput because it takes the load on each AP into consideration, but our CU-based protocol displays the smallest standard deviation for all cases. This suggests that the CU protocol can achieve the best load balance among the three protocols. The same trend is shown in Fig.  which plots the cumulative distribution function (CDF) of the

To further demonstrate the load-balancing advantage of CU-based association, we present the cumulative fraction of the achieved throughput of each contending station in Fig.3.10. CU-based association (the solid line) shows steeper slope of curve, meaning that most stations achieve well-balanced throughput, while SSF-based association yields throughput spanning from 0.14Mbps to 0.32Mbps for all the stations, and LLF associations from 0.15Mbps to 0.31Mbps.

In addition to per-station throughput, we also present the overall system-wide throughput in Fig. 3.11. The CU protocol achieves the highest system throughput. As the system load goes up, the throughput first increases and then remains at 3.2Mbps (the system capacity is 3.4Mbps). The maximum achievable throughput for SSF and LLF is much lower, 2.3Mbps and 2.5Mbps respectively.

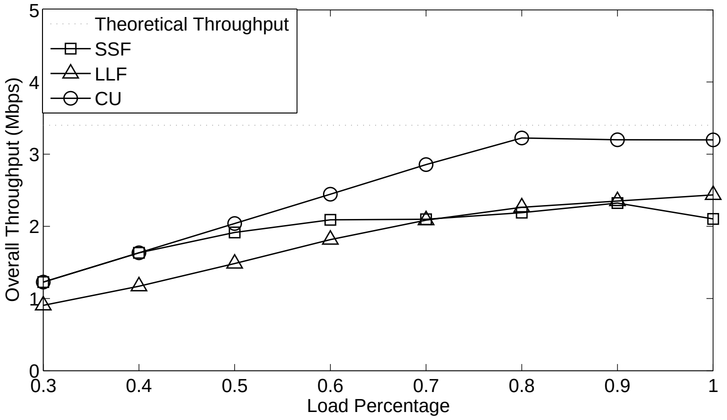


Figure 3.11: System throughput when the number of stations is 10

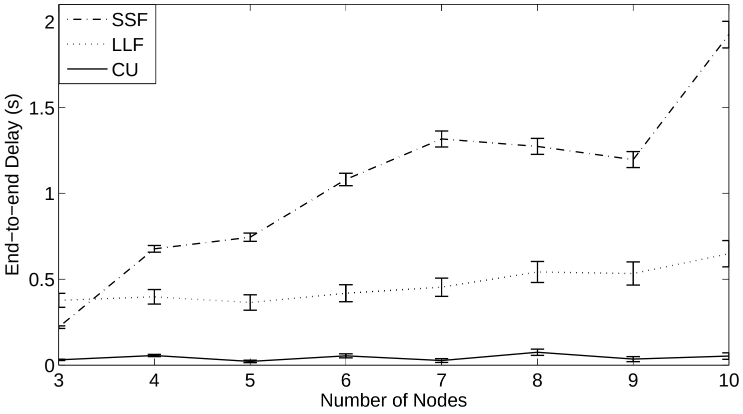


Figure 3.12: Delay and Jitter among contending stations

### Delay and Jitter

Another important metric for protocols supporting applications with QoS requirements is packet arrival delay and jitter. For applications such as audio and video streaming, high jitter, for example, having some packets taking 20msec and others taking 30msec to arrive will give an uneven quality to the sound or movie. Although QoS applications generally have jitter control components on higher layer to deal with the uneven packet arrival rate, large end-to-end delay and jitter of packet arrival at lower level generally prolongs the processing time of the jitter control components and therefore jeopardize the users' experience. Besides, although in some applications, such as video on demand, jitter can be eliminated by buffering at the receiver and then fetching data for display from the buffer

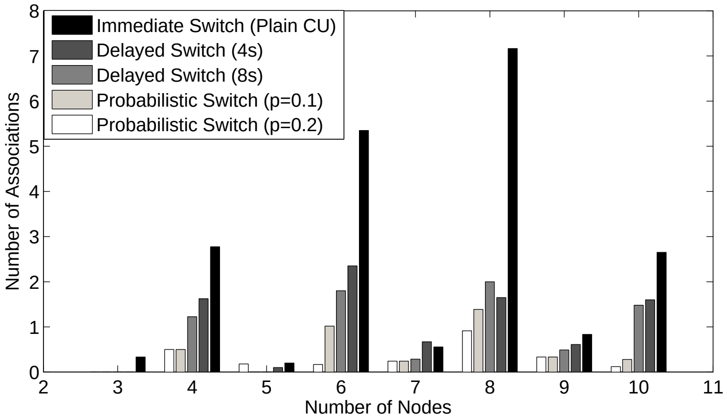


Figure 3.13: Average Number of Per-node Reassociations with CU and its APS variants

instead of from the network in real time. However, for other applications, especially those that require real-time interaction between people such as Internet telephony and video conferencing, the delay inherent in buffering is not acceptable. We compare the end-to-end delay and jitter (defined as the variance of the delay) for the three protocols in Fig. 3.12. Among the three protocols, the CU protocol is the best. It generates very low end-to-end delay (0.044s in average) and jitter (0.015s in average). SSF fares the worst, with its delay varying dramatically from 0.22sec to 1.92sec, with an average jitter of 0.038sec. LLF has an average delay of 0.47sec, and average jitter of 0.05sec.

### AP Switching Policy

Any association protocol can potentially lead to an excessive number of reassociations, dropped packets, or even thrashing. The CU protocol has this potential problem as well, and we can alleviate this problem by controlling when or whether to switch, such as in Delay Switch and Probabilistic Switch discussed in Section 3.4.3. Here we look at their effectiveness in reducing the number of reassociations and dropped packets in Fig. 3.13 and Fig. 3.14. we can see these two switching policies are quite effective. For example, Probabilistic Switch with p=0.2 can reduce the average number of reassociations by 91%, and can reduce the number of dropped packets by 90%.

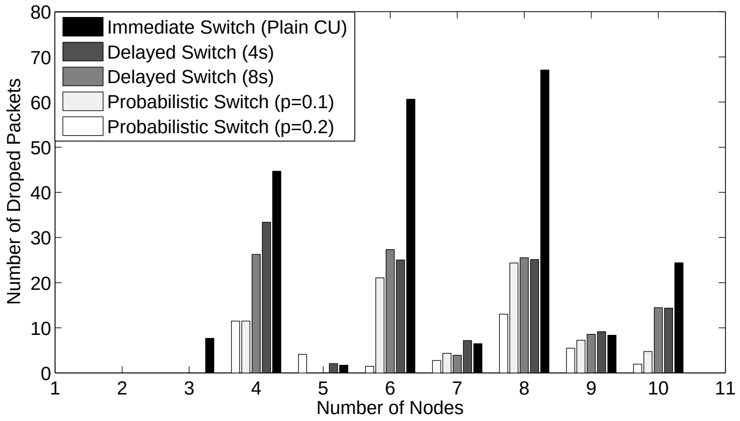
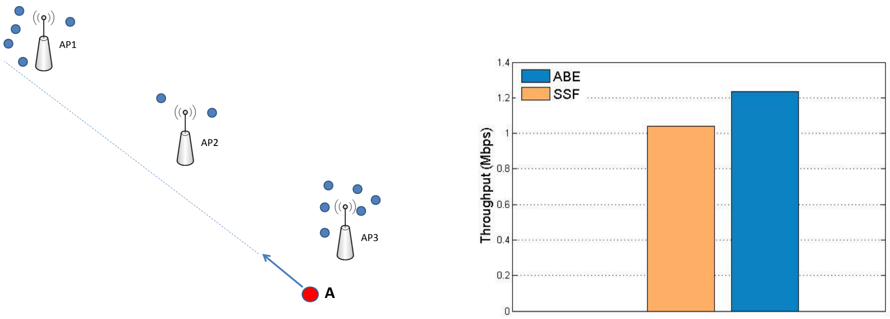


Figure 3.14: Average Per-node Packets Drop with CU and its APS variants



(a). Mobility Scenario.

(b). Average throughput using CU and SSF.

Figure 3.15: (a). The simulation set up for the mobility scenario. (b). The comparison of the average throughput of CU and SSF.

### 3.5.1 Mobility Scenario

In this subsection, we are testing the performance of CU protocol in mobility scenarios. Fig. 3.15(a) demonstrate the simulation topology. During a simulation run, node A travels through the coverage range of 3 APs. Among the 3 APs, AP1 and AP3 have heavy load, while AP2 has light load. The radio coverage range of the adjacent AP pairs (AP1 and AP2, AP2 and AP3) have overlap. For PHY settings, we use 802.11b, with 2Mbps transmission rate.

Fig. 3.15(b) shows the average throughput achieved using SSF and CU. We are able to achieve approximately 20% improvement using CU. The reason is using CU, the mobile node can identify the load on the different APs and is able to stay associated longer with the light-load AP to achieve better throughput.

## 3.6 Enhancing the CU Protocol through Reinforcement Learning

The main advantage of CU over SSF is that the former protocol can make a node start probing for a better AP much earlier. The implicit assumption is that a node can find a better AP by probing. This assumption, however, does not always hold. If no better AP is available, then probing would not solve the problem but further hurt the performance. This adverse effect will be more pronounced when all the APs in the system are heavily loaded.

To address this issue, we enhance the CU protocol by adopting reinforcement learning in determining whether a node needs to probe for a better AP when the available bandwidth of the current AP is not sufficient for the node.

### 3.6.1 Reinforcement Learning Model

Formally, the basic reinforcement learning model, as applied to MDPs (Markov decision process), consists of: a set of environment states S, a set of actions A, and a set of scalar "rewards" R.

At time t, the agent perceives its state $s_t \in S$ and the possible actions $A(s_t)$. It chooses an action $a \in A(s_t)$ and receives from the environment the new state $s_{t+1}$ and a reward $r_t$. Based on these interactions, the reinforcement learning agent must develop a policy $\pi : S \times T \to A$ (where T is the set of possible time indexes) which maximizes the quantity $R = r_0 + r_1 + \cdots + r_n$ for MDPs which have a terminal state, or the quantity $R = \sum_{t=0}^{\infty} \gamma^t r_t$ for MDPs without terminal states (where $0 \leq \gamma \leq 1$ is some "future reward" discounting factor).

Reinforcement learning has the strengths of: (1) It does not require a exact model of the environment, such as transition probabilities between states, etc, and (2) It can do on-line training, and the trained models is more adaptive to the environment. Besides, reinforcement learning is particularly well suited to problems which include a long-term versus short-term reward trade-off, which is exactly the focus of our problem.

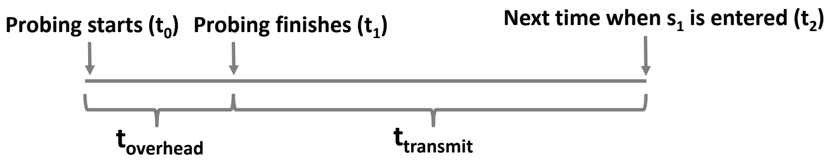


Figure 3.16: Illustration of transmission gain and association overhead when probing occurs.

### 3.6.2 Reinforcement CU

In this section, we build the reinforcement learning model according to our problem scope. We build the following reinforcement model:

1. $S = (s_1, s_2)$. $s_1$ represents the state in which the available bandwidth of the current AP falls below the required level, while $s_2$ represents the opposite state.
2. $A = (a_1, a_2)$, where action $a_1$ is to probe neighboring nodes and action $a_2$ is not to probe.

Next we define the reward functions $r(s_i, a_i)$. The reward function for state $s_2$ is straightforward: For state $s_2$, when the estimated CU is above the bandwidth requirement, we consider it not necessary to start probing. Thus to encourage action $a_2$ when state is at $s_2$, we define the following:

$$\begin{aligned} r(s_2, a_1) &= 0 \\ r(s_2, a_2) &= 1. \end{aligned}$$

We use this reward definition to refrain probing when the available bandwidth of the current association can meet the bandwidth requirement.

Next let us look at the reward functions at state $s_1$. The reward for each action is determined by the data transmission gain since last time when the same action was taken. In this way, we are predicting the reward of the near future using the reward of the near past. This is illustrated in Fig. 3.16. Suppose that at time $t_2$, we are in state $s_1$, and need to decide whether to probe other APs. We have recorded the time when last probing started ($t_0$) as well as the time when it finished ($t_1$). Thus we can derive the probing overhead $t_{overhead} = t_1 - t_0$, and the transmission time since last

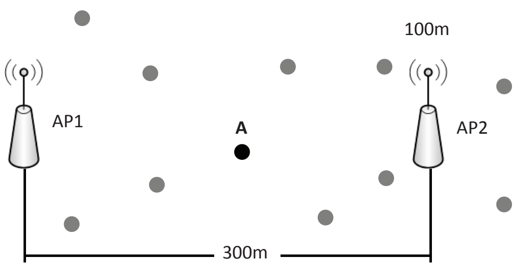


Figure 3.17: Simulation topology of 2 APs with variable load.

probing $t_{transmit} = t_2 - t_1$. We have the following reward function definitions:

$$\begin{aligned} r(s_1, a_1) &= t_{transmit} \times ABE \\ r(s_1, a_2) &= (t_{transmit} + t_{overhead}) \times ABE' \end{aligned}$$

Here, $ABE$ is the estimated bandwidth since last probing, and $ABE'$ is the estimated bandwidth before last probing. Therefore, $r(s_1, a_1)$ is the transmission gain due to last probing, and $r(s_1, a_2)$ is the transmission gain that could have been achieved if we chose not to probe at time $t_0$.

We use function Q to calculates the quality of a state-action combination:

$$Q : S \times A \rightarrow \mathbb{R} \tag{3.13}$$

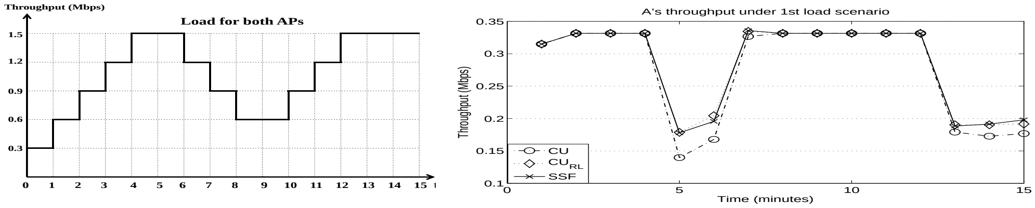
Before learning starts, Q returns a fixed initial value. Then, every time when the agent is given a reward (due to a state change), new values are calculated for every state/action pair. The core of the algorithm is a value iteration update. It assumes the old value and makes corrections based on the updated information. Q is updated based on the following equation:

$$Q(s_i, a_j) = (1 - \alpha)Q(s_i, a_j) + \alpha r(s_i, a_j), \tag{3.14}$$

where $\alpha (0 < \alpha \leq 1)$ is the learning rate. It determines to what extent the newly acquired information will override the old information.

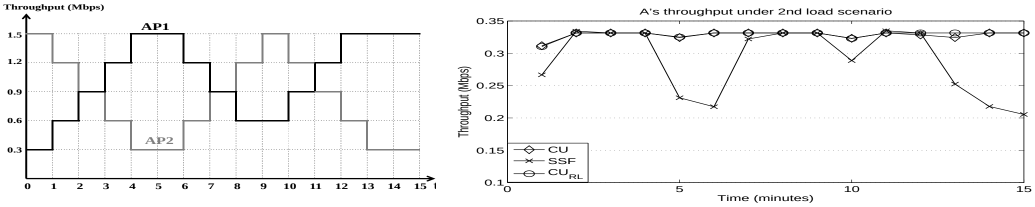
Next, we give the $\epsilon$-greedy learning policy for our Reinforcement CU protocol:

**Policy:** At each state $s_i$, the agent picks the action $a_j = \text{argmax}_a \, Q(s_i, a)$ with probability 1-$\epsilon$, and picks a random action with probability $\epsilon$.



(a) AP load pattern 1 (b) A's throughput

Figure 3.18: Throughput under load scenario 1.



(a) AP load pattern 2 (b) A's throughput

Figure 3.19: Throughput under load scenario 2.

### 3.6.3 Performance Evaluation

#### Static Scenarios

Next we compare the performance of the enhanced CU algorithm (referred to as RL-CU) with CU and SSF. Fig. 3.17 shows the simulation topology with 2 APs and a few wireless nodes. All nodes except A are used to create background traffic on the two APs and their associations are fixed. A can change its association according to different association protocols, and we report the performance observed on node A.

In the first experiment, the two APs have the same background load pattern, as shown in Fig. 3.18(a). It is intuitive that there won't be much gain to switch AP associations in this case. From Fig. 3.18(b), we observe that around the time minute 5 and 15, the throughput drops due to the increase of background traffic. When the network is congested, CU suffers from large probing cost, and it performs worse than SSF. However, RL-CU can avoid this problem and delivers the best performance among the three.

Fig. 3.19(a) shows another scenario in which the load on one AP is always noticeably heavier than the other. In such scenarios, CU has advantage over SSF since a node can choose a suitable

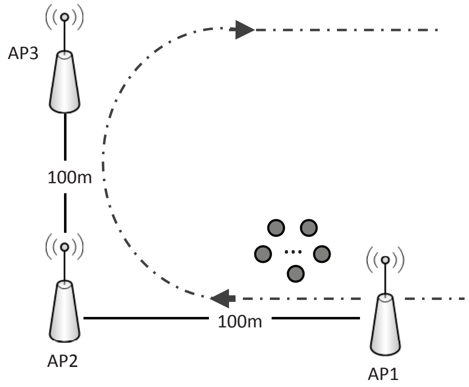


Figure 3.20: Simulation topology for mobile nodes with bursty traffic.

AP to associate. In Fig. 3.19(b), we can see RL-CU can perform just as well as CU, while the performance for SSF is much worse than the other two. Therefore, to summarize, by using RL-CU to adjust probing schedules, we can achieve the best performance of both worlds.

### Mobile Scenarios

Next, we use another set of experiments to demonstrate the performance in mobile scenarios. In these experiments, a group of 15 nodes will in turn visit the 3 access points. The simulation topology pattern is shown in Fig. 3.20. Again, we tested 2 traffic patterns, which are both shown in Fig. 3.21. The first traffic pattern represents continuous traffic, with each node continually contributes an individual traffic of 0.33Mbps. The second pattern represents bursty traffic. The traffic generated by each node has a mean inter-arrival rate of 30 seconds and each traffic burst lasts for 10 seconds. For both scenarios, there are existing background traffic for the 3 APs (0.1Mbps, 0.1Mbps and 0.2Mbps for A, B, C respectively). For each scenario, we tested the performance of three protocols, SSF, LLF, and CU_ML. For LLF, we vary the probing interval to three different values: 30s, 60s, 90s.

The result for continuous traffic is shown in Fig. 3.22. CU_ML's packet delivery rate reaches 83.18%. The performance for LLF varies with the probing intervals. The delivery rate decreases

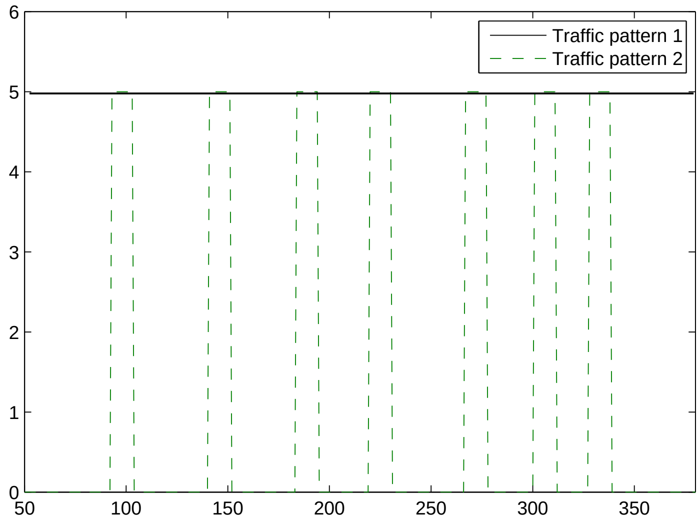


Figure 3.21: Traffic pattern for mobile scenarios.

from 79.13% to 72.55% when the probing interval increases from 30s to 60s, indicating that the performance suffers when the probing is not frequent enough to discover better gateways to associate. The performance for SSF is the worst of all three, with only 30% delivery rate. This is because all nodes associate with the gateway that they approach first, which is AP1, and ignore other gateways that appear later and may become better choices for association.

The result for bursty traffic is shown in Fig. 3.23. CU_ML's packet delivery rate reaches 87.45%, which is significantly higher than the SSF(37.84%) and LLF(from 37.75% to 46.06%). The reason for SSF's suboptimal performance is that the group of nodes tend to choose the same AP as their common strongest signal strength AP, thus it's always the one and only AP that get flooded. LLF's performance is caused by the bursty nature of the traffic. Since the duration of the traffic is short, the periodic probing in LLF is less able to capture the burst of traffic. In contrast, CU_RL is able to constantly adjust its probing on-demand, therefore delivers the best results.

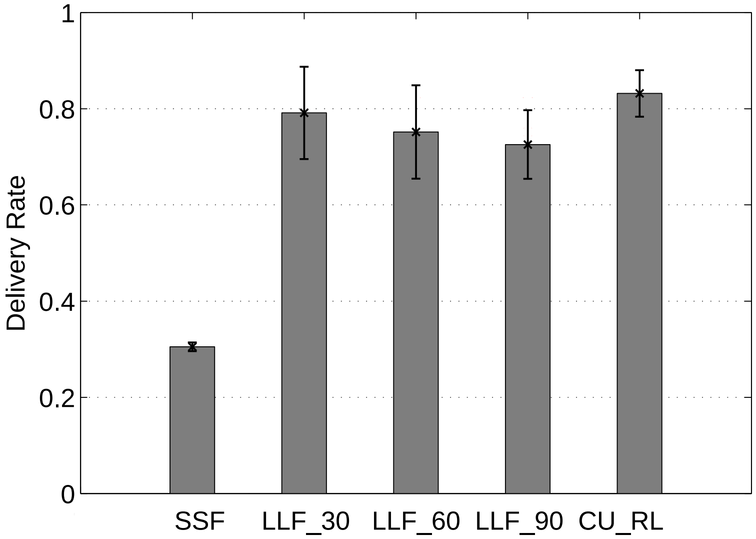


Figure 3.22: Delivery rate for mobile continuous traffic scenario.

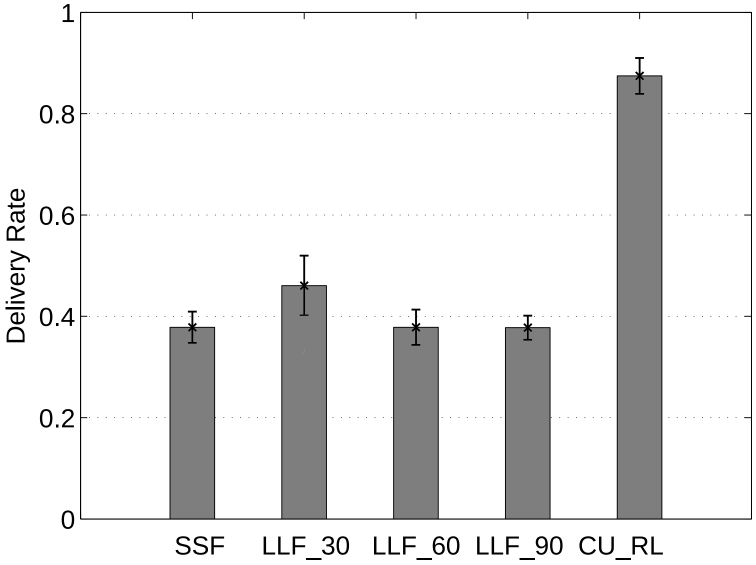


Figure 3.23: Delivery rate for mobile bursty traffic scenario.

## 3.7 Related Work

**Bandwidth Estimation.** Various bandwidth estimation schemes have been proposed for 802.11 networks. The bandwidth estimation method EVA in [46] is similar to our ABE concept. However, EVA can only be used for evaluating data traffic. During probing, the station needs to dedicate a long channel sensing time (10 seconds) in order to estimate EVA from on-going traffic of other stations. By contrast, the ABE scheme can easily incorporate probing frames into uplink/downlink data traffic. To the best of our knowledge, we are the first to propose a complete set of bandwidth estimation methods covering all the data exchange scenarios in the association process. Unlike the EVA protocol which always favors APs with the best bandwidth, we compare the estimated bandwidth with the individual client's bandwidth requirement, which further reduces the reassociation overhead. In [48], the authors propose to evaluate the potential AP bandwidth by passively measuring beacon timings from a particular AP. It uses the estimated MAC-layer bandwidth offered by different wireless networks in the vicinity. We note that using beacons can only provide downlink bandwidth estimates, which usually gives inaccurate evaluation of the overall channel condition. Beside, this paper didn't address probing and reassociation decisions using the estimation.

**Load Balancing.** The problem of load balancing has been extensively studied. In [6], explicit channel switching and network-directed roaming are used to provide hot-spot congestion relief while maintaining pre-negotiated user bandwidth agreements with the network. IEEE 802.11e is an extension to the base IEEE standard to address QoS issues. It defines changes to the operation of the IEEE 802.11 MAC to enable prioritization and classes of service over a WLAN. It defines the hybrid coordination function (HCF) to replace the legacy DCF and PCF in a STA implementing IEEE 802.11e. An AP in 802.11e will announce its load and capacity information in beacons, in the form of STA populations, available time slot, etc. In [49], two access point selection algorithms are proposed to maximize the average throughput and minimum throughput of stations. This paper uses a simplified communication model and bases the decision of AP selection on the calculated "throughput". This work as well as many others require significant modification to the current communication protocol. Another group of works require a centralized server to collect and distribute statistics from the network and distribute to AP and clients, such as [7][50][51][52]. In [7], a network operation center (NOC) is needed to make association decisions, as well as balancing load of all the AP. Similarly in [50], proportional fair (or time-based fair scheduling) provides a balanced tradeoff between fairness and network throughput. The function is implemented in a central management server, and the approximation algorithms can be used for periodic offline optimization. In [51], the authors propose two selection mechanisms which are decentralized in the sense that the decision is performed by each station. A few bytes of status information have to be added to the beacon. In [52], three metrics pertaining to wireless channel quality, AP capacity in the presence of interference, and client contention, are proposed. Unlike the above mentioned works, the ABE protocol we proposed allows the clients to make distributed decisions and requires no modifications to the current infrastructure.

**Lower Bandwidth Utilization.** Another class of related work focus on improving the bandwidth utilization by reducing the channel scanning delay, either by reducing scanned channels, or reducing the time consumed in scanning each channel (active scanning). In [53], the paper adopts proactive association to avoid losing connectivity. The scanning phase is shortened in the association process by only scanning the APs on the same or overlapping channels, and in-band scanning and switching is given higher priority in order to reduce channel switching time. In [45], the authors propose to decouple scanning from actual handoff by interleaving scanning into data transmission, during which a sleep request is sent to the AP to buffer packets before the scan finishes. In [54], vehicular mobility is considered in MESH networks with a scalable multi-tier architecture. A set of policies that employ smoothed AP-client signal quality coupled with per-AP quality scores are designed, which characterize the inherent inability of the mesh architecture to provide uniform bandwidth to all spatial locations. The above mentioned work mainly use RSSI-based metrics and did not address the load-balancing problems associated with it.

## 3.8 Conclusion and Future Work

As wireless LAN hotspots become more prevalent and experience many more users, an efficient access point association protocol is in a great demand. Most of existing association protocols rely on either the received signal strength of the access point, or the load of the access point, without considering the bandwidth requirement of the user nor the available bandwidth at the access point. Those schemes that do consider the bandwidth factors, however, are mostly centralized schemes. In this thesis, we set out to fill this void by designing a distributed access point association protocol based upon the bandwidth situation.

Our association protocol is centered around light-weight and accurate estimation of bandwidth and channel utilization (CU). We also adopt techniques that can avoid unnecessary reassociations among multiple access points. Further, we employ reinforcement learning techniques to determine whether we need to probe other access points when the current one seems to have insufficient bandwidth. Considering these factors, we compare our scheme with existing ones through extensive simulation studies. Our simulation results show that the proposed CU scheme can outperform existing schemes in many situations. In stationary scenarios, the basic CU association scheme can improve the average per-node throughput by 38.4% compared to the signal-strength based scheme, and a factor of 29.1% by the load-based scheme. In bursty mobile traffic scenarios, the reinforced CU scheme achieves a packet delivery rate of 87.45%, which is significantly higher than the signal strength based scheme (37.84%) and the load based scheme (from 37.75% to 46.06%).

# Chapter 4

# Identifying Association Protocols through Learning

## 4.1 Introduction

Many network services are being proposed that utilize administration information, which can be obtained either a-priori or on the fly. The information may either be public, or the client may obtain the information regarding a particular service from the provider as needed.

This approach, however, has been noted to be easily affected by a malicious entity providing fake administration information to clients and perform attacks once the client is connected. For example, the methods by which clients associate with access points (APs) and gateways can be used by malicious party to perform network attacks [55]. Association attacks take advantage of the security risks of the association protocol used by the clients. Various follow-up attacks can be easily performed by passing false administration frames once a client has been tricked to switch its association to a compromised gateway. Therefore, it becomes increasingly important that the information utilized by these services is trustworthy. Notably, before an entity gained access to a certain gateway, it is essential that administration information be verifiable. One approach to verify the service provider is to witnessing physical (e.g. signal strength [56] or time of arrival [57]) or network properties (e.g. hop count [58]) associated with that service, and learn a pattern associated with that property.

The ability to learn key characteristics in a network is also beneficial in cases where it is difficult or impossible to obtain the information a-priori. For example, a client may use the traffic information, such as traffic volume or delivery delay to estimate the service quality from a provider [59]. A network monitoring portal may learn certain network activity pattern for early detection of security breaches [55]. It is especially helpful, yet challenging to learn an accurate estimate of such information on-the-fly in todays increasingly diverse and complex networks. In this thesis, we investigate the learning techniques to identify several popular association protocols by looking at their key properties. In particular, we examine the following association protocols, which represent a broad selection of common association methods:

- SSF (Signal Strength First [45]): Association is based on the received signal strength, and an AP with the strongest signal strength will be chosen. When the signal strength drops below a certain threshold, the client starts probing neighboring APs.
- LLF (Least Load First [46]): A client probes neighboring AP periodically and associate with the AP with the least load.
- CU (Channel Utilization [60]): Association is based on channel utilization, a characteristic that combines AP's available load and the client's bandwidth requirement. When channel utilization drops below a certain threshold, the client start probing neighboring APs.
- CU_RL(Channel Utilization with Reinforcement Learning [61])): Association is not only based on channel utilization, but also based on feedback from previous associations. A reinforcement learning agent is used to trigger the probing process. It evaluates history information from previous associations and predicts the benefit of switching to other APs. A change in association only takes place when the predicted benefit of switching outweighs the overhead.

Being able to identify the association protocols used by nearby clients may provide a mobile node great advantage in choosing its own association approach. Once the nearby clients' association protocols are identified, the client may decide its own association protocol, by evaluating its need or the network condition. For example, the SSF protocol is known to work well in more dynamic environment, but does not emphasize on load balancing, where the CU-based protocol does well in balancing the load of relatively stable networks, but may cause more frequent switching. CU-based protocol works best when the traffic of nearby clients are more static, especially if more neighbors are using SSF protocol. If the neighbors are mostly using CU-based protocol, the entire network is more prone to the thrashing effect. [61] It appears to be a wise choice to use CU-based protocol is fewer neighboring clients are using CU-based protocol, and stick with the SSF protocol if the network becomes more dynamic. Therefore, it becomes important to identify the association protocol being used by nearby client through learned knowledge of the network, such as traffic pattern or probing pattern.

The remainder of the chapter is organized as the following: Section 4.3 gives details of the learning-based identification methods. Section 4.4 evaluates the performance of the methods on different association protocols. Section 4.5 discusses the related work, and Section 4.6 concludes the chapter.

## 4.2 System Overview

### 4.2.1 System Model and Assumptions

We begin by describing our underlying system model and some assumptions that we will use throughout this thesis. In this model, we assume that in a mobile ad hoc network, all mobile nodes are capable of running multiple association protocols. We consider a multi-AP multi-client scenario, in which the clients can be either static or mobile. Fig. 4.1 is the scenario picture with 2 APs and several clients. Both APs can be reached by all the clients. Suppose all clients can run both SSF and CU protocol. When client A comes to the network and wants to associate with an AP, it not only needs to decide which AP to associate with, but also choose which association protocol to use. Suppose the existing clients marked in red are running CU protocol, and the existing clients marked in green are running SSF protocol. After evaluating the network load and identifying the neighboring clients, A decides the load on the both APs are satisfactory, but in AP1's network, all 4 nodes are running SSF, while in AP2's network, 2 nodes are running CU and 1 node is running SSF. According to [61], A decides AP2's network are more prone to thrashing effect, therefore choose to associate with AP1. And since none of the existing clients with AP1 is using CU, A decides

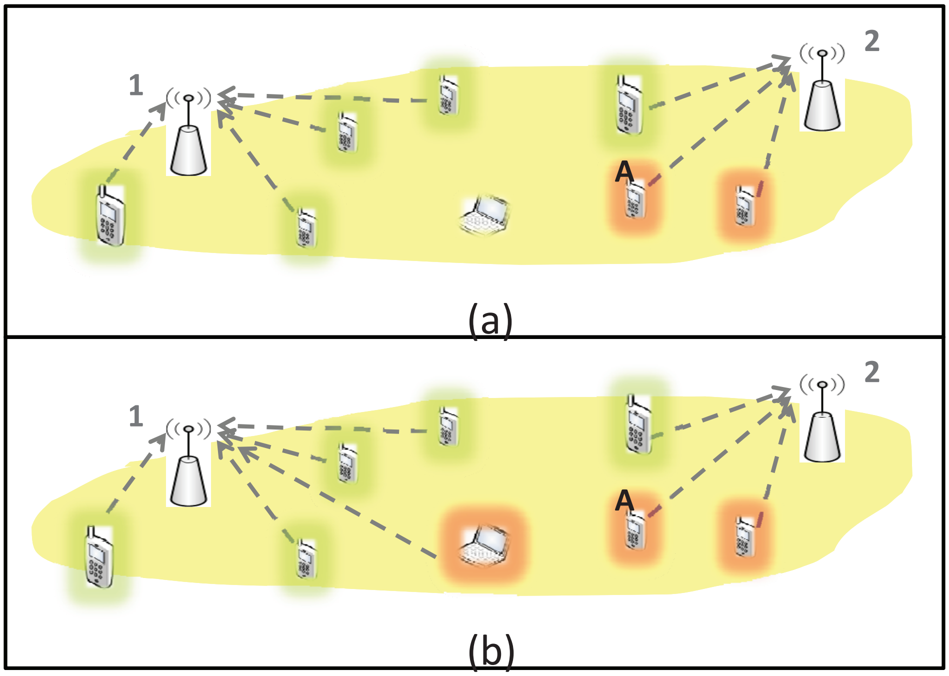


Figure 4.1: System Overview

to use CU which works best in more static network environment. In reality, the switching criteria and network scale are more up to individual's choice. In this thesis, we focus on investigating the techniques to identify the association protocol used by each individual client.

## 4.3 Identification Approaches

In this section, we look at approaches to identify association protocols. To understand the properties of a particular association protocol, we use the following simulation to study and elaborate on the underlying behavior of the probing pattern involved in the association protocol.

**Studied Scenario** Fig. 4.2 illustrates the simulation scenario. In this experiment, 10 nodes travel in a group and move in the area surrounded by 6 access points using the random waypoin mobility pattern [62]. The 6 APs have the total throughput of 10.2 Mbps. Each client has 0.3 Mbps bandwidth requirement. We vary the association protocols and run each simulation 10 times for the results.

First, we look at the probing patterns for different protocols, which is shown in Fig. 4.3. In this

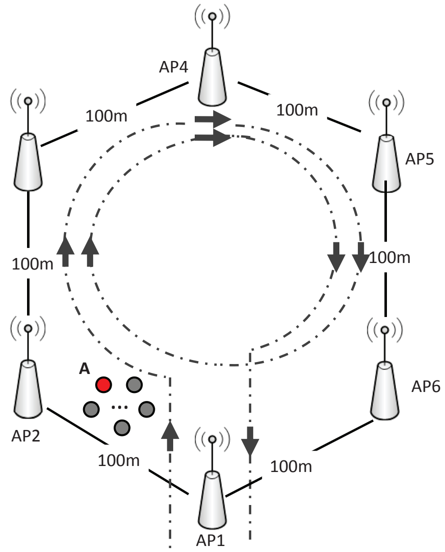


Figure 4.2: Simulation topology for mobile nodes.

case, we are interested in the probing interval, which is the time between two consecutive probings. We can see the probing patterns of the protocols are significantly different. To better understand the relationship between a protocol and its probing pattern, we use a 2-step approach to analyze the probing intervals. The approach is given as:

1. Clustering - Apply k-means clustering on the probing interval values.
2. Gaussian Fitting - Fit the interval values in each cluster into a Gaussian distribution.

We will explain each step in detail in the following subsections.

### 4.3.1 K-means Clustering

We use the k-means clustering algorithm [63] to group the probing values into clusters. Given a set of observations $(x_1, x_2, \ldots, x_n)$, where each observation is a d-dimensional real vector, $k$-means clustering aims to partition the n observations into $k$ sets $(k \leq n) S = S_1, S_2, \ldots, S_k$ so as to minimize the within-cluster sum of squares:

$$\underset{S}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{x_j \in S_i} \left\| x_j - u_i \right\|^2$$

where $u_i$ is the mean of points in $S_i$.

There are 3 steps involved in the clustering process.

1. Initialization. Commonly used initialization methods are Forgy Partition, Random Partition and Uniform Partition [64]. Here, we use uniform partition. Given an initial set of k means $m_1^{(1)}, \ldots, m_k^{(1)}$ (see below), the algorithm proceeds by alternating between the following two steps:

2. Assignment step. Assign each observation to the cluster with the closest mean.

$$S_i^{(t)} = \{x_j : \left\| x_j - m_i^{(t)} \right\| \leq \left\| x_j - m_{i^*}^{(t)} \right\| \text{for all } i^* = 1, \ldots, k\}$$

3. Update step: Calculate the new means to be the centroid of the observations in the cluster.

$$m_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j$$

After applying 4-means clustering to the above probing patterns, we have the results shown in Fig. 4.3.

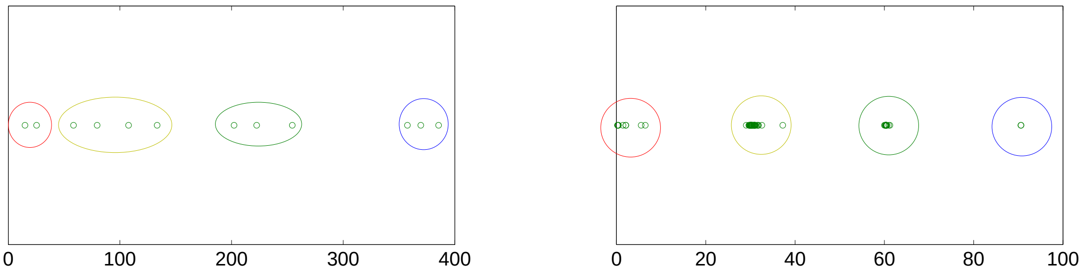
Next, we use gaussian fitting to determine the loose/dense clustering in the results from the previous step.

### 4.3.2 Gaussian Fitting by Maximizing the Likelihood Function

For a single real-valued variable x, the Gaussian distribution is defined by

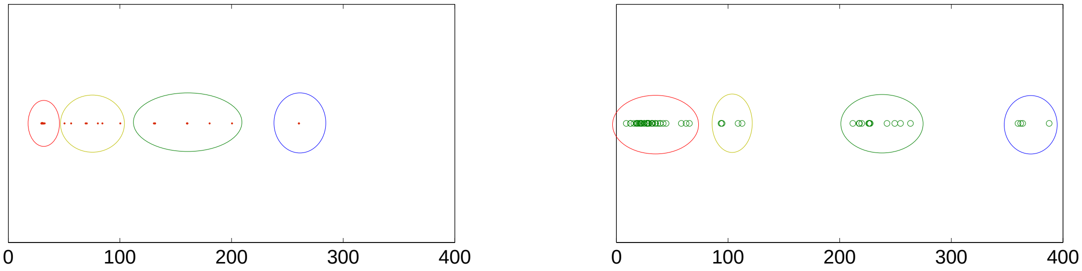
$$N(x|\mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} exp\{-\frac{1}{2\sigma^2}(x-\mu)^2\}$$

Given a data set of observations $x = (x_1, ..., x_N)^T$, suppose the observations are drawn independently from a Gaussian distribution with unknown mean $\mu$ and variance $\sigma^2$. Data points drawn independently from the same distribution are independent and identically distributed (i.i.d). The



(a). SSF probing interval pattern.

(b). LLF probing interval pattern.



(c). CU probing interval pattern.

(d). CU_RL probing interval pattern.

Figure 4.3: Probing intervals for different association protocols.

joint probability of two independent events is given by the product of the marginal probabilities for each event separately. Given $\mu$ and $\sigma^2$, the probability of an i.i.d. data set $x$ is:

$$p(x|\mu,\sigma^2) = \prod_{n=1}^{N} N(x_n|\mu,\sigma^2)$$

which is also the likelihood function for the Gaussian, when viewed as a function of $\mu$ and $\sigma^2$.

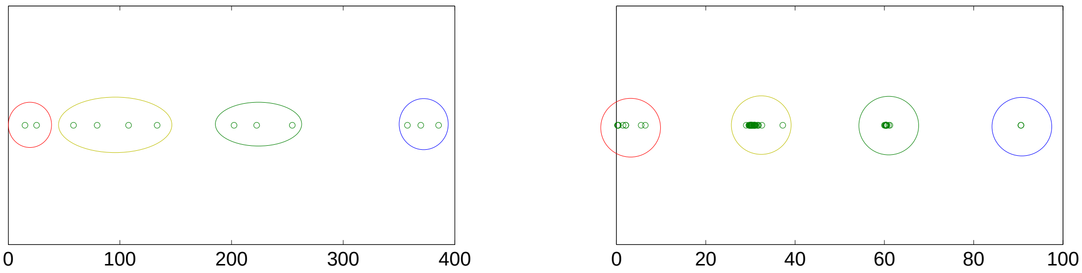
One common criterion for determining the parameters in a probability distribution using an observed data set is to find the parameter values that maximize the likelihood function. In practice, it is more convenient to maximize the log of the likelihood function [65]. The log likelihood function can be written as:

$$\ln p(x|\mu,\sigma^2) = -\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2 - \frac{N}{2}\ln\sigma^2 - \frac{N}{2}\ln(2\pi)$$

Maximizing ln $p(x|\mu,\sigma^2)$ with respect to $\mu$ and $\sigma^2$:

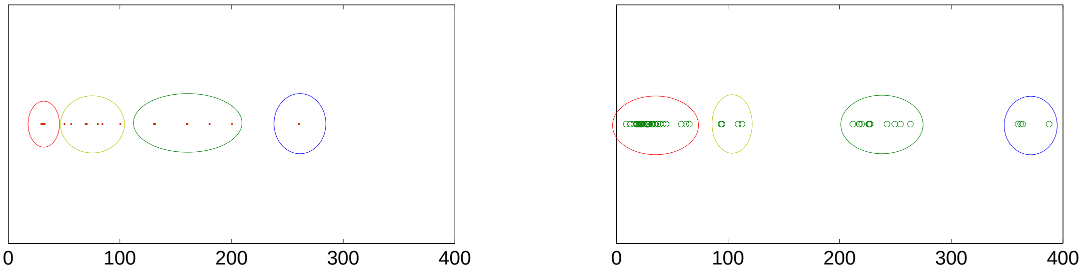
$$\mu_{ML} = \frac{1}{N}\sum_{n=1}^{N} x_n$$

$$\sigma^2_{ML} = \frac{1}{N}\sum_{n=1}^{N}(x_n-\mu_{ML})^2$$



(a). SSF probing pattern.

(b). LLF probing pattern.



(c). CU probing pattern.

(d). CU_RL probing pattern.

Figure 4.4: Gaussian fitting for different association protocols.

By applying Gaussian fitting to the probing intervals of LLF and CU_RL, we have the results shown in Fig. 4.4

We can see in Fig. 4.4(b), the variances of the last three clusters are very small. We will call cluster with a small variance a *dense cluster*. The reason for the 3 dense clusters in in Fig. 4.4(b) (for the LLF scheme) is that the probings are scheduled at a fixed interval (30 seconds in this case). The client may occasionally miss one or two probings, and for this reason there are two more clusters with a mean of 60 seconds and 90 seconds. The three dense clusters also display strong periodic pattern with approximate periodic mean values. Later, well introduce a technique to quantify the likelihood of a protocol being periodic probing protocol.

In the CU protocol, the client initiates probing when the available bandwidth is below the bandwidth requirement needed to stay connected with an access point. However, under conditions when the network is congested, the client may not find other suitable APs within the WLAN to provide better throughput, which will cause the client to go back to normal data transmission and then initiate another probing after waiting for a fixed delay. Consequently, we expect to see a lot of periodic probings when the network is congested. This results in a lot of fixed-interval-probings. But the protocol also generates probings at random intervals when not all APs within the WLAN are severely congested. This observation explains why there is one dense cluster in Fig. 4.4(c) centered around the system defined delay value, and 3 other loose clusters, with no obvious pattern.

For SSF and CU_RL, the probing is mostly generated randomly and on-demand. So there is no dense cluster or obvious patterns in Fig. 4.4(a) and Fig. 4.4(d). Later we will discuss how to distinguish between the protocols using other metrics.

Next, we look at the problem of using Confidence Interval (CI) to quantify the likelihood of a protocol being periodic probing protocol.

### 4.3.3 Using Confidence Interval (CI) to Evaluate Detection Accuracy

In statistics, a confidence interval (CI) is a particular kind of interval estimate of a population parameter and is used to indicate the reliability of an estimate. Confidence intervals are constructed based on a confidence level selected by the user. It measures how confident we want to be for the random variable to lie within the confidence interval. It's usually given in the form of $1 - \alpha$, where $\alpha$ is a small nonnegative number, close to 0.

For example, for $a = 0.05$, we take $1 - \alpha = 0.95$. For standard normal distribution $Z(\mu = 0, \sigma = 1)$, we have:

$$P(-z \leq Z \leq z) = 1 - \alpha = 0.95$$

Using the CDF of the standard normal distribution $\Phi$, we have:

$$\begin{aligned} \Phi(z) &= P(-z \leq Z \leq z) = 1 - \frac{\alpha}{2} = 0.975 \\ z &= \Phi^{-1}(\Phi(z)) = \Phi^{-1}(0.975) = 1.96 \end{aligned}$$

and we get

$$\begin{aligned} 0.95 &= 1 - \alpha = 1 - P(-z \leq Z \leq z) \\ &= P(-1.96 \leq \frac{\bar{X} - \mu}{\sigma} \leq -1.96) \end{aligned}$$

and,

$$P(\bar{X} - 1.96\sigma) \leq \mu \leq P(\bar{X} + 1.96\sigma)$$

Therefore, the 0.95 confidence interval becomes:

$$(\bar{X} - 1.96\sigma, \bar{X} + 1.96\sigma)$$

We define $L_{N2,N1,k}(l)$ as the likelihood of N2 being k-periodic of N1 with confidence level of $l$. N1 and N2 are normal distributions $N(\mu_1, \sigma_1)$ and $N(\mu_2, \sigma_2)$. For $l = 0.95$, we have:

$$L_{N2,N1,k}(0.95) = \Phi(\frac{\frac{\mu_2}{k} - \mu_1}{\sigma_1} + 1.96) - \Phi(\frac{\frac{\mu_2}{k} - \mu_1}{\sigma_1} - 1.96)$$

where $\Phi$ is the cumulative standard normal distribution function.

We now give the criteria for determining protocols with periodic probing patterns:

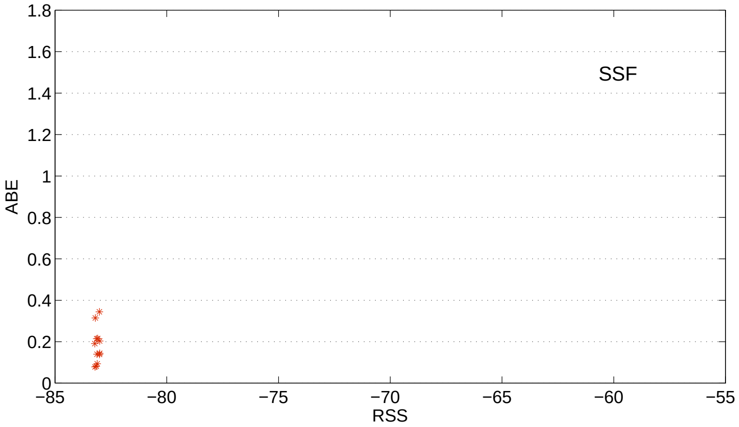
1. After applying k-means clustering, each cluster should have small variance, which we call a dense clustering.
2. The likelihood values for the cluster pairs should be high.

Applying the above criteria on the previous results of CU_RL and LLF, we have the following results:

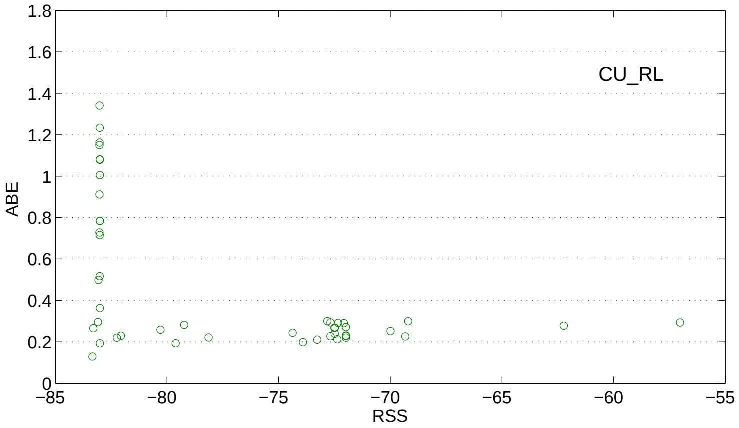
- For CU_RL, the clusters' variance values are: $s_1 = 12.14, s_2 = 9.21, s_3 = 16.49, s_4 = 12.97$. It's against the first criteria with the large variance values, we therefore determine CU_RL is not a periodic probing protocol.
- For LLF, $s_1 = 0.67, s_2 = 0.211, L_{N2,N1,2}(0.95) = 0.8077$. It meets the above two criteria, so we determine it's a periodic probing protocol based on the interval observation.

## 4.4 Identifying between SSF and CU_RL

To identify between SSF and CU_RL, we evaluate the received signal strength (RSS) and available bandwidth estimation (ABE) at a node when probing is triggered. Fig. 4.5 shows the RSS and ABE at node A when probing is triggered.



(a) SSF



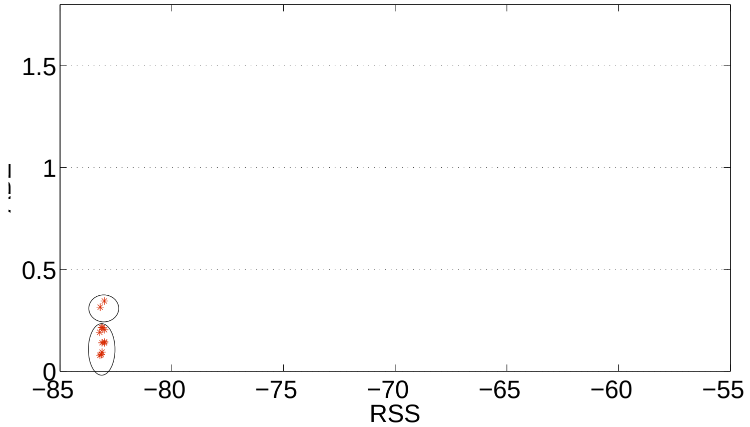
(b) CU_RL

Figure 4.5: ABE vs. RSS for SSF and CU_RL protocols

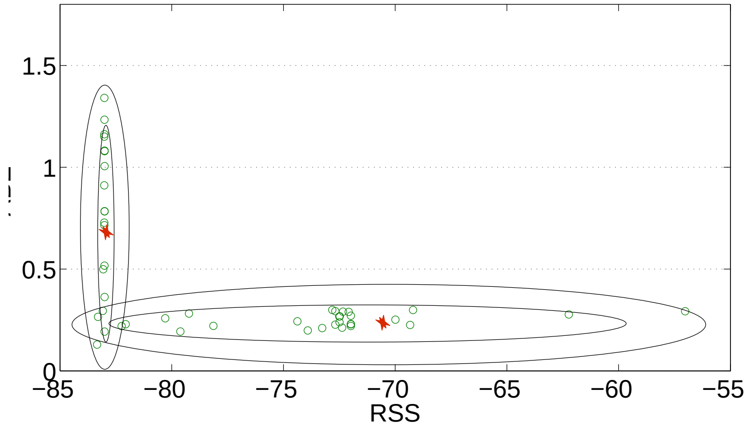
We can see SSF has lower ABE, and the RSS when probing occurs is within a small range; while CU_RL has higher ABE, initiates more probing, and has a wider range of triggering RSS.

Another observation is, for SSF, all probing occurred when and only when the RSS is below a certain threshold. Therefore, the triggering {RSS, ABE} pairs forms only 1 major cluster; while for CU_RL, the probing occurred not only when the RSS is below a certain threshold, but also when the ABE is below a certain threshold. Therefore, the {RSS, ABE} pairs in LLF forms 2 main clusters. To identify the two protocols based on the triggering {RSS, ABE} pairs, we again use the 2-step approach discussed before:

- Process the data pairs using 2-means clustering



(a) SSF



(b) CU_RL

Figure 4.6: Clustering and Gaussian fitting for SSF and CU_RL.

- Fit the data in the 2 generated clusters into Gaussian distributions using maximum likelihood.

The clustering and fitting results are shown in Fig. 4.6.

We can also see that for SSF, the two clusters have centers with similar RSS values. They can actually be considered as a single cluster. The RSS values of the cluster centers indicate the threshold of the probing signal strength.

$$RSS_{C1} \approx RSS_{C2} \approx RSS_{threshold}$$

For CU_RL, the data set is divided into two distinct clusters. The Signal strength and available bandwidth thresholds are given by the two centers of the clusters respectively.

$$RSS_{C1} \approx RSS_{threshold}$$

$$ABE_{C2} \approx ABE_{threshold}$$

## 4.5 Related Work

**Attack on Wireless Channel.** Checkoway et al. [66] presented an analysis of vulnerabilities of automotive short range wireless communications (Bluetooth), and long-range wireless communications (cellular). Francillon et al. [67] demonstrated relay attacks against keyless entry systems, and [68] [69] [70] also employed attacks on the RFID-based protocols used by engine immobilizers to identify the presence of an owners ignition key. Clark et al. [71] analyzed the security of P25 systems against both passive and active adversaries and showed that even when encryption is used, much of the basic meta-data is sent in the clear and is directly available to a passive eavesdropper. AMR systems being studied in this paper differ in several aspects from prior studied systems. In [72], defensive jamming has been proposed to protect medical devices. Although sharing similar concepts, AMR meters involve a different physical layer technology (frequency hopping), which makes jamming harder to perform. Differing from prior work [72], the focus of our paper is to provide insight from both attack and defense sides. [73] developed robust statistical methods to make localization attack-tolerant. [74] presents two methods to tolerate malicious attacks against beacon-based location discovery in sensor networks. [75] proposed to use time of arrival to resist position and distance spoofing attacks. The method measures distance from verifiers to the prover with RF first, then uses geometric method to validate the location claim. However, unlike these works, which employ timing information, our verification involves signal strength and load estimation as the underlying physical property.

**Non-intrusive Load Monitoring (NILM).**NILM systems monitor the total load at an electric meter to extract individual appliance profiles. NILM algorithms can be divided into two categories based upon the signatures they use: steady-state and transient [76]. Transient techniques require high frequency measurements (e.g., Msps) [77] [78], while steady-state techniques utilize low frequency measurements and perform edge detection to identify appliances [76] [79]. Recent work examined power consumption in the frequency domain [80], extending the capabilities of traditional transient solutions by empowering differentiation between similar appliances. Researchers also investigated privacy leakage by employing NILM systems. Mikhail et al. [81] proposed a method to infer a residents activities from demand-response systems. They first employed an existing NILM algorithm to recognize the running time schedules of various appliances. Then, extraction routines were used to determine occupancy schedules, sleeping cycles, and other activities. In their earlier work [82], they investigated the impact of sampling rate on the accuracy of personal activity inference. They showed that even with 20-minute time resolution, attackers could still get meaningful estimates of a users activities with 70% accuracy. To preserve consumer privacy from load monitoring, a protection system called NILL was proposed recently in [83]. They used an in-residence battery to mask the variance in load to counter potential invasions of privacy. We used prior work [76] to evaluate the privacy breach of AMR meters.

**Reverse Engineering.** Researchers have used reverse engineering methodology to expose security loopholes in systems when the designers tried to secure the system by obscurity. Rouf et al. [84] used a similar methodology to discover security and privacy risks of tire pressure monitoring systems. Nohl et al. [85] used reverse engineering to reveal ciphers from a cryptographic RFID tag that is not known to have a software or micro-code implementation. With some prior knowledge of the cipher, researchers used a black box approach [68] for cryptanalysis of ciphers. Bortolozzo et al. [86] used reverse engineering to extract sensitive cryptographic keys from commercially available tamper resistant cryptographic security tokens by exploiting vulnerabilities in their APIs.

## 4.6 Conclusion

In this thesis, we focused on using the timing patterns related to clients probing an environment for APs as a means to identify different association protocols. Specifically, we investigated methods to identify four association protocols, and we proposed an approach that combines k-means clustering and Gaussian fitting to classify the association protocols based on probing patterns. The designed schemes were tested on synthetic traffic traces for a test network scenario. To further quantify the likelihood and accuracy of the identification, we combined the usage of confidence intervals into the identification process. Results show that the proposed method both correctly identifies association protocols, as well as identify certain important metrics of the clients̈ chosen association protocol.

# Chapter 5

# Conclusion and Future Work

## 5.1 Conclusion

In this thesis, we discussed trajectory-based boomerang protocol to periodically make available data at certain geographic locations in a highly mobile vehicular network. The boomerang protocol returns the Geocache through nodes traveling toward the anchor location. To ensure high packet return rate, the carrier node records its trajectory while moving away from the anchor location. Then the best nodes to return the Geocache are selected based on the trajectory (RevTraj). We evaluated the effectiveness of the boomerang protocol by comparing this scheme with a shortest-distance georouting scheme MaxProgress. Using realistic traffic modeling, we demonstrated that the boomerang scheme outperforms its counterpart with a significantly increased return probability. Additionally, we further improved the boomerang protocol to satisfy more stringent anchoring requirements(e.g, returning the Geocache within specified time frame). We proposed the use of Q-learning to adaptively adjust the initial handoff time based on the return time history, and results show the learning-based boomerang protocol can achieve high in-time return probability, as well as more accurate average return time.

Secondly, we proposed the distributed access point association protocol based on bandwidth estimation. We examined the performance of multi-AP multi-node systems where static or mobile nodes access the AP for data services. Given limited number of access points and an abundance of service requests from the nodes, the load imbalance is not negligible and will cause tremendous performance degradation when the system supports many users simultaneously. We used the data received/transmitted by a node within a certain time period for bandwidth estimation, and formulated a criterion for optimizing the association strategy. We first proposed an heuristic-based association scheme to avoid excessive reassociation. And then we presented the reinforcement-learning-based protocol to optimize the timing for probing. Based on simulation studies, we have shown that the bandwidth estimation scheme has better performance than two baseline strategies, and that the learning-based association scheme can achieve better throughput and delivery rate than the baseline methods.

Thirdly, we investigate the methods for identifying association protocols. Specifically, we investigated methods to identify four association protocols, and we proposed an approach that combines k-means clustering and Gaussian fitting to classify the association protocols based on probing patterns. The designed schemes were tested on synthetic traffic traces for a test network scenario. To further quantify the likelihood and accuracy of the identification, we combined the usage of confidence intervals into the identification process. Results show that the proposed method can accurately identify the association protocols, as well as revealing certain important metrics of the client's chosen association protocol.

## 5.2 Future Work

We have evaluated the performance of our proposed methods under simulation settings. In the future, we would like to re-evaluate our proposed research methodology in more practical systems and applications. First of all, a practical location-based system needs to be implemented on a WiFi or a cellular network, so that our boomerang protocol and association protocol could be verified in a real system. As for the boomerang protocol, we plan to use the data collected from the system for our analysis. In particular, we would like to examine the formulation of the return probability and path model. Further, the packet delivery rate and expected return time would be recalculated, and compared to the results in the thesis. Similarly, since our boomerang strategy makes use of the movement pattern of users in a location-based system, it is important to make an accurate description of the moving patterns. As for the improved access point association protocol, we note that our evaluation is performed in simulations, different real life experiment scenarios should be considered to get a comprehensive evaluation.

In this thesis, we examined three wireless services (data management, load management, and identification services) and focused on improving the efficiency and security of these systems. We believe that more problems need to be addressed in these three services. In particular, our learning-based association protocol currently is used to address problems associated with a node causing too much overhead when switching among APs. To address other vulnerabilities in a WLAN, we would extend our association techniques and evaluate these algorithms on real-time testbed like the WINLAB Orbit. In addition, we would evaluate other forms of feeback other than the bandwidth estimation, and use these to formulate a more accurate association criteria that can address a broader variety of threats facing a WLAN.

# References

[1] B. M. Oki, M. Pfluegl, A. Siegel, and D. Skeen, “The information busan architecture for extensible distributed systems,” in *Proc. of the 14th ACM Symposium on Operating Systems Principles*, 1993, pp. 58–68. 1, 5

[2] A. Akella, G. Judd, S. Seshan, and P. Steenkiste, “Self-management in chaotic wireless deployments,” in *Proc. of the 11th Annual International Conference on Mobile Computing and Networking*, 2005, pp. 185–199. 1, 45

[3] D. Hadaller, S. Keshav, and T. Brecht, “MV-MAX: Improving wireless infrastructure access for multi-vehicular communication,” in *Proc. of the ACM SIGCOMM Workshop on Challenged Networks*, September 2006. 1, 45

[4] Y.-F. Wen, F.-S. Lin, and K.-W. Lai, “System throughput maximization subject to delay and time fairness constraints in 802.11 WLANs,” in *Proc. of the 11th International Conference on Parallel and Distributed Systems*, vol. 1, July 2005, pp. 775–781. 1, 45

[5] Bychkovsky,, Vladimir and Hull,, Bret and Miu,, Allen and Balakrishnan,, Hari and Madden,, Samuel, “A measurement study of vehicular internet access using in situ Wi-Fi networks,” in *Proc. of the 12th Annual International Conference on Mobile Computing and Networking*, 2006, pp. 50–61. 1, 45

[6] A. Balachandran, P. Bahl, and G. Voelker, “Hot-spot congestion relief in public-area wireless networks,” in *Proc. of the 4th Workshop on Mobile Computing Systems and Applications*, 2002, pp. 70–80. 2, 45, 48, 80

[7] Y. Bejerano, S.-J. Han, and L. E. Li, “Fairness and load balancing in wireless LANs using association control,” in *Proc. of the 10th Annual International Conference on Mobile Computing and Networking*, 2004, pp. 315–329. 2, 45, 48, 81

[8] B. Yu, J. Gong, and C. Z. Xu, “Catch-up: a data aggregation scheme for vanets,” in *Proc. of the 5th ACM International Workshop on VehiculAr Inter-networking*, 2008, pp. 49–57. 2, 45

[9] J. Zhao, T. Arnold, Y. Zhang, and G. Cao, “Extending drive-thru data access by vehicle-to-vehicle relay,” in *Proc. of the 5th ACM International Workshop on VehiculAr Inter-Networking*, 2008, pp. 66–75. 2, 45

[10] V. Navda, A. P. Subramanian, K. Dhanasekaran, A. T. Giel, and S. R. Das, “Mobisteer: Using steerable beam directional antenna for vehicular network access,” in *Proc. of the 5th International Conference on Mobile Systems, Applications, and Services*, June 2007. 2, 45

[11] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein, “Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet,” *ACM SIGOPS Operating Systems Review*, vol. 8, pp. 96–107, 2002. 5, 41

[12] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. Miu, E. Shih, H. Balakrishnan, and S. Madden, "Cartel: a distributed mobile sensor computing system," in *Proc. of the 4th international conference on Embedded networked sensor systems*, 2006, pp. 125–138. 5, 41

[13] U. Lee, E. Magistretti, B. Zhou, M. Gerla, P. Bellavista, and A. Corradi, "Mobeyes: Smart mobs for urban monitoring with a vehicular sensor network," *Wireless Communications, IEEE*, vol. 13, pp. 52–57, 2006. 5, 41

[14] B. Hoh, M. Gruteser, R. Herring, J. Ban, D. Work, J. Herrera, A. Bayen, and Q. J. M. Annavaram, "Virtual trip lines for distributed privacy-preserving traffic monitoring," in *Proc. of the 6th International conference on Mobile Systems, Applications, and Services*, 2008, pp. 15–29. 5

[15] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan, "The pothole patrol: Using a mobile sensor network for road surface monitoring," in *Proc. of the 6th International conference on Mobile Systems, Applications, and Services*, 2008, pp. 29–39. 5

[16] M. Mauve and J. Widmer, "A survey on position-based routing in mobile ad hoc networks," *IEEE Network*, vol. 15, pp. 30–39, 2001. 12

[17] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, "Ght: A geographic hash table for datacentric storage," in *Proc. of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, 2002, pp. 78–87. 41

[18] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tag: a tiny aggregation service for ad-hoc sensor networks," in *Proc. of the 5th Annual Symposium on Operating Systems Design and Implementation*, 2002, pp. 131–146. 41

[19] Y. Ni, U. Kremer, A. Stere, and L. Iftode, "Programming ad-hoc networks of mobile and resource-constrained devices," in *Proc. of the 2005 ACM SIGPLAN conference on Programming language design and implementation*, 2005, pp. 249–260. 41

[20] I. Vasilescu, K. Kotay, and D. Rus, "Data collection, storage, and retrieval with an underwater sensor network," in *Proc. of the 3rd International conference on Embedded networked sensor systems*, 2004, pp. 154–165. 41

[21] U. Lee, E. Magistretti, M. Gerla, P. Bellavista, and A. Corradi, "Dissemination and harvesting of urban data using vehicular sensing platforms," *IEEE Transactions on Vehicular Technology*, vol. 58, pp. 882–901, 2009. 41

[22] H. Kargupta, R. Bhargava, K. Liu, M. Powers, P. Blair, S. Bushra, J. Dull, and K. Sarkar, "Vedas: A mobile and distributed data stream mining system for real-time vehicle monitoring," in *Proc. of the SIAM International Conference on Data Mining*, 2004, pp. 309–311. 41

[23] T. Nadeem, S. Dashtinezhad, C. Liao, and L. Iftode, "Trafficview: traffic data dissemination using car-to-car communication," *ACM Mobile Computing and Communications Review*, vol. 8, pp. 6–19, 2004. 41

[24] J. Li, J. Jannotti, D. Couto, D. Karger, and R. Morris, "A scalable location service for geographic ad hoc routing," in *Proc. of the 6th annual international conference on Mobile computing and networking*, 2000, pp. 120–130. 42

[25] W. Zhao, M. Ammar, and E. Zegura, “A message ferrying approach for data delivery in sparse mobile ad hoc networks,” in *Proc. of the 5th ACM international symposium on Mobile ad hoc networking and computing*, 2004, pp. 187–198. 42

[26] J. Burgess, B. Gallagher, D. Jensen, and B. Levine, “Routing for vehicle-based disruption-tolerant networks,” in *Proc. of the 25th IEEE International Conference on Computer Communications*, 2006, pp. 1–11. 42

[27] Q. Li, D. Rus, M. Dunbabin, and P. Corke, “Sending messages to mobile users in disconnected ad-hoc wireless networks,” in *Proc. of the 6th annual international conference on Mobile computing and networking*, 2000, pp. 44–55. 42

[28] L. Briesemeister and G. Hommel, “Role-based multicast in highly mobile but sparsely connected ad hoc networks,” in *Proc. of the 1st ACM international symposium on Mobile ad hoc networking and computing*, 2000, pp. 45–50. 42

[29] H. Takagi and L.Kleinrock, “Optimal transmission ranges for randomly distributed packet radio terminals,” *IEEE Communications*, vol. 32, pp. 246–257, 1982. 42

[30] T. Hou and V.O.K.Li, “Transmission range control in multihop packet radio networks,” *IEEE Communications*, vol. 34, pp. 38–44, 1986. 42

[31] R. H. Frenkiel, B. R. Badrinath, J. Borres, and R. D. Yates, “The infostations challenge: balancing cost and ubiquity indelivering wireless data,” *Personal Communications, IEEE*, vol. 7, pp. 66–71, 2000. 42

[32] Y. cai and T. Xu, “Design, analysis, and implementation of a large-scale real-time location-based information sharing system,” in *Proc. of the 6th International conference on Mobile Systems, Applications, and Services*, 2008, pp. 106–117. 42

[33] H. Lu, N. Lane, S. Eisenman, and A. Campbell, “Bubble-sensing: A new paradigm for binding a sensing task to the physical world using mobile phones,” in *Proc. of the International Workshop on Mobile Device and Urban Sensing*, 2008. 42

[34] L. Chisalita and N. Shahmehri, “A peer-to-peer approach to vehicular communication for the support of traffic safety applications,” in *Proc. of the 5th IEEE International Conference on Intelligent Transportation Systems*, 2002, pp. 336–341. 42

[35] Y. B. Ko and N. H. Vaidya, “Flooding-based geocasting protocols for mobile ad hoc networks,” *Mobile Networks and Applications*, vol. 7, pp. 471–480, 2002. 42, 43

[36] T. Small and Z. J. Haas, “The shared wireless infostation model: a new ad hoc networking paradigm (or where there is a whale, there is a way),” in *Proc. of the 4th ACM international symposium on Mobile ad hoc networking and computing*, 2003, pp. 233–244. 42

[37] R. Morris, J. Jannotti, F. Kaashoek, J. Li, and D. Decouto, “Carnet: A scalable ad hoc wireless network system,” in *Proc. of the 9th ACM SIGOPS European Workshop*, 2000, pp. 61–65. 42, 43

[38] C. Maihofer, T. Leinmller, and E. Schoch, “Abiding geocast: Time-stable geocast for ad hoc networks,” in *Proc. of the 2nd ACM international workshop on Vehicular ad hoc networks*, 2005, pp. 20–29. 42

[39] C. Schwingenschlogl and T. Kosch, “Geocast enhancements of aodv for vehicular networks,” *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 6, pp. 96–97, 2002. 42

[40] J. LeBrun, C.-N. Chuah, D.Ghosal, and M.Zhang, “Knowledge-based opportunistic forwarding invehicular wireless ad hoc networks,” in *Proc. of the 61st IEEE conference on Vehicular Technology*, 2005, pp. 2289–2293. 43

[41] I. Leontiadis and C. Mascolo, “Geopps: Geographical opportunistic routing for vehicular networks,” in *Proc. of the IEEE Workshop on Autonomic and Opportunistic Communications*, 2007, pp. 1–6. 43

[42] ——, “Opportunistic spatio-temporal dissemination system for vehicular networks,” in *Proc. of the 1st international MobiSys workshop on Mobile Opportunistic networking*, 2007, pp. 39–46. 43

[43] A. J. Nicholson, Y. Chawathe, M. Y. Chen, B. D. Noble, and D. Wetherall, “Improved access point selection,” in *Proc. of the 4th International Conference on Mobile Systems, Applications and Services*, 2006, pp. 233–245. 45

[44] J. Ott and D. Kutscher, “Drive-thru internet: IEEE 802.11b for ”automobile” users,” in *Proc. of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 1, March 2004, pp. 362–373. 45

[45] H. Wu, K. Tan, Y. Zhang, and Q. Zhang, “Proactive scan: Fast handoff with smart triggers for 802.11 wireless LAN,” in *Proc. of the 26th IEEE International Conference on Computer Communications*, May 2007, pp. 749–757. 46, 59, 81, 84

[46] H. Lee, S. Kim, O. Lee, S. Choi, and S.-J. Lee, “Available bandwidth-based association in IEEE 802.11 wireless LANs,” in *Proc. of the 11th International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, 2008, pp. 132–139. 46, 67, 80, 84

[47] M. A. Visser and M. E. Zarki, “Voice and data transmission over an 802.11 wireless network,” in *Proc. of the 6th IEEE Symposium on Personal, Indoor and Mobile Radio Communications*, September 1995, pp. 648–652. 52

[48] S. Vasudevan, K. Papagiannaki, C. Diot, J. Kurose, and D. Towsley, “Facilitating access point selection in IEEE 802.11 wireless networks,” in *Proc. of the 5th ACM SIGCOMM Conference on Internet Measurement*, 2005, pp. 26–26. 59, 80

[49] A. Fujiwara, Y. Sagara, and M. Nakamura, “Access point selection algorithms for maximizing throughputs in wireless LAN environment,” in *Proc. of the 13th International Conference on Parallel and Distributed Systems*, 2007, pp. 1–8. 80

[50] L. E. Li, M. Pal, and R. Yang, “Proportional fairness in multi-rate wireless LANs,” in *Proc. of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies*, April 2004, pp. 1004–1012. 81

[51] M. Abusubaih, J. Gross, S. Wiethoelter, and A. Wolisz, “On access point selection in IEEE 802.11 wireless local area networks,” in *Proc. of the 31st IEEE Conference on Local Computer Networks*, 2006, pp. 879–886. 81

[52] K. Sundaresan and K. Papagiannaki, “The need for cross-layer information in access point selection algorithms,” in *Proc. of the 6th ACM SIGCOMM Conference on Internet Measurement*, 2006, pp. 257–262. 81

[53] V. Mhatre and K. Papagiannaki, “Using smart triggers for improved user performance in 802.11 wireless networks,” in *Proc. of the 4th International Conference on Mobile Systems, Applications, and Services*, 2006, pp. 246–259. 81

[54] A. Giannoulis, M. Fiore, and E. W. Knightly, “Supporting vehicular mobility in urban multi-hop wireless networks,” in *Proc. of the 6th International Conference on Mobile Systems, Applications, and Services*, 2008, pp. 54–66. 81

[55] R. Bejtlich, *The Tao of network security monitoring: beyond intrusion detection*. Pearson Education, 2004. 83, 84

[56] K. Langendoen and N. Reijers, “Distributed localization in wireless sensor networks: A quantitative comparison,” 2003. 83

[57] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, “The cricket location-support system,” in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom ’00, 2000, pp. 32–43. 83

[58] D. Niculescu and B. Nath, “Ad hoc positioning system (aps) using aoa,” in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 3, 2003, pp. 1734–1743 vol.3. 83

[59] G. Davies, M. Hardt, and F. Kelly, “Come the revolutionnetwork dimensioning, service costing and pricing in a packet switched environment,” *Telecommunications Policy*, vol. 28, no. 5, pp. 391–412, 2004. 83

[60] T. Sun, W. Trappe, and Y. Zhang, “Improving access point association protocols through channel utilization and adaptive probing,” in *Proc. of the 8th IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, 2011, pp. 155–157. 84

[61] ——, “Improved ap association management using machine learning,” *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 14, pp. 4–6, November 2010. 84, 85

[62] T. Camp, J. Boleng, and V. Davies, “A survey of mobility models for ad hoc network research,” *Wireless Communications and Mobile Ccomputing: Special Issue on Mobile Ad Hoc Networking: Research, Trend and Applications*, vol. 2, pp. 483–502, 2002. 86

[63] J. A. Hartigan and M. A. Wong, “Algorithm as 136: A k-means clustering algorithm,” *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, pp. 100–108, 1979. 87

[64] P. L. J.M. Pena, J.A. Lozano, “An empirical comparison of four initialization methods for the k-means algorithm,” *Pattern Recognition Letters*, vol. 20, pp. 1027–1040, 1999. 88

[65] C. M. Bishop, *Pattern Recognition and Machine Learning*. The MIT Press, October 2007. 89

[66] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno, “Comprehensive experimental analyses of automotive attack surfaces,” in *Proceedings of the 20th USENIX Conference on Security*, ser. SEC’11, 2011. 95

[67] T. Yang, L. Kong, W. Xin, J. Hu, and Z. Chen, “Resisting relay attacks on vehicular passive keyless entry and start systems,” in *Fuzzy Systems and Knowledge Discovery (FSKD), 2012 9th International Conference on*, 2012, pp. 2232–2236. 95

[68] S. C. Bono, M. Green, A. Stubblefield, A. Juels, A. D. Rubin, and M. Szydlo, “Security analysis of a cryptographically-enabled rfid device,” in *Proceedings of the 14th Conference on USENIX Security Symposium - Volume 14*, ser. SSYM’05, 2005. 95, 96

[69] T. Eisenbarth, T. Kasper, A. Moradi, C. Paar, M. Salmasizadeh, and M. Shalmani, “On the power of power analysis in the real world: A complete break of the keeloq code hopping scheme,” in *Advances in Cryptology  CRYPTO 2008*, ser. Lecture Notes in Computer Science, 2008, vol. 5157, pp. 203–220. 95

[70] S. Indesteege, N. Keller, O. Dunkelman, E. Biham, and B. Preneel, “A practical attack on keeloq,” in *Advances in Cryptology  EUROCRYPT 2008*, ser. Lecture Notes in Computer Science, 2008, vol. 4965, pp. 1–18. 95

[71] S. Clark, T. Goodspeed, P. Metzger, Z. Wasserman, K. Xu, and M. Blaze, “Why (special agent) johnny (still) can’t encrypt: A security analysis of the apco project 25 two-way radio system,” in *Proceedings of the 20th USENIX Conference on Security*, ser. SEC’11, 2011. 95

[72] S. Gollakota, H. Hassanieh, B. Ransford, D. Katabi, and K. Fu, “They can hear your heartbeats: Non-invasive security for implantable medical devices,” *SIGCOMM Comput. Commun. Rev.*, vol. 41, pp. 2–13, 2011. 95

[73] Z. Li, W. Trappe, Y. Zhang, and B. Nath, “Robust statistical methods for securing wireless localization in sensor networks,” in *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, 2005, pp. 91–98. 95

[74] D. Liu, P. Ning, and W. Du, “Attack-resistant location estimation in sensor networks,” in *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, 2005, pp. 99–106. 95

[75] S. Capkun and J.-P. Hubaux, “Secure positioning of wireless devices with application to sensor networks,” in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 3, 2005, pp. 1917–1928 vol. 3. 95

[76] G. Hart, “Nonintrusive appliance load monitoring,” *Proceedings of the IEEE*, vol. 80, no. 12, pp. 1870–1891, 1992. 95, 96

[77] S. Leeb, S. Shaw, and J. Kirtley, J.L., “Transient event detection in spectral envelope estimates for nonintrusive load monitoring,” *Power Delivery, IEEE Transactions on*, vol. 10, no. 3, pp. 1200–1210, 1995. 96

[78] C. Laughman, K. Lee, R. Cox, S. Shaw, S. Leeb, L. Norford, and P. Armstrong, “Power signature analysis,” *Power and Energy Magazine, IEEE*, vol. 1, no. 2, pp. 56–63, 2003. 96

[79] M. Marceau and R. Zmeureanu, “Nonintrusive load disaggregation computer program to estimate the energy consumption of major end uses in residential buildings,” *Energy Conversion and Management*, vol. 41, no. 13, pp. 1389 – 1403, 2000. 96

[80] S. Gupta, M. S. Reynolds, and S. N. Patel, “Electrisense: Single-point sensing using emi for electrical event detection and classification in the home,” in *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*, ser. Ubicomp ’10, 2010, pp. 139–148. 96

[81] M. Lisovich, D. Mulligan, and S. Wicker, “Inferring personal information from demand-response systems,” *Security Privacy, IEEE*, vol. 8, no. 1, pp. 11–20, 2010. 96

[82] M. Lisovich and S. Wicker, “Privacy concerns in upcoming residential and commercial demand-response systems,” in *2008 Clemson University Power Systems Conference*. Clemson University, March 2008. 96

[83] S. McLaughlin, P. McDaniel, and W. Aiello, “Protecting consumer privacy from electric load monitoring,” in *Proceedings of the 18th ACM Conference on Computer and Communications Security*, ser. CCS ’11, 2011, pp. 87–98. 96

[84] I. Rouf, R. Miller, H. Mustafa, T. Taylor, S. Oh, W. Xu, M. Gruteser, W. Trappe, and I. Seskar, “Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study,” in *Proceedings of the 19th USENIX Conference on Security*, ser. USENIX Security’10, 2010, pp. 21–21. 96

[85] K. Nohl, D. Evans, S. Starbug, and H. Plötz, “Reverse-engineering a cryptographic rfid tag,” in *Proceedings of the 17th Conference on Security Symposium*, 2008, pp. 185–193. 96

[86] M. Bortolozzo, M. Centenaro, R. Focardi, and G. Steel, “Attacking and fixing pkcs #11 security tokens,” in *Proceedings of the 17th ACM Conference on Computer and Communications Security*, ser. CCS ’10, 2010, pp. 260–269. 96