6.267 Design Project 1

Colleen Josephson
cjoseph@mit.edu

2Lt Dave Adams, USAF
dcadams@mit.edu

Nov. 8, 2013

1 Introduction

Wireless networks introduce many difficult challenges, and the increasing mobility of hosts only exacerbates these issues. Users want to be able to walk from place to place with continuous and automatic network connectivity. Most modern wireless networks have quite poor support for moving hosts, however. The connection must usually be re-established when the host finally becomes stationary. The problem is that by the time a moving host establishes a connection with an access point, they are moving out of range.

Wireless access points have a relatively stable topology in most buildings. This information can be used to figure out where a host is likely to go next so the access point can be ready to interact with the host seamlessly. Most currently deployed schemes do not take advantage of these properties. With this protocol we aim to change that, and create a system that can predict where a host will be so that continuous communication is possible as a host moves across the network.

Next we discuss the assumptions our protocol was designed with. In section two we discuss the goals of our protocol and outline the protocol design. In section three we analyze performance of both average operating parameters and edge cases, particularly focusing on the overhead necessary to maintain continuous communication. In section four we conclude our work and discuss possible extensions.

1.1 Assumptions

Throughout the design process, we worked with few important assumptions, which we believe are reasonable. These are outlined below.

- The wired network is much faster than the wireless network, and the links are bi-directional
- Movement of the devices will typically be no faster than 5mph (a slow jog)

- Computation is cheap at the gateway, slightly less cheap at access points
- All access points are one hop away from the gateway
- The MAC protocol allows for several access points to all simultaneously communicate with multiple devices without collision or interference issues (really good MIMO)
- There will be a pre-computed table of the nearest neighbors to any access point, developed from floor plans and kept updated by the network manager

2 Design

In this section we discuss our design goals, and outline how our system meets these goals. We discuss the details of the routing protocol and provide examples of typical operating scenarios.

2.1 Goals

The primary goals of this system is to minimize the amount of host intervention that is necessary, and to maximize the connectivity. We aim to never have the host unable to send or receive packets while transitioning from access point to access point, unless the user enters an area that has no coverage, such as a tunnel or elevator. If a host does lose coverage, we aim to have the host rapidly reconnected as soon as they are in range again. We do not to waste too much capacity ensuring these smooth coverage area transitions–our goals above could be met by simply broadcasting the packets from every access point, but this would lead to terrible throughput.

2.2 Protocol Outline

Floor plans are used to create a list of nearest neighbors to any access point, and this nearest neighbor list is a basis for a Markov chain that outlines how likely transitions from one access point to another are. The initial Markov chain is further developed by the gateway observing patterns of where hosts tend to move to from a given state. This is particularly helpful for cases where nearest neighbors are not necessarily where a host will transmit next, as in the case of elevators or tunnels.

Brief protocol outline:

1. When joining the network, each host will be assigned a unique number with which to announce themselves every τ seconds when not already transmitting. The access point that the host joins on becomes the **principal** node, i.e. the node that the host has the strongest connection to (by SNR, etc). In a Markov sense, the principal node is the state a host is in.
2. The access point includes this new host in the next routing update it sends to the gateway.

3. If the host does not already have a routing entry, the gateway creates a new one, assigning the host a **neighborhood** of nearest access points, centered around the principal. This neighborhood will broadcast packets addressed to the host.

4. The access points will periodically send routing updates that include information about all of the hosts it can hear, and how well it can hear them.

5. If the signal strength becomes stronger at some node other than the principal, the principal node changes for that host. This corresponds to a state transition in the Markov chain. The gateway will then send out updates, adding new access points and removing access points to the neighborhood. The gateway will also adjust transition probabilities in the Markov graph based on the state transition it just observed.

Steps 4 and 5 repeat infinitely until a host disconnects from the network or the prediction model makes an error and a host is left outside of the range of access points broadcasting its packets. In those cases, the host can simply begin again at step 1. We attempt to minimize prediction errors, but no probabilistic scheme can be 100% accurate. The re-association process will be about as inconvenient as it is in a typical network today.

In the remainder of this section we will extrapolate on specific aspects of the protocol.

2.3 Principal Selection

Here we formally define the **principal** node of a host to be the access point that corresponds to which state a host is at in the Markov chain.

Let $\mathcal{N} \triangleq \{1, \ldots, N\}$ be the set of all N access points. For a given host, let $f(i,t)$ be the estimated SNR of the host's transmission at access point i at time t, $i \in \mathcal{N}$. Then the principle of the host at time t is given by

$$x(t) \triangleq \arg\max_{i \in \mathcal{N}} \{f(i,t)\}$$

2.4 Markov Model

Assume that the motion of each host is a wide-sense stationary random process, and that noise is also wide-sense stationary. Then if we pick a sample period τ, we can define a Markov model based on the transition probabilities within period τ:

$$P_{ij}^{\tau} = P[x(t+\tau) = j | x(t) = i]$$

2.5 Neighborhood Selection

The **neighborhood** is a subset of nodes that a host can transition to from its current state. The neighborhood of a principal i is the smallest subset of $\mathcal{N}$ such that the summed transition probabilities from i is least $1 - \epsilon$.

$$U_i^{(\epsilon,\tau)} \triangleq \min_{\mathcal{S}}\{\mathcal{S} : \sum_{j\in\mathcal{S}} P_{ij}^{\tau} \geq 1-\epsilon\,, \mathcal{S} \subseteq \mathcal{N}\}$$

Note that $U_i^{(\epsilon,\tau)}$ will always contain the principals most likely to follow i, as any subset of $\mathcal{N}$ containing less probable states would require a larger number of states to reach the $1-\epsilon$ probability-mass limit.For each host whose principal is i, we multicast information to $U_i^{(\epsilon,\tau)}$. Then $m = \max_i |U_i^{(\epsilon,\tau)}|$.

From the definitions above, we see that there is probability at most ϵ that a host will go to a state that is not in the neighborhood. Intuitively, we want to minimize ϵ so that the network will be prepared for the host's transition as often as possible. There exists a key tradeoff between τ and ϵ and m. For $\tau' > \tau$, that $U_i^{(\epsilon,\tau')} \supseteq U_i^{(\epsilon,\tau)}$, and thus $m' \geq m$. Similarly, $\epsilon' > \epsilon \implies U_i^{(\epsilon',\tau)} \supseteq U_i^{(\epsilon,\tau)}$, so again, $m' \geq m$. In other words, the smaller ϵ is or the larger τ is, the larger the average neighborhood will have to be.

It would not be prudent for $\epsilon = 0$, because this would require the inclusion of all edges leaving a state. This is potentially a very large number of edges, especially for a gateway that has been alive for long periods of time. Networks are unpredictable, and very strange transitions can and will happen. For example, a host turn off its RF switch, and then turn it on again somewhere down the hallway, causing a discontinuous transition, geography wise. This specific case is very unlikely to happen often, so having the node down the hall broadcast packets would be wasteful. Thus, very low probability transitions should be excluded from the neighborhood via the ϵ technique.

2.6 Chirps

Hosts send out chirps at least once every τ seconds so the access points can maintain accurate liveness and SNR data for hosts who are not transmitting continuously. The chirps can even be piggybacked onto other necessary wireless maintenance packets, such as wireless rate selection probe packets, to reduce overhead.

If an access point does not hear a chirp for the entire period between routing updates, then the access point indicates that the SNR was zero. The access point continues to broadcast packets until explicitly told not to by the gateway, however.

To ensure that the network is not sending packets to hosts who are no longer present, we define a timeout of β seconds. If no access points hear a host chirp for β seconds, the host is considered to be off the network. The gateway informs any access points that may still be broadcasting to the host to stop, and removes the host from the routing table. At this point, the host will have to re-associate. This grace period should be long enough to accommodate reasonable outages from obstacles like elevators or tunnels.

3 Analysis

In this section we analyze how well our design performs. We also discuss engineering tradeoffs, and possible solutions to new problems introduced by our approach.

3.1 Performance

The first thing to note about this system is the likelihood of a host being dropped due to improper routing. Assuming that **a)** the MAC is perfect (i.e., no collisions or congestion) and **b)** the predictive model being implemented is accurate, then the probability of any host getting "lost" by the gateway is, by construction, $\leq \epsilon$.

The price for this reliability, however, is paid in redundancy and overhead. Since each host is receiving from up to m towers, the amount of traffic can be increased by as much as a factor of m. A performance hit is unavoidable if each host is suddenly causing more packets to be broadcast across the system. If an average host has 3 access points broadcasting to it simultaneously, then the system can support only a third of the hosts it did before. A performance cut of small constant factor is quite reasonable, however. It can be somewhat ameliorated by simply adding more access points. We say somewhat because suddenly increasing density of the network could have unexpected consequences, such as destructive interference if the frequency space is not carefully arranged. In most systems, additional access points will probably only need to be added in areas that tend to get very crowded, such as a lecture hall.

Additionally, hosts are announcing themselves every τ seconds, and although these announcements are brief, they can add up if the number of hosts in the system is large. There is some flexibility here, though. Adopting a second-order Markov model would decrease m for a given τ and ϵ at the cost of higher computational loads at the gateway, essentially trading off throughput for computation. This is discussed further in the **Extensions and Conclusion** section.

Another primary concern is the amount of routing traffic necessary to track hosts satisfactorily. We have designed the system to perform reliably up to movements of 5 mph, which is a slow jog. 5 mph translates to approximately 2.24 meters per second. If we assume that access points are typically 10 or more meters apart, then a host will be reasonably tracked if the access points send one update packet per second. If moving continuously down a well-covered hallway, a host moving less than 5mph will take 4 or more seconds to pass through an access point's coverage area.

The overhead incurred by sending one routing update every 2 seconds is small. If the gateway responds to each update period before the next period starts, this means an average of one packet of routing traffic per access per second. This is not much overhead. Assuming that the routing information can fit in a packet of 1.5MB, if the wired links are 10Gbit then routing packets take up 1.2ms. Thus, on the wired links, routing information takes up at only 0.12% of the transmission time, if we assume that the wired links deliver the routing

information without errors. This should not be hard, assuming the quality of the links are good, and the distances are small enough for attenuation to be small. Additionally, there will never be channel contention because each access point has direct bi-directional links to the gateway.

Finally, the fact that there are multiple access points broadcasting to the same host leads to interference concerns. One possible way to reduce interference concerns is to use a network coding scheme so that the access points are not sending identical packets. See the **Extensions and Conclusion** section for further discussion.

3.2 Edge Cases

In general, failure of the network to accurately track a host means that packets are either sent to a sub-optimal neighborhood of access points or, worse, that no packets intended for the host actually reach it. In the latter case, a host must re-initiate a connection with the gateway, and we assume that this process is costly in terms of time, power, etc. The following list includes some circumstances when the tracking function may break down.

- *Fast Hosts:* If a host is traveling fast enough that it can move through several access points in the span of τ seconds, then the predictive element of this scheme is less effective. However, our procedure still provides for more coverage in this case than routing to one access point: the host is still likely to move through access points in the neighborhood of its last calculated principal. In extreme cases, however, the host moves out of the neighborhood before its next principal is calculated, and re-initiation breaks down because the host will not stay near an access point long enough for the process to complete.

- *Gateway Failure:* Prolonged gateway failure is catastrophic. There is no way around this fact when dealing with a highly centralized system such as this one. However, if the outage is brief (less than a few τ), then the gateway can attempt to use its last routing table, and most hosts will still be within range of the neighborhood of access points last associated with them. If hosts left the network failure, then the timeout period will eventually remove those entries from the routing table so that the network is not broadcasting to ghost users.

- *Transient Conditions:* One admitted weakness of this strategy is its dependence on the accuracy of the Markov model to determine neighborhoods. In an environment like MIT, traffic is heavily time-dependent. This does not necessarily mean that the Markov model is a poor way of determining neighborhoods, though. For access points with few neighbors, like an access point in the middle of a long hallway, the time of day has little impact on which region the host is going to enter next. By contrast, an access point covering the lobby of Building 36 would have a much different distribution governing transition probability in the morning than it would in the evening.

- *Poor Coverage:* A network is only as good as its infrastructure. Unfortunately, if an area does not have coverage, this scheme cannot remedy that.
- *Access Point Failure:* A benefit of multicasting to a host from multiple access points is robustness. In the event of an access point going offline, if its neighbors are able to communicate to the host, then the information routed through the neighbors will still make it through. Upon the next calculation, since no SNR will be received from the incapacitated access point, one of its neighbors will be deemed the new principal of that host. This may cause packets to be routed to a sub-optimal neighborhood, but communication will not be severed. If an access point goes offline and none of its neighbors can reach the host, then the previous bullet point applies.

4 Extensions

The following variations on the design proposed above also appear as promising solutions to the problem of tracking hosts throughout a physical space.

4.1 Alternative Predictive Models

As mentioned above, the motion of hosts throughout the network, while random, is hardly stationary. A variant of this scheme that would provide for greater degree of accuracy when predicting motion would be to create several Markov models corresponding to different times of day with distinct traffic patters, or a time-dependent Markov model. Another variant might be a second- or third-order Markov model, which might be far more accurate than the original design in that such a model would have a sense of the trajectory and inertia of a host. The benefit of a more accurate model would be to reduce the entropy of each transition, shrinking the size of each $U_i^{(\epsilon,\tau)}$ and thus decreasing the amount of hosts connected to each access point and the overall amount of traffic. The cost of these modifications would be to increase the complexity of the routing calculations, placing additional computational loads on the gateway.

4.2 Network Coding

One interesting idea to boost the throughput of this design would be to employ network coding with multiplexing of coded packets between the nodes of a neighborhood. Using network coding, we no longer have multiple access points transmitting identical information to one host. Instead, the access points are transmitting complementary degrees of freedom, so that any combination of correctly received transmissions can be used to decode. For example, if the rate of each wireless link is R, the length of a data block is U, and the host receives from m access points, then with no network coding, the amount of data transmitted is mU and the delay (assuming perfect transmission) is U/R. With network coding and multiplexing, however, we can lower the amount of transmitted data to $U + \Delta$, where Δ is the coding overhead. Furthermore, the total rate we have to work with is mR, so our delay is reduced to $\frac{U+\Delta}{mR}$. With imperfect links, the benefits of network coding become even more pronounced, but such analysis is outside the scope of this write-up.

5 Conclusion

In this paper we have proposed a centralized system for automatically updating and maintaining the routing information of a specific wireless network topology. We employ statistical models to route packets not only to where a host was last detected, but also to where it is likely to go. In this way, we exchange excess bandwidth (for a lightly loaded system) for a certain level of confidence that a host will maintain its connection for the duration of its session, regardless of its path through the local area. We have provided a mathematical framework to model this system's performance and gone through some basic analysis, identifying a few of the core strengths and weaknesses. Finally, some adaptations have been listed which could seriously enhance the performance of the design with respect to various criteria.

On the first pass we are encouraged that, under our operating assumptions, this design not only seems to perform well with respect to tracking the hosts, but is also general enough to be tweaked and optimized for specific circumstances. Perhaps most satisfying is that the cost of the tracking ability, in terms of quality of service, appears to be relatively cheap.