Using Smart Triggers for Improved User Performance in 802.11 Wireless Networks

Vivek Mhatre
Intel Research, Cambridge, UK
vivek.p.mhatre@intel.com

Konstantina Papagiannaki
Intel Research, Cambridge, UK
dina.papagiannaki@intel.com

ABSTRACT

The handoff algorithms in the current generation of 802.11 networks are primarily reactive in nature, because they wait until the link quality degrades substantially to trigger a handoff. They further rely on instantaneous signal strength measurements when choosing the best AP. This approach leads to handoff delays on the order of 1-2 seconds that are unacceptable for delay sensitive applications such as VoIP. We propose a fundamentally new approach to handoffs that is based on *continuous monitoring* of wireless links. In our approach, a client measures the beacon strengths of all the APs operating on the current, and the overlapping channels, and makes its handoff decisions based on the long-term, and short-term trends in these signals. We show through experiments in a campus wireless network that our proposed algorithms result in more than 50% reduction in average handoff delays, while having the potential to improve overall user performance. Our algorithms have been implemented in today's hardware, and unlike other proposed roaming algorithms in the literature, need no infrastructure support.

Categories and Subject Descriptors: C.2.1 [Computer-Communication Networks]: Wireless Communication

General Terms: Performance, Measurement, Experimentation.

Keywords: 802.11, roaming, handoff, triggering, continuous monitoring.

1. INTRODUCTION

The ease of deployment and low cost of wireless infrastructure networks has made them one of the most popular access mechanisms to the Internet. Such popular demand has led to dense urban and corporate wireless networks, wherein a wireless client may have multiple choices in terms of the Access Point to which it can affiliate [1]. High density not only ensures that the network can meet the overall user demand, but also guarantees that the users can be always connected even if they are highly mobile.

Support for mobile users within an 802.11-based network infrastructure necessitates efficient techniques for seamless handoff between APs, as the user moves from one area to the next. Such a process can be broken down into four stages, i) triggering, ii) discovery, iii) AP selection, and iv) commitment. The "triggering" stage corresponds to the points in time when a wireless client identifies the need to look for other AP affiliation options. Any triggers generated from this stage lead to the "discovery" stage where the client collects information about the nearby APs and their associated performance metrics. The "AP selection" stage identifies the one AP among the list of available APs that meets the client's performance requirements. During the "commitment" stage, the client performs disassociation with its current AP, and re-association with the newly identified AP.

The fundamental problem behind today's handover mechanisms can be attributed to the fact that wireless clients trigger a handover event upon loss of connectivity or poor and unsustainable performance. Thus, when the client is about to handoff, it has already been experiencing poor performance for some time. Furthermore, today's handoff mechanisms involve a lengthy measurement stage. During this stage, the client scans all channels of 802.11b/g (and 802.11a, if it is supported by the card) to collect information about all the APs operating in its neighborhood. Previous research has shown that the entire handoff takes up to 2 seconds, and the scanning stage amounts to about 80% of the entire handoff duration [2, 3].

In this work, we advocate that clients *should not wait until they lose connectivity or experience poor performance* to seek alternative APs. In other words, the clients should be proactive, and not reactive to poor performance. Secondly, we propose a mechanism to reduce average handoff delays. For this, we note that in typical urban and enterprise environments, the AP density is fairly high, and this presents the client with many opportunities to handover *within its current channel.* We exploit this observation in designing our handoff algorithms.

In our proposed algorithms, the client driver *continuously monitors* the performance of all the APs operating on the client's current channel, and also all the APs operating on its overlapping channels, by measuring their beacon signal strengths. Such an operation incurs minimal computational overhead, and can be trivially supported by today's hardware (at least the Intel 2915ABG card used in our experiments). Furthermore, since the client firmware attempts to decode all the received frames by default, exposing all the received beacons to the driver incurs no additional energy overhead. This way, the stage of discovering APs (at least the ones operating on the current channel) has zero delay overhead, while providing the card with a continuous stream of measurement that the client can use to trend AP performance. In typical 802.11 b/g deployments (as well as in high density deployments of 802.11a), the limited number of available channels (3 for 802.11b/g and 11 for 802.11a) ensures substantial overlap in the coverage of co-channel APs, allowing the client visibility into multiple co-channel APs as it roams. When no better APs can be found on the current or the overlapping channels, the client falls back to scanning. If the client supports power save mode, there will be periods when continuous monitoring of beacons is not feasible. Further work is required to assess the impact of power saving on the performance of our algorithms.

There are several choices in the design of triggering algorithms. We identify four such choices and present a taxonomy of triggering algorithms according to their:

- Use of information about other APs operating in the neighborhood,
- Criterion that ultimately leads to a trigger for an AP transition (hysteresis-based, threshold-based, prediction-based, etc.)
- Use of instantaneous vs historical information about the performance of the APs, and
- Operating region, i.e., whether triggers are generated even when the client operates at the highest possible quality.

We then propose three new algorithms that attempt to cover the design space which has not been previously explored. The first algorithm is similar in spirit to what is typically employed in cellular networks. It triggers a transition as soon as the signal of another AP (on the same or overlapping channel) is better than the signal of the current AP plus some hysteresis factor. Our second algorithm extracts trends from the time series of RSSI (Received Signal Strength Indicator) of each AP (on the same or overlapping channel). It then issues a trigger when the current AP features a downwards trend of at least α, while another AP in the neighborhood exhibits a similar-sized upwards trend. Lastly, we propose a predictive scheme that issues a trigger when the predicted RSSI measurement of one AP (on the same or overlapping channel) exceeds the predicted RSSI measurement of the current AP. The algorithm fits a linear regression model in the RSSI measurements of each AP, and attempts to predict the future RSSI value.

Through experimental validation of our proposed algorithms, we show that it is beneficial for a triggering mechanism to:

- Operate on smoothed signals that show long term trends in signal strengths of APs. This can be achieved by filtering out rapid fluctuations in measured signals caused by shadow-fading
- Exploit knowledge of *all* the APs operating on the client's current and overlapping channels.

Our proposed algorithms have the following desirable features:

- Average handoff delays are reduced by more than 50% (140-250 ms as compared to 860 ms with today's mechanisms).
- Considerably reduced packet loss for VoIP traffic (0.41% over the uplink and 0.72% over the downlink, as compared to 2.54% over the uplink and 3.36% over the downlink in the state of the art).
- Lower packet delay variation during handoffs for VoIP traffic.
- Require a few minor modifications to the driver code of the client, but *unlike other proposed roaming algorithms in the literature, require no infrastructure support.*

The remainder of the paper is structured as follows. In Section 2 we present background information on the four stages in a 802.11 handoff process. In Section 3 we advocate the need for a new generation of triggering mechanisms. The proposed algorithms are described in Section 4. We implement the new algorithms, and test their performance using experiments in a real environment in Section 5. Limitations in the experimental environment are explored using simulations in Section 6. We conclude in Section 7.

2. BACKGROUND

In this section, we describe each of the four stages in 802.11 handoff in detail. We also briefly discuss the research proposals addressing performance issues specific to each stage. Notice that in what follows, we study the *Layer 2* handover process. For solutions dealing with *Layer 3* see [4] and references therein.

2.1 Stage I: Triggering

Implementation of appropriate triggering mechanisms is typically left to the wireless card manufacturer, and are therefore proprietary (usually implemented in the firmware). To the best of our knowledge, the state of the art triggering mechanisms include reaction to (i) a manufacturer specific number of unacknowledged frames [2], (ii) loss of a certain number consecutive Beacon frames [5, 6], and (iii) Beacon frame loss or quality degradation beyond specific threshold values [6]. In the last case, the driver starts collecting information on neighboring APs when the quality falls below what is referred to as the "cell search threshold". A transition to a new AP is made, however, only if the difference in quality exceeds a predefined threshold. The aforementioned algorithm may get overridden by the loss of four consecutive Beacon frames.

The use of Beacon frame loss attempts to capture disconnection due to the client being out of the AP's range, while quality thresholding attempts to capture unsustainable performance. Recent work has shown that today's triggering mechanisms lead to detection times of 900 to 1600 ms, between the time when channel quality degradation begins and the time when the appropriate trigger is generated [2]. A further suggestion was made for the use of three unacknowledged frames as triggers [2], leading to only 3ms of detection time.

2.2 Stage II: Discovery

This stage has received significant attention in the recent past as a reaction to experimental measurements revealing that 80% of the handoff duration is due to the way neighbor knowledge is acquired through scanning [3]. Discovery of 802.11 Access Points in the neighborhood of a wireless client can be performed in two different ways: (i) actively or (ii) passively.

In the active scanning case, the wireless client tunes itself to a specific channel, and solicits beacon messages using the "Probe Request" frame. The client then waits for responses on each channel. according to two timers: *MinChannelTime* (about 20ms), and *MaxChannelTime* (about 30 ms). *MinChannelTime* corresponds to the time a client should wait on an idle channel. If no response is received within *MinChannelTime*, and no traffic has been observed, then the channel is declared empty, and the client moves to the next channel. The second timer, *MaxChannelTime*, is the maximum amount of time a client should wait to collect all responses in a used channel. The second timer becomes effective only when traffic, or a response, has been obtained within *MinChannelTime* on that channel. The total latency of an active scanning process has been quantified in [2, 7], and has been shown to last between 50 and 360 milliseconds, depending on the number of active channels and incumbent APs. If the card supports 802.11a in addition to 802.11b/g, then this delay is even higher.

Passive discovery of a client's neighborhood requires a client to tune on to each individual channel, and stay on that channel for up to 100 ms, such that it receives all periodic beacon frames of all incumbent APs. The total latency of such a process sums up to 1.1 seconds for the 11 channels of the IEEE 802.11b/g band. Once again, if the card supports 802.11a in addition to 802.11b/g, then this delay is even higher.

Given the dramatic impact of scanning on the duration of the handoff process, researchers have come up with recommendations on ways in which scanning latency can be reduced. Within the IEEE 802.11k task force specification, APs will not only advertize their presence to potential wireless clients using the beacon frames, but they will also provide "neighbor reports" upon request. These reports will contain a list of all the APs operating in the neighborhood of a given AP, as well as their operating channels for expedited scanning [7]. Moreover, in certain cases the client is informed of the exact identities of APs in each channel, and can unicast Probe Request messages. Thus the client can collect information on all APs in that channel, and move to the next one without having to wait up to *MaxChannelTime*.

Another solution was recently proposed in [8]. *SyncScan* addresses the issue of high scanning times by "synchronizing" APs collocated in the same channel in their transmission of beacon frames. If the wireless client knows the time instant when beacon frames are scheduled in each channel, it can optimize its scanning procedure so as not to wait up to *MaxChannelTime*.

All the above solutions for improving *Stage II* require some kind of infrastructure support; whether it comes in the form of "Neighbor Reports", or synchronization among channel-collocated APs. To avoid possible adoption barriers that arise once one necessitates infrastructure support, our work targets solutions that can be implemented in today's networks, requiring *modifications at the client side alone*.

2.3 Stage III: AP selection

The problem of AP selection has also received a significant amount of interest in the research community. The objective being that of providing clients with the intelligence to identify better performing APs. Currently implemented AP selection mechanisms typically select APs according to the Received Signal Strength Indicator (RSSI) on the client, i.e. if a client has a variety of APs to select from, it will pick the one with the strongest received signal. The fundamental limitation with such a scheme is that it ignores the network load of the selected AP. Consequently, a wireless client may receive worse performance from a closeby, but overloaded AP, than from an AP that has lighter loaded, but is relatively farther away from the client. This is because the throughput that a client receives from an AP depends not just on the RSSI, but also on the network load (number of other clients, and their traffic) of the AP. Solutions proposed to address this limitation typically require changes in the infrastructure [9, 10], while others attempt to offer similar performance while obtaining measurements passively [11]. Further efforts are carried out within the IEEE and as collaborations between AP and wireless client manufacturers [12].

2.4 Stage IV: Commitment

During the commitment phase a wireless client commits its decision to affiliate with another AP in its neighborhood. Specific functions performed in this stage include disassociation from the current AP and association with the AP identified in *Stage III*. Stage IV in the handover process typically lasts a few milliseconds [2] if it does not involve sophisticated security mechanisms such as 802.1x, also confirmed in our measurements. Furthermore, within the IEEE 802.11r task force, procedures for fast and efficient handoff capable of supporting voice applications are standardized.

The focus of this work is on stages I and II of the handoff, i.e., triggering and discovery. Our approach to triggering departs from previous approaches, and is discussed in the next section.

3. SMART TRIGGERING

An important desirable feature of a good triggering scheme is that it should result in minimal client service disruption. We complement this notion with an additional objective; the triggering mechanism should also be capable of notifying the wireless client about the existence of other APs in its neighborhood that may offer better performance - the client ultimately decides whether or not to transition. Note that service disruption incorporates both the effects of inability to communicate due to poor quality of the current AP, as well as disruption due to the internal operations of the triggering mechanism itself. The latter can be attributed to scanning which may imply packet loss or high delay variability when traffic is buffered for the duration of the scanning cycle.

Consequently, any triggering mechanism needs to balance the tradeoff between its reactiveness to channel conditions and its overhead. Moreover, one should ensure that the proposed algorithm does not lead to situations where the client switches between APs for short periods of time or bounces between a small set of APs (e.g. the ping-pong effect). There are two main limitations of the current approach to triggering. Firstly, the wireless client typically waits until it observes substantial performance degradation before triggering the discovery phase. Secondly, only after the initiation of the discovery phase does the client obtain knowledge about alternative APs in its neighborhood. To address the above concerns, we advocate a fundamentally different approach that relies on the following four principles:

- We exploit the broadcast nature of the wireless medium to *continuously monitor* the RSSI of the beacons of all APs operating on client's current and overlapping channels. The latter is possible only in 802.11b and 802.11g physical layers when there are APs operating on overlapping frequencies.
- We continuously monitor the RSSI of all "visible" APs, extract the long term trends in the performance of these APs, and make handoff decisions based on these trends. This is unlike the current approaches which rely on instantaneous measurements to make handoff decisions.
- Global scanning is used as a last resort only when local knowledge cannot improve the client performance. Inherently, "in-band" handoffs (within the client's channel) are significantly faster than their "out-of-band" counterparts.
- The objective of the handoff algorithm is no longer to avoid disconnection or unsustainable performance, but to improve client performance in the presence of better performing APs.

The proposed strategy works particularly well in scenarios with high AP density as we demonstrate in our experimental results. Recent work has shown that high density is not an artifact of our environment, but can be observed at a larger scale [1].

3.1 Collecting performance indicators

In the previous section we mentioned that our goal is to augment current triggering mechanisms with algorithms that not only react to disconnection, but also guide the wireless card to make affiliations that may improve its performance.

The performance of the service between a client and an AP depends on the quality of the wireless link between the client and the AP, as well as the load of the AP. Until recently the only metric available to the client to make its affiliation decision was the Received Signal Strength Indicator (RSSI), or the Signal to Noise Ratio (SNR). These metrics are usually used to identify the "best" AP to affiliate with. Lately, within the IEEE 802.11k standard, APs also provide information about their load to potential clients. The algorithm proposed in this work can function on any of the above performance indicators, as long as they are continuously collected.

Measurements have shown that in equally loaded environments, the relationship between the client throughput and the SNR can be well approximated by a piecewise linear law [13]. RSSI is further defined as SNR plus the ambient noise power. In what follows, we use RSSI measurements simply because the noise measurements reported by our Intel 2915ABG card tend to be constant and equal to -90dB. All the arguments can be extended to SNR by simply subtracting -90dB from RSSI.

Fig. 1 shows the achievable UDP throughput as a function of RSSI for the Intel 2915ABG in a campus or enterprise environment. We can clearly identify three operating regions in terms of RSSI, 1) the bad region, below which no communication is possible, 2) the intermediate region, where the throughput is linearly related to the RSSI, and 3) the excellent region, where the client receives the maximum throughput rate regardless of the exact RSSI value. We denote the threshold for the bad region as Q_{min}, and the threshold that leads to the excellent region as Q_{max}. Fig. 1 allows us to link user throughput with RSSI. This is going to prove useful in Section 6, where we establish the scheme performance through event-driven simulations.

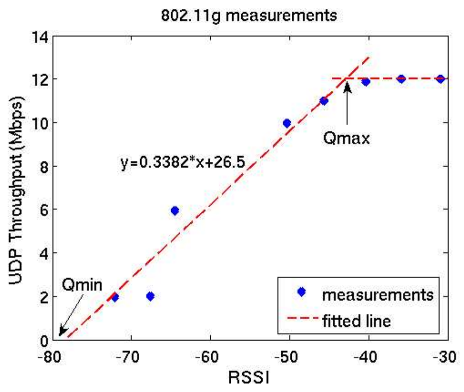


Figure 1: Relationship between throughput and RSSI for Intel 2915ABG card in 802.11g mode (CTS-to-self turned ON on the AP).

Note that the throughput achieved in the excellent region is far from the physical layer data rate of 54Mbps. This is because of the MAC layer overhead, as well as the fact that the APs used in this experiment use "CTS-to-self" to allow 802.11b devices to co-exist with 802.11g devices. The additional signaling required from the AP fo co-existence of 802.11b devices results in lower throughput [14].

3.2 Frequency of collected measurements

In our scheme, the client affiliated with an AP on a specific channel can sample performance measurements for the quality of the link between itself and any other AP in the same, and the overlapping channels when it receives the respective beacon frames. Beacon frames are sent out periodically (once every 100 ms) by APs to advertize their presence, thereby offering a stream of measurements that the client can use to trend the performance of different APs.

3.3 Collection mechanism

In the previous section we argued in favor of using information on all beacon frames exchanged in a client's channel as well as its overlapping channels. Such an operation is not supported by default in today's hardware. More specifically for the Intel 2915ABG card that we use for our experiments, beacon frames that do not belong to the current BSSID are filtered at the microcode level. However, allowing those beacons to pass into the driver space involves a minor modification to the driver source code (turning on the "management promiscuous" mode of the card by setting `accept_all_mgtm_frames = 1`). Assuming such a modification is feasible for the majority of wireless adaptors, the client not only gains visibility into the existence of other APs in its operating channel, but also those APs occupying overlapping channels.

While it is possible to decode the beacons from the overlapping channels, these beacons are attenuated by varying factors (depending on channel separation), since the receive filter of the client overlaps only partially with the adjacent channels. However, similar to the results presented in [15], we can determine the attenuation factors for each of the adjacent channels through measurements, and then correct the RSSI values of the beacons received on those channels.

Lastly, an entirely non-obtrusive collection process can be accomplished if nodes feature multiple radios, in which case one radio collects all the necessary information while the second radio acts. A solution assuming two radios has also been proposed in [16].

4. ALGORITHMS

The fundamental motivation behind our work stems from the fact that triggering performance is likely to be significantly improved if clients obtain visibility into the long-term trends of the performance of the APs within range before initiating a handoff. For instance, if the current AP is seen to exhibit a decreasing performance trend while another AP within range has been consistently improving its performance indicators, then the client may be better off handing off. Current triggering schemes wait for signs of disconnection, thus guaranteeing that the user will observe poor performance until the triggering mechanism takes action.

4.1 Input and Output

The input to our algorithms is the set of Beacon RSSI measurements from all APs in range. Given the state of the art, throughout this work we use RSSI but any metric concretely capturing performance can be used (one example is potential throughput based on beacon measurements as shown in [11]). The output is a set of APs that could offer better performance. A transition is initiated to the AP in the candidate set that features the best performance as per the criterion of the triggering algorithm. If no AP meets the triggering criterion, then the wireless card remains connected to the current AP until it goes out of range, and then falls back to the default scan-based handoff scheme.

4.2 Deriving appropriate filters

In Fig. 2, we show the RSSI measurements collected for each AP (y-axis) across time (x-axis) when we walked with the test laptop along a fixed path in our test environment. The laptop was set on a fixed channel, and the RSSI values of all the captured beacons were recorded. No traffic was sent either over the uplink or over the downlink. The experiment was carried out three times; once each for channels 1, 6 and 11. For the subsequent experiments, we follow the same path (further details in Section 5). Areas of lighter color in the plot correspond to higher RSSI values for the AP at that time instant. Consequently, a triggering scheme would lead to better client-perceived performance if the wireless card stayed in the "white" connectivity islands for as long as possible, while making the smallest number of transitions. One property of the RSSI measurements that may complicate such a task is that they are highly time varying.

The short-lived transients present in Fig. 2 are due to two reasons:

- APs may not always generate one beacon frame every 100 ms due to high load or queueing delays, since beacon frames may need to queue behind other packets and contend for access to the medium; consequently, RSSI time series are bound to have missing values,
- The radio environment is highly time varying due to shadowing and fading;

Indeed, in cellular networks, handover operations are by default based on smoothed signals [17]. Due to the two aforementioned effects, triggering mechanisms exploiting the time behavior in the signal measurements of different APs need to pre-process the time series for missing values, and apply some kind of smoothing filter that can remove the high frequency component in the signal. Due to their simplicity, we use the two following solutions:

- We replace the missing values in the time series by -80dB, a value that corresponds to no effective communication channel as per Fig. 1.
- We smooth the resulting time series using an exponential weighted moving average (EWMA) filter.

We use an EWMA filter since: i) it requires a small amount of historical information, limited to the previous value of the filter itself, ii) it can efficiently discount the contribution of older values in the running value (unlike a moving average filter). Based on the above two operations, APs that miss several beacons will feature smooth RSSI, but these RSSI values will gradually approach -80dB. We use this criterion for flushing out-of-range APs from the list of active APs.

An exponential weighted smoothing filter is expressed as:

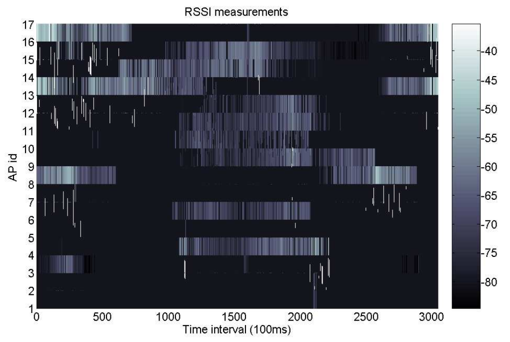
$$y_t = \alpha y_{t-1} + (1-\alpha)x_t, \tag{1}$$

where y_t is the value of the filter at time t, x_t is the measurement collected at time t, and α is the smoothing parameter controlling the impact that the current measurement has on the value of the filter; large values of α will lead to a smooth evolution of y_t, while small values of α lead to a highly responsive y_t that reacts to abrupt changes in the behavior of x_t.

In Fig. 3, we present the behavior of the RSSI time series for one of the APs in our traces. The original measurements appear to be very noisy featuring multiple spikes and drops. However, by using a weighted moving average filter with $\alpha = 0.9$, we can smooth the overall signal capturing the long term behavior of the AP across time. Also note that the missed beacon RSSI values are replaced with -80dB. We believe that triggering algorithms that attempt to capture long-term AP performance will need to operate on this kind of smoothed signal to avoid instability due to channel fluctuations.

4.3 State of the Art

The state of the art in the area of triggering can be summarized in two schemes. The first scheme uses the number of consecutively lost beacons (or unacknowledged MAC layer frames) to issue a trigger and is implemented in the Intel



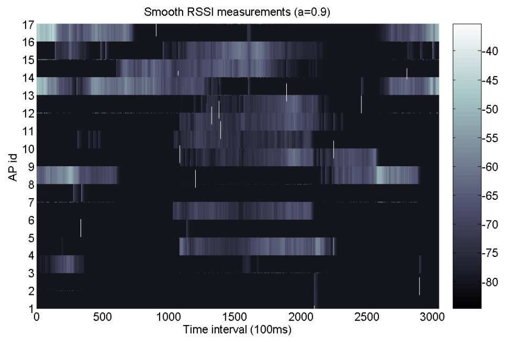


Figure 2: RSSI matrix (in dB) for all the APs (global view). Without and with smoothing.

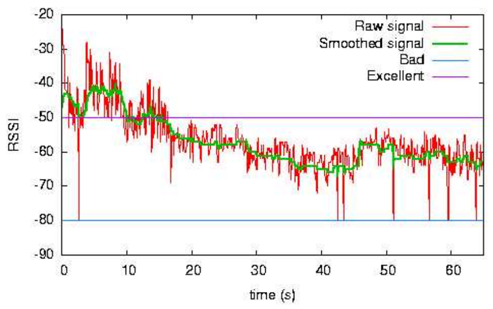


Figure 3: Time evolution of RSSI for AP id 14 on channel 11.

2915ABG cards (both Linux and Windows drivers), which we call the *Beacon* scheme. According to this scheme, a wireless card issues a trigger as a reaction to potential disconnection. It then scans all the frequencies to identify a new AP for affiliation. The second scheme reacts to degrading performance and issues a trigger when the signal quality falls below a specific threshold [6]. Due to the volatile nature of instantaneous RSSI measurements, it appears that manufacturers typically prefer to capture disconnection through the loss of frames rather than through poor quality signals. In fact, [6] also uses loss of consecutive Beacon frames to detect cases when the client has stepped out of the AP's range. We label the second scheme as *Threshold* scheme and explore its performance in response to the raw and the smooth signal strength measurements in Sections 5 and 6 respectively.

Nonetheless, when handover decisions can be made proactively and using knowledge on the client's neighborhood then the design space for triggering mechanisms is significantly enlarged. In what follows we present a taxonomy of triggering algorithms in this new framework.

4.4 A taxonomy of triggering algorithms

From previous discussion we have motivated the need for triggering mechanisms to not only react to disconnection and unacceptable performance, but also to improve the client's throughput. Continuous tracking of the signal strength received by a multitude of APs in the client's vicinity can provide the necessary input for such an objective to be met.

Scheme	Operating region	History	Criterion	Neigh-borhood
Beacon	N/A	Yes	Lost beacons	No
Threshold	No	Yes/No	RSSI value	No
Hysteresis	Yes/No	Yes/No	RSSI difference	Yes
Trend	Yes/No	Yes	RSSI trend	Yes
LSE	Yes/No	Yes	RSSI prediction	Yes

Table 1: Different triggering mechanisms and their features

Once such information is available to the wireless card, then there are several choices that can be made in the design of new triggering mechanisms:

1. whether they trigger transitions across the entire range of the analyzed performance metric or only when the performance falls below for instance excellent levels, e.g., Q_{max}. Transitions across APs that both live in the "excellent" region will have no impact on user throughput.

2. whether they incorporate history or not, e.g. if they use instantaneous values of the performance metric or smoothed measurements.

3. whether they use neighborhood knowledge or simply rely on current AP performance.

4. whether they use simple criteria, such as absolute RSSI values being less than a prescribed threshold value, or use trends in the overall signal received by an AP.

Table 1 lists the different dimensions in the design of triggering mechanisms. The default *Beacon* scheme uses as its criterion the number of consecutive lost Beacon frames, and thus incorporates historical information. It operates without knowledge on the performance of neighbors. Given that triggers are issued upon signs of disconnection, there is no option for this algorithm to trigger handover while the client is operating in the "excellent" region.

Similarly, the *Threshold* scheme uses no knowledge about the neighboring APs and makes decision using local information alone. Its criterion is *absolute RSSI value*. This algorithm typically triggers a transition when the client enters the "bad" operating region and therefore never issues a trigger when the client is already experiencing good performance. Notice that the *Threshold* scheme can be easily made neighborhood-aware, or can be made to operate on smoothed RSSI measurements. In addition, its threshold could be set to values above Q_{max}.

In what follows we complement the two currently implemented schemes with three new triggering algorithms. The flowchart of these algorithms can be seen in Fig. 4. Common functions among the three new algorithms are the use of smoothed measurements, and the condition that if the newly identified AP does not feature a quality index of at least Q_{min} then no transition is issued. In addition, if no AP meets the selection criterion of the algorithm in question, the algorithm selects the AP with the maximum quality signal, as long as it exceeds Q_{min}. In what follows we look at the selection criterion of each individual algorithm.

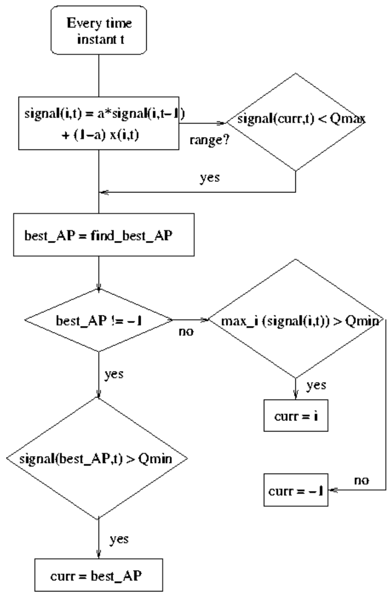


Figure 4: Overall flow chart for algorithms that incorporate neighbor knowledge and smooth measurements.

4.4.1 The Hysteresis(Δ) algorithm

In this algorithm, the client uses knowledge on all the APs operating on the current and overlapping channels, and issues triggers when the RSSI of an AP exceeds the RSSI of the current AP plus a hysteresis factor of Δ. This scheme is in line with handover schemes implemented in cellular networks [17]. The algorithm can operate on raw signal measurements as collected in our experiments, or on corrected and smoothed signal measurements. The *Hysteresis* algorithm can be allowed to issue triggers at any point in the RSSI range or only when current RSSI values fall below the "excellent" operating region. A flowchart of the algorithm operation is shown in Fig. 5(a).

4.4.2 The Trend(L,Δ) algorithm

The *Hysteresis* algorithm will lead to a transition between two APs, when the smooth (or not) signal of one AP *instantaneously* exceeds the smooth (or not) signal of another AP. Given the inherent volatility in the medium one way to reduce excessive transitions between APs is through the use of trending information that captures recent behavior, instead of instantaneous behavior. To demonstrate this effect we use an example from one of our traces shown in Fig. 6.

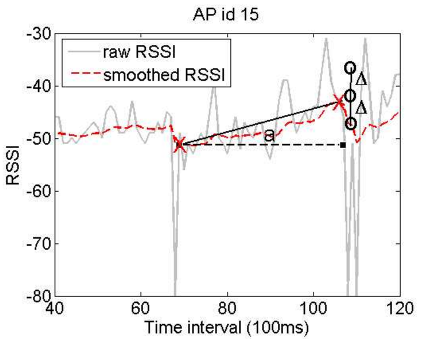


Figure 6: Time series of RSSI of AP 15: raw and smoothed signal. Demonstrating rate of change and LSE predictor.

Fig. 6 presents the RSSI recorded for one AP in our trace for time intervals 40 to 120. The RSSI measurements collected for this particular AP are highly time varying. The smoothing operation is capable of extracting the overall increasing trend in the AP's signal. In order to quantify the long-term trend in the signal we can use two entities; the rate of change of the signal, or the linear relationship between time and smoothed RSSI using linear regression. For both operations we need to define the amount of time over which trend is going to be defined (parameter L in our algorithm). Once L is defined, rate of change is defined as follows:

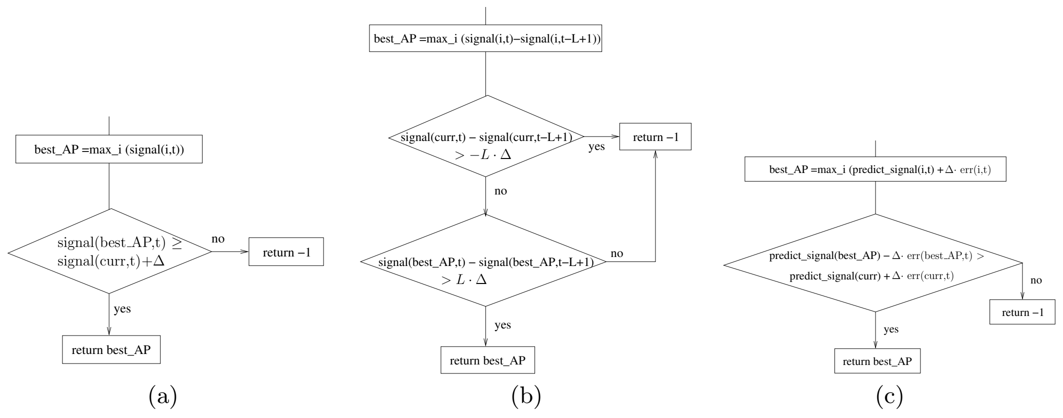
$$a = \frac{y_t - y_{t-L+1}}{L} \qquad (2)$$

APs with increasing trends will feature positive a values, as our example in Fig. 6. When a new AP appears in the client's neighborhood, the client can decide to transition only if the new AP features a positive rate of change, while the current AP features a negative rate of change. In order to further reduce transitions, the previous statement can be changed to "... the client can decide to transition only if the new AP features at least a positive rate of change of Δ, while the current AP features at least a negative change of Δ.". For simplicity we use the same threshold for both upwards and downwards change. A flowchart of the algorithm is presented in Fig. 5(b).

4.4.3 The LSE(L,Δ) algorithm

Quantification of the trend in the RSSI evolution using the rate of change metric is sensitive to the current RSSI measurement, as well as the interval L, since the rate of change is defined based on the values of the signal at time t and time $t - L + 1$. The impact of such a simplification is likely to be small if the algorithm operates on smoothed signal measurements. However, given the collected measurements one could also consider extracting the linear relationship that may exist in the collected measurements. The advantage of the latter approach stems from the fact that linear regression will depend on all L measurements in the interval.

The most commonly used method for linear regression is the least squares method. We show the line fitted through



best_AP =max_i (signal(i,t))
signal(best_AP,t) ≥ signal(curr,t)+Δ
no
return −1
yes
return best_AP
(a)
best_AP =max_i (signal(i,t)−signal(i,t−L+1))
signal(curr,t) − signal(curr,t−L+1) > −L · Δ
yes
return −1
no
signal(best_AP,t) − signal(best_AP,t−L+1) > L · Δ
no
yes
return best_AP
(b)
best_AP =max_i (predict_signal(i,t) + Δ· err(i,t)
predict_signal(best_AP) − Δ· err(best_AP,t) > predict_signal(curr) + Δ· err(curr,t)
no
return −1
yes
return best_AP
(c)

Figure 5: Flow chart for the criteria employed by the (a) Hysteresis, (b) Trend and (c) LSE algorithms.

the measurement for AP 15 in Fig. 6. Once a line has been fitted through the data one can use the least squares estimator (LSE) to predict the value of the signal in the next time interval $\hat{y}_t = \alpha * x_t + \beta$ and the associated error $\delta\hat{y}_t$. Consequently, at this point one potential triggering mechanism is to allow a transition to a new AP (z) only if the least squares estimator for the new AP minus its associated error still exceeds the least squares predictor of the current AP (y) plus the associated error, i.e.,

$$\hat{z}_t - \delta\hat{z}_t > \hat{y}_t + \delta\hat{y}_t \tag{3}$$

Essentially, we require that the lowest predicted value for the new AP is higher than the highest predicted value for the current AP. We call this scheme *LSE* (Least Square Estimator). Its criterion is based on RSSI predictions and can be applied on the "excellent" client operating regions or not. One could design such an algorithm to operate on the raw signal measurements, but given the volatility in the original measurements, predictions tend to be very poor with large errors. Consequently, we define *LSE* to operate only on corrected, smoothed measurements. In addition, we control the highest and lower predicted values using parameter Δ and redefine the criterion as:

$$\hat{z}_t - \Delta \cdot \delta\hat{z}_t > \hat{y}_t + \Delta \cdot \delta\hat{y}_t \tag{4}$$

Note that in absence of better APs on the current (or the overlapping channels), all three of the above schemes do not force any action. In other words, if there are no better APs operating on the current channel, then the above schemes do no trigger a handoff event. However, if the default handoff mechanism of the driver detects serious link deterioration, then a scan-based handoff is triggered.

5. EXPERIMENTATION IN A REAL ENVIRONMENT

In this section, we first describe the setup for the experimental evaluation of different triggering schemes described in the previous section. We then present measurement results for a comparative study of these schemes.

5.1 Testbed

The triggering experiments were performed using an IBM Thinkpad (T30) laptop that has Fedora Core 4 operating system and Linux Kernel 2.6.11-1.1369. We installed an Intel 2915ABG card on the laptop. We implemented the triggering algorithms in the linux driver ipw2200-1.0.6 (driver is open source, available at [5]). No changes were necessary at the low level microcode or firmware of the card. Also, no changes were made at the infrastructure side of the network. The infrastructure side of the wireless network consisted of 18 APs managed by the Computer Laboratory of University of Cambridge. There are four APs on channels 1, seven APs on channel 6, and seven APs on channel 11. All the APs use 802.11g. For AP locations and channel number assignment see [18]. The network does not use link layer encryption. However, only those clients are allowed to associate with the network whose hardware address is registered with the Computer Lab.

The experiments were performed either very early in the morning, or very late at night to minimize the impact of background traffic, movement of people, and interference caused by microwave and other such appliances. During each experiment, we carry the laptop, and walk along the first floor of the Computer Lab, and return to the starting point (see [18]). The walk takes about 5 minutes. Throughout each experiment we generate two way UDP traffic between a remote wired machine and the test laptop, emulating a VoIP call. The traffic consisted of fixed sized UDP datagrams (180 bytes each) spaced by about 22 milliseconds to have an overall data rate of about 64 Kbps to mimic the VoIP traffic generated by the G711 codec [19].

5.2 Implementation of Algorithms in the Driver

To implement the triggering schemes proposed in the previous section, we change the driver of the Intel 2915ABG card to instruct the firmware to pass on all the management frames that can be successfully decoded by the physical layer by setting the `accept_all_mgtm_frames` bit during the card initialization. We use this to capture the beacons of all the APs on our operating channel[1]. We add modules to the driver to continuously monitor the strength of the beacons received from all the active APs in our neighborhood. How-

[1]We can also capture the beacons of the APs operating on the overlapping channels. However, all the APs in our experiments operate on orthogonal frequencies, and hence only the beacons of APs operating on our current frequency were "visible" to the driver.

ever, we do not pass all these beacons to the 802.11 module. Only the beacons of the associated AP are passed to the ieee 802.11 module. We compute the EWMA value of the RSSI of all visible APs (equation (1)) and store its value for the past five seconds (about 50 beacons, e.g. LSE(50)). The RSSI information is updated on reception of beacons.

Since the beacons are transmitted once every 100 ms, an absence of a beacon message after this duration implies that either the signal was too weak to decode due to shadow-fading, or we are moving out of range of the AP. Hence, for a missed beacon, we replace the signal strength by -80 dB. This is done to make sure that the moving average value of RSSI of an AP decreases as the client moves out of range of the AP. We flush an AP entry from the data structure when its moving average RSSI value falls below -79 dB. This is because at RSSI values below -80dB, the channel becomes extremely lossy and unreliable.

In the following, we discuss the choice of parameters for different triggering schemes that we implement. Although the experiments were performed with a fixed set of parameters, we tested a wide range of parameters for each of the algorithms using measurement-driven simulations (more details in Section 6).

5.2.1 *Threshold based algorithm*

This algorithm triggers a handoff when the EWMA RSSI value of the associated AP falls below a fixed threshold (-70dB for our experiments). When the average RSSI falls below this threshold, the driver triggers a scan, and then chooses the new AP based on the highest received RSSI in the scan responses.

5.2.2 *Hysteresis (Δ) algorithm*

As described in Section 4, the algorithm uses hysteresis for switching from one AP to another when there are multiple APs in the client's neighborhood. In our experiments, the hysteresis factor Δ was 5, i.e., the new AP should have an average RSSI of at least 5dB higher than the currently associated AP. If so, we disassociate from the current AP, and associate with the new AP.

5.2.3 *Trend (L, Δ)*

For this algorithm, we use $L = 50$, i.e., we compare the current RSSI value with the RSSI value 50 beacon periods ago. We use $\Delta = 0.09$. Thus, if the RSSI of the current AP is decreasing at a rate of at least 0.9dB per second, and if the RSSI of another AP is increasing at a rate of at least 0.9dB per second, then we trigger a handoff. This rate of change of RSSI was chosen based on our empirical studies in which we found that at typical pedestrian speeds, when a client moves away from an AP, the RSSI decreases at a rate of about 1 to 2 dB per second. Using a high value of Δ results in lack of responsiveness of the algorithm, while a low value results in frequent transitions which cause firmware errors.

5.2.4 *LSE (L, Δ)*

For this algorithm, we use $L = 50$ and $\Delta = 1$ (see equation (4)). When the driver notes that the predicted RSSI value of an AP is better than the predicted RSSI value of the current AP (even after considering the possible error in estimation), the driver triggers a handoff to the new AP. Among all the algorithms, this is the most computationally intensive algorithm.

From Fig. 1 we note that if the RSSI of the current AP is more than -50 dB, then we do not have much to gain by handing off to a new AP. Hence, we use a threshold of -50dB for the operating region. In other words, our algorithms do not trigger a handoff as long as the average RSSI value of the associated AP is more than -50dB.

5.3 Results

In this subsection, we present the comparative performance of different triggering algorithms. For each scheme, the RSSI of the associated AP, and the index of the associated AP is shown in Fig. 8. In these plots, the y-axis on the right side corresponds to AP id. APs on channel 1 have ids from 0 to 3, APs on channel 6 have ids from 4 to 10 and APs on channel 11 have ids from 11 to 17. The out-of-band handoffs are highlighted with vertical dashed bars. We observe from these figures that Beacon-based and Threshold-based schemes are reactive in nature, because they trigger a handoff when the RSSI of the associated AP degrades considerably. However, the Hysteresis, Trend and LSE schemes are proactive in nature, because they are constantly on the lookout for a better AP. Note from Fig. 8 that our proposed schemes have most of the handoffs taking place in-band, since these handoffs are not scan-based. Also note that even the beacon based scheme has in-band handoffs, but it still has to go through the time consuming scanning phase.

To demonstrate how our proposed algorithms choose the best AP on the current channel at all times, in Fig. 7, we plot the RSSI seen by the client for all the APs operating on channel 11 when the Hysteresis algorithm is used. We also plot the id of the AP that the client is associated with as a function of time. We observe that all the handoffs in the first 250 seconds are within channel 11: AP 14 (purple square) to AP 11 (red plus) to AP 12 (green cross) to AP 16 (brown circle) to AP 13 (blue star). At around 280s, the client can no longer find any better AP on channel 11, and so it opts for a scan of all the channels. Thus, our proposed algorithm succeeds in identifying, and associating with the best quality AP within its operating channel through "in-band" handoffs.

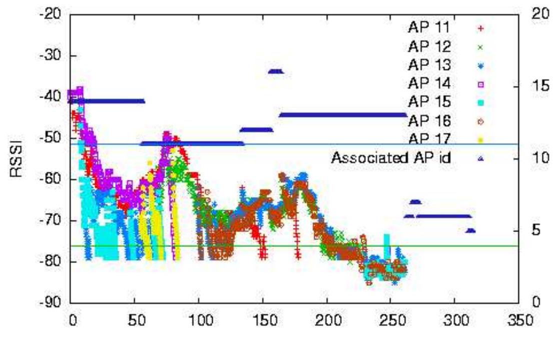


Figure 7: RSSIs as seen by the client when operating on channel 11, and corresponding handoffs with Hysteresis scheme.

Such improvement in terms of client roaming decisions is likely to be better observed through the performance of VoIP traffic. There are three important metrics to study with respect to VoIP traffic (i) average handoff delay, (ii)

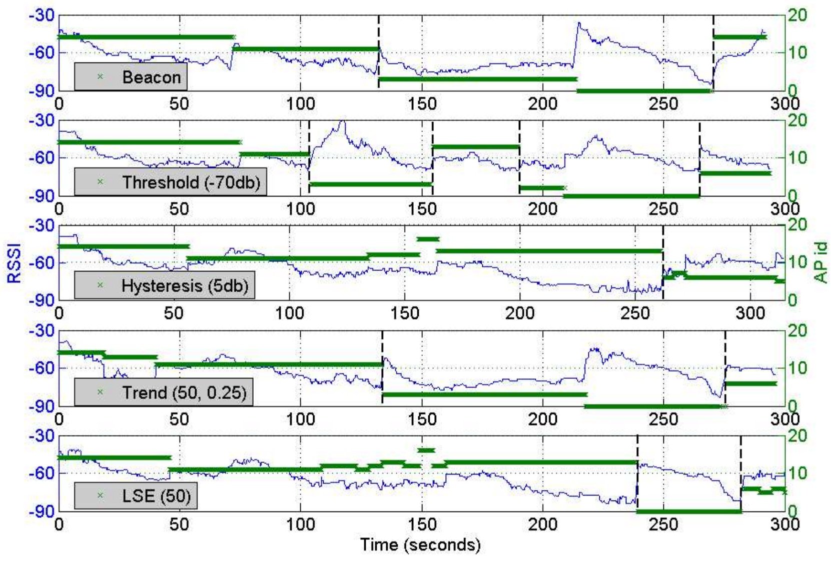


Figure 8: Time evolution of RSSIs of all the "visible" APs as seen by the client during the experiment, and associated AP transitions.

packet loss, and (iii) delay variation. We summarize these results in Table 2 and Table 3.

Both *Beacon* and *Threshold* schemes lead to average hand-off delays between 530 and 860 ms since they rely entirely on scan-based handoffs. As Table 2 shows, a scanning based handoff results in 50 to 90 packet losses per handoff for VoIP traffic. This amounts to close to 100% packet loss over the downlink for the handoff duration. These results are consistent with VoIP packet losses for handoffs in 802.11b WLANs as presented in [20]. On the other hand, our proposed schemes (Hysteresis, Trend and LSE) try to avoid the scan-based handoffs. Instead, if a better AP is available within the same channel, the driver hands off to this AP. This merely involves disassociation, authentication and association delays. We can see from Table 2, that an in-band hand-off requires merely 24 ms. However it is possible that sometimes there is no better AP operating on the current channel, and this means that we do not have any other option, but to scan all the channels. In spite of this, Table 3 shows that all three proposed schemes (Hysteresis, Trend and LSE) have relatively low average handoff delays (140 to 450 ms) as compared to Beacon and Threshold schemes.

Since an in-band handoff is very short, it results in just 1-2 packets lost per handoff over the downlink (see Table 2). Consequently, as Table 3 shows, the average packet loss over the entire experiment is considerably lower for proposed schemes as compared to the Beacon and the Threshold schemes.

During each handoff, the driver buffers the uplink packets since the default gateway entry in the IP table of the client does not change during the handoff. Consequently, fewer packets are lost over the uplink as compared to the downlink. However, if the handoff duration is too long, then buffering could result in substantial delay variation. We plot the packet inter-arrival times as seen at the receivers in Fig. 9. Note that the y-axis is on a log scale. We note that whenever there is a scan-based hand-off, there is a large variation (spikes) in the inter-arrival time as seen by the receivers (especially the wired receiver, i.e., the uplink traffic). For the Hysteresis, Trend and LSE schemes, we note that such high delay variability regions appear only during the scan based handoffs (see Table 3 for the number of scan-based handoffs). The in-band handoffs do not even appear on the delay-variability plot. This goes to show that the in-band handoffs are truly seamless, in that they result in very low delay variation and very low packet loss.

Besides the regions of very high delay variability that can be attributed to scanning, there are some other fairly long durations of medium to high delay variability. These regions can be attributed to link level retransmissions due to consistently poor channel conditions. For example, for Hysteresis just before the first scan-based handoff (at around 260s), the channel is in a poor state, and this results in several link level retransmissions.

5.4 Implications and Limitations of Experiments

From the above experiments, we conclude that in-band handoffs are preferable to scan-based handoffs, especially for delay sensitive applications such as VoIP. This is because in-band handoffs are more than one order of magnitude shorter than scan-based handoffs, and therefore result in very low packet loss and delay variability. We also observe that continuous monitoring of APs enables the client to make smart handoff decisions. Our proposed schemes (Hysteresis, Trend and LSE) use these in-band handoffs by continuously monitoring all the APs on the current channel.

We acknowledge that choosing the best AP simply based on information about the APs operating on the current channel is sub-optimal. For example, there could be an AP with a much stronger RSSI operating on another channel, but we might hand-off to an AP with relatively weaker RSSI

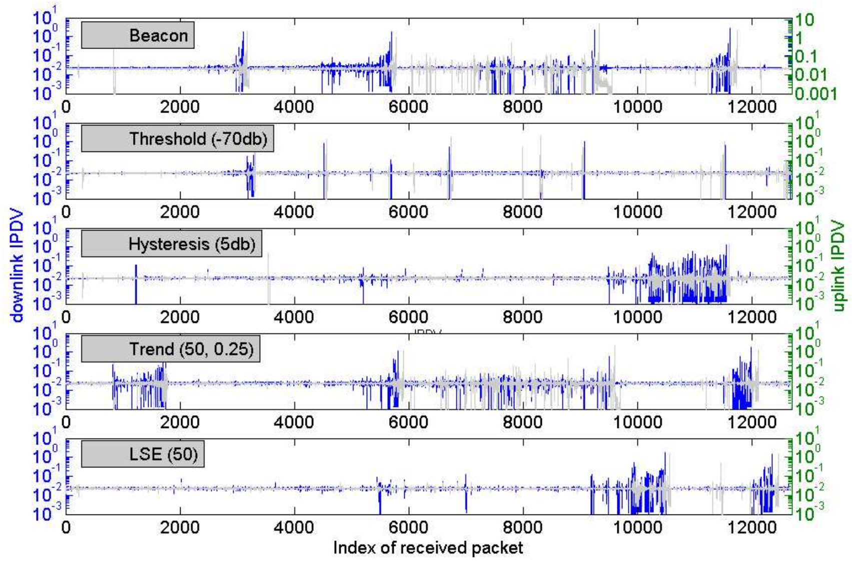


Figure 9: Time evolution of Inter-packet delays.

Type of handoff	Average duration (ms)	Downlink packets lost per handoff	Resultant Interpacket delay variation
In-band	24	1-2	30-40 ms
Scan-based	600-1000	50-90	1-9 seconds

Table 2: Handoff scheme comparison for VoIP traffic

just because it is in our frequency band. However, we believe that for delay sensitive applications, scanning delays on the order of 1000 ms are unacceptable. Our proposed schemes considerably reduce these delays at the cost of potentially lower throughput. Since our experiments primarily focused on delay sensitive traffic, we only talk about delay and packet loss. We comment on the aspects of achievable throughput through different schemes in the next section.

We tried to compare the performance of all the triggering schemes in terms of throughput by generating high traffic on the uplink, and later on the downlink. However, we found that the process of in-band handoff is disrupted when the client is sending or receiving high traffic. This is a known problem with Intel 2915ABG card [21], and we suspect that the problem lies in the firmware. We observe that under very high traffic conditions, the driver disassociates with the old AP and associates with a new AP just as required by the algorithm. However within a few milliseconds, the driver receives a deauthentication message from the firmware, thereby triggering a new scan, and a new association. This could potentially be due to the fact that some of the control messages get lost under high traffic conditions. We are currently addressing this issue.

6. SIMULATION RESULTS

In the previous section we evaluated all five triggering algorithms using an actual implementation in a campus environment. Based on this implementation of the algorithms we were able to assess handoff delays involved in each case, as well as quantify the application performance a user should expect while implementing these schemes. For this, we studied the performance of the algorithms with VoIP traffic. However, given that we used a real card, there were specific aspects of the algorithms that we could not test, and which are mainly an artifact of today's technology. Firstly, we could not observe the APs operating on non-overlapping frequencies while the client is on a given frequency (since we cannot scan all the channels continuously). Due to this our triggering criterion functioned on local, rather than global knowledge. If one had the capability of *continuously monitoring* the RSSI measurements of APs operating over *all* the channels, then globally optimum handoff decisions can be made. This could be the state of the art in the future. Secondly, we could not test excessively resource intensive algorithms due to the inability of the card firmware to perform such handoff actions very quickly. Consequently, our experimental results needed to test "stable" algorithms. Hence the experiments did not address the performance of the algorithms when smoothing is not present. Thirdly, we could not experimentally calibrate the algorithm parameters due to the excessive amount of time such a task would take. Finally, we could not assess the performance impact in terms of user throughput due to the bug reported in [21].

In order to address the above issues we use event driven simulations. The input to our simulator is a beacon trace collected along the exact same path as the one followed in Section 5. In order to gain visibility into all frequencies op-

	Scan-based handoffs	In-band handoffs	Total handoffs	Average scan-based handoff delay (ms)	Average Inband handoff delay (ms)	Overall average handoff delay (ms)	Uplink % packet loss	Downlink % packet loss
Beacon	4	-	4	867	-	867	2.54	3.36
Threshold	6	-	6	534	-	534	2.29	2.04
Hysteresis	1	7	8	1102	24.7	159	0.41	0.72
Trend	2	3	5	1110	41	468	0.96	1.01
LSE	2	12	14	850	23.9	142	0.97	1.66

Table 3: Delay and Packet Loss comparison of different triggering schemes

erational in the building we put the 802.11g wireless client in monitor mode and walked around the building three times, having the client tuned to channel 1, 6, and 11 respectively and maintaining the same speed. The superposition of the three beacon traces provides us with an approximate global view of the state of the frequencies throughout the walk.

Our methodology comprises the following steps. Using the "global" trace as input we run each algorithm and identify the AP selected at each point in time. While there is no real transition between APs, such a task provides us with a time-series of RSSI measurements for the client (see Fig. 10). These RSSI time series can be converted to throughput according to the relationship captured in Fig. 1. If there is a transition, then we need to take into account the switching cost from the previous AP to the new AP. If the transition is "in-band" then our experimental results showed that it takes 20 ms for the client to disassociate from the previous AP and associate with the new AP in the same frequency. If the transition is "out-of-band" we additionally need to take into account the time it takes to tune to a different channel, which was found to be equal to 5 ms in our experiments. Given that both times are smaller than 100 ms, our operating time scale, we account for these effects by adjusting for the client rate as (i) 80% of the nominal rate for "in-band" transitions, and (ii) 75% of the nominal rate for "out-of-band" transitions. The aforementioned methodology captures the behavior of the system at the observed time scale of 100 ms. Notice that effects such as shorter time scale variations and rate adaptation (e.g., Auto Rate Fallback) cannot be taken into account due to the coarse scale of the input measurements. The reason for using such a "global" view of the neighboring APs, is to compare the performance of our triggering algorithms (which use local information) with the best possible scheme (which uses global information). With the current technology, it is not possible to obtain the global view at all times, but such a comparison at least shows how far off we are from the optimum.

The evaluation of the different schemes covers the following dimensions: (i) Number of transitions; smaller number of transitions implies less "service disruption" since it corresponds to less amount of time when the client does not do real work; a duration that can get even larger in the presence of complex authentication/authorization schemes. (ii) Average client throughput. (iii) Average amount of time the clients stays affiliated with any one AP. (iv) Fraction of time when the client cannot communicate (RSSI=-80db). (v) Total amount of time spent in handoffs.

6.1 The need for smoothing

We process the collected data set to derive the time-series of signal strength measurements per AP within reach for a 5 minute walk around the building. The matrix of measurements has been shown in Fig. 2. Notice that towards the middle of our experiment excellent performance can be offered by a set of APs that exhibit almost similar performance; these APs were found in the same (x,y) location but on different floors. This phenomenon may cause multiple transitions if instantaneous performance is taken into account with small hysteresis factors, or if trending is performed with small Δ values.

Out of the five schemes studied, only the *Threshold* and *Hysteresis* schemes can operate on raw measurements. Even the *Beacon* scheme incorporates a notion of history since it counts the number of lost beacons. We study the performance impact of smoothing on the *Threshold* and *Hysteresis* schemes in Table 4.

We notice that the *Threshold* and *Hysteresis* schemes lead to an excessive number of transitions when operating on raw RSSI measurements. Transitions are such that the average affiliation time with any AP is only 4 time intervals. Each interval is 100ms long. The positive side effect of such agility is that the average throughput is very high since the algorithms are capable of exploiting the best APs at each point in time. However, it is questionable whether any actual wireless card can achieve such high performance when it needs to transition every 400 ms.

On the other hand, smoothing of RSSI measurements can lead to a dramatic reduction in the number of transitions, in excess of 95%. The average affiliation with an AP increases to more than 100 time intervals for both algorithms while the average throughput is reduced by less than 9% in both cases. In addition the total handoff delay has dropped from more than 11 seconds to 460-860 ms.

The *Beacon* scheme performs worse than the *Threshold* scheme as well as the *Hysteresis* scheme. The reason for that may be due to the fact that a client remains associated with the AP for 8 beacon intervals before it initiates the AP selection algorithm. Direct implications of such a design choice is higher stability, in terms of average AP affiliation times, and smaller number of in-band and out-of-band transitions.

6.2 The impact of the criterion

Once triggering schemes operate on smooth measurements, significant behavioral differences are expected due to the use of different criteria. From Table 4, we can see that all new proposed schemes outperform the *Beacon* scheme in terms of throughput with the exception of *Trend*. Moreover, such a performance limitation is not due to the parameter calibration. We explored the space of parameter setting for *Trend*, and as long as $\Delta * L < 6$, the performance was consistent. We believe that the reason behind *Trend*'s poor performance is due to the fact that it tends to stay affiliated with one AP for long periods of time (on average 300 time intervals, see

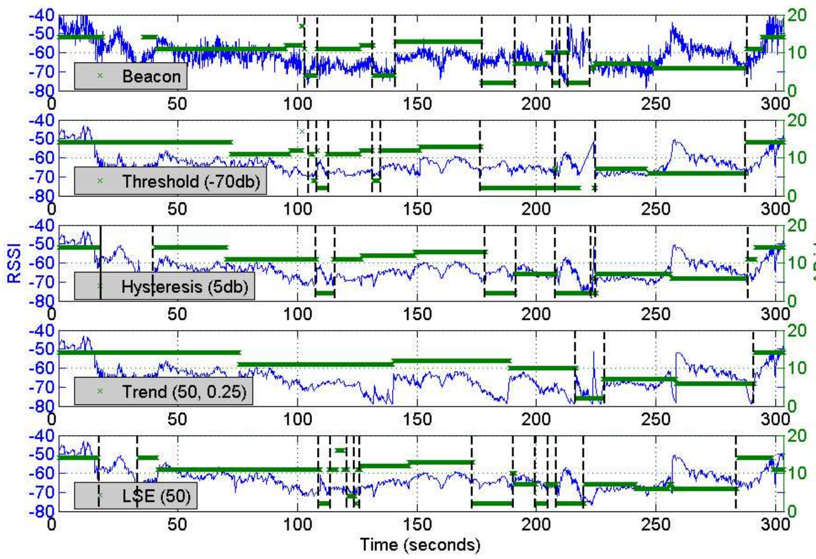


Figure 10: Time evolution of RSSI achieved through the different schemes and associated AP transitions (event-driven simulation). Vertical dashed bars indicate out-of-band transitions.

Scheme	In-Band	Out-of-Band	Disruption	Average rate	Average stay	Total handoff delay (s)
				without History		
Threshold	246	265	5%	5.39 Mbps	4.42	11.54
Hysteresis of 5	262	265	1.57%	5.56 Mbps	4	11.86
				with History		
Beacon	13	20	17%	4.64 Mbps	103.79	0.76
Threshold + History	12	25	15%	4.92 Mbps	108.91	0.86
Hysteresis of 5	8	12	12%	5.08 Mbps	137.2	0.46
Trend (50, 0.25)	4	5	23%	4.28 Mbps	288.5	0.2
LSE (50,1)	11	15	11%	5.17 Mbps	99.15	0.59

Table 4: Impact of history. Smoothing schemes use 0.9 as a smoothing factor. Average rate is computed using Fig. 1.

Fig. 10), which gives it little flexibility to discover better performing APs, i.e. it operates on a rather strict criterion. On the other hand, *LSE* presents the best throughput while leading to total handoff delays comparable to the ones of *Hysteresis*. We believe that *LSE* offers the best point in the tradeoff between overhead and reactiveness. *LSE* leads to a small number of transitions, lasting 0.59 seconds in total. Moreover, it is capable of identifying sequences of APs that feature very low signal for only 11% of the time, while achieving 5.17 Mbps on average. Notice that the low signal time intervals do not necessarily correspond to inability to communicate as mentioned in Section 3.

6.3 The impact of operating region

Lastly, we look into the impact of the use of the operating region on the performance of the triggering algorithms. Our results indicate that when the triggering algorithms operate on smoothed measurements, then the operating region parameter has no evident impact. On the other hand, if the triggering algorithms operate on raw measurements (e.g. *Threshold*, *Hysteresis*) then looking at the operating region of the current AP can prevent the card from temporarily selecting other APs in its neighborhood with similar signal strength measurements. This became evident in our experiments, since the APs were laid out in the building in such a way that the same position (x,y) was occupied by an AP on all three floors, and these three APs used the same frequency. Consequently, when our client was close to any of these APs, it tended to briefly affiliate with the APs on the other floors. The operating range criterion can prevent such a ping-pong effect.

6.4 Sensitivity Analysis of LSE

In the previous section we identified *LSE* as the algorithm that offers the best tradeoff between reactiveness and overhead. Such an observation was further made in comparison to the most favorable setting of the rest of the algorithms in the way they balance transitions and throughput. *LSE* itself features two parameters capturing the amount of history used and the sensitivity to prediction errors. In Fig. 11, we show the simulated throughput achieved by *LSE* versus the different combinations of (L, Δ), as well as *LSE*'s performance in terms of in-band and out-of-band transitions. We notice that a value of $\Delta = 1$ is capable of taking into account inherent prediction errors without greatly delaying *LSE*'s reaction to environmental changes. In addition, a value of $L = 50$ (e.g 5 seconds) provides a good tradeoff between throughput and number of transitions for walking users. In faster environments, smaller values of L are likely to be better at exploiting the short term wireless opportunities.

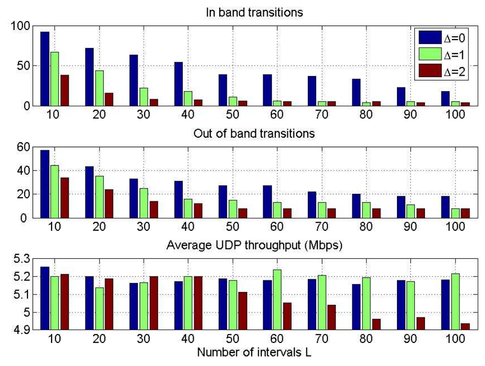


Figure 11: Impact of history duration, and error term on number of transitions and average UDP throughput.

7. CONCLUSION

In this work, we note that most state of the art handoff algorithms in 802.11 networks wait until the loss of connectivity or poor and unsustainable performance before triggering a handoff. These algorithms further rely on instantaneous signal strength measurements when choosing a new AP during the handoff. We propose a fundamentally different approach, in which the client *continuously monitors* the quality of the links of all the APs operating on the client's current and overlapping channels. The ability to capture the *long term trend* in the link quality enables us to design handoff algorithms that are more robust to channel variations. We propose, implement, and experimentally study a range of handoff algorithms within this framework. The current generation of wireless cards can only monitor APs on the current and the overlapping channels. However, we show through experiments in a campus network that even with this limited information, the average handoff delays can be reduced by more than 50%. Our proposed algorithms can be easily implemented in the driver of a wireless card, and *unlike other proposed roaming algorithms in the literature, require no infrastructure support*.

Due to the configuration of the wireless network featured in our measurements, we could not test our algorithms in the presence of APs operating on overlapping channels. We intend to study the above case in our future work. In addition, we would like to conclude by saying that our results clearly demonstrate that there is a tradeoff between handoff delay and achievable throughput. We believe that the ultimate tuning of our algorithms will have to be a function of the performance requirements of the client.

8. REFERENCES

[1] A. Akella, G. Judd, P. Steenkiste, and S. Seshan, "Self management in chaotic wireless deployments," in *ACM Mobicom*, Cologne, Germany, Aug. 2005.

[2] H. Velayos and G. Karlsson, "Techniques to Reduce IEEE 802.11b MAC Layer Handover Time, Tech. Rep. TRITA-IMIT-LCN R 03:02, Apr. 2003.

[3] A. Mishra, M. H. Shin, and W. Albaugh, "An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process," *ACM SIGCOMM Computer Communication Review*, vol. 3, pp. 93–102, Apr. 2003.

[4] S. Seshan, H. Balakrishnan, and R. Katz, "Handoffs in Cellular Wireless Networks: The Daedalus Implementation and Experience," *Wireless Personal Communications: An International Journal*, vol. 4, pp. 141–162, Mar. 1997.

[5] "Intel PRO/Wireless 2200bg driver for linux."

[6] I. Lucent Technologies, "Roaming with orinoco/ieee 802.11," Tech. Rep., Dec. 1998.

[7] H. S. Kim, S. H. Park, C. S. Park, J. W. Kim, and S. J. Ko, "Select Channel Scanning for Fast Handoff in Wireless LAN using Neighbor Graph," in *International Technical Conference on Circuits Systems, Computers and Communications*, Sendai/Matsusima, July 2004.

[8] I. Ramani and S. Savage, "SyncScan: Practical Fast Handoff for 802.11 Infrastructure Networks," in *IEEE Infocom*, Miami, FL, USA, March 2005.

[9] A. Balachandran, P. Bahl, and G. Voelker, "Hot-spot congestion relief and service guarantees in public-area wireless networks," *SIGCOMM Computer Communication Review*, vol. 32, no. 1, 2002.

[10] Y. Bejerano, S. Han, and L. Li, "Fairness and load balancing in wireless lans using association control," in *Proceedings of ACM Mobicom*, Philadelphia, PA, USA, Oct 2004.

[11] S. Vasudevan, K. Papagiannaki, C. Diot, J. Kurose, and D. Towsley, "Facilitating Access Point Selection in IEEE 802.11 Wireless Networks," in *ACM Sigcomm IMC*, Berkeley, CA, USA, Oct. 2005.

[12] A. Gilbert, "Intel, cisco fine-tune wireless networking," ZDNET.com, August 2005.

[13] C. Na, J. Chen, and T. Rappaport, "Hotspot traffic statistics and throughput models for several applications," in *IEEE Globecom*, Dallas, TX, USA, Nov. 2004.

[14] Broadcom, "The new mainstream wireless lan standard," [Online].

[15] A. Mishra, E. Rozner, S. Banerjee, and W. Arbaugh, "Exploiting Partially Overlapping Channels in Wireless Networks: Turning a Peril into an Advantage," in *ACM SIGCOMM IMC*, Berkeley, CA, USA, Oct. 2005.

[16] V. Brik, A. Mishra, and S. Banerjee, "Eliminating handoff latencies in 802.11 WLANs using multiple radios: Applications, experience, and evaluation," in *ACM SIGCOMM IMC*, Berkeley, CA, USA, Oct. 2005.

[17] G. P. Pollini, "Trends in handover design," *IEEE Communications*, vol. 34, pp. 82–90, Mar. 1996.

[18] "Ap locations for 802.11 handover experiments." [Online]. Available: http://www.cambridge.intel-research.net/~kpapagia/handover/

[19] D. Chen, D. Gu, and J. Zhang, "Supporting real-time traffic with qos in ieee 802.11e based home networks," Mitsubishi Electric Research Laboratory Technical Report, TR-2004-006, Tech. Rep., Feb. 2004.

[20] A. Cabellos-Aparicio, R. Serral-Gracia, L. Jakab, and J. Domingo-Pascual, "Measurement based analysis of the handover in a wlan mipv6 scenario," in *PAM*, Boston, MA, USA, Mar. 2005.

[21] Bugzilla, "Firmware error during large ftp transfers." [Online]. Available: http://bughost.org/bugzilla/show_bug.cgi?id=800