TCP Fairness in 802.11e WLANs

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 $AIFS$, $TXOP$ and CW_{min} parameters to ensure fairness between competing TCP uploads and downloads.**

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 paper considers TCP uploads and downloads, and mixtures of both.

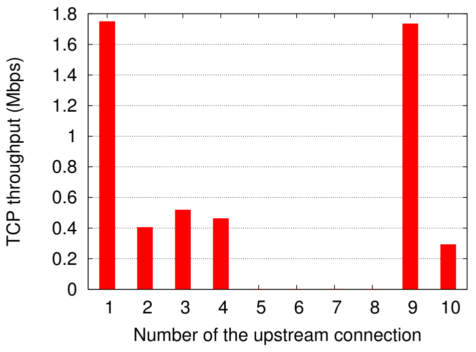


Fig. 1. Throughput of competing TCP uploads (NS simulation, 10 upload TCP flows, infrastructure mode 802.11b WLAN, TCP SACK).

II. TCP Unfairness Over 802.11 WLANs

We consider, in turn, unfairness between competing TCP upload flows and between competing upload and download flows in 802.11 WLAN's.

A. *Unfairness between competing TCP upload flows*

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).

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. 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.

B. Unfairness between competing TCP upload and download flows

In 802.11b there is asymmetry between the upload data packet flow and the returning flow of TCP ACKs as discussed above. Asymmetry also exists between competing upload and download TCP flows that can create significant unfairness. This is illustrated for example in Figure 2 where it can be seen that upload flows achieve nearly two orders of magnitude greater throughput than competing download flows.

To understand this behaviour, consider the situation where we have only TCP downloads. Download data packets are transmitted by the AP and on receiving a data packet a wireless station generates a TCP ACK (we ignore delayed acking for the moment to streamline the present discussion). Importantly, wireless stations only generate TCP ACK packets on receipt of a TCP data packet and otherwise do not contend for medium access. Consequently, TCP downloads typically exhibit a quasi-polled behaviour. Namely, the AP transmits a data packet to a wireless station which then responds with a TCP ACK while the other wireless stations remain silent. Hence, regardless of the number of TCP download flows, generally only *two* stations (the AP and the most recent destination wireless station) contend for medium access at any time. This behaviour has also been noted by [7].

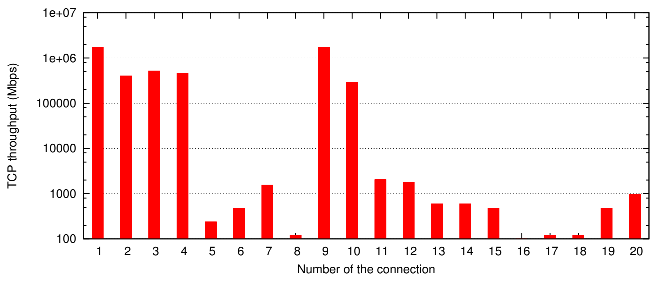


Fig. 2. Throughput of competing TCP uploads and downloads. Flows 1-10 are uploads, flows 11-20 are downloads (NS simulation, 10 upload TCP flows, 10 download TCP flows, infrastructure mode 802.11b WLAN, TCP SACK).

Considering now a mix of competing upload and download TCP flows, suppose we have n_u upload flows and n_d download flows. Owing to their quasi-polling behaviour, we have that the download flows (regardless of the number n_d of download flows) gain transmission opportunities at the roughly same rate as a *single* TCP upload flow. That is, roughly $1/(n_u + 1)$ of the channel bandwidth is allocated to the download flows and $n_u/(n_u + 1)$ allocated to the uploads. As the number n_u of upload flows increases, gross unfairness between uploads and downloads can result.

III. RESTORING FAIRNESS BETWEEN TCP UPLOADS

Existing approaches to alleviating the gross unfairness between TCP flows competing over 802.11 WLANs work within the constraint of the current 802.11 MAC, resulting in 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. We initially consider the case where we only have competing TCP upload flows.

A. TCP ACK Prioritisation

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. The maximum duration that a station may transmit on the wireless medium is specified by the $TXOP$ parameter; packet bursting (consecutive transmission of several packets during a single transmission opportunity) is permitted provided the overall duration remains below the value specified by $TXOP$. 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), CW_{min} and $TXOP$ 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.

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 its $AIFS$ 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.

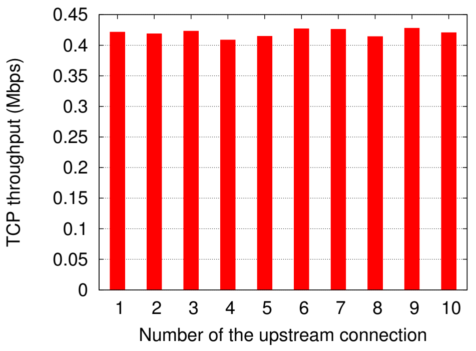


Fig. 3. Throughput of competing TCP uploads with 802.11e AP prioritisation (NS simulation, 10 upload TCP flows, TCP ACKs prioritised at AP with $AIFS = 50\mu s$ and $CW_{min} = 2$, wireless stations $AIFS = 90\mu s$ and $CW_{min} = 32$, infrastructure mode WLAN, 11Mbs PHY).

B. *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. 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.

(i) The wireless channel is the TCP bottleneck link and hence data packets are queued at the wireless stations.
(ii) 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).
(iii) The AP is prioritised using the 802.11e $AIFS$ and CW_{min}.
(iv) TCP timeouts can be neglected (we return to the validity of this assumption later).

It follows from Assumption III-B 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,

$$\begin{aligned} P(i,k,0|i,k+1,0) &= P_s, & (1)\\ P(i,k+1,1|i,k+1,0) &= 1-P_s, & (2)\\ P(i,k+1,d+1|i,k+1,d) &= P_{s1}, \quad d\in[1,D-1] & (3)\\ P(i,k,0|i,k+1,D) &= P_{s1}, & (4)\\ P(i,k+1,1|i,k+1,d) &= 1-P_{s1}, \quad d\in[1,D] & (5)\end{aligned}$$

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,

$$\begin{aligned} P(i,0,1|i,0,0) &= 1, & (6)\\ P(i,0,d+1|i,0,d) &= P_{s1}, \quad d\in[1,D-1], & (7)\\ P(i,0,1|i,0,d) &= 1-P_{s1}, \quad d\in[1,D], & (8)\\ P(0,k,0|i,0,D) &= \frac{P_{s1}(1-p)}{W}, & (9)\\ P(i+1,k,0|i,0,D) &= \frac{P_{s1}p}{2^{i+1}W}, & (10)\\ P(m,k,0|m,0,D) &= \frac{P_{s1}p}{2^m W} & (11)\end{aligned}$$

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

$$\tau_2 = \frac{2(1-2p)}{(1-2p)(W+1)+pW(1-(2p)^m)}. \quad (12)$$

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

$$\begin{aligned} p &= 1-(1-\tau_2)^{n-1}, & (13)\\ P_{s1} &= (1-\tau_1), & (14)\\ P_s &= (1-\tau_1)(1-\tau_2)^{n-1}. & (15)\end{aligned}$$

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

$$P_{hold} = 1 - \sum_{i=0}^{m}\sum_{k=0}^{W-1} \frac{(2^iW-k)p^i}{2^iW} b(0,0,0), \quad (16)$$

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

Remark: While P_{hold} does not directly enter the expressions for τ_2 and p (which are identical to those for the Bianchi model), 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

$$\begin{aligned} Q(0,0) &= (1-\tau_1)(P_{hold}+(1-P_{hold})(1-\tau_2)^n) & (17)\\ Q(1,0) &= \tau_1 & (18)\\ Q(0,1) &= (1-\tau_1)(1-P_{hold})n\tau_2(1-\tau_2)^{n-1} & (19)\end{aligned}$$

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.

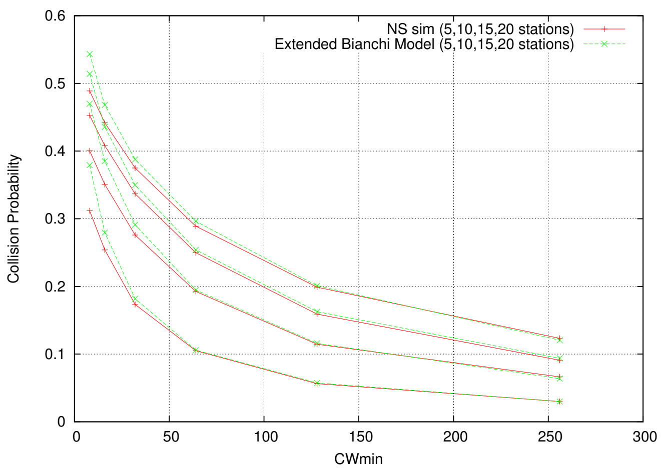
$$\tau_1 = (1-\tau_1)(1-P_{hold})n\tau_2(1-\tau_2)^{n-1} \quad (20)$$

Solving equations (12)-(20) 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

$$S_{TCP} = \frac{Q(0,1)E}{Q(0,0)\sigma + Q(1,0)T_{s_2} + Q(0,1)T_{s_1} + Q_cT_c}$$

where $Q_c = 1 - Q(0,0) - Q(0,1) - Q(1,0)$, 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. 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 [8] 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. The behaviour of the full and simplified models is illustrated in Figure 4. 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.



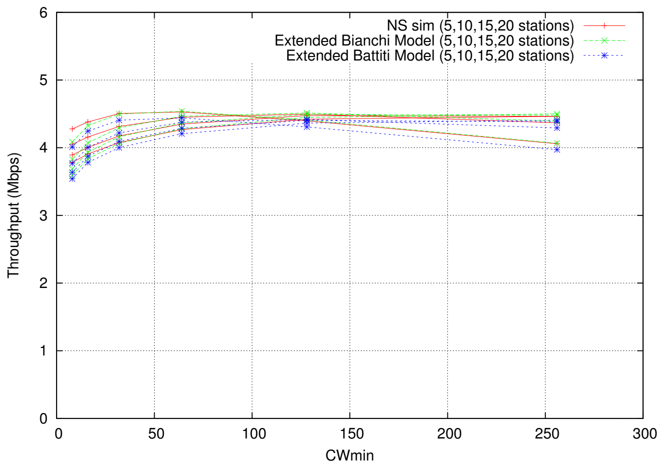


Fig. 4. 802.11e theory versus NS simulation for varying numbers of upload flows, 11Mbs PHY.

IV. Achieving fairness between TCP upload and download flows

We consider now the case of competing TCP upload and download flows. Recall that the primary source of unfairness arises from the quasi-polling behaviour of TCP downloads which means that if we have n_u uploads and n_d downloads then the download flows roughly win only a $1/(n_u+1)$ share of the available transmission opportunities. This suggests that to restore fairness we need to prioritise the download data packets at the AP so as to achieve an $n_d/(n_u+n_d)$ share.

While we might prioritise download data packets by using an appropriate value of CW_{min} at the AP for TCP data packets, the utility of CW_{min} is constrained by the availability of only a coarse granularity (CW_{min} can only be varied by powers of two in 802.11e). The $AIFS$ parameter might also be used, but seems better suited to strict prioritisation rather than proportional prioritisation. Instead, we propose that the $TXOP$ packet bursting mechanism in 802.11e provides a straightforward and fine grained mechanism for prioritising TCP download data packets. Since the download TCP data traffic gains a $1/(n_u+1)$ share of transmission opportunities, by transmitting n_d packets (one packet to each of the n_d download destination stations) at each transmission opportunity it can be immediately seen that we restore the $n_d/(n_u+n_d)$ fair share to the TCP download traffic.

Specifically, we queue TCP data packets in a separate traffic class at the AP. By inspecting this queue we can determine both the current number n_d of distinct destination stations and the first packet due to be transmitted to each destination. This information changes in real-time but can readily determined solely by inspection of the AP interface queue, with no requirement for monitoring of the wireless medium activity itself. When the traffic class wins a transmission opportunity, we use a $TXOP$ value of n_d packets and transmit one packet to each of the destination stations.

The effect is to dynamically track the number of active TCP download stations and always ensure the appropriate prioritisation of TCP download traffic. Hence, this approach accommodates both bursty, short-lived traffic such as HTTP and long-lived traffic such as FTP in a straightforward and consistent manner (see later for examples).

Remark: With this $TXOP$ approach the AP transmits n_d packets in a single burst. For n_d large, this can result in the AP occupying the channel for a substantial consolidated period of time and this may, for example, negatively impact competing delay sensitive traffic. We can address this issue in a straightforward manner by using multiple smaller bursts instead of a single burst. When using smaller packet bursts, it is of course necessary to ensure a corresponding increase in the number of transmission opportunities won by the AP. This can be achieved by using a smaller value of CW_{min} for the TCP data packet traffic class at the AP. It is shown in [6] that competing traffic classes gain transmission opportunities approximately in inverse proportion to their values of CW_{min}. Let k denote the ratio of the wireless station TCP data class CW_{min} value to that of the AP TCP data class. Scaling k with the number of transmission opportunities required provides coarse (recall that in 802.11e k is constrained to be a power of two) prioritisation of download TCP flows. We then complement this with use of $TXOP$ for fine grained adjustment of the packet burst lengths, scaling $TXOP$ with $1/k$. Hence fine grained prioritisation can be achieved while avoiding unduly large packet bursts.

In addition to prioritisation of download data packets at the AP, in line with the discussion in Section III-A it is also necessary prioritise the TCP download ACKs using $AIFS$ to mitigate queueing and loss of TCP ACKs. While in the case of TCP uploads the TCP ACKs are queued only at the AP and hence there is no contention (i.e no collisions) between the TCP ACKs of competing TCP flows in accessing the wireless channel, with TCP downloads the TCP ACK packets are queued at the wireless stations and thus can contend with each other. The 802.11 standard value of 32 for CW_{min} is therefore suggested for TCP download ACK traffic as providing a reasonable balance between number of collision and channel idle time.

Remark: We can verify that this choice of CW_{min} is sufficient, in combination with using an $AIFS$ value of zero, to prevent a backlog of TCP download ACKs building at the wireless stations. A sustained backlog will occur if, on average, the transmission rate of TCP download ACKs

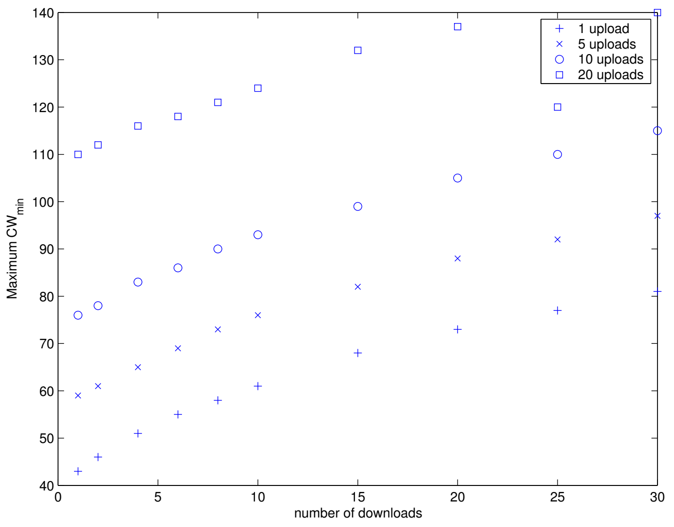


Fig. 5. Maximum CW_{min} vs number of upload and download stations, 802.11b PHY.

		$AIFS$ (slots)	CW_{min}	TX_{op} (packets)
AP	Upload ACKS	0	2	1
	Download data	4	32	n_d
wireless station	Download ACKS	0	32	1
	Upload data	4	32	1

TABLE I

TCP 802.11E MAC PARAMETERS WITH 11MBS PHY

on the wireless channel is less than the transmission rate of TCP download data packets (neglecting delayed acking for simplicity). In this situation, the stations sending TCP download ACKs are in a so-called saturated condition where they always have a packet to send, and hence can be modelled using the approach in [6]. By starting with a large value of CW_{min} for the TCP ACK traffic (so that the TCP ACK's are backlogged) and reducing CW_{min} until the TCP data transmission rate just equals the TCP ACK transmission rate we can determine the stability boundary for TCP ACK queueing. The stability boundary determined in this way is shown in Figure 5, and provides an upper bound on the value of CW_{min} for TCP ACK traffic. It can be seen that a value of 32 lies within the stability region across the range of operating conditions of interest.

Revisiting the example in Figure 2, the impact of the proposed prioritisation approach can be seen in Figure 6. Evidently, fairness is effectively restored between the competing TCP flows. We have also obtained similar fairness with other numbers of flows, including with different numbers of upload and download flows, although these are not included here owing to space constraints. The 802.11e MAC parameter settings used in this example (with an 11Mbs PHY) for both TCP uploads and downloads are summarised in Table I.

V. 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

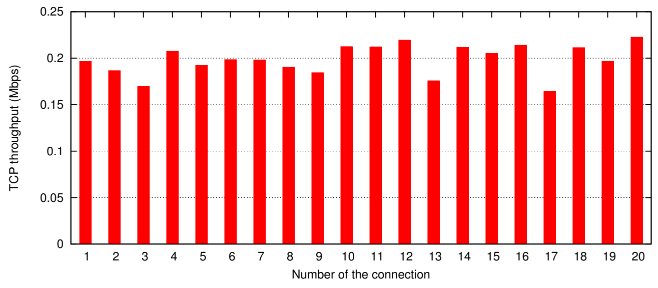


Fig. 6. Throughput of competing TCP uploads and downloads; first 10 flows are TCP uploads, remainder are TCP downloads. (NS simulation: 802.11e parameters as in Table I).

that uses the 802.11e $AIFS$, $TXOP$ and CW_{min} parameters to ensure fairness between competing TCP uploads and downloads. The computational burden of the proposed approach is very low (online adaptation is limited to inspection of one AP interface queue to determine the value for $TXOP$).

VI. 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] R. Bruno, M. Conti, E. Gregori, "Throughput Analysis of TCP Clients in Wi-Fi Hot Spot Networks", WONS 2004, 21-23 January, Trento, Italy.
[8] G. Bianchi, "Performance analysis of IEEE 802.11 distributed coordination function", *IEEE Journal on Selected Areas in Communications*, 18(3):535-547, March 2000.