Peeling the 802.11 Onion: Separating Congestion from Physical PER

Malik Ahmad Yar Khan
ARC Special Centre for Ultra-Broadband Information Networks (CUBIN), an affiliated program of National ICT Australia (NICTA), Dept. of Electrical and Electronic Engineering University of Melbourne, Australia a.malik@ee.unimelb.edu.au

Darryl Veitch
ARC Special Centre for Ultra-Broadband Information Networks (CUBIN), an affiliated program of National ICT Australia (NICTA), Dept. of Electrical and Electronic Engineering University of Melbourne, Australia dveitch@unimelb.edu.au

ABSTRACT
An ability to accurately classify observed packet errors according to their root cause: physical layer or MAC layer contention, in 802.11 networks, opens up many opportunities for performance improvement at the both the MAC and IP layers. We investigate two orthogonal ways to achieve this, and present a methodology, the 'map', for clearly visualizing the results free of packet size and rate 'bias'.

Categories and Subject Descriptors
C.2.3 [Network Operations]: Network monitoring; C.2.5 [Local and Wide-Area Networks]: Ethernet (e.g., CSMA/CD)

General Terms
Experimentation, Measurement, Verification.

Keywords
IEEE802.11, passive monitoring, packet error rate, congestion, cross layer design, measurement.

1. INTRODUCTION
It is widely recognized that the fact that there are two significant sources of packet error in paths containing a 802.11 wireless hop, one due to the wireless physical layer, and the other due to congestion over all hops, is an inherent cause of performance difficulties, and one that calls for cross layer solutions.

This problem has been noted in particular in the context of TCP performance [1, 2, 3]. TCP, which senses network state primarily through a packet loss signal, assumes it to be caused by congestion, and cannot distinguish between congestion and physical losses. Retransmission at the wireless link layer only partially addresses the problem, as the resulting increase in packet latency (replacing loss with extra delay) also affects TCP, and link layer losses can still occur. Another example is rate control in the 802.11 MAC itself. Algorithms currently in use sense and act upon (either implicitly or explicitly) total packet error over the wireless link, and cannot distinguish between errors due to a failure to decode at the physical layer, or collisions resulting from the random backoff of the CDMA mechanism. The effects can be significant because of the wide range of transmission rates selected by these algorithms (an example is given in Figure 7). A third area is the primitive state of active probing techniques. In wireline networks, a link can be successfully characterised by a constant link capacity, FIFO packet scheduling responsible for congestion, and a Packet Error Rate (PER) which is so low that it can be ignored. This basic model has led to much development in wireline networks of probing methods to measure end-to-end bottleneck bandwidth, available bandwidth, cross traffic and more. In contrast, in 802.11 networks the capacity (i.e. transmission rate) varies widely depending on conditions, scheduling is contention based, and interference and contention effects result in highly variable total PER (TPER). Because neither physical PER (PPER) nor contention related PER (CPER) can be ignored, and because they are strongly coupled (increased PPER triggers changes in channel capacity), developing active probing based inference is far more difficult.

The above examples show that there would be considerable benefits, at multiple layers in the protocol hierarchy, if independent, reliable information were available for each of PPER and CPER for any given 802.11 link in a path. In this paper we investigate two different approaches for isolating and measuring these for a single wireless hop, each based on passive measurements made at a client station. The first is based on the observation that packet errors measured during periods when the channel load is low will be closer to the fragmentation approach holds. Using it, we can clearly observe the interplay between PPER and CPER without interference from rate and packet size dependencies. The map, and our approach in general, is station specific. We are not attempting to characterise the entire network, but only to measure PPER and CPER pertaining to a given station. This information could be used by the station itself to adjust its transmission rate intelligently (we investigate this application in depth in a separate work [4]), be passed to a higher layer for TCP optimization or other purposes, or be exploited to perform bandwidth estimation.
The paper is structured as follows. Section 2 briefly describes relevant prior work, and Section 3 introduces the 802.11 standard and our testbed. Section 4 introduces our validation methodology and introduces the map concept. Section 5 investigates the utilisation based approach of measuring PER, and in so doing provides information of the relative size of PER and CPER and illustrates the pitfalls of the existing approaches. Section 6 introduces and evaluates the fragmentation method. We conclude and discuss directions for future work in Section 7.

2. RELEVANT WORK

There is little work which focuses directly on how to isolate PPDR and CPER based on low level measurements in the way we do here. To the best of our knowledge, we are the first to use fragmentation to classify errors as physical or contention related. Recently however CARA [5] has been proposed which can (in simulations) distinguish between physical errors and collision errors by employing RTS/CTS. In [6] a variant of CARA was implemented. We demonstrate the actual improvements achieved by employing RTS/CTS and fragmentation based techniques on the rate selection algorithms SampleRate and AMRR using real implementations in a testbed in [4].

Detection of congestion or non-congestion related errors at the TCP layer using end-to-end measurements over wireless ad-hoc network are discussed in [2, 3], while [1] considers the impact of TCP performance on a wireless multi-hop link. In contrast, we present an inexpensive local isolation of congestion errors, which can then be used by wireless-aware TCP implementations (like [2]).

In [7] a scheme is devised to distinguish collisions from physical errors which requires comparing frames transmitted from the AP to those received at the client station. The need for cooperation from the AP severely limits the practical applicability in real networks.

Most of the rate adaptation techniques such as Auto Rate Fall back (ARF) [8] and Adaptive Autorate FallBack (AMRR) [9] use error based thresholds to change the transmission bit-rate of the card. SampleRate [10] chooses the data rate which gives the highest throughput and samples the channel with different rates to get throughput related statistics. Recently a study of these rate control algorithms in a congested environment in the ORBIT testbed [11] was presented in [12], which shows that all error-threshold based algorithms suffer from contention errors and unnecessarily decrease the throughput. Rate based algorithms which measure RSSI information only (e.g. [13, 14]) use a physical layer model and/or threshold driven decision algorithm to change the rate whenever RSSI values fluctuate. Many researchers including ourselves have found that RSSI information and physical layer models are very poor predictors of performance in indoor environments.

Transmit Power Control (TPC) allows a station to conserve energy by transmitting at a lower power level and can also enhance the spatial reuse of 802.11 based networks [15]. Collision errors can force the stations to always transmit at high power, hampering the usefulness of TPC. Very significant improvements in TPC based algorithms can be achieved if they could only base their decisions on physical errors.

3. BACKGROUND

3.1 The 802.11 Wireless LAN Standard

We will describe details of the 802.11{a,b,g,n} MAC for the widely used infrastructure mode, where a central Access Point (AP) bridges the data between all participating (or associated) stations (STAs).

The Distributed Co-ordinated Function (DCF), the default MAC mechanism to transmit and receive frames, uses CSMA/CA. When a data frame accesses the channel, the destination station transmits an acknowledgment (ACK) frame back to the source after a Short Inter-Frame Space (SIFS) interval. Frames may be corrupted due to physical layer issues, collisions of contending data frames, or collisions of ACK and data frames involving hidden terminals. In any case, if the data frame does not have a valid Cyclic Redundancy Code (CRC), or if the sender does not receive a valid ACK, then transmission is unsuccessful. The MAC has a maximum frame size which we call $F_{max}$. If an IP packet is too large, it will be split into multiple fragment frames sent close together with no backing off. A gap only SIFS wide is left between ACK from the previous fragment and the next fragment (see Figure 4).

3.2 Testbed

Our testbed consists of wireless clients, access points, PCs and traffic generators. We chose Linux as the operating system in order to benefit from open source driver code, and focussed on 802.11a,e to avoid interference from neighboring 802.11{b,g} networks. Experiments were conducted in an office environment, both during the day and at night. We use the 802.11e extension to 802.11a in best effort mode only, where it behaves essentially like 802.11a.

The possible rates for the data part of the data packets are $r = \{6, 9, 12, 18, 24, 36, 48, 54\}$.

The wireless hardware consists of Linksys [16] and Netgear wireless cards, based on Atheros chipsets [17], which can operate in each of 802.11{a,b,g} networks. We used the Multiband Atheros Driver for Wireless Fidelity (MADWIFI-ng) [18], a Linux kernel driver for Atheros-based Wireless LAN devices. The PCs are each Pentium III or higher, running Fedora Core 5, kernel version 2.6.18.

We make extensive use of accurate passive monitoring. Most wireless card manufacturers (including Linksys and Netgear) provide a monitor mode, similar to promiscuous mode in wired Ethernet, whereby all valid packets within range of the card on a given channel are captured, even those not destined for it. The MADWIFI driver supports client or AP mode being used simultaneously with monitor mode. Monitored IP packets have MAC headers and physical layer PRISM headers appended, and are subsequently caught via the libpcap capture library and stored. The packet traces are later filtered by tcpdump and processed by programs we wrote in C++ and Matlab to access the information we require at different protocol levels.

The per-packet Received Signal Strength Indicator (RSSI) is available in all IEEE 802.11 compliant cards. The mapping between RSSI to SNR is not standardized by IEEE, but is supplied by vendors. For our cards, RSSI = SNR, and S = SNR + W where S is signal strength and W is room temperature thermal noise taken to be $-95$ dBm.

Because there are known problems with monitor mode [20, 21] (essentially monitors can fail to see all packets due to resource constraints), we installed an additional sniffer (which sends no traffic) near the target station to confirm our results. The sniffer uses a Linksys card, whereas the client station uses Netgear.

We use the ‘resent packet counter’ (the number of times the packet has been sent) to calculate Packet Error Rate (PER) precisely in client stations (at the sender side). The MAC retry bit is used for PER estimation of packets captured by the monitor but originating from other stations. The sniffing station is constrained to use this (less accurate) method.

To simulate competing background IP traffic we use D-ITG [22] to generate TCP and UDP traffic of various types.
and when a per-packet given no collision error occurs PPER is that we can now define CPER as the statistic which describes the conditional form of PPER. To describe it, first note that a packet can fail to be transmitted correctly on the first attempt. We adopt a packet-focused definition of packet error, rather than one based on throughput. In other words, we consider that our station attempts to send a (typical) packet in a given time slot, and define TPER as:

**TPER**: the proportion of time a data frame (and its ACK) fail to be transmitted correctly.

A time-averaged view of loss (closely related to throughput) would require that we keep track both of the number of attempts needed for successful transmission as well as how often and when the station makes the attempt (a function of backoff dynamics). In contrast, our definition is essentially conditional and only provides one (albeit, key) building block on the way to throughput. It asks the question: given that a packet transmission is attempted now, what is the probability of immediate success?

The chief motivation for a conditional definition is its simple relationship to PPER. To describe it, first note that a packet can fail to be received either due to collision or through suffering a physical error. One can certainly argue that both can occur simultaneously, however defining such an event precisely can be problematic. To avoid this, we consider that a station can only suffer a physical error conditional on the fact that it does not encounter a collision. This results in physical (PPER) and contention (CPER) errors being mutually exclusive, a convenient separation of error types which facilitates measurement. Hence we have

**PPER**: PPER = TPER given no collision error occurs.

As we see shortly, the key to the separate measurement of PPER and CPER is the isolation of PPER, and so it makes sense to settle on a definition of TPER which is in effect PPER-centric in this way. Another powerful reason for basing our definitions on a per-packet and conditional form of PPER is that we can now define CPER as simply:

**CPER**: CPER = TPER − PPER

Analytic results describing CSMA/CA dynamics also acknowledge the importance of conditional error probabilities. In an environment free of physical error, the classic result of Bianchi [23] call our TPER=CPER the ‘conditional collision parameter’ and expresses the system throughput in terms of it. Extensions have been derived (e.g. [24]) which include PPER in the definition of p in a way consistent with our TPER definition.

Having defined the errors, the two key problems now are:
(i) how to obtain PPER measurements which are reliable and free of contention effects,
(ii) how to deal with their dependence on other parameters.

We address (i) by performing mapping experiments, consisting of running packets between the station and the AP with no other cross traffic (CT) streams present. The map is a tool to measure packet errors under controlled conditions. We therefore disable rate control and make separate measurements for each fixed data rate. In practice we use a single greedy UDP source, simply to provide the opportunity to observe many errored packets quickly (such a source may generate some packet loss due to buffer overflow at the IP layer, but for our purposes this does not matter as there are no packets competing at the MAC level which is where we make our measurements). Our station has high enough performance to ensure that even under high load, packet drops by the monitor are minimal. We confirmed this in diverse ways, including using our sniffer node which monitored the same packets. Furthermore, we performed the same experiments (taking much longer) at low packet rates and found the same results, confirming that the hardware does not perform differently at high load.

Point (ii) is a non-trivial issue because PPER, far from being a single, is a quantity which varies according to packet size l, transmission rate r, and of course SNR and more generally the environment of the network. Furthermore, the relationship between these can be very different from what might be expected by simple models, because of the complexity of real radio environments. One widely recognised consequence (see for example [25]) is that RSSI can be a poor indicator of achievable throughput.

For our problem it is essential that we measure what is going on, rather than being influenced by the assumptions of particular models. Accordingly, we use an empirical approach, letting the data tell us what the station specific mapping between PER, RSSI, packet size and rate is. To achieve this in a way that does not obscure important details by excessive averaging, we view the data as effectively a scatter plot in the four dimensions of RSSI, l, PER and r, which we call the map.

In practice we can only populate a few points in this large space, and to make it ‘human readable’ we view it via a two-dimensional projection, of which many are possible. We found the most useful to be a projection into the (PER, r) plane, as shown in Figure 1. To generate a point, for a given radio environment (characterised by a single average value of RSSI) we fix (l, r), perform a mapping experiment as above, and record the measured average PER, resulting in the point (PER, l, PER, r). We repeat this for (l, l) values taking l in {48, 540, 1500} for each r in {6, 9, 12, 18, 24, 36, 54} collecting data for a total of 5 minutes for each point, built up by combining 10 intervals of 30 seconds each (this is done to avoid possible non-stationary effects such as fading from distorting the results).

In Figure 1 the plot axes give us the PER and r components of a point (RSSI, l, PER, r), RSSI is distinguished by color (two different radio environments are given), and packet size is indicated by the disk radius. The map represents a memory of the physical parameter relationships found in the environments experienced by the AP-station pair. The complexity of the relationships recorded in the map (for example PER is often not monotone in r and/or l as one might naturally expect) is not the product of non-stationarity.
nor of small sample size. We found maps to be reproducible on a time scale of several hours (or more) in our testbed, located in an office environment, and this was corroborated by corresponding SNR and PER timeseries which appeared very stationarity. This consistency also gives us additional confidence in the performance of the station monitor. Finally, we repeat that the map of Figure 1 was generated with no cross traffic, and hence PER in this case is in fact PPER. Using the map, we can see at a glance what the effect of congestion is on PER across the dimensions of \( l, r, \) and RSSI. In the remainder of this paper however, we focus on a single radio environment with an approximately constant RSSI. Note that mapping experiments are not a rate selection algorithm, nor are we suggesting here that they be executed over live networks. Rather, the map is a tool to systematically measure, verify and analyze error dependencies, which in this paper allows us to investigate the benefits of different techniques for the isolation of PPER.

5. PPER VIA CONDITIONAL UTILIZATION

If there is no traffic from other stations, then there will be no contention and measured PER will correspond to PPER. Even when the network supports multiple users, there are periods where, from the point of view of a given client station, the CT from other stations is low or even absent. Our approach is to exploit the natural burstiness of traffic to measure PER at those times.

We measure CT by measuring channel utilisation over contiguous intervals of bin width \( \delta \), to obtain a time series \( \rho_\delta(k) \) of local utilisation values. We denote the marginal distribution of this series by \( f_\delta(x) = Pr(\rho_\delta(k) = x) \) and the average value by \( E[\rho_\delta(k)] = \overline{\rho}_\delta \). The station measures \( \rho_\delta(k) \) by individually accounting the channel busy time due to each frame seen by the station monitor. We use a small base value of \( \delta \) which enables the calculation of statistics for larger values of interest via aggregation. The per bin measurement includes the SIFS and DIFS guard bands, ACKs frames, and the MAC and physical layer headers.

We begin by testing the basic idea in a simple setting where the entire time series is used. Figure 2 shows a sequence of maps where \( \overline{\rho}_\delta \) is increased from zero (top plot corresponding to PERR) to \( \overline{\rho}_\delta = 0.36 \) using \( \delta = 1 \) sec. Here a single Poisson UDP cross traffic stream from a second station is used in addition to the stream from the AP to the client station. The PER values monotonically increase with \( \overline{\rho}_\delta \) as expected, and dramatically so (note the logarithmic scale). The main observations is that even for very small \( \overline{\rho}_\delta \) values, the PPER measurement is significantly polluted by CPER.

Next, we use a single experiment and exploit burstiness to create maps indexed by \( \rho \). More specifically, we divide \( \rho \) into ranges and calculate maps corresponding to errors observed in the bins (these are spread over time) corresponding to them. To increase the probability that \( \rho_\delta(k) \) can be found low, we increase the CT burstiness by using a TCP instead of UDP. For space reasons we omit these maps, but our observations are as discouraging as those of Figure 2.

In a further experiment, we use an even richer cross traffic in-
volving several CT streams (described in Table 1). To determine more precisely how low $\rho(k)$ has to be to obtain accurate measures of PPER, and the probability of such values, we play with the timescale $\delta$. Figure 2 gives a sequence of maps corresponding to (from top to bottom) $\rho(k) \in [0, \rho^*]$, with $\rho^* = \{0, 0.1, 0.5, 0.01\}$ for respectively $\delta = \{1000, 1000, 100, 100\}$ [ms]. As we can see, there is no real improvement in the maps in either of the parameters $\rho^*$ and $\delta$. To understand in more detail the effect of this $\rho$ based filtering, Figure 3 gives histograms of the local per-bin $\rho$ values corresponding to the two top plots in Figure 2. They reveal that when filtering is applied (the $\rho^* = 0.1$ case) we succeed in isolating very effectively those bins with low $\rho$, as intended. It is remarkable that, despite this, the improvement in the maps is negligible. We speculate that this is due to the following subtle effects: (i) we do not measure PPER based on errored packets directly, but perform inference based on their retransmissions, which occur after the errors themselves. The intervals selected based on $\rho$ may therefore be translated in time from the ones which are relevant to the errors measured. (ii) we may succeed in excluding surrounding traffic by filtering on measured $\rho$, however this never includes those CT packets which are actually responsible for a collision based loss. Such packets are invisible to $\rho$, and moreover cannot be removed by selecting smaller $\rho^*$ or $\delta$. In extreme cases low utilization could be due to a small number of successful transmissions due to high numbers of these ‘invisible’ collisions, leading to the $\rho$ based selection choosing high contention instead of low contention periods!

In any event, our results indicate that only regions which are essentially entirely free of cross traffic can be used to measure PPER, and to identify them, one must track utilisation on a fine timescale. This implies a high computational load, and clearly becomes very difficult under high congestion, where the time scale required to measure the PPER would very likely be long enough for non-stationarity to enter in. On the bright side, the principle of the measurement is insensitive to non-stationarity as such, and it may be possible to remove some of the undesirable conditioning effects described above by further processing. However, overall this approach seems to suffer from too many serious problems.

### 6. PPER VIA FRAGMENTATION

It is possible to set a parameter to limit the size of MAC frames, causing the fragmentation of IP packets into two or more frames. For simplicity, let us consider the case where the maximum IP size is $F_m = (1500 - 28)/2 = 736$ bytes (780 bytes including head-

<table>
<thead>
<tr>
<th>Sta</th>
<th>IP type</th>
<th>l</th>
<th>Traffic</th>
<th>Pkt/s</th>
<th>Dirn</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>UDP</td>
<td>1500</td>
<td>54M</td>
<td>Gamma</td>
<td>300</td>
</tr>
<tr>
<td>2</td>
<td>UDP</td>
<td>1500</td>
<td>54M</td>
<td>Poisson</td>
<td>266</td>
</tr>
<tr>
<td>3</td>
<td>TCP</td>
<td>1500</td>
<td>54M</td>
<td>Poisson</td>
<td>366</td>
</tr>
<tr>
<td>4</td>
<td>TCP</td>
<td>1500</td>
<td>54M</td>
<td>Poisson</td>
<td>366</td>
</tr>
</tbody>
</table>

Table 1: Cross traffic parameters used in the experiment.

Figure 3: Histograms of local $\rho$ values in the range $\rho = [0, \rho^*]$ corresponding to the top two figures of Figure 2.

Figure 4: Fragmentation of an IP packet into two MAC fragments, separated by SIFS.

Figure 5: Fragment dependence of PER. For each rate, the 2nd fragment has a much lower PER than the 1st. Rates are cycled from 54 to 6 Mpbs.
Figure 7: An example of a current rate selection algorithm reducing rate as contention increases, despite constant PPER. Here several different experiments (indexed by colour) are superimposed.

has to be retransmitted one or more times, this does not invalidate the measurement as it contributes to the same event above. Thus, absolute measurements of PER for second fragments correspond directly to a measurement of PPER corresponding to the size of the second fragment.

The additional MAC header overhead for the second fragment will incur a throughput penalty which is more prominent at higher data rates. This throughput loss can be mitigated by fragmenting only the number of packets required for an adequate inference of physical error. Fragmentation at a station can be performed periodically and be adaptively configured.

Another issue is how to obtain measurements for different packet sizes in a convenient way. Theoretically a frame of size of 2313 bytes is permitted by the MAC, so a 1150 bytes second fragment is possible, however the IP layer does not generally allow a packet of size more than 1500 bytes.

Figure 5 shows the PER trace of the experiment having the cross traffic scenario specified in Table 1. The traffic of the station fragmenting the packets is Poisson and sends UDP packets towards the AP. The station rate is periodically cycled from 54 to 6 Mbps. There are three key observations:
(i) the PER for the second fragment is much lower than that of the first, as expected,
(ii) rate has an impact on PER, again as expected
(iii) results for each given rate are similar over time, confirming the stationarity of the environment (and therefore the true map based on PPER).

We now compare directly fragment specific maps obtained in experiments with richer CT behaviour. Specifically, we use 4 stations generating a mix of bursty UDP traffic (Gamma renewal process with a high coefficient of variation) and UDP Poisson traffic, and TCP traffics. The station using fragmentation is sending UDP Poisson packets while cycling the rates from 54 to 6 Mbps. Further details are given in table 1.

Figure 6 shows four maps superimposed from three experiments conducted back to back. Each map uses a different plotting symbol and colour (Note: here colour does not refer to RSSI, all experiments here are conducted in the same radio environment). The black circles on the left show the PPER measured in a mapping experiment (see section 4 using \( l = 1500 \) (large circle) and \( l = 780 \) byte packets. For the second experiment the CT is switched on. The map on the right (triangles) corresponds to the TPER map again with \( l = 1500 \) bytes. In the third experiment fragmentation is switched on, so that each \( l = 1500 \) byte packet is fragmented into frames 780 bytes long. The map on the right offset upwards (stars) is calculated back on the first fragments, and the diamonds on the left are for the second fragments.

The results for the first fragments are similar to those for all packets: highly polluted by CPER. The second fragments are much closer to the reference ones. Although they are still larger in most cases, this confirmed that this method has real potential to access PPER.

Figure 6: Fragment dependent maps. Experiment 1: circles show baseline PPER with \( l = 780, 1500 \) bytes. Experiment 2: with CT, triangles give TPER with \( l = 1500 \) bytes unfragmented packets. Experiment 3: packets fragmented. Maps based on (1st,2nd) fragments given by (stars, diamonds).
We do not fully understand as yet why the results are not identical to PPER. However, after conducting hundreds of experiments we have learnt that the conformance of cards, drivers, and Hardware Adaptation Layers (HALs) to the 802.11 standard is often less than perfect. This may reflect a simple lack of compliance, bugs, and/or a lack of testing under conditions of high congestion and/or packet fragmentation, which we focus on here.

Figure 7 shows an example of rate adaption at work, lowering rate as congestion is increased. Five separate experiments are given here where the number of CT stations (sends Poisson traffic) is increased. The thin full-height black lines indicate the actual data rates. The results for each experiment are offset for clarity and placed on the right side of the corresponding black line. In each case where there is CT, the total CT offered load is $\bar{\rho} = 0.6$, with rate fixed at $r = 54$ Mbps. The client station uses SampleRate, and offers 8Mbps ($l = 1500$ bytes). We clearly see that SampleRate recommends rate deduction, more and more so as the amount of contention increases (through having more stations). However, the PPER has not changed, and is consistent with the (black) PPER map of Figure 6, the rate reduction is therefore mistaken.

7. CONCLUSIONS AND FUTURE WORK

We have pointed out the multiple applications of an ability to clearly separate contention loss (CPER) from physical loss (PPER). Using concrete examples from a testbed, we investigated two methods for isolating and measuring PPER, and hence separating PPER and CPER, in practice, based on a per-packet conditional definition of loss closely related to PPER. The first is based on focusing error measurement on short periods where competing traffic is low. We found that cross traffic effectively ‘polluted’ physical error measurements even if present in trace amounts, resulting in measurement times which are likely to be excessive in most cases. Furthermore the computational cost of a utilisation based method may be high. The second method is based on packet fragmentation to exploit the contention avoidance mechanism built into the MAC for fragments. The method improves the PPER estimates dramatically, enough to have a significant impact, but there is still room for improvement. In future work, we will consider the cross layer impact of our approach on Transmit Power Control (TPC) algorithms, especially in congested scenarios.

8. REFERENCES


