The Pennsylvania State University

The Graduate School

College of Engineering

ASSOCIATION CONTROL IN VEHICULAR WIRELESS NETWORKS

A Thesis in

Computer Science & Engineering

by

Alejandro César Barreto

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

August 2010

The thesis of Alejandro César Barreto was reviewed and approved* by the following:

Guohong Cao
Professor of Computer Science & Engineering
Thesis Advisor

Thomas La Porta
Distinguished Professor of Computer Science & Engineering

Raj Acharya
Department Head and Professor of Computer Science & Engineering
Head of the Department of Computer Science & Engineering

*Signatures are on file in the Graduate School

Abstract

Association control algorithms play a significant factor in the performance of mobile clients traversing a dense wireless mesh network. In this paper, we introduce TAR, a novel association control algorithm which uses trip prediction alongside previous knowledge of the wireless network topology in order to select the APs which the client will associate with. We note that different applications have different wireless requirements. Interactive applications will demand a minimum amount of bandwidth and the least number of handoffs which introduce unwanted hiccups in live service (e.g. VoIP or live video). Non-interactive applications have no time demands but desire the maximum throughput possible. With cross-layer communication, the application can notify TAR about its bandwidth requirements and whether it wishes to perform handoff reduction or maximum throughput. Furthermore, we propose a distributed fairness mechanism around a weighted max-min scheme to allow interactive users to gain priority during the association cycle, perform a sort of load balancing, and act as a quality-of-service admission control. This can either be done with assistance from the AP or through peer-to-peer coordination with a mobile peer-to-peer database of client-AP associations. We then show that these algorithms perform better than some commonly known and deployed association control algorithms, which tradeoff simplicity for poor performance in mobile scenarios.

Table of Contents

List of Figures

List of Tables

Acknowledgements

The author would like to acknowledge Yang Zhang for his invaluable advice and guidance throughout the implementation of the ideas presented in this paper. The author would also want to acknowledge Professor Guohong Cao, for which without his support and ideas, this project would never have been pursued.

Chapter 1:
INTRODUCTION

In the past decade, the popularity and low cost of 802.11 wireless products have ushered a new age of commodity wireless communications. Organizations worldwide have moved toward wireless networks to provide their constituents with network access wherever they may be located. For example, a metropolitan area may provide free Internet access to its citizens downtown and a university may provide wireless access to its students in all its campus classrooms. Wireless networks have also proven themselves in crisis situations. During Hurricane Katrina, relief services established wireless networks to better communicate amongst themselves after the local infrastructure was knocked offline [1].

A wireless network that spans a wide geographic area with a multiplicity of access points (AP) is called an infrastructure-based wireless mesh network. These networks are normally comprised of hundreds or thousands of APs that are stationed to provide the greatest network coverage within a target geographic area. While 802.11 products make great mesh networks due to their popularity among client devices, their limited transmission range introduces new challenges.

The typical range of an AP to client association is 250 meters [2]. If the client is stationary, limited wireless range is a nonissue; the client merely associates with the AP with the highest signal strength. 802.11 dynamic bitrate scaling ensures that associating with APs with a higher signal-to-noise ratio leads to better transmission bitrates. However, recent interest in networks with mobile clients may invalidate this simple association control scheme. For instance, a wireless client on a vehicle moving 80 kph will receive at most 9 seconds of useful connectivity from a single AP [2]. That figure also assumes the AP is road-side and within line-of-sight; the plethora of radio obstacles in an urban environment further reduces this range. Given such constraints, the mobile client must reassociate itself with new APs frequently to sustain network access, placing a burden on applications requiring constant access.

With every AP handoff, the wireless stack must reassociate with the new access point, reauthenticate itself, transfer any state, and notify its applications of the change and possibly new IP address. Handoff durations can range from a few hundred milliseconds to

several seconds, especially when 801.11i authentication is being used. During this time, the mobile client cannot use any network resources. The length and frequency of this sort of service interruption is generally unacceptable for interactive or streaming applications such as VoIP and live video. Moreover, handoffs are also energy intensive thereby serving as an impediment to battery-operated devices. Finally, handoff penalty is constant, so the faster the mobile client is moving, the bigger it will seem compared to actual association time.

In a sparse wireless mesh network, there could be geographic locations where there is limited network coverage. When a mobile client navigates these regions, it cannot do anything but wait until serviceable coverage is available again. Association control algorithms cannot deal with these situations, nonetheless, there has been work in extending coverage by using cooperative, multi-hop, ad-hoc networks amongst the mobile clients [3]. On the other hand, in a properly-designed, dense wireless mesh network, a single geographic location may be covered by several APs. Often times, these regions are along the fringe of connectivity of the APs in question. These regions provide ample opportunity for a mobile client to reassociate itself with a new AP in its current trajectory; these are known as handoff regions.

When there are many APs to choose between, association control algorithms must be smart enough not to select APs that will lead to unnecessary handoffs in the future. This decision is especially difficult when the client does not know where it is in relation to the APs or where it expects to go. At the same time, it is desirable to exploit dynamic bitrate scaling and select APs with the greatest signal strength. As network applications evolve to consume greater bandwidth, it is important for an association control algorithm to also ensure that the AP it is connected to can supply the needed bitrates. But if an AP is satisfying a client with consistent bandwidth demand, there is no need to reassociate even if there are APs that can provide a better bitrate.

The tradeoff is clear: To maintain high bitrates, it is necessary to switch networks often so that the client has the highest possible signal strength at every location along its trajectory. Along the same lines, in order to reduce the number of handoffs, it is necessary to sacrifice the overall signal strength as the client will choose to stay connected to far-away access points for longer periods of time.

In addition, an association control must be mindful of other clients and the topology of APs in the immediate area. If every client is making the same decision about which APs to associate with, then they will all suffer from increased contention and overloading at those APs. Similarly, if a flash crowd occurs, clients will need to coordinate in some fashion to distribute their collective load on all the APs in the vicinity. Flash crowds are not uncommon; e.g. one can occur at a busy intersection blocked with a lengthy red light. This paper defines fairness as distributing the total available bandwidth in an area to all clients in an equal manner, while placing an importance to servicing those with a minimum bandwidth requirement.

In this paper, we introduce Trajectory-Aware Roaming (TAR), an association control algorithm that optimally handles this tradeoff and performs especially well in vehicular wireless networks. With location information (GPS) and channel scan information, each vehicle creates a wireless coverage map of the roads it travels. Using trip prediction, each vehicle can map out which APs will be visible in the foreseeable future in order to make the best decisions about how to switch between them. The application also communicates with the wireless stack in a cross-layer scheme to define network requirements, such as bandwidth and handoff scheme. TAR will then maintain an actively-updated AP association schedule that will satisfy application requirements for the duration of the user's trip. Furthermore, TAR employs weighted max-min to perform load balancing, a process that could affect AP selection. Two distributed weighted max-min implementations are suggested: allocating bandwidth to individual clients at the AP, and allocating bandwidth to clients in a peer-to-peer manner over a vehicular ad-hoc network with a novel mobile database.

Related Work

Association control algorithms play an important part in network performance. The majority of wireless clients will collect AP beacons for a short duration and associate with the one with the highest signal strength. The client can also reject APs that are too busy [4]. In [12], clients and APs could cooperate to satisfy bandwidth requirements for all users, sometimes advising clients to physically move. In the real world, however, the wireless stack cannot expect the driver to change his route to achieve better wireless performance.

In [11], a centralized algorithm provides max-min fairness by assigning users to APs. While a centralized algorithm can benefit from global knowledge, it is impractical to implement and requires existence and maintenance of such a service. Clients could also maintain different associations for unicast and broadcast communications to exploit the differences between them [10].

Reducing the time of handoffs has been studied in MIT's CarTel project by tuning timeout and retry parameters to vehicular speeds, performing association and authentication asynchronously, and scanning frequently used channels first [8]. In [9], clients perform multiple associations according to a centralized selection procedure. In addition, a new 802.11 amendment, 802.11r-2008, aims to provide faster BSS transitions on 802.1X-based authentication systems by reducing the number of messages needed to reassociate [14]. These techniques require significant changes and are anyway orthogonal to the improvements in AP association discussed in this paper.

MobiSteer uses a steerable beam directional antenna and a history of traces to select the beam that leads to the highest signal strength, while reducing handoff time [5]. However, this is only in an attempt to maximize throughput by beam selection; interactive or streaming network applications such as video conferencing will suffer from the *number* of handoffs. Furthermore, it is not always necessary to have the maximum amount of throughput, rather, the necessary amount required by the application. One IBM work argues that optimal handoff reduction will maximize the expected association time, and provides a rudimentary association control algorithm based on average association time given the set of APs currently visible [6]. But these algorithms do not maximize throughput.

Trip prediction is discussed in [7] in an effort to efficiently route query packets over a vehicular ad-hoc network (VANET). Another prediction scheme based on a Markov-modeled mobility grid, Breadcrumbs, provides connectivity forecasts and notes the importance of keeping an individual history over centralized efforts [13]. Mobility prediction is discussed in [15] to predict which APs will be visible to avoid the time it takes to scan either activity (during associated time) or passively.

Chapter 2:
TRAJECTORY-AWARE ROAMING (TAR)

Given mobile client roaming through a mesh wireless network, the TAR association control algorithm actively predicts the client's route and selects new APs along the way in an effort to maximize throughput while minimizing handoff time (*throughput maximization mode*). We can also configure it to minimize the total number of handoffs since handoffs during interactive applications have penalties not associated with throughput, such as delays, unwanted interruption of service, or even connection failures (*handoff reduction mode*). TAR optimizes three metrics: throughput, handoff time, and handoff count. An association control algorithm that covers all three has yet to be studied.

The are many benefits in knowing the client's route. For example, an AP with high signal strength may only be in range for a short time due to physical obstructions in the environment. Meanwhile, there could be an AP with weaker signal strength but that would be available for a longer time.

The best example would be a commuter driving to work along a straight road. At any instant, her wireless client could scan and connect to the access point with the highest signal strength. On average, this commuter would have 125 meters of coverage before she would have to find a new access point. Now, over a period of several trips to-and-from work, her wireless client has saved enough scan histories along this road. Upon an association decision, her client may select an AP with a weaker signal but that would be available for twice as long as she moved closer to the access point and then away. Instead of 125 meters of coverage, she now has 250 meters, reducing the number of handoffs.

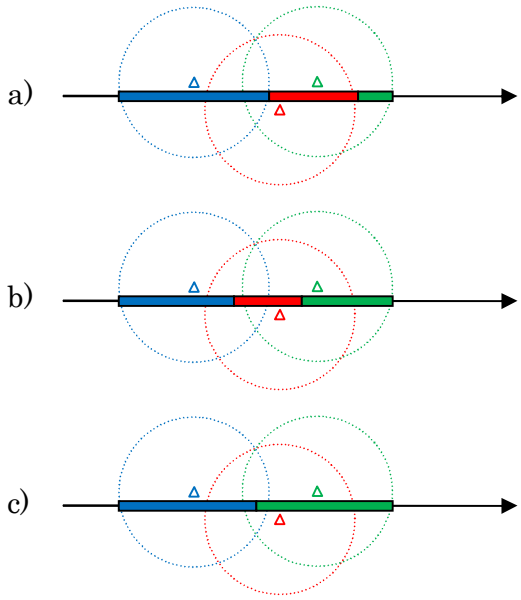

Figure 1: A vehicle moving through a region with three APs. (a) select-high-and-retain association control; (b) actively-select-high; (c) TAR.

Figure 1(a) shows *select-high-and-retain*, an association control which selects the AP with highest signal strength and retains it until it is no longer available (most popular scheme, termed). It performs two handoffs. The last AP may not even be usable by the time it has completed the association because of low signal quality and bitrates. Subfigure (b) shows *actively-select-high*, an association control that is actively scanning and selecting the AP with the best signal strength, and it also performs two handoffs. Subfigure (c) shows TAR in handoff reduction mode efficiently selecting only two APs for one handoff. By the time the second association is complete, it will have very good bitrates for a longer period of time than the other two algorithms. TAR would also perform similar to (b) if handoff reduction mode is off, depending on the handoff time penalty.

Furthermore, if the client knows it will be spending an average of 40 seconds at an intersection, it may as well disconnect from an AP with poor signal quality and reassociate itself with an AP with higher signal quality for the duration of the intersection stay. Otherwise, in a naïve handoff reduction algorithm, the client would have remained associated with the weaker AP because it may not have crossed the threshold for roaming.

Knowing the client's trajectory means it can do *pre-emptive* roaming to improve overall quality and throughput, especially if a handoff is required anyway.

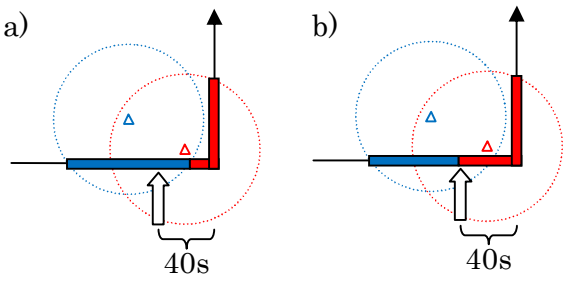

Figure 2: A vehicle stopped for 40 seconds before an intersection. (a) A naïve handoff reduction algorithm; (b) TAR.

Figure 2(a) shows a naïve handoff reduction algorithm that wastes an opportunity for higher throughput because it based on short-term handoff reduction. Subfigure (c) shows TAR performing a pre-emptive association to the second AP to take advantage of 40 seconds of high throughput while waiting at the intersection. This would happen regardless of which mode TAR is configured under: throughput maximization or handoff reduction.

Cross-Layer Cooperation

To determine how TAR should be configured, we shall categorize applications as either *interactive* or *non-interactive*. A cross-layer communication between the application and the wireless stack will establish these requirements. For example, if the application is being used interactively such as in VoIP, it may demand a minimum amount of bandwidth and smallest number of handoff interruptions possible (handoff reduction). On the other hand, if the network is being used non-interactively such as in email or in a system upgrade, the application can request the highest overall bandwidth while optimizing handoffs for smallest download time (throughput maximization).

Communicating with the association algorithm in such a manner ensures that it will make the best decisions based on how the network will actually be used. Even in an *actively-select-high* association control algorithm, a cross-layer interaction can ensure that

the client does not attempt to reassociate at every possible moment, but instead only when the minimum bandwidth requirement has been violated.

In Linux, a sort of cross-layer communication already exists in the power management module. The pm-qos kernel module allows applications to specify their desired network throughput and latencies so the network drivers can power their devices adequately when necessary and power them down when they are not needed [16]. A similar protocol can be used to guide an association control algorithm to better decisions.

Trip Prediction

The next essential step is predicting the client's route. It can be easily observed that vehicles generally follow common everyday routes; e.g. work, school, gym, friend's house, etc. Even buses have a specific route they follow every day. The Mobi*drive* project found these sorts of regular activities in the 30,000 trips performed by 320 correspondents over a six-week study [17]. This fact allows TAR to predict where the client will be in the future to make better association decisions.

TAR presents a new way of trip prediction based on the decisions made at every junction or intersection. This has the advantage of being fast and saving storage space over grid-based approaches like [13]. Furthermore, it does not explicitly predict your destination as in [7], opting to perform incremental prediction instead. Therefore it does not rely on a routing algorithm to determine a future route, which could be wrong if the user takes shortcuts or other colloquial paths. Also, if the vehicle begins to take a different route halfway through the trip, it makes no difference as this algorithm will continue to incrementally predict. Lastly, our prediction algorithm tracks travel time.

Building the Trip Database

The prerequisites for the client are a GPS device and a map of the area. Many mobile clients are now equipped with GPS by default, such as the iPhone or navigation systems in cars. The map may be preloaded on the client, or downloaded on demand and cached locally from a central service. The map is specially parsed into roads, lanes, and junctions. A road is any path between two junctions in which a vehicle cannot stray. A set of lanes is a

direction along the road; a one-way road only has one set of lanes while a two-way road has two sets of lanes. A junction is any location where two or more roads meet, such as a stop sign, traffic lights, or a highway exit. In other words, the map is parsed into a graph where roads are edges, lanes represent the edge directions, and the junctions are nodes. Roads retain information about their type (service, residential, tertiary, secondary, primary, motorway), name, and their speed limit.

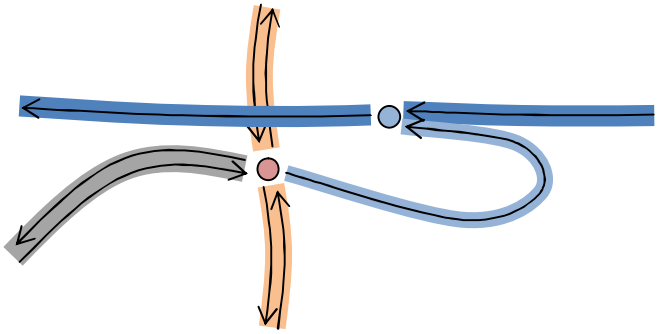

Figure 3: Excerpt of a map graph. It contains a primary road leading underneath a westbound highway segment, with a traffic light connecting a residential road an on-ramp to the highway.

The algorithm keeps track of lane changes as the client leaves a junction by incrementing a counter associated with the change *previous lanes* → *current lanes*. It also ages all other possible changes from *previous lanes* to ensure that new routes will eventually replace old routes. Additionally, it keeps track of the duration of travel spent along a set of lanes with an exponential moving average. Two duration averages are kept: one for pass-throughs and another for stop times. Some junctions could affect the duration of travel on incoming lanes in a significant, variable manner, such as a traffic light. In these cases, the client keeps track of junction approach velocity. If the vehicle's approach velocity as it nears the junction (within 50 meters) is below a threshold (16 kph), then it is determined that the vehicle is stuck at the intersection and the duration is averaged into the stopping duration. Otherwise it is averaged into the pass-through duration. The stopping frequency is also stored. Some traffic lights are rarely green, such as a residential road leading into a primary road.

All this data is binned into time slots since trip information could be very different in the morning or in the afternoon. Specifically, we make each bin 2 hours long: e.g. 8am-10am, 10am-12pm, 12pm-2pm, and so on for 24 hours.

Predicting a Route

To predict the route, the algorithm pulls data from the current time slot. Given the current set of lanes, it will select its next set of lanes based on the change that has the highest probability (or counter). Then it will predict how long the vehicle will spend on those lanes by the following equation:

$$duration = stop_{freq} \times stop_{duration} + (1 - stop_{freq}) \times pass_{duration} \qquad (1)$$

Since the trip prediction algorithm may be called at any time, durations are interpolated for the current road if the vehicle is already travelling on it. This is especially useful to correct travel times as a vehicle approaches a traffic light. If the vehicle is stopped, then $stop_{duration}$ is used outright instead of equation (1), and likewise for $pass_{duration}$ if the vehicle does not seem to be stopping. Lastly, if the predicted time for the current set of lanes is less than the minimum time it would take to get there given the client's current speed and acceleration, then the duration is bounded by this minimum time instead.

The lane prediction procedure may be run many times to produce enough future points as necessary.

```
TripRediction Algorithm:
  → Given (Lane, DistanceOnLane, Status):
  → Initial: TotalDuration = 0, TotalDistance = 0
  → Returns: [(NextLanes, Duration), … ]

If lane == None:
  Return [(None, 0)]

LD = LaneDurations[Lane]
Case status:

  When STOPPING:
    Duration = LD.Stopping

  When Status == PASSING_THROUGH:
    Duration = LD.PassThrough

  When Status == UNKNOWN:
    Duration = LD.StopFreq * LD.StopDuration +
               (1 - LD.StopFreq) * LD.PassDuration

DistanceLeft = Lane.Distance - DistanceOnLane
Duration *= DistanceLeft / Lane.Distance

TotalDuration += Duration
If TotalDuration > MAX_DURATION:
  Return [(Lane, Duration)]

TotalDistance += DistanceLeft
If TotalDistance > MAX_DISTANCE:
  Return [(Lane, Duration)]

NextLanes = Maximize Counter in LaneChanges[Lane]
Return [(Lane, Duration)] +
       TripPrediction(NextLanes, 0, UNKNOWN)
```

Figure 4: TripPrediction Algorithm.

Default Prediction

When vehicles are moving along new routes, the prediction cannot go further than the upcoming junction due to lack of data. In these situations, the algorithm will look at other time slots for the strongest prediction data for this route (the highest counter). Otherwise, the algorithm assumes the vehicle will continue to travel in a straight-line trajectory. The predicted route will continue on the set of roads that share the same name and lane direction. This default prediction may eventually be wrong; however, as demonstrated in chapter 3, making a prediction allows for more scan histories to be considered and leads to better algorithm performance.

Creating a Scan History

Besides route information and application requirements, TAR needs to know about the set of APs it will encounter along its path. When the wireless client is idle, it will scan the

wireless medium for APs and tag this scan on the spot on the road it is currently travelling along. This is a continuously running process. Since a scan takes several seconds to complete, a comprehensive coverage map of a road can take several passes. Nevertheless, it can be observed that vehicles travel on habitual routes and over time such a comprehensive coverage map could be compiled. This means that the routes a vehicle travels often will be well covered.

Alternatively, a coverage map may be downloaded on demand and cached locally, or it can be shared amongst vehicles on the road over an ad-hoc network. It does not matter how it is created or obtained, but it is necessary for AP selection, otherwise, the algorithm degrades to either select-high-and-retain or actively-select-high. The former is used in handoff reduction mode, the latter in maximum throughput mode.

Selecting APs

The goal of the AP selection process is to create a schedule for AP associations containing the distances or coordinates along the travelled roads when the client should switch to the new AP. First, trip prediction is performed to obtain 300 meters or distance of 40 seconds of duration, whichever one leads to the greatest predicted distance. Next, it collects all the scans it stored along the predicted route, and calculates from the prediction data how long it will spend in the vicinity of each scan. If no scan is available for a route segment, a null AP with zero bandwidth capacity is assigned for that length.

Each scan represents a layer in a directed-acyclic graph, with complete bipartite connections between each layer. Each layer is comprised of the APs seen during that scan, with incoming edges from the previous layer and outgoing edges to the next layer. There are two additional layers that are added artificially: a *start* and *end* layer with a single AP each. The start layer contains the currently associated AP. The end layer contains a null AP. See figure 5 for an example.

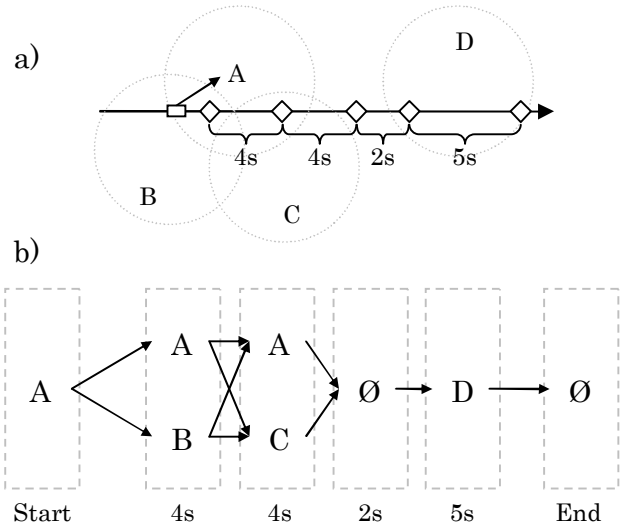

Figure 5: A vehicle travelling along a road with a scan history, and the corresponding graph. (a) A vehicle—currently associated with A—travelling along a road with previously recorded scans a given time interval from each other. This time interval is calculated by the trip prediction component. (b) The corresponding graph using 4 future scans. Each scan is a layer in the graph with complete bipartite connections between all their APs. The Ø symbol represents a null AP.

The weight for edge (u, v) represents the benefit that is obtained by using AP v for the duration of its scan, coming from AP u. More specifically:

$$B(u,v) = v_{bitrate} \times \left(v_{duration} - H(u,v)\right) - M(u,v) \quad (2)$$

$v_{bitrate}$ in equation (2) is estimated from v's signal strength during the scan. It can also be learned from experience. For example, a client may learn that AP has poor performance because of an inadequate backhaul connection. Instead of relying solely on signal strength, the client may update the AP's $v_{bitrate}$ from the client's past experience.

The function H(u, v) is the handoff time penalty for switching between APs u and v. If $u = v$, there is no penalty; otherwise, it is the time spent to reassociate to v and begin useful communication. This function may be variable depending on AP v as different APs may have different association performance. It also may depend on u if switching between

APs in the same network is quicker. This function may be hardcoded, downloaded from a central authority with knowledge of association performance, or learned from experience.

The function M(u, v) is the mode penalty for handoffs, which will make TAR favor either maximum throughput or handoff reduction. In maximum throughput mode, equation (3) is applied. In handoff reduction mode, equation (4) is applied.

$$M(u,v) = 0 \tag{3}$$

$$M(u,v) = \begin{cases} -K & u \neq v \text{ or } v_{bitrate} < min_{bitrate} \\ 0 & otherwise \end{cases} \tag{4}$$

In equation (4), the quantity −K is a very large negative constant used to give a large incentive for the algorithm to choose the least number of handoffs when maximizing the benefit. It is applied when there is a handoff or when the bitrate for AP v in this scan would not satisfy the application-specified minimum bandwidth requirement. Nonetheless, between two different AP schedules with the same number of handoffs or unsatisfactory scans, it will optimize for maximum throughput because –K is so largely negative and because they will cancel out in equally bad paths.

Maximizing Benefit

To maximize the benefit in the AP graph is an instance of the well-known longest path problem. It can be solved in linear time using dynamic programming. We will feed the algorithm described in figure 6 a list of APs sorted in ascending topological order by layer (the first layer comes first, then the second, etc). The same AP in two different layers do not share the same graph node—they are two different AP objects in different scans, but compare equally for handoff penalty and mode purposes.

```
MaximizeBenefit Algorithm:
  → Given (topologically sorted APs):
  → Initial: Benefit[*] = -∞, Path[*] = None

For u in APs:
  For each edge u → v:
    If Benefit[v] ≤ Benefit[u] + B(u,v):
      Benefit[v] = Benefit[u] + B(u,v)
      Path[v] = (u, coordinate for u)
```

Figure 6: MaximizeBenefit Algorithm.

To determine the final AP schedule, we follow *Path* backward from the final null AP in the end layer. We then have a list of APs in the order we will associate with them and when to make the switch.

When to Select APs

The selection algorithm is a computationally intensive process, especially if there are many APs in every scan. However, it only needs to be run when the distance left in the schedule is less than some minimum threshold (100 meters), if a road on the predicted route was incorrect, or if road conditions change. A condition change could be a deacceleration when approaching a known traffic light intersection.

Weighted Max-Min Fairness

To implement fairness, TAR implements a weighted max-min algorithm to distribute load among APs in the area, with coordination with the APs or with adjacent vehicles over a vehicular ad-hoc network. With max-min, the client must advertise its bandwidth request. Max-min equally divides the available bandwidth among all clients while redistributing surplus bandwidth to clients that requested more than the equal allocation. Each client is allocated at least a minimum share, but never more than it requested. Max-min is fair for the majority of clients who request similar amounts of bandwidth while attempting to satisfy clients with large requests; it *max*imizes the *min*imum share of a client that is not fully satisfied [18].

A weighted max-min algorithm may give higher priority to the bandwidth requests of some clients. Each client now advertises a weight alongside its bandwidth request; the higher the weight, the better satisfied this client will be compared to clients with smaller weights. For a detailed explanation, the reader is referred to [19]. In this implementation, a higher weight is assigned to clients in handoff-reduction mode since those are the clients who are interested in fewer interruptions and TAR should attempt to guarantee their bandwidth over other clients. That means that a VoIP application will have a higher weight than simple downloads and will have a higher priority over an AP's bandwidth.

It is the client's responsibility to restrict its transmission to its allocated share; if every client does this then there will be less overall contention on the wireless channel. The appendix contains a comprehensive proof showing that if clients follow a max-min bandwidth allocation, it will always perform better than the naïve random allocation scheme, as would be observed if neither the client nor the AP operated with any quality-of-service mechanisms. It is important to note that this type of fairness is orthogonal to 802.11e (commonly known as the WiFi Wireless Multimedia Extensions). 802.11e provides quality-of-service to clients based on their advertised priority class once the client is associated to the AP. This usage of max-min fairness is primarily used to determine how well an AP can satisfy a client's bandwidth request *before* the association takes place, acting as an upper limit for $v_{bitrate}$ in equation (2). A highly loaded AP will have a lower max-min calculated upper limit for $v_{bitrate}$ and therefore, clients will tend to associate with other APs, achieving a form of load balancing. It is important to have admission control at the AP that distinguishes between different types of applications; a weighted max-min approach does just that. Allowing a best-effort type of client into a moderately loaded AP is more acceptable than allowing a client that requires a minimum bandwidth requirement. Performing quality-of-service mechanisms only after the client has associated itself with the AP is inefficient for the client and unfair for the previously associated clients.

Note that even though the client may not know this upper limit for every AP in its foreseeable future, if it knows the upper limit for every AP in its current vicinity, then it will be able to make a locally good choice achieving incremental load balancing in its current location.

AP-Centered Max-Min

The most obvious way to implement max-min fairness is at the AP. Some APs already broadcast their current load, but with weighted-max-min, the algorithm must be re-run to take account different client weights. Upon an association decision, a non-associated client can ask all the APs in its vicinity the amount of bandwidth it would be allocated if it were to associate with it, given its bandwidth request and weight. The APs, knowing their bandwidth requests and weights of the clients currently associated with it, will re-run the weighted max-min algorithm including the requesting client's bandwidth request and

weight to determine the new allocations. Each AP will then reply with the client's bandwidth should it associate with it, serving as the upper bound for $v_{bitrate}$. This is the simplest scheme but requires modification to the 802.11 protocol.

Peer-to-Peer Max-Min

Another approach that has not been studied in the literature is to allow clients to coordinate in sharing a common dumb resource instead of relying on an authority to allocate shares. In this case, vehicles will coordinate among themselves over a wireless ad-hoc network to share AP bandwidth, which has no knowledge of such a sharing scheme. Basically, each client will need to know the bandwidth requests and weights of every other client currently associated with the APs it is interested in. This is a challenge if the AP is not involved since clients may not be in wireless range of each other. Without any loss of generality, this paper presents an ad-hoc mobile database to solve this problem while reducing the overall number of coordination messages over the ad-hoc network. Truly, the same mobile database can be used to share other data such as traffic conditions on a road, accident information, or even to extend the range of a particular stationary node (e.g. a movie theater advertising its movie show times to nearby vehicles). It is useful for local information which must be shared by all clients within the geographic area.

Data in this mobile database is segmented into contexts. A context is a group of client data that is available to all clients interested in that context. In this case, a context is an AP and the data are the bandwidth requirements and weights of each client associated with the AP—although contexts and data can be interpreted to mean anything in a general sense. To post data into a context, a client must be interested in that context, but interested clients are not required to post data. For example, associated clients will post their bandwidth data, but clients that are not yet associated with that AP may become interested in that context—thereby receiving all the context data—to run max-min to calculate $v_{bitrate}$. One simple approach is that each client becomes interested in the contexts for the APs that are currently in range. When an association decision comes up, it will already have the bandwidth data for the clients associated with the surrounding APs in order to calculate $v_{bitrate}$.

Coordination within the Mobile Database

A requirement of this mobile database is the knowledge of each client's neighbors and the contexts they are interested in. Clients may transmit this data within a beacon packet in common practice with wireless networks. To post data into a context, a client will flood an *update*-packet containing its data to all the clients interested in the context. An *update*-packet contains the context, previous receivers, source client, and the source's data for the context. The previous receivers list contains all the clients that *update*-packet has seen during its flooding. This is to prevent future clients from sending the *update*-packet to a client that has already received it, and reduces the number of retransmissions. Clients may discard any *update*-packets it has already seen, but it must at least store the most recent *update*-packet from each source client that has transmitted one. Those clients uninterested in the context have no obligation to forward packets to other interested clients in their vicinity, but they may choose to do so altruistically if their wireless devices are idle.

To enter a context, a client sends a *sync*-packet to the neighboring clients that are interested in that context. The *sync*-packet simply requests the client to retransmit all the *update*-packets it has received for that context. If this client is bridging two previously unconnected groups of clients interested in the same context, the *update*-packets will be forwarded to the opposite group of clients by the nature of the flooding technique. This ensures the property that all clients interested in a context have the entirety of the data associated with that context, as long as they there is a wireless route to each client.

To leave a context, a client simply removes the context from its interested list. It will no longer be forwarded *update*-packets and it may discard all its stored packets. If it had posted data to the context at some point, it should send an *update*-packet with null data to remove its data, unless the data has an expiry date and/or it is unnecessary to void the data.

Transmission Reduction Optimizations

One optimization to further reduce the number of messages transmitted during a flood requires one-hop knowledge of every client's neighbors. Upon a retransmission, clients will add all its neighbors to the previous receivers list, and only select a few clients to actually

retransmit the message to their neighbors. The choice of which neighbors will retransmit the *update*-packet to their neighbors is the minimum set cover problem, which is NP-complete, but can be approximated with a greedy algorithm [20]. A further optimization is to only send the *sync*-packet to clients that have differing sets of data (the stored *update*-packets). This can be done by broadcasting a hash of the data for each context a client is interested in with the beacon packet. Clients wishing to enter a context would only send *sync*-packets to clients with different hashes.

Chapter 3:
EVALUATION

A simulator was used to evaluate TAR against other common association algorithms, specifically select-high-and-retain (SHR) or actively-select-high (ASH). Three requirements were needed from a simulator:

1. Since a major component of TAR is trip prediction with individual vehicle histories, vehicles must have realistic travel patterns day-after-day. Each vehicle must behave as it would if it were commuter, student, or any other category of driver. Furthermore, it should select destinations from a pool of possible locations for that driver (pool locations can be updated over time). For example, a driver may only have one work location but three lunch locations.

2. The simulation of hundreds or thousands of vehicles with unique travel patterns to test the fairness aspects of TAR. In the absence that many real-world vehicle traces, a simulator must create realistic patterns.

3. Since TAR also does time prediction to determine when to proactively switch APs, vehicles must have realistic traffic behavior complete with lane changing, stop signs, traffic lights, yielding junctions, acceleration, deacceleration, varying types of driving styles (e.g. convervative vs. aggressive), all the while preventing vehicle collisions and maintaining some level of inter-vehicle distance depending on driver type.

No simulator was found that satisfied these requirements to an adequate level, so a new discrete-time simulator was developed. Maps are exported from the OpenStreetMap project [21] in their open XML format, parsed into a graph of intersections and roads, reduced to their largest strongly-connected component, and saved into an intermediary file format for fast future access.

Simulation Parameters

Any OpenStreetMap map may be used in the simulator; we chose downtown Lancaster, PA and State College, PA as graphed in figure 7. We used the smaller State College map for fairness simulations to increase the amount of AP contention. The simulator is also fed configuration data to create realistic travel patterns modeled after the distributions found from studying 30,000 trips in the Mobi*drive* project [22]. Figure 8 shows the trip simulation of a single worker and student vehicle, simulated for a period of 365 weekdays (weekends were not considered in this simulation). Different vehicles may have slightly different distributions; e.g. different worker types may leave for work at varying times. The simulator then runs whatever experimental module the user desires. Common simulation parameters are enumerated in table 1. All simulations were run using the same random seed to ensure that results are comparable.

a)

b)

Figure 7: Maps used in our simulation (not shown to scale). (a) Map of downtown Lancaster, PA, used in our simulation, shown without vehicles. Triangles represent APs; orange circles are stop signs; red circles are traffic lights; all other junctions are yielding. (b) Map of small area of downtown State College, PA.

Table 1: Common simulation parameters. None of these parameters are known by the client. Interactive trips stands for the percentage of trips that are done in handoff-reduction mode.

Name	*Value*
Path Loss Exponent [23]	3.35
Receiver Sensitivity	-91.0 dbm
AP Bandwidth Capacity	8.0 Mbps – 11.0 Mbps
Association Time	0.1 secs – 2.0 secs
AP Count (randomly placed)	1,000 for Lancaster, PA 150 for State College, PA
Vehicle Number	1,000 for Lancaster, PA 400 for State College, PA
Interactive Trips	50%
Interactive Bandwidth	1.0 Mbps – 3.0 Mbps
Interactive Max-Min Weight	3.0
Interactive Mode	Handoff Reduction
Non-interactive Bandwidth	3.0 Mbps – 7.0 Mbps
Non-interactive Mode	Maximum Throughput

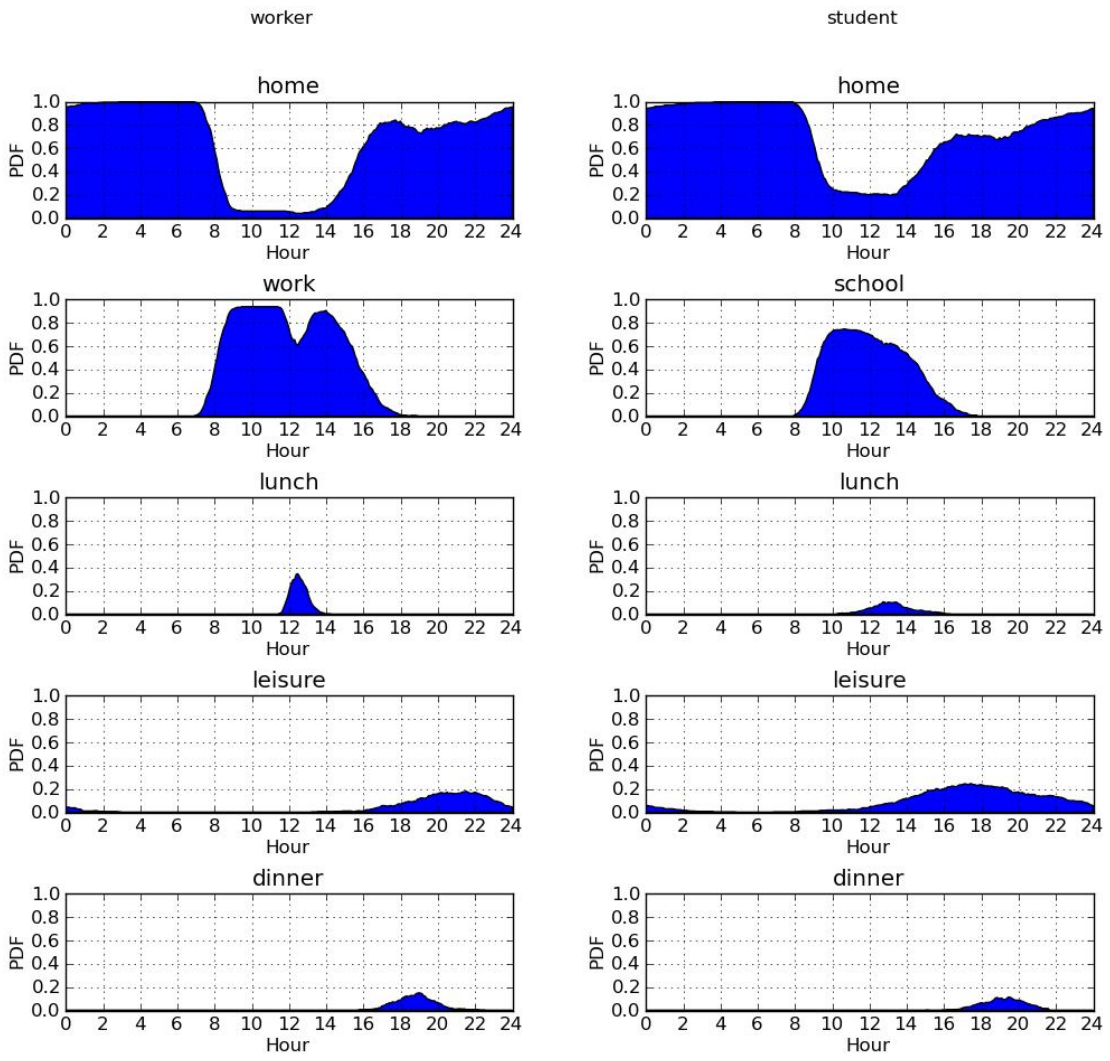

Figure 8: Travel pattern distributions for a worker and student vehicle.

To simplify the simulator, we assumed vehicles could generate comprehensive scan histories for every road on a single pass. In other words, scan histories were available for every 15 meter stretch after a single pass on the road, making the results presented in this paper for TAR how it would perform in the ideal case. Generating a comprehensive scan history in a single pass is not possible in practice unless the vehicle is moving slowly, but it does not change the fundamental results of our experiment. Instead, TAR would perform incrementally better until it created a comprehensive scan history over several passes on the road.

Trip Prediction

Figure 9(a) shows the prediction accuracy of the trip prediction component of TAR. Recall that default prediction entails looking at trajectories in other time slots or estimating a straight-line trajectory on the current road if no prediction data currently exists. Subfigure (b) shows prediction accuracies for the travel duration on each road.

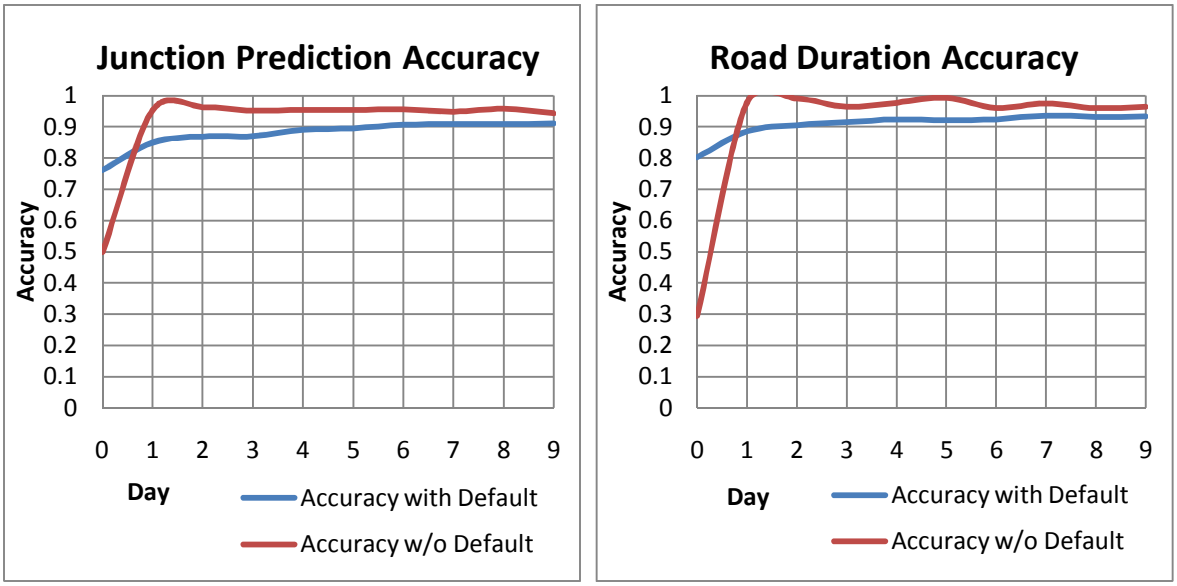

Figure 9: Prediction accuracies for the next road at every junction.

Trip prediction with default prediction is expected to perform worse than prediction without it because it is almost always going to be wrong at some point in the future; e.g. vehicles do not typically maintain a straight-line trajectory without any turns during their trips. Nonetheless, performing default predictions allows TAR to select more scan histories in the foreseeable future; without it, it would not select scan histories beyond the last available prediction. The more scan histories that are available, the further in the future TAR can see to determine the best association path. If a prediction is wrong, TAR simply reevaluates its path under its new trajectory. Table 2 demonstrates that TAR with default prediction performs better than without it, as compared to the optimal TAR algorithm which uses the actual route of the vehicle instead of predicting it. In practice, the actual route of the client

is generally unknown to the association control algorithm, unless the driver has programmed it into a GPS; e.g. for longer trips or to previously untraveled destinations.

Table 2: TAR performance with different trip models.

Algorithm	*Satisfaction*	*Association Duration*
TAR-TP w/o Default	0.776	8.97
TAR-TP w/ Default	0.805	9.81
TAR-ROUTE	0.826	10.61

Non-Interactive Applications on TAR

Figure 10(a) shows the satisfaction performance of the three association algorithms evaluated. Satisfaction is the ratio of bandwidth received vs. bandwidth requested. The satisfaction ratio is computed for every association performed by every client. TAR outperforms in terms of satisfaction because it is able to predict which APs a client will encounter in the future, and chooses the best association path to maximize throughput in non-interactive mode. Subfigure (b) shows the association duration—the higher the association duration, the fewer the number of handoffs the association control algorithm forced. SHR outperforms all other algorithms because it is designed to retain an association until it is no longer available; however, that design choice is not conducive for satisfaction as is seen in subfigure (a). ASH has the worst association duration because it is frequently reassociating to maintain the highest signal strength between a client and an AP. Recall that it is inconsequential how TAR performs in association duration in non-interactive mode since it was throughput that was being maximized (satisfaction) and not the number of handoffs (association duration).

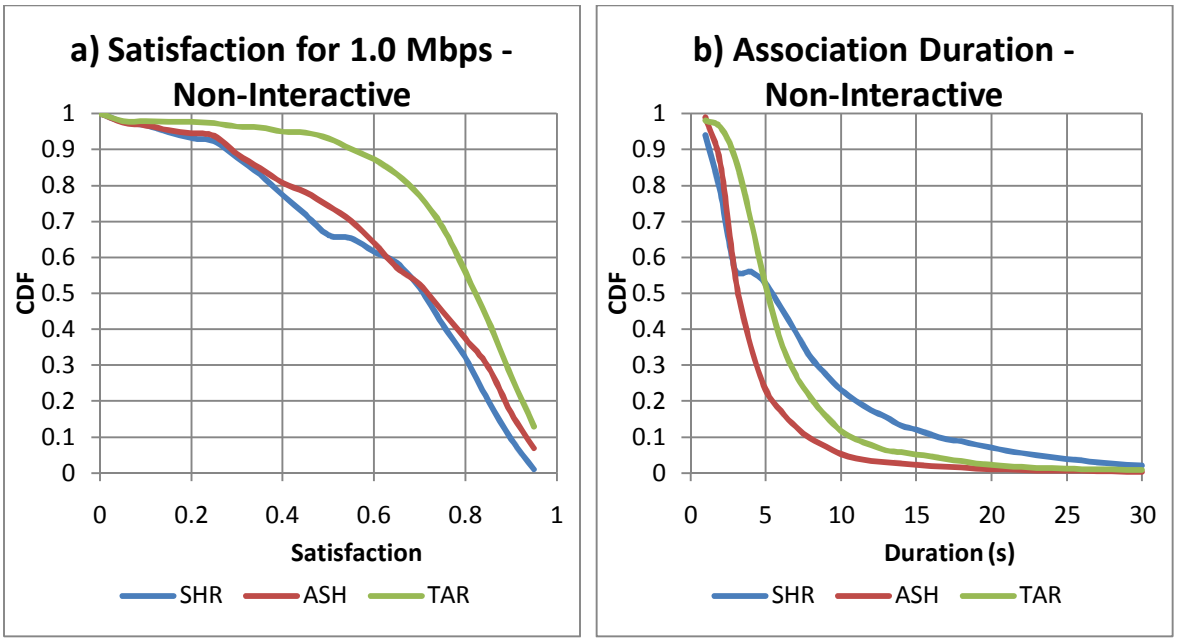

Figure 10: Performance of association control algorithms for non-interactive applications.

Interactive Applications on TAR

Figure 11(a) shows the satisfaction performance of the three association algorithms while subfigure (b) shows the association duration performance for applications in interactive mode. It is expected that SHR and ASH will perform similarly in interactive mode vs. non-interactive mode since they make no distinction about the application using the network resources. However, TAR's cross-layer communication allows the association algorithm to reconfigure itself easily to interactive applications. TAR here is configured in handoff reduction mode and the association duration for TAR is far greater than those of SHR and ASH. At the same time, TAR also performs better in satisfaction because while it is primarily optimizing for handoff reduction, it also performs throughput maximization as a secondary goal. Moreover, the longer association durations allow a relatively larger transfer window within an AP's coverage, especially with larger bandwidth requirements (i.e. lower association overhead vs. usable time).

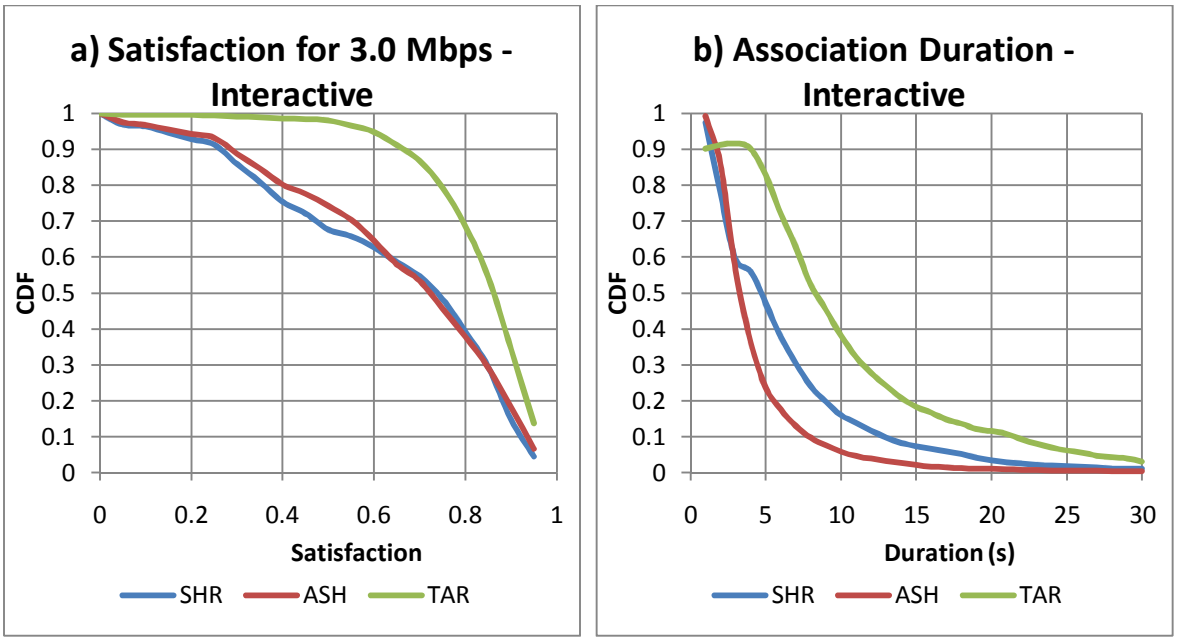

Figure 11: Performance of association control algorithms for interactive applications.

It should be argued that the differences between handoff reduction mode and maximum throughput mode for TAR become more pronounced as the AP association time decreases. As AP associations become quicker, the frequent switching of the maximum throughput mode will lead to greater satisfaction ratios over handoff reduction as the association overhead goes to nil. Certainly, there has been a trend toward faster AP associations as evidenced by the 802.11r effort [14]. At the same time, TAR in handoff reduction mode will always be needed to prevent incessant, unwanted hiccups in service for interactive applications such as VoIP or live video.

Fairness

To evaluate the fairness component of TAR, we used a smaller map (State College, PA) and ensured that there would be enough traffic to ensure a heavy load on the APs in the area. We evaluated SHR, ASH, and TAR without any modification, and added TAR-P2P and TAR-AP, the two distributed max-min schemes discussed in chapter 4. Figure 12(a) shows the satisfaction for non-interactive uses; subfigure (b) shows the satisfaction for interactive users. As can be noted, the satisfaction for the three unmodified association control

algorithms run similar performance specifications because the area is heavily loaded and only the closest APs to the road are being used. These algorithms have no notion of other clients or the load on the AP. On the other hand, TAR-P2P and TAR-AP show considerable improvement in the satisfaction for interactive users. Recall that these algorithms perform load balancing to spread the load among all the APs in the area, giving preference to clients with interactive applications. Although the satisfaction for non-interactive clients is negatively affected (TAR-AP or TAR-P2P vs. regular TAR), a significant positive change in interactive client satisfaction is observed—in accordance to our goals.

TAR-AP serves as the upper bound for TAR-P2P since in TAR-AP, the AP manages the max-min algorithm and is always present for any client that wishes to request a bandwidth estimate. In TAR-P2P, a context may be split into several unconnected groups of clients outside collective wireless range which therefore fail to share bandwidth information because they cannot communicate with each other. Nonetheless, that is a rare occurrence as TAR-P2P performs very similarly to TAR-AP.

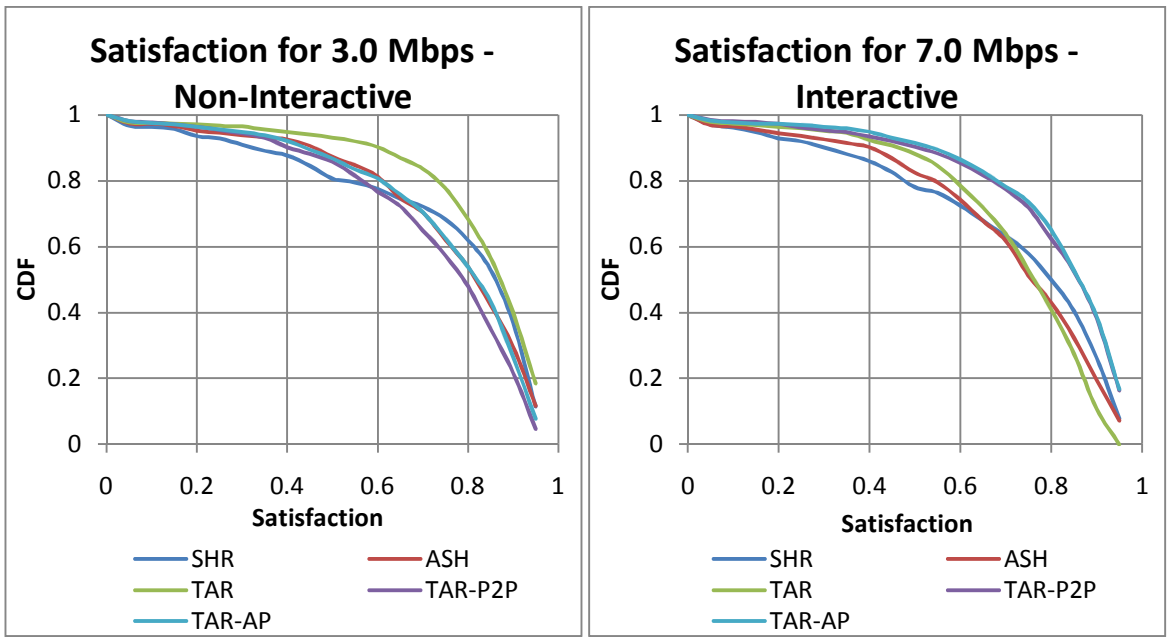

Figure 12: Satisfaction for association control algorithms in a heavily congested area.

Figure 13 shows the association duration for the association control algorithms. The max-min versions of TAR perform better in association duration with non-interactive clients

because before an AP is used from a previously determined AP association path, the path is reevaluated from that point using the current load conditions in that location. Since the area is heavily congested, it is less likely for TAR to force a handoff between two heavily-loaded APs if the client still has a good connection with the original AP. This is due to trying to reduce the handoff penalty and lost throughput during this penalty (the association time). TAR will then only force a handoff when the throughput expected from moving to another heavily-loaded AP would be greater than staying at the current AP as it goes out of range, hence increasing the average association time. For interactive clients, the association time remains unchanged because TAR is already optimizing for handoff reduction, regardless of load conditions.

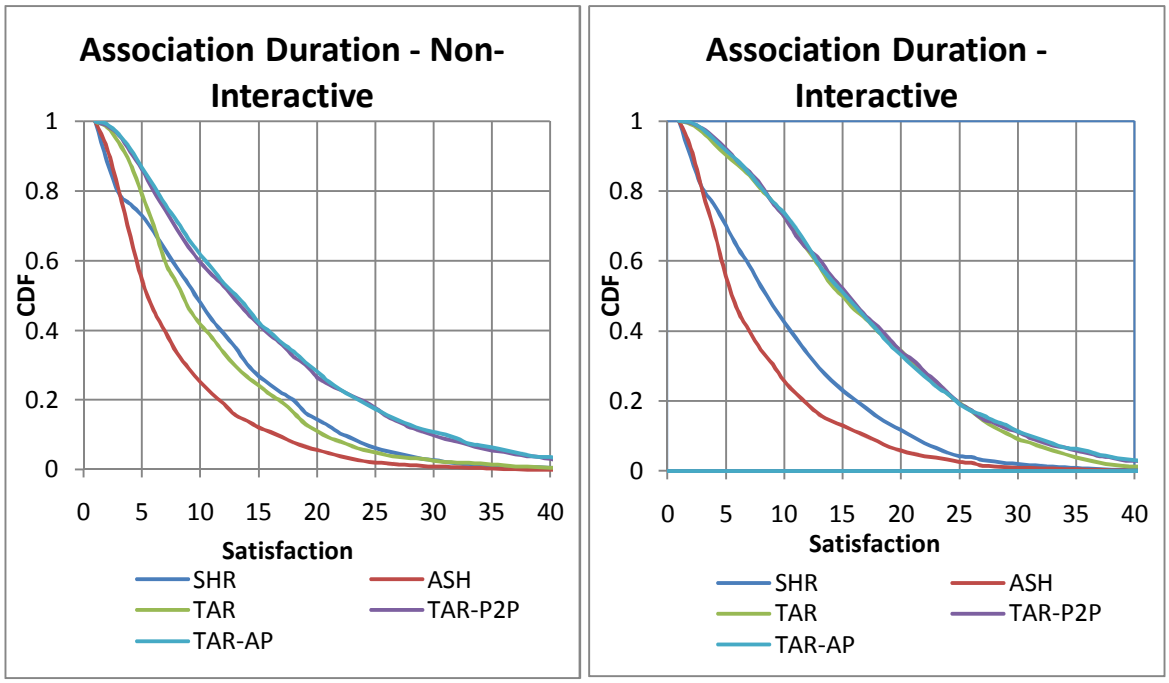

Figure 13: Association duration for association control algorithms in a heavily congested area.

Chapter 4:
CONCLUSIONS

In this paper, we have introduced TAR, a novel association control algorithm which uses trip prediction alongside previous knowledge of the wireless network topology in order to select the APs which the client will associate with. With cross-layer communication, the application can notify TAR about its bandwidth requirements and whether it wishes to perform handoff reduction or maximum throughput. Furthermore, we also proposed a distributed fairness mechanism around a weighted max-min scheme to allow interactive users (e.g. VoIP or streaming video) to gain priority to bandwidth during the association cycle and to do a form of load balancing. This can either be done with assistance from the AP or through peer-to-peer coordination with a mobile database of client-AP associations. We have then shown that these algorithms perform better than the commonly known and deployed select-high-and-retain and actively-select-high association control algorithms, which tradeoff simplicity for poor performance in mobile scenarios.

References

1. T. Green; *New Orleans' Wi-Fi network now a lifeline*. Computerworld. March 17, 2006.

2. J. Ott, D. Kutscher. *Drive-thru Internet: IEEE 802.11b for Automobile Users*. IEEE Infocom 2004 Conference. 2004.

3. J. Zhao, T. Arnold, Y. Zhang, and G. Cao. *Extending Drive-Thru Data Access by Vehicle-to-Vehicle Relay*. Proc. of ACM VANET 08', in conjunction with Mobicom 08', San Francisco. September 2008.

4. *Aggressive Load-Balancing on Wireless LAN Controllers (WLCs)*. Cisco Systems, Inc. July 8, 2008.

5. V. Navda, A. P. Subramanian, K. Dhanasekaran, A. Timm-Giel, and S. R. Das. *MobiSteer: Using Steerable Beam Directional Antenna for Vehicular Network Access*. Proc. ACM MobiSys 2007. June 2007.

6. M. Kim, Z. Liu, S. Parthasarathy, D. Pendarakis, and H. Yang. *Association Control in Mobile Wireless Networks*. IEEE Infocom. 2008.

7. N. Liu, M. Liu, J. Cao, G. Chen, and W. Liu. *When transportation Meets Communication: V2P over VANETs*. 30th International Conference on Distributed Computing Systems (ICDCS 2010). June 21-25, 2010.

8. J. Eriksson, H. Balakrishnan, and S. Madden. *Cabernet: Vehicular Content Delivery Using WiFi*. Proc. 14th ACM MOBICOM, San Francisco. September 2008.

9. S. Pack, Y. Choi. *Fast inter-AP handoff using predictive-authentication scheme in a public wireless LAN*. Proc. IEEE Networks 2002, Atlanta. August 2002. pp. 15-26.

10. D. Lee, G. Chandrasekaran, M. Sridharan, and P. Sinha. *Association Management for Data Dissemination over Wireless Mesh Networks*. Elsevier Computer Networks. 2007.

11. Y. Bejerano, S. Han, and L. Li. *Fairness and load balancing in wireless LANs using association control*. IEEE/ACM Transactions on Networking (TON), v.15 n.3, p.560-573. June 2007.

12. A. Balachandran, P. Bahl, and G. M. Voelker. *Hot-Spot Congestion Relief in Public-Area Wireless Networks*. Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications, p.70. June 20-21, 2002.

13. A.J. Nicholson, B.D. Noble. *Breadcrumbs: Forecasting Mobile Connectivity*. Proc. ACM MobiCom. Sept. 2008.

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

15. P. Deshpande, A. Kashyap, C. Sung, and S. Das. *Predictive methods for improved vehicular WiFi access*. ACM SIGMOBILE, pages 263—276. 2009.

16. M. Gross. *Power Management Quality of Service (PM_QOS).* Intel Corporation. 2008.

17. S. Schönfelder, U. Samaga. *Where do you want to go today? - More observations on daily mobility*. 3rd Swiss Transport Research Conference, Ascona. March 2003.

18. I. Marsic. *Computer Networks: Performance and Quality of Service*. Rutgers. 2010.

19. S. Keshav. *An Engineering Approach to Computer Networking*. Addison-Wesley, Reading, MA. 1997. pages 215-217.

20. R. Hassin, A. Levin. *A better-than-greedy approximation algorithm for the minimumset cover problem*. SIAM J. Comput., 35(1): 189–200. 2005.

21. OpenStreetMap. http://www.openstreetmap.org.

22. A. König. *Graphic description of travel behavior using the multiweek Mobidrive travel diary*. Arbeitsberichte Verkehrs- und Raumplanung, 50, Institut für Verkehrsplanung, Transporttechnik, Stassen- und Eisenbahnbau (IVT), ETH, Zürich. Oct. 2000.

23. *Primer on WiFi Range*. Rubicon Communications. 2004.

Appendix:
Max-Min vs. Random Allocation Proof

We shall prove that a contention-free max-min bandwidth allocation is better than randomly allocating bandwidth to users at every contention window, a scenario that would occur if clients are competing for AP bandwidth without any quality-of-service provisions. Some notes:

1. Multiple access point problem is equivalent to single access point problem. So in this proof we assume there is one access point with a capacity X bandwidth.

2. This proof can be extended to the case with 2+ vehicles.

Theorem: The performance on satisfaction degree of MAX-MIN bandwidth allocation algorithm will not worse than that of the random-selection algorithm. Satisfaction is defined as the ratio of bandwidth received versus bandwidth requested from the AP.

Proof

We assume that the bandwidth can be provided is X. There are two vehicles competing for the bandwidth, and their requirements are x1 and x2, respectively. Without loss of generality, we assume x1>x2.

Case 1: X>x1+x2

In this case, obviously both vehicles can be satisfied. Therefore, the satisfaction degrees in both algorithms are 100%. The performance of MAX-MIN is not worse than the random-selection algorithm.

Case 2.1: X<x1+x2, and X>x1>x2, and x1>X/2>x2

We use S to denote the satisfaction degree. Then, in this case, the expected satisfaction degree of random-selection algorithm can be calculated as,

$$E(S_{rand}) = \left[\frac{\frac{1}{2}x1 + \frac{1}{2}(X - x2)}{x1} + \frac{\frac{1}{2}x2 + \frac{1}{2}(X - x1)}{x2}\right]/2$$

And similarly, the expected satisfaction degree of MAX_MIN can be calculated as,

$$E(S_{MM}) = \left[\frac{X - x2}{x1} + \frac{x2}{x2}\right]/2$$

Then,

$$\begin{aligned} E(S_{MM}) - E(S_{rand}) &= \left[\frac{X - x2}{x1} + \frac{x2}{x2} - \frac{\frac{1}{2}x1 + \frac{1}{2}(X - x2)}{x1} - \frac{\frac{1}{2}x2 + \frac{1}{2}(X - x1)}{x2}\right]/2 \\ &= \left[\frac{X - x2}{2x1} - \frac{X - x1}{2x2}\right]/2 \\ &= \left[\frac{Xx2 - x2x2 - Xx1 + x1x1}{x1x2}\right]/4 \\ &= \left[\frac{(x1 + x2 - X)(x1 - x2)}{x1x2}\right]/4 \end{aligned}$$

Since x1+x2>X and x1>x2,

$$\begin{aligned} &E(S_{MM}) - E(S_{rand}) > 0 \\ &\Rightarrow E(S_{MM}) > E(S_{rand}) \end{aligned}$$

Case 2.2: X<x1+x2, and X>x1>x2, and x1>x2>X/2

Similar to case 2.1,

$$E(S_{rand}) = \left[\frac{\frac{1}{2}x1 + \frac{1}{2}(X - x2)}{x1} + \frac{\frac{1}{2}x2 + \frac{1}{2}(X - x1)}{x2}\right]/2$$

But,

$$E(S_{MM}) = \left[\frac{X}{2x1} + \frac{X}{2x2}\right]/2$$

Then,

$$E(S_{MM}) - E(S_{rand}) = \left[\frac{X}{2x1} + \frac{X}{2x2} - \frac{\frac{1}{2}x1 + \frac{1}{2}(X - x2)}{x1} - \frac{\frac{1}{2}x2 + \frac{1}{2}(X - x1)}{x2}\right]/2$$

$$= \left[\frac{x2 - x1}{x1} - \frac{x1 - x2}{x2}\right]/2$$

$$= \left[(x1 - x2)(\frac{1}{x2} - \frac{1}{x1})\right]/2$$

Since x1>x2, $(x1 - x2) > 0$ and $\left(\frac{1}{x2} - \frac{1}{x1}\right) > 0$. Therefore,

$$E(S_{MM}) - E(S_{rand}) > 0$$

$$\Rightarrow E(S_{MM}) > E(S_{rand})$$

Case 3.1: X<x1+x2, x1>X and X/2>x2

Similarly, in this case, the satisfaction degree of random-selection algorithm can be calculated as,

$$E(S_{rand}) = \left[\frac{\frac{1}{2}X + \frac{1}{2}(X - x2)}{x1} + \frac{\frac{1}{2}x2}{x2}\right]/2$$

And the satisfaction degree of MAX-MIN is still the same as that in case 2.1. Therefore,

$$E(S_{MM}) - E(S_{rand}) = \left[\frac{X - x2}{x1} + \frac{x2}{x2} - \frac{\frac{1}{2}X + \frac{1}{2}(X - x2)}{x1} - \frac{\frac{1}{2}x2}{x2}\right]/2$$

$$= \left[\frac{1}{2} - \frac{x2}{2x1}\right]/2$$

> 0 because x2<x1

$\Rightarrow E(S_{MM}) > E(S_{rand})$

Case 3.2: X<x1+x2, x1>X and x2>X/2

$$E(S_{rand}) = \left[\frac{\frac{1}{2}X + \frac{1}{2}(X - x2)}{x1} + \frac{\frac{1}{2}x2}{x2}\right]/2$$

And the satisfaction degree of MAX-MIN is still the same as that in case 2.2. Therefore,

$$\begin{aligned} E(S_{MM}) - E(S_{rand}) &= \frac{X}{2x1} + \frac{X}{2x2} - \frac{\frac{1}{2}X + \frac{1}{2}(X - x2)}{x1} - \frac{\frac{1}{2}x2}{x2} \\ &= \frac{x2 - X}{2x1} + \frac{X - x2}{2x2} \\ &= \frac{(X - x2)}{2}\left(\frac{1}{x2} - \frac{1}{x1}\right) > 0 \\ \Rightarrow E(S_{MM}) &> E(S_{rand}) \end{aligned}$$

Case 4: X<x1+x2, and x1>x2>X

$$\begin{aligned} E(S_{MM}) - E(S_{rand}) &= \frac{X}{2x1} + \frac{X}{2x2} - \frac{\frac{1}{2}X}{x1} - \frac{\frac{1}{2}X}{x2} \\ &= 0 \end{aligned}$$

Sum up case 1-4, we can conclude that$E(S_{MM}) \geq E(S_{rand})$. Therefore, the performance of MAX-MIN will not be worse than the random-selection algorithm. ▪