Using the 802.11e EDCF to Achieve TCP Upload Fairness over WLAN Links

D.J. Leith, P. Clifford
Hamilton Institute, NUI Maynooth

Abstract

We investigate the use of the 802.11e MAC EDCF to address transport layer unfairness in WLANs. A simple solution is developed that uses the 801.11e $AIFS$ and CW_{min} parameters to ensure fairness between competing TCP uploads. An analytic model of TCP transport over the modified channel is developed in order to study the fairness properties of the proposed scheme. In addition to fairness between competing TCP flows, consideration is extended to include characteristics of TCP flows such as RTT unfairness and responsiveness and we observe that TCP flows with a wireless bottleneck link exhibit quite different properties from flows with a wired bottleneck.

I. Introduction

In recent years, 802.11 wireless LANs have become pervasive. While providing wire-free connectivity at low cost, it is widely recognised that the 802.11 MAC layer requires greater flexibility and the new 802.11e standard consequently allows tuning of MAC parameters that have previously been constant. While the 802.11e standard provides adjustable parameters within the MAC layer, the challenge is to use this flexibility to achieve enhanced network performance.

Existing work on 802.11e tuning algorithms is largely informed by the quality of service requirements of newer applications such as voice over IP. However, network traffic is currently dominated by data traffic (web, email, media downloads, etc.) carried via the TCP reliable transport protocol and this situation is likely to continue for some time. Although lacking the time critical aspect of voice traffic, data traffic server-client applications do place significant quality of service demands on the wireless channel. In particular, within the context of infrastructure WLANs in office and commercial environments there is a real requirement for efficient and reasonably fair sharing of the wireless capacity between competing data flows.

Unfortunately, cross-layer interactions between the 802.11 MAC and the flow/congestion control mechanisms employed by TCP typically lead to gross unfairness between competing flows, and indeed sustained lockout of flows. While the literature relating to WLAN fairness at the MAC layer is extensive, this issue of transport layer TCP fairness has received far less attention. Early work by Balakrishnan and Padmanabhan [1] studies the impact of path asymmetries in both wired and wireless networks, while more recently Detti et al.[2] and Pilosof et al.[3] have specifically considered TCP unfairness issues in 802.11 infrastructure WLANs and Wu et al. [4] study TCP in the context of single-hop 802.11 ad hoc WLAN's. With the exception of [4], all of these authors seek to work within the constraints of the basic 802.11 MAC and thus focus solely on approaches that avoid changes at the MAC layer. However, as we shall see, the roots of the problem lie in the MAC layer enforcement of per station fairness. Hence, it seems most natural to seek to resolve this issue at the MAC layer itself.

In this paper we investigate how we might use the flexibility provided by the new 802.11e MAC to resolve the transport layer unfairness in infrastructure WLANs. The focus in the present paper is on TCP uploads; consideration of mixed TCP uploads/downloads is the subject of ongoing work.

The paper is organised as follows. In section 2 we describe the transport layer unfairness problem in 802.11 WLANs. Section 3 discusses solutions to this problem that make use of new features provided by the 802.11e MAC. An analytic model of network fairness and throughput is developed in section 4. The characteristics of TCP flows using the modified wireless channel are investigated in more detail in section 5 and the conclusions summarised in section 6.

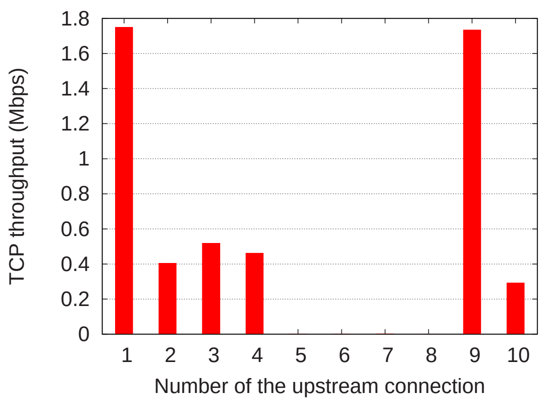


Fig. 1. Throughput of competing TCP uploads (NS simulation, 10 upload TCP flows, single cell infrastructure mode 802.11b WLAN, TCP SACK variant).

II. TCP Upload Unfairness Over 802.11 WLANs

Figure 1 illustrates the behaviour of competing TCP upload flows over an 802.11b WLAN. Gross unfairness between the throughput achieved by competing flows is evident. The source of this highly undesirable behaviour is rooted in the interaction between the MAC layer contention mechanism (that enforces fair access to the wireless channel) and the TCP transport layer flow and congestion control mechanisms (that ensure reliable transfer and match source send rates to network capacity).

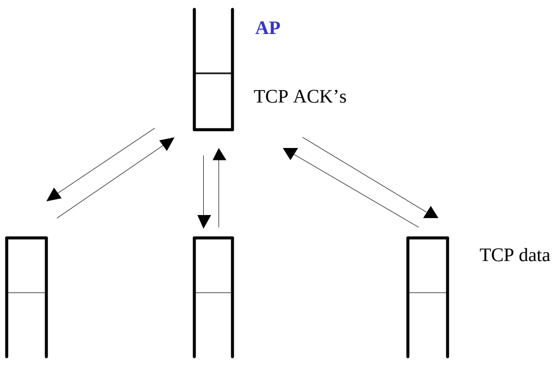


Fig. 2. TCP uploads. TCP data packets are queued at the wireless stations pending transmission to the access point. TCP ACK packets travel from a shared queue in the access point to the wireless stations.

At the transport layer, to achieve reliable data transfers TCP receivers return acknowledgement (ACK) packets to the data sender confirming safe arrival of data packets. During TCP uploads, the wireless stations queue data packets to be sent over the wireless channel to their destination and the returning TCP ACK packets are queued at the wireless access point (AP) to be sent back to the source station, see Figure 2. TCP's operation implicitly assumes that the forward (data) and reverse (ACK) paths between a source and destination have similar packet transmission rates. The basic 802.11 MAC layer, however, enforces station-level fair access to the wireless channel. That is, n stations competing for access to the wireless channel are each able to secure approximately a $1/n$ share of the total available transmission opportunities [2]. Hence, if we have n wireless stations and one AP, each station (including the AP) is able to gain only a $1/(n+1)$ share of transmission opportunities. By allocating an equal share of packet transmissions to each wireless node, with TCP uploads the 802.11 MAC allows $n/(n+1)$ of transmissions to be TCP data packets yet only $1/(n+1)$ (the AP's share of medium access) to be TCP ACK packets. For larger numbers of stations, n, this MAC layer action leads to substantial forward/reverse path asymmetry at the transport layer.

Asymmetry in the forward and reverse path packet transmission rate is a known source of poor TCP performance in wired networks, e.g. see [1]. If the reverse path ACK transmission rate is k times slower than the forward path data packet transmission rate, the reverse path is liable to become congested before the forward path causing TCP ACK packets to be dropped. On average, only one ACK will get through for every k data packets transmitted. This degrades performance in a number of ways. First, each ACK packet will on average acknowledge k data packets, thereby disrupting the ACK clocking within TCP and typically leading to increased burstiness in the rate at which the TCP sender transmits data packets. Second, infrequent ACKs can hamper congestion window growth at the TCP sender and hence interfere with the TCP congestion control algorithm that is seeking to match the TCP send rate to the available network capacity. Third, a pathological interaction with the TCP timeout mechanism is often created, which can be understood as follows.

A TCP sender probes for extra bandwidth until a data packet is lost or a timeout occurs. A timeout is invoked at a TCP sender when no progress is detected in the arrival of data packets at the destination - this may be due to data packet loss (no data packets arrive at the destination), TCP ACK packet loss (safe receipt of data packets is not reported back to the sender), or both. TCP flows with only a small number of packets in flight (e.g. flows which have recently started or which are recovering from a timeout) are much more susceptible to timeouts than flows with large numbers of packets in flight since the loss of a small number of data or ACK packets is then sufficient to induce a timeout.

Hence, on asymmetric paths where ACK losses are frequent a situation can easily occur where a newly started TCP flow loses the ACK packets associated with its first few data transmissions, inducing a timeout. The ACK packets associated with the data packets retransmitted following the timeout can also be lost, leading to further timeouts (with associated doubling of the retransmit timer) and so creating a persistent situation where the flow is completely starved for long periods; this is particularly prevalent in wireless networks, see for example Figure 1.

III. TCP ACK Prioritisation

Existing approaches to alleviating the gross unfairness between TCP upload flows over 802.11 WLANs work within the constraint of the current 802.11 MAC, resulting in relatively complex adaptive schemes requiring online measurements and, perhaps, per packet processing. We instead consider how the additional flexibility present in the new 802.11e MAC might be employed to alleviate transport layer unfairness.

The current 802.11 MAC defines a contention mechanism used by wireless stations to gain access to the wireless medium. Briefly, on detecting the wireless medium to be idle for a period $DIFS$, each station initializes a counter to a random number selected uniformly from the interval [0,CW-1]. Time is slotted and this counter is decremented each slot that the medium is idle. An important feature is that the countdown halts when the medium becomes busy and only resumes after the medium is idle again for a period $DIFS$. On the counter reaching zero, the station transmits a packet. If a collision occurs (two or more stations transmit simultaneously), CW is doubled and the process repeated. On a successful transmission, CW is reset to the value CW_{min} and a new countdown starts for the next packet. The new 802.11e MAC enables the values of $DIFS$ (called $AIFS$ in 802.11e) and CW_{min} to be set on a per class basis for each station i.e. traffic is directed to up to four different queues at each station, with each queue assigned different MAC parameter values. (Note that the 802.11e standard specifies further MAC parameters in addition to $AIFS$ and CW_{min} that may also be adjusted, but these are not considered here).

[

In this paper we argue for the combined use of the 802.11e $AIFS$ and CW_{min} parameters to restore path symmetry at the transport layer. Stations with a smaller CW_{min} will generally gain more transmission opportunities than stations with a larger value of CW_{min} as they have a shorter countdown procedure. To understand the influence of the $AIFS$ parameter recall that the MAC countdown halts when the wireless medium becomes busy and resumes after the medium is idle again for a period $AIFS$. In addition to the initial delay of $AIFS$ before countdown starts, a station accumulates an additional $AIFS$ delay for every packet sent on the medium by other stations, leading to a reduction in the number of transmission opportunities that can be gained by a station as $AIFS$ is increased. While the impact of this behaviour is generally complex, here we are interested specifically in a network configured such that the AP has small values of $AIFS$ and CW_{min} and other stations have standard or larger values. This prioritisation approach is straightforward and does not require online adaptation of the MAC parameters. With this configuration the AP effectively has unrestricted access to the wireless medium while the other stations divide the channel capacity not used by the AP fairly amongst themselves as per the standard 802.11 mechanism (this behaviour is analysed in detail in the next section). Rather than allowing unrestricted access to all traffic sent by the AP, recall that in 802.11e the MAC parameter settings are made on a per class basis. Hence, we propose collecting TCP ACKs into a single class (i.e. queue them together in a separate queue at the AP) and confine prioritisation to this class.

The rationale for this approach to differentiating the AP makes use of the transport layer behaviour. Namely, allowing TCP ACKs unrestricted access to the wireless channel does not lead to the channel being flooded. Instead, it ensures that the volume of TCP ACKs is regulated by the transport layer rather than the MAC layer. In this way the volume of TCP ACKs will be matched to the volume of TCP data packets, thereby restoring forward/reverse path symmetry at the transport layer. When the wireless hop is the bottleneck, data packets will be queued at wireless stations for transmission and packet drops will occur there, while TCP ACKs will pass freely with minimal queuing i.e. the standard TCP semantics are recovered. The performance of this scheme is illustrated in Figure 3 where it can be seen that fairness between TCP uploads is achieved.

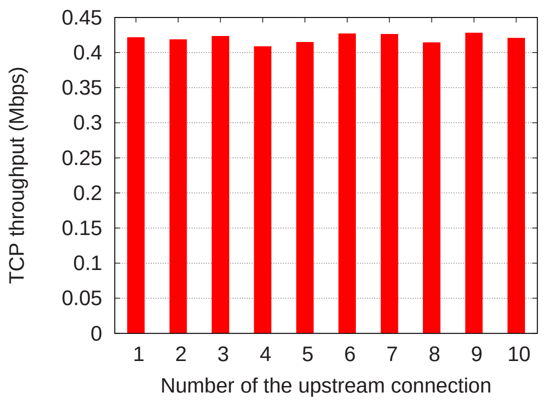


Fig. 3. Throughput of competing TCP uploads with 802.11e AP prioritisation (NS simulation, 10 upload TCP flows, TCP ACKs prioritised at AP with $AIFS = 50\mu s$ and $CW_{min} = 1$, wireless stations $AIFS = 90\mu s$ and $CW_{min} = 32$, single cell infrastructure mode WLAN, 11Mbs PHY).

IV. Analytic Modelling

Modelling of TCP over wireless is challenging due to the interactions between the TCP congestion control action, the interface queue dynamics and the MAC layer channel contention mechanism. While some initial analytic work has been reported relating to the pseudo-polling behaviour of TCP download traffic over 802.11 [8], to the authors' knowledge no models have been developed for TCP upload traffic (or for mixed upload/download situations). In this section we discuss how the symmetry created by prioritising TCP ACKs can be exploited to yield a tractable quantitative model of TCP uploads.

We proceed by first distinguishing between the n wireless stations that are TCP data senders and the wireless AP which transmits TCP ACKs. We initially make the following assumptions.

1) The wireless channel is the TCP bottleneck link and hence data packets are queued at the wireless stations.
2) The interface queues at the wireless stations are sized to be at least the delay-bandwidth product for the corresponding TCP path (i.e, in accordance with standard queue provisioning guidelines for data traffic).
3) The AP is prioritised using the 802.11e $AIFS$ and CW_{min}.
4) TCP timeouts can be neglected (we return to the validity of this assumption later).

It follows from Assumption 2 that the interface queues do not empty following backoff of the TCP congestion window $cwnd$ and so the n wireless stations are saturated (i.e. always have a packet to send). This greatly simplifies analysis as it obviates consideration of queuing dynamics and traffic arrival rates for these stations (although not for the AP). Similarly to Battiti and Li [6], we therefore model each wireless station by a triple of integers (i, k, d). The backoff stage, i, starts at 0 at the first attempt to transmit a packet and is increased by 1 every time a transmission attempt results in a collision, up to some maximum value m. It is reset after a successful transmission. The counter, k is initially chosen uniformly between $[0, W_i - 1]$, where $W_i = 2^i W$ is the range of the counter (where here we are following standard notation and denoting CW_{min} by W). Time is slotted and while the medium is idle the counter is decremented at each slot. When the medium is busy, the countdown is halted until the medium has been idle for a period of $AIFS_1$ time slots. Since the AP has a smaller $AIFS$ value, denoted $AIFS_0$, it will recommence its countdown a time $D = AIFS_1 - AIFS_0$ slots before the wireless stations. We model this behaviour using the parameter d, which counts off a sequence of hold states that the lower priority wireless stations occupy following a channel busy period. Transmission is attempted when $k = 0$. Using this model yields the following transition probabilities.

Before packet transmission,

$$\begin{aligned}
P(i,k,0|i,k+1,0) &= P_s, \\
P(i,k+1,1|i,k+1,0) &= 1-P_s, \\
P(i,k+1,d+1|i,k+1,d) &= P_{s1}, \quad d \in [1, D-1], \\
P(i,k,0|i,k+1,D) &= P_{s1}, \\
P(i,k+1,1|i,k+1,d) &= 1-P_{s1}, \quad d \in [1, D],
\end{aligned}$$

where P_s is the probability that no station transmits given that the considered station is not in a hold state (i.e. $d = 0$). P_{s1} is the probability that the AP does not transmit.

After packet transmission,

$$\begin{aligned}
P(i,0,1|i,0,0) &= 1, \\
P(i,0,d+1|i,0,d) &= P_{s1}, \quad d \in [1, D-1], \\
P(i,0,1|i,0,d) &= 1-P_{s1}, \quad d \in [1, D], \\
P(0,k,0|i,0,D) &= \frac{P_{s1}(1-p)}{W}, \\
P(i+1,k,0|i,0,D) &= \frac{P_{s1}p}{2^{i+1}W}, \\
P(m,k,0|m,0,D) &= \frac{P_{s1}p}{2^m W},
\end{aligned}$$

where p is the packet collision probability for the wireless stations (i.e. the TCP data sources).

After some manipulation, we obtain the following expression for the per station per slot transmission probability (conditioned on the station not being in a hold state) τ_2 of the wireless stations

$$\tau_2 = \frac{2(1-2p)}{(1-2p)(W+1)+pW(1-(2p)^m)}. \tag{1}$$

Letting τ_1 denote the per slot transmission probability of the AP, we have

$$\begin{aligned}
p &= 1-(1-\tau_2)^{n-1}, & (2)\\
P_{s1} &= (1-\tau_1), & (3)\\
P_s &= (1-\tau_1)(1-\tau_2)^{n-1}. & (4)
\end{aligned}$$

The probability, P_{hold} of a station being in a hold state is

$$P_{hold} = 1 - \sum_{i=0}^{m} \sum_{k=0}^{W-1} \frac{(2^i W - k)p^i}{2^i W} b(0,0,0), \tag{5}$$

where b(0,0,0) is the probability of a station being in state (0,0,0).

Remark: The expressions we obtain for the collision probability p and transmission probability τ_2 are identical to those obtained by Bianchi [9] for a saturated 802.11 network with the standard 802.11 MAC without 802.11e extensions. The difference from Bianchi lies in the presence of the hold states and the associated probability P_{hold} which is not present in his model. While P_{hold} does not directly enter the expressions for τ_2 and p, τ_2 is conditioned on not being in a hold state and as we will see below P_{hold} plays a central role in determining the AP transmission probability and station throughputs.

Define $Q(0,0)$ to be the probability that there are no stations transmitting within a randomly selected slot, $Q(1,0)$ the probability that only the AP is transmitting and $Q(0,1)$ the probability that a single wireless station is transmitting and the AP is silent. We have that

$$\begin{aligned} Q(0,0) &= (1-\tau_1)(P_{hold}+(1-P_{hold})(1-\tau_2)^n), && (6)\\ Q(1,0) &= \tau_1, && (7)\\ Q(0,1) &= (1-\tau_1)(1-P_{hold})n\tau_2(1-\tau_2)^{n-1}. && (8)\end{aligned}$$

In order to complete the model, it remains to establish the per slot transmission probability τ_1 of the AP. The AP traffic is not saturated. However, we can proceed by exploiting the symmetry in the network i.e. we make use of the fact that number of TCP ACK packets transmitted is on average equal to the number of data packets successfully transmitted (or half that number if delayed acking is used). This assumption is equivalent to the assertion $Q(1,0) = Q(0,1)$ i.e.

$$\tau_1 = (1-\tau_1)(1-P_{hold})n\tau_2(1-\tau_2)^{n-1}. \qquad (9)$$

Solving equations (1)-(9) yields predictions for the transmission probabilities τ_1 and τ_2, hold probability P_{hold} and collision probability p (note that no collisions are possible between TCP ACK packets since the AP is the only node that transmits TCP ACKs).

Finally the TCP data throughput is given by

$$S_{TCP} = \frac{Q(0,1)E}{Q(0,0)\times\sigma + Q(1,0)T_{s_2} + Q(0,1)T_{s_1} + [1-Q(0,0)-Q(0,1)-Q(1,0)]T_c}.$$

where E is the time spent transmitting TCP payload data, σ is the time slot duration, T_{s_1} is the time taken for a successful data transmission, T_{s_2} the time taken for a successful TCP ACK transmission and T_c is the time taken by a packet collision. Note that the denominator of this fraction is the expected duration of a state in the Markov chain in real-time. In more detail, in the basic scheme without RTS/CTS we have that

$$\begin{aligned} T_{s_1} &= PHY_{hdr} + MAC_{hdr} + E + SIFS + \delta + ACK + DIFS + \delta,\\ T_{s_2} &= PHY_{hdr} + MAC_{hdr} + TCP\_ACK + SIFS + \delta + ACK + DIFS + \delta,\\ T_c &= PHY_{hdr} + MAC_{hdr} + E + ACK\_Timeout + DIFS,\end{aligned}$$

where PHY_{hdr} denotes the PHY header duration (synchronisation preamble plus PLCP), MAC_{hdr} denotes the overhead due to MAC frame encapsulation (MAC header, frame CRC), ACK is the duration of an 802.11 ACK (PHY_{hdr} plus ACK frame), δ is the propagation delay and TCP_ACK is the time taken to transmit a TCP ACK packet.

We note that when the propagation delay of the wired component of the TCP paths is small, TCP ACKs are ready for transmission at the AP shortly after the corresponding TCP data packet is received by the AP, creating a strong correlation in time between these events. In this case the hold state modelling in the foregoing model can be considerably simplified. Essentially, we can use the standard Bianchi model [9] and simply replace the time spent to send a TCP data packet by the time spent to send the data packet and receive the TCP ACK. This yields the following expression for TCP throughput

$$S = \frac{P_s P_{tr} E}{(1-P_{tr})\sigma + P_{tr}P_sT_s + P_{tr}(1-P_s)T_c},$$

where $P_{tr} = 1-(1-\tau_2)^n$, $P_s = n\tau_2(1-\tau_2)^{n-1}/P_{tr}$, τ_2 is as before and

$$T_s = 2\times[PHY_{hdr} + MAC_{hdr} + SIFS + 2\delta + ACK + DIFS] + E + TCP\_ACK.$$

The behaviour of the full and simplified models is illustrated in Figures 4 and 5. We observe excellent agreement between analysis and simulation except when the collision probability is high (greater than about 0.3, corresponding to more than 30% of packet transmissions failing due to collisions). When the collision probability is high, multiple TCP backoff and timeout events become frequent, violating the assumptions on which our model is based. However, it can be seen from the figures that such high collision probabilities are associated with CW_{min} values less than the 802.11 standard value of 32, and so are of little relevance in the present context.

Remark: Although not investigated further in the present paper, we note from Figures 4 and 5 that the throughput efficiency of the wireless channel is dependent on the number of upload stations and can evidently be optimised by tuning CW_{min}.

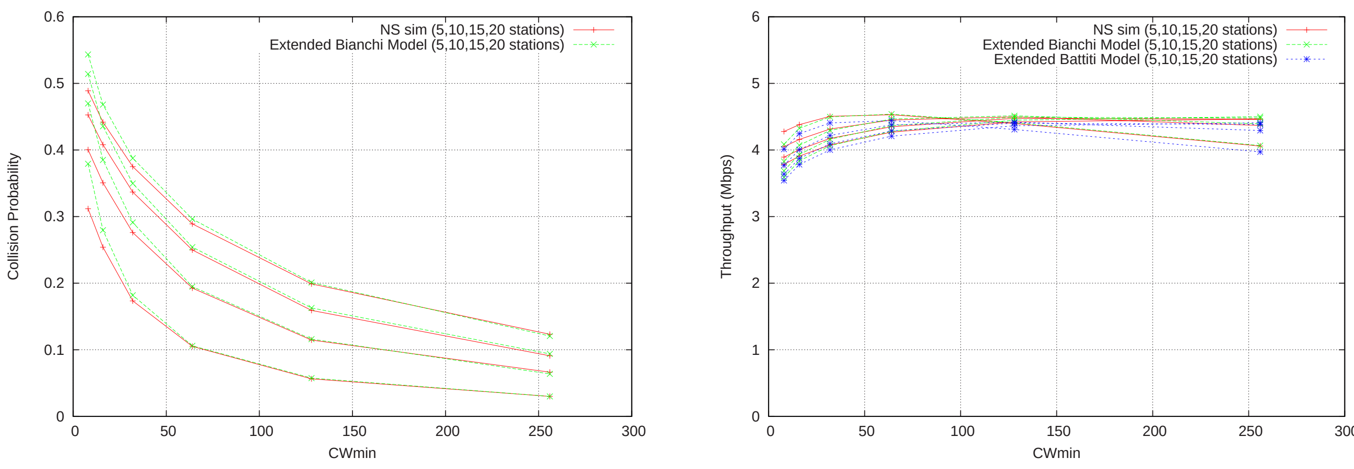


Fig. 4. 802.11e theory versus simulation, varying numbers of upload flows, topology as in Figure 2, delayed acking **not** used in TCP sink (which is in the access point).

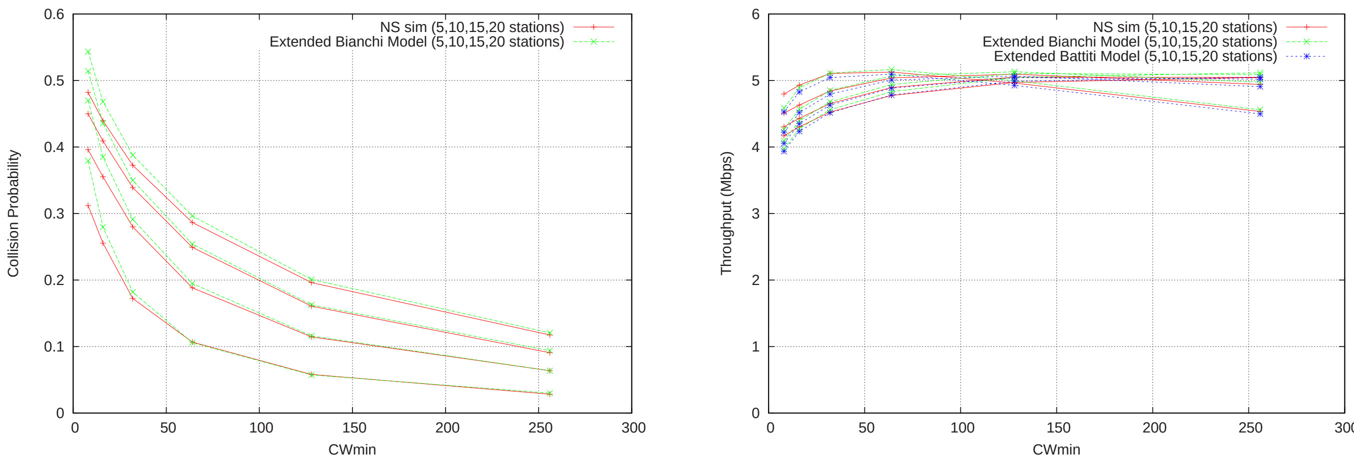


Fig. 5. 802.11e theory versus simulation, varying numbers of upload flows, topology as in Figure 2, delayed acking used in TCP sink (which is in the access point).

V. Characterising TCP Over Wireless Links

Much of the existing 802.11 literature has focussed on the fairness properties of TCP flows sharing a wireless hop, understandably so in view of the potentially catastrophic nature (sustained lockout, etc.) of the TCP unfairness in 802.11 WLANs. Once a degree of fairness is restored in WLANs, it becomes important to investigate other key TCP characteristics. Aspects of TCP behaviour that are known to be of importance and interest in wired networks include round-trip time unfairness and responsiveness/convergence rates.

In the rest of this section we will use the topology shown in Figure 6. By varying the bandwidth B of the wired link, the bottleneck in the network can be varied between the wired and wireless hops. An 802.11b PHY is used, in which case the wired link acts as the bottleneck when its bandwidth B is less than about 5Mbs, whereas the wireless hop acts as the bottleneck for higher values of B.

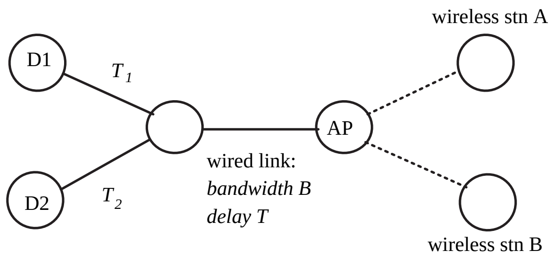


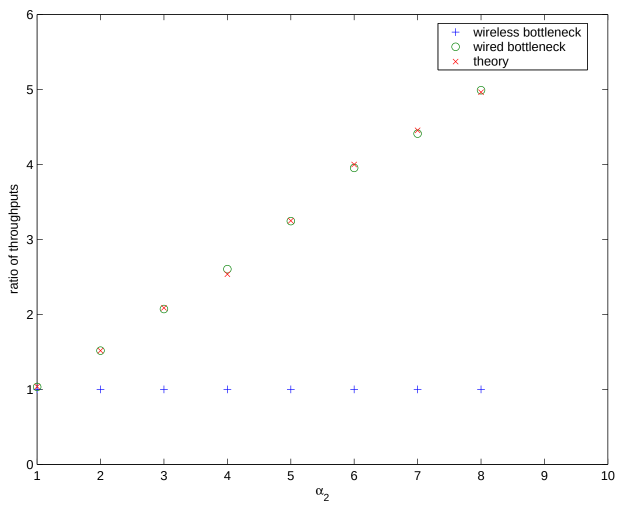
Fig. 6. Network topology used to compare characteristics of TCP uploads with wired/wireless bottleneck link. (Network parameters: $T = 10ms$, $T_1 = 0ms$, $T_2 = 40ms$, wired link queue $0.0001B$, 802.11b PHY, TCP uploads from stn A to D1 and from stn B to D2.)

A. RTT Unfairness

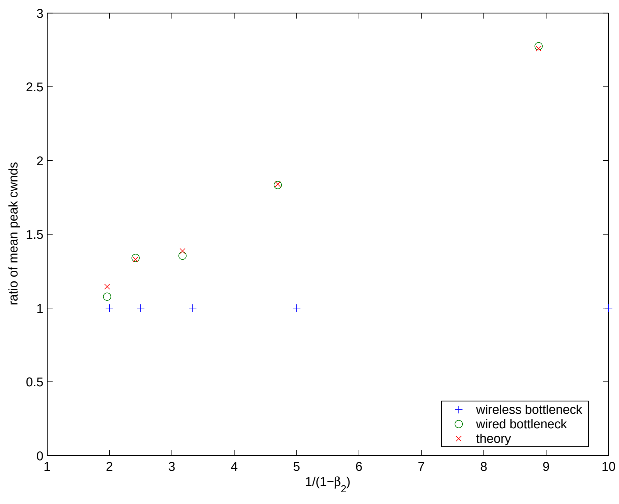
In wired networks, it is known that the sharing of bandwidth between competing TCP flows depends on each flow's round-trip time (RTT). For long-lived flows it has been shown [10] that the mean peak TCP congestion window, $cwnd_i$, of the i'th flow is proportional to $\alpha_i/\lambda_i(1-\beta_i)$, where α_i is the TCP additive-increase parameter (approximately $1/RTT_i$), β_i is the TCP multiplicative decrease parameter (0.5 in standard TCP) and λ_i is the probability of flow i detecting a loss event when the bottleneck queue overflows[1]. With $B = 4Mbs$ so that the wired link is the bottleneck, the impact of varying α and β can be seen in Figure 7 together with the corresponding analytic predictions.

We are interested in the corresponding behaviour when the wireless link is the bottleneck. A key difference between the wired and wireless situations is that for TCP uploads over a 802.11 wireless channel the TCP data packets are queued separately at each wireless station (see Figure 2), whereas in a wired bottleneck link flows compete via a shared queue. Using the scheme proposed in Section III, and assuming appropriately sized interface queues, access to the wireless channel is regulated by the ability of the wireless stations to secure transmission opportunities for their data packets. The MAC enforces fair per station access independent of the AIMD parameters of the competing TCP flows and hence it can be expected that the bandwidth share achieved by TCP uploads is invariant with respect to α and β. This behaviour is confirmed by simulation results, see Figure 7. An immediate consequence is that wireless uploads do not suffer from the RTT unfairness that is ubiquitous in wired TCP networks. To demonstrate this behaviour, Figure 8(a) shows simulation results as the bandwidth, B, of the wired bottleneck is varied. When the wired link bandwidth is low, the wired link acts as the bottleneck and unfairness exists between the competing TCP flows as a result of their different round-trip times. When the wired link bandwidth is increased, thereby shifting the bottleneck to the wireless link, this unfairness disappears. It can be seen that the transition between these regimes is quite abrupt, as might be expected. Further confirmation of the insensitivity of fairness to RTT when the bottleneck link is the wireless hop is provided in Figure 8(b).

[1]Hence, other things being equal (in particular the loss event probabilities), the ratio $cwnd_i/cwnd_j$ of mean peak congestion windows is given by the inverse ratio of the flow RTT's, RTT_j/RTT_i. Since the throughput of flow i is approximately $cwnd_i/RTT_i$, we have that the corresponding ratio of flow throughputs is approximately $(RTT_j/RTT_i)^2$

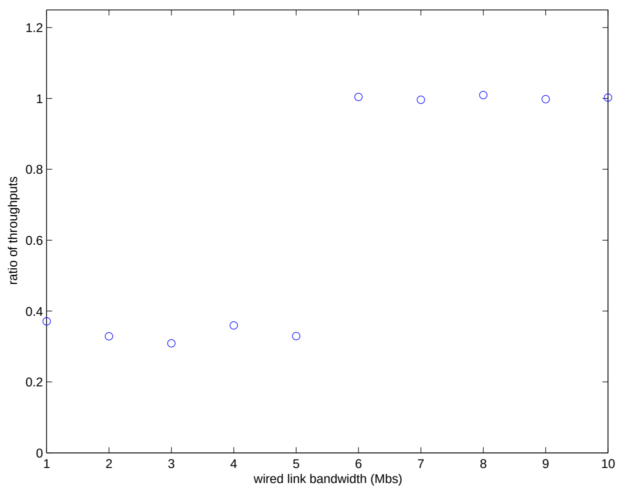


(a) varying α_2

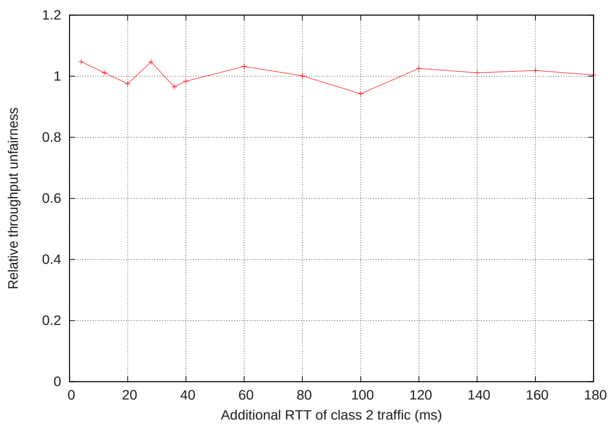


(b) varying β_2

Fig. 7. Impact on fairness of AIMD parameters α and β and bottleneck link location. (NS simulations, topology as in Figure 6, bandwidth B=4Mbs yields wired bottleneck, B=10Mbs yields wireless bottleneck, AIMD parameters have the standard TCP values unless otherwise stated).



(a) varying wired link bandwidth B (wired link is network bottleneck when B is less than around 5Mbs and wireless link is bottleneck for larger values).



(b) varying RTT with wireless bottleneck (bandwidth B=10Mbs)

Fig. 8. Impact on fairness of bottleneck link location and RTT. (NS simulation, topology as in Figure 6).

B. Convergence Rate

The convergence rate, or responsiveness, of a network of TCP flows is a measure of the time that the network takes to reach steady state following start-up of a new flow or other such disturbance. In wired networks the data packets for all flows share a common bottleneck queue, with packet drops largely arising from the aggregate action of the competing TCP flows. Hence, in studying convergence rates in wired networks it is necessary to consider the network as a whole. Using such approaches, it is known [10], [11] that the convergence rate measured in congestion epochs is determined by the AIMD backoff factors β_i of the competing TCP flows, with the convergence time increasing exponentially as β is increased. When the backoff factors are all 0.5 (the situation with standard TCP) the

95% convergence time is 4 congestion epochs [10], [11]; see, for example, Figure 9. The duration of the congestion epochs is dependent on the AIMD increase parameters α_i. In general, TCP flows need not experience synchronised packet drops unlike in this example. However, the previous convergence results still hold provided that we work in terms of ensemble averages. For example, for the topology in Figure 6 and a wired bottleneck, Figure 10 shows ensemble average time histories of the TCP flow congestion windows following the startup of a second flow (the '+' symbols mark the end of each congestion epoch and indicate the average congestion window at that time). The impact of the AIMD backoff factor β on the convergence time measured in congestion epochs is evident.

As noted previously, a key difference between the wired and wireless situations is that for TCP uploads over a 802.11 wireless channel the TCP data packets are queued separately at each wireless station, whereas in a wired bottleneck link flows compete via a shared queue. In the wireless case dropping of TCP data packets (and associated backoff of the TCP send rate) only occurs due to either (i) the queue at a wireless station overflowing (and this can occur solely from the action of TCP flows originating at that station) or (ii) repeated collisions or corrupted packets on the wireless channel. The latter source of drops is generally less important than the former, provided the MAC CW_{min} parameters are not too small. Hence, on startup a new TCP flow will typically not experience any data packet drops until its probing action has led to the interface queue at its own station filling. In wireless networks convergence following startup of a new flow is therefore largely *independent* of the aggregate action of the network of TCP flows and in this respect is fundamentally different from a wired network. Figure 11 illustrates the convergence in a wireless network following the startup of a second TCP upload flow. It can be seen that, in contrast to the wired case (see Figure 10), the new flow increases its congestion window monotonically and experiences no packet drops until its steady state value is reached. The latter is determined by the interface queue size and the delay-bandwidth product of the path. An immediate consequence of this behaviour is that convergence time measured in congestion epochs in the wireless case is largely insensitive to the AIMD backoff parameter β, see Figure 11.

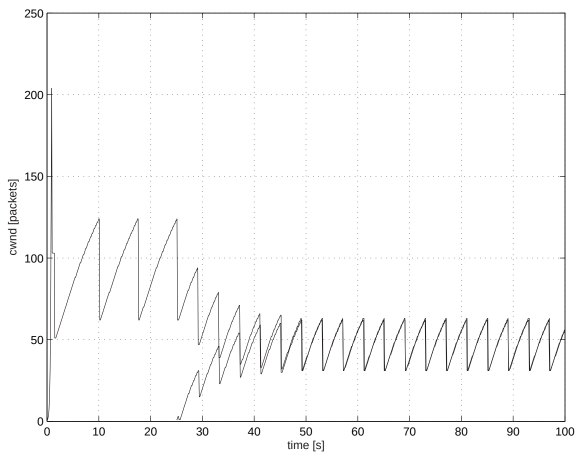
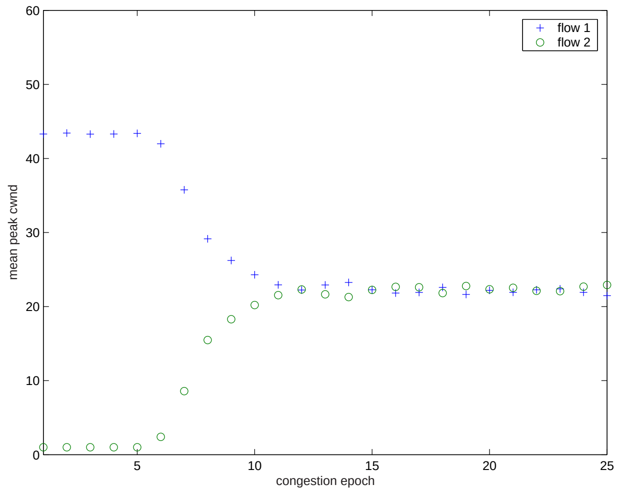
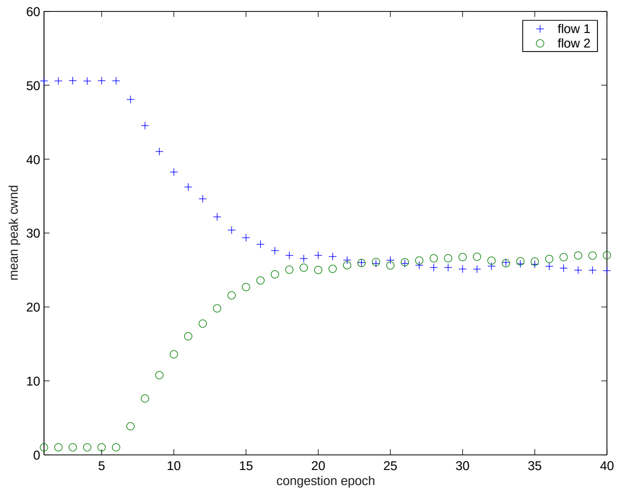


Fig. 9. Two sources competing for bandwidth with $\alpha_i = 1$, $\beta_i = 0.5$. 95% convergence is achieved in 4 congestion epochs. (NS simulation results, network parameters: 10Mb bottleneck link, 100ms delay, queue 40 packets)

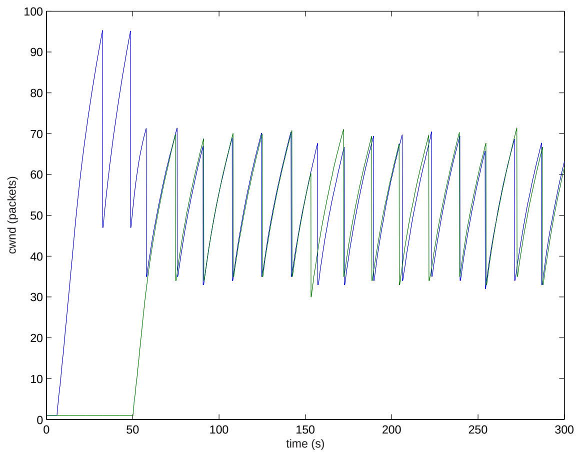


(a) β_2=0.5

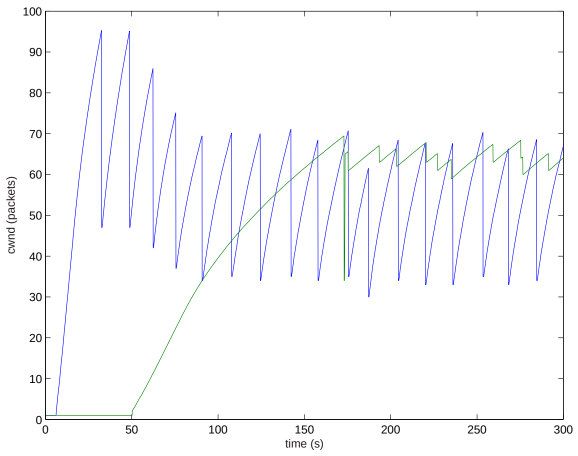


(b) β_2=0.95, α_2=0.2

Fig. 10. Convergence of competing TCP flows with a wired bottleneck link. (Topology is as in Figure 6 with B=4Mbs, results are ensemble averages over 50 runs; the + and o symbols mark the end of each congestion epoch and indicate the average congestion window at that time).



(a) $\alpha_2 = 1$, β_2=0.5



(b) β_2=0.95, α_2=0.2

Fig. 11. Convergence of TCP congestion windows of two TCP upload flows as AIMD α and β parameters of second flow are varied (NS simulations, topology is as in Figure 6 with B=10Mbs yielding wireless bottleneck).

VI. Conclusions

In this paper we investigate how we might use the flexibility provided by the new 802.11e MAC to resolve the transport layer unfairness in WLANs. A simple solution is developed that uses the 802.11e $AIFS$ and CW_{min} parameters to ensure fairness between competing TCP uploads. The computational burden of the proposed approach is very low (no online adaptation is required).

An analytic model of TCP transport over the modified channel is developed in order to study the fairness properties of the proposed scheme. TCP traffic modelling is difficult in general since traffic is bursty and flow depends on interaction between queue, transport layer and MAC dynamics. However, the decoupling action of the proposed prioritisation scheme is shown to greatly simplify the modelling task.

In addition to fairness between competing TCP flows, consideration is extended to other characteristics of TCP flows such as RTT unfairness and responsiveness. We observe that TCP flows with a wireless bottleneck link exhibit quite different properties from flows with a wired bottleneck. For example, RTT unfairness is absent in wireless networks and convergence rates are insensitive to the AIMD backoff parameter in TCP.

Future work includes the consideration of mixed upload/download traffic and mixed TCP/UDP traffic.

VII. Acknowledgements

This work was supported by Science Foundation Ireland grant 03/IN3/I396.

References

[1] H. Balakrishnan, V. Padmanabhan, "How Network Asymmetry Affects TCP". IEEE Communications Magazine, April 2001, pp60-67.
[2] A. Detti, E. Graziosi, V. Minichiello, S. Salsano and V. Sangregorio, "TCP fairness issues in IEEE 802.11 based access networks", submitted paper.
[3] S. Pilosof, R. Ramjee, Y. Shavitt, P. Sinha, "Understanding TCP fairness over Wireless LAN", INFOCOM 2003, 1-3 April 2003, San Francisco, USA.
[4] H. Wu, Y. Peng, K. Long, S. Cheng, J. Ma, "Performance of Reliable Transport Protocol over IEEE 802.11 Wireless LAN: Analysis and Enhancement", INFOCOM 2002, 23-27 June 2002, New York, USA.
[5] The Network Simulator - ns-2, http://www.isi.edu/nsnam/ns/
[6] R. Battiti, B. Li, "Supporting service differentiation with enhancements of the IEEE 802.11 MAC protocol: models and analysis", QOFIS 2003, 1-3 October 2003, Stockholm, Sweden .
[7] Sven Wiethlter, Christian Hoene, An IEEE 802.11e EDCF and CFB Simulation Model for ns-2, http://www.tkn.tu-berlin.de/research/802.11e_ns2/, Technical Report TKN-03-019, Telecommunication Networks Group, Technische Universitt Berlin, November 2003.
[8] R. Bruno, M. Conti, E. Gregori, "Throughput Analysis of TCP Clients in Wi-Fi Hot Spot Networks", WONS 2004, 21-23 January, Trento, Italy.
[9] G. Bianchi, "Performance analysis of IEEE 802.11 distributed coordination function", *IEEE Journal on Selected Areas in Communications*, 18(3):535-547, March 2000.
[10] R. Shorten, F. Wirth, D. Leith, "A Positive Systems Model of TCP-like Congestion Control: Asymptotic Results". Hamilton Institute Technical Report 2004-1, April 2004.
[11] R. Shorten, D. Leith, "Analysis and Design of Congestion Control in Synchronised Networks". Automatica, in press.