This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2007 proceedings.

AQM with Dual Virtual PI Queues for TCP Uplink/Downlink Fairness in Infrastructure WLANs

Qiuyan Xia, Xing Jin and Mounir Hamdi
Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
Kowloon, Hong Kong, China
Email: {xiaqy, csvenus, hamdi}@cse.ust.hk

***Abstract*— In this paper we address the unfairness problem between uplink and downlink TCP flows in the IEEE 802.11 infrastructure WLANs. It has been shown that TCP flows exhibit resource allocation (e.g., bandwidth) inequalities mainly due to two reasons: TCP's asymmetric reactions towards data and ACK losses when the downlink buffer overflows at the AP; and the AP's inability to distinguish itself from other contending stations when accessing the medium with the 802.11 DCF MAC. To overcome this problem, we propose an AQM (Active Queue Management) approach (V2PI AQM) to be implemented at the AP, which utilizes two virtual queues, namely the data queue and the ACK queue, with their lengths controlled by the PI controllers. As a result, data losses can be reduced since the AP's downlink buffer is no longer overwhelmed by the ACK packets destined to the uplink stations. In addition, when a discrepancy is observed by the AP monitoring the traffic intensity on the wireless link and the lengths of both queues, it uses an AIMD (Additive Increase Multiplicative Decrease) approach to adjust its contention window size, which compensates for the unfairness induced by the DCF MAC. We demonstrate using ns-2 simulations that fair bandwidth sharing between uplink and downlink TCP flows can be achieved by the AP's buffer management in conjunction with contention window adaptation.**

I. Introduction

In recent years, WLAN (Wireless Local Area Network) technology has been evolving at a rapid pace. Most of the commercial WLAN products are based on the IEEE 802.11 standard [1] working under the infrastructure mode, where an AP (Access Point) is responsible for providing wireless stations with access to the wired network. All traffic going through or inside the WLAN is handled by the AP. According to the direction of the traffic flow, we use TCP downlink flow to denote traffic flowing from the wired network towards the wireless station, whereas uplink is used to refer to traffic flowing from the wireless station to the wired network. This is illustrated in Fig. 1, where wireless stations (TCP flows) are labelled as UP_STAs or DN_STAs (uplink or downlink).

Since the wireless link from the AP to all the receiving stations has limited capacity offered by the 802.11 standards compared to its wired counterparts, it may become a bottleneck and congestions happen easily at the AP. Previous research work showed that TCP is unfair towards uplink and downlink flows [2]: the sending stations (UP_STAs) obtain substantially larger bandwidth than the receiving stations (DN_STAs), which tend to starve and sometimes could not start altogether. Similar to [3], we refer to the uplink/downlink unfairness as the "critical unfairness" and that's the problem we try to solve in this paper.

Both MAC and TCP factors account for the critical uplink/downlink unfairness [4]. The AP must share the scarce wireless link bandwidth with other transmitting stations. While these stations contend the medium for their own traffic, the AP contends the medium for the whole downlink traffic to multiple terminals. However, the currently used 802.11 DCF (Distributed Coordination Function) at the MAC layer does not provide a higher priority to the AP than the wireless stations. Theoretically, when N wireless stations try to access the channel, the AP has a probability of accessing the channel equal to $1/(N+1)$. Since users typically download more data than they upload, the AP becomes a bottleneck, which needs to transmit downlink packets heading towards multiple stations. As a result, the downlink flows are highly penalized by the uplink flows originated from various stations. The reasons are as follows: packets belonging to multiple downlink TCP flows are buffered inside the AP's radio interface; besides, each receiver of the uplink flows sends the TCP ACKs which are stored at the AP's downlink buffer as well. Since the AP has no privileged access to wireless medium compared to the uplink stations, each receiver of the uplink flows is able to send the TCP ACKs with a comparable rate as the TCP downlink data packets, which fill up the AP's downlink buffer quickly (the typical value of the buffer size is $50 \sim 100$ packets). Hence, congestions and packet losses occur at the AP. However, TCP reacts differently to data and ACK losses [3], [4].

For downlink flows, a packet loss in the downlink buffer means a TCP data segment loss; at the wired sender, either duplicate ACKs are received (if the losses are not in a burst), or an RTO (Retransmission Time Out) event occurs. Since a data loss due to buffer overflow cannot be recovered by link layer retransmission, it has to be retransmitted by the TCP sender. Moreover, due to the burstiness of the TCP traffic, segments arrived at a full buffer are dropped in a burst fashion. Therefore, most packet losses are discovered by RTO. RTO triggers TCP's congestion control to reduce the congestion window to only a few packets and hardly increase any more, which in turn, causes a significant decrease of the downlink throughput and even a service outage. For uplink flows, a packet loss at the downlink buffer means the loss of a TCP ACK. Due to the cumulative nature of TCP ACKs, even if

1525-3511/07/$25.00 ©2007 IEEE

one or several ACKs are lost because of buffer overflow, it may not trigger the TCP's congestion control at the UP_STA, as long as some latter ACK with a higher sequence number is received by the UP_STA before a retransmission timeout.

Given the above analysis, "critical unfairness", characterized by complete starvation of some TCP flows or even, the inability of some TCP flows to start altogether, need to be avoided for satisfying the application's QoS requirements. The challenges involved for solving this problem are twofold: (1) controlling the number of TCP ACKs that prevents them from consuming all buffer space and causing bursty data losses; and (2) providing the AP some controlled priority to access the WLAN bandwidth.

For the first challenge, we propose an AQM (Active Queue Management) using two "virtual" queues to separate the data and ACK packets. The length of each virtual queue is regulated by the PI (Proportional Integral) controller. Note that we do not maintain two physical queues; actually, only one physical FIFO queue is implemented at the AP's downlink buffer. The virtual PI queue is a data structure which is kept track of only by its length. There is no real enqueue/dequeue operation on it; instead, such an operation is achieved by increasing/decreasing the queue length. When the queue length approaches the target value, early drop may be performed. Hence, bursty data losses at the AP's downlink buffer can be reduced significantly.

However, the control of the queue lengths alone can not achieve satisfying fairness between the uplink and downlink flows, due to the AP's inability to gain more access opportunities though it needs to send much more traffic than individual stations. The solution we adopt is for the AP to monitor the uplink/downlink traffic intensity (i.e., traffic rate, number of flows) and dynamically set the target queue length of the two virtual queues. In addition, whenever the data queue is detected to be under-utilized even though both queues are under control, we conclude that the AP cannot fill the data queue as it is allowed to, so we intentionally increase the AP's access probability to the medium. Through all of the above strategies, we show that the uplink/downlink unfairness can be reduced to a very low level.

The rest of the paper is organized as follows. In Section II, we briefly introduce the DCF in the IEEE 802.11 standard and discuss related work. The system model and details of our proposed AQM are presented in Section III. Section IV gives simulation set up and the main results of our approach. Finally, this paper concludes with Section V.

II. Background and Related Work

A. IEEE 802.11 DCF

The DCF defined in the 802.11 MAC layer is a mandatory, contention-based access protocol with CSMA/CA (Carrier Sense Multiple Access/Collision Avoidance). The basic access mode works as follows: before a station starts a frame transmission, it checks the medium status by carrier sensing. If the medium is sensed idle, the transmission may proceed; if the medium is sensed busy, the station defers its transmission until the medium is determined to be idle for the DIFS (DCF Inter Frame Space) and a random backoff procedure is invoked, where the duration of the timer is a function dependent on the physical layer and the contention window parameter cw. Once the backoff timer reaches zero, the station can transmit. For each successful reception of a frame, the receiver immediately sends an ACK after the SIFS (Short Inter Frame Space). If the ACK is correctly received, the sender defers for the DIFS and another random backoff procedure before transmitting the next frame. The DCF also defines an optional RTS/CTS mechanism to reserve the channel. We assume the basic access mode throughout this paper.

B. Related Work

Extensive work has been done regarding TCP's fairness issue. Most of them deal with the unfairness in the same direction, either uplink or downlink. Some investigate the unfairness between uplink and downlink. In [2], the authors showed that the MAC induced asymmetry of channel access can build up a large queue at the AP, where TCP ACKs take up most of the AP's buffer space. In [5], joint channel-aware scheduling and contention window adaptation was proposed to give the AP higher priority to access the channel. The authors of [6] investigated the interaction between WLAN link layer parameters or AP buffer provisioning with uplink/downlink fairness. It then proposed a simple scheduling discipline at the AP's buffer to control the aggressiveness of the uplink flows. The paper [7] proposed an AQM adjusting the TCP advertised window to slow down TCP senders so that they do not overflow the AP's queue. In [3], a rate control mechanism operating at the IP level before the AP uplink buffer was proposed to purposely introduce packet losses. In [8], the authors addressed both the "critical unfairness" and finer per-connection fairness. However, it has the main drawback in terms of implementation complexity. In [4], each TCP sender adjusts its sending rate based on the "link access cost", which prevents packet losses due to buffer overflow and assure per-station fairness. In [9], the authors evaluated the impact of EDCA (Enhanced Distribution Channel Access) on TCP applications with different traffic mix. By studying the effect of the setting of the EDCA parameters and the AP's buffer size on the performance of TCP applications and the employed admission strategy, it showed that using EDCA MAC the uplink/downlink unfairness can be tuned. In [10], the authors extended the control principles of the PI controller which first appeared as an AQM for wired networks. The proposed AQM adapts the AP's MAC layer target queue size according to the congestion level of the wireless channel.

In this paper, we propose solutions to avoid "critical unfairness" (i.e., uplink/downlink unfairness where downlink flows starve) by means of AQM with dual virtual PI queues and enforcing a fair sharing of WLAN bandwidth between the AP and the wireless stations. Our solution works on the aggregate TCP flows crossing the AP. Consequently, we do not provision a perfect per-station/flow fairness. However, our solution is simple to implement, and robust against variable traffic loads.

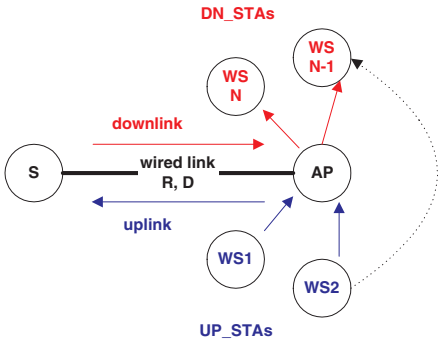

Fig. 1. Infrastructure WLAN model. The AP is connected to a wired station. There are N_{up} UP_STAs and N_{dn} DN_STAs with equi-distant from and close enough to the AP.

III. V2PI Active Queue Management

In this section, we present our approach to implement the dual virtual PI AQM, "V2PI AQM", at the AP's downlink buffer. We describe in detail the overall system model, the buffer allocation strategy (i.e., the assignments of target queue lengths), the AQM policy, and the way of informing and adapting the AP's access probability to the wireless medium. In this research, our main goal is to achieve satisfying uplink/downlink fairness without sacrificing the link utilization, and the ease of implementation. We expose the dependence between AP's buffer provisioning and fairness under different TCP traffic mix and traffic intensity.

A. System Model

The system model is shown in Fig. 1. A number of static wireless stations are associated with an AP and establish TCP connections with a wired host in a high-speed fixed network. In particular, we consider N_{dn} wireless stations downloading data from the wired host (downlink flows) and N_{up} wireless stations uploading data to the wired host (uplink flows). The AP is connected to the wired station via a 100 Mbps link with a 25 ms propagation delay. We assume that the wireless stations are equi-distant from and close enough to the AP for an error free channel. The stations and AP are working under 802.11b transmitting at the rate of 11 Mbps.

We evaluate the TCP throughput for each flow. In order to quantify the fairness performance, we use the metric "average per-station throughput ratio", $\gamma = R_{up}/R_{dn}$, where R_{up} (R_{dn}) denotes the average per-station throughput of UP_STAs (DN_STAs). Unfairness between TCP uplink and downlink flows occurs when $\gamma << or >> 1$, assuming that both uplink and downlink TCP sources are "greedy". The performance figures that we consider are listed in Table I.

B. The V2PI AQM Approach

In this section we explain our approach to handle TCP unfairness between uplink and downlink flows. Specifically, we introduce an integrated solution for fairness provisioning by active queue management with two virtual PI queues, and adjustment of the AP's access probability by adapting its contention window size. Our solution is based on the following observations:

- By controlling the number of TCP ACKs, the chance of buffer overflow and data packet losses is reduced; hence, congestion control at the TCP sender is less likely invoked.
- The choice of two virtual queues for ACK packets and data packets are flexible and simple. We do not perform exact queue operations; instead, only the queue length of each virtual queue is traced to decide the drop probability of the incoming packets.
- While TCP induced asymmetry towards uplink and downlink flows can be alleviated by buffer provisioning, the asymmetry induced by the MAC layer has to be solved by adjusting the access probability of the AP.

The framework of our approach is illustrated in Fig. 2. The implementation details are given below.

1) Dual Virtual PI Queues: The dual virtual PI queues, i.e., the ACK queue and the data queue, are the queue management entities (structures) associated with the AP's downlink buffer. By "virtual" we means that no buffer space accommodating packets is allocated; no real enqueue or dequeue operation is performed. In fact, there is only one real queue implemented at the AP's downlink buffer; however, we maintain two sets of parameters to trace the numbers of the ACK and data packets in the real queue, and regulate the enqueue/dequeue behavior of ACK and data packets.

To describe the V2PI AQM scheme we first define the

TABLE I
Performance Figures for the System Model

$R_{uptotal}$	sum of the uplink throughput
$R_{dntotal}$	sum of the downlink throughput
$R_{total} = R_{uptotal} + R_{dntotal}$	sum of uplink and downlink throughput
$R_{up} = R_{uptotal}/N_{up}$	mean throughput of uplink flows
$R_{dn} = R_{dntotal}/N_{dn}$	mean throughput of downlink flows
$\gamma = R_{up}/R_{dn}$	mean per-flow throughput ratio

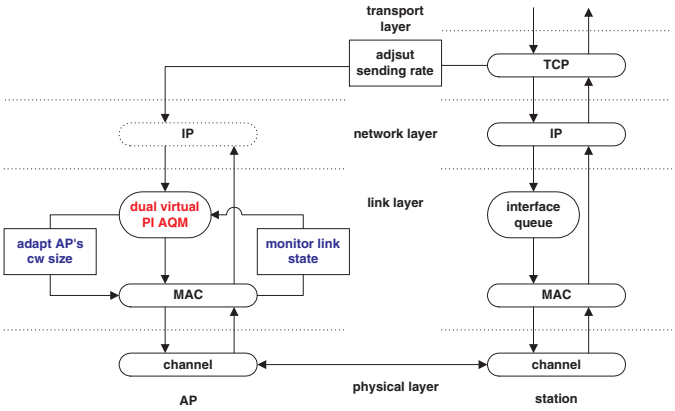

Fig. 2. Dual virtual PI AQM architecture [4] in ns-2 [11].

following terms in TABLE II. At the time a new packet is received at the LLC (Logical Link Control) layer, a packet classifier inspects the TCP header and refers to the corresponding virtual PI queue management entity. The goal of the PI controller is to drive the queue size around the reference value. It periodically (with frequency ω) calculates the packet drop probability ($qa/qd.prob$) as a function of the current/old/reference ACK/data queue length ($qa/qd.len$, $qa/qd.old$, $qa/qd.ref$), and the previous drop probability: $prob = a*(len-ref)-b*(old-ref)+prob$, where a and b are the PI control parameters. If the calculated $qa/qd.prob$ is high, the packet may be dropped with a high probability; otherwise, the packet is inserted to the tail of the physical buffer. This algorithm is detailed in Algorithm 1.

The problem that remains is how to set the reference queue size of the virtual ACK/data queue properly. We assume that each wireless station has only one flow to transfer and all flow rates are identical. We consider the following two simple cases:

1. $N_{up} = N_{dn} = N/2$;
2. $N_{up} = 1$, $N_{dn} = N-1$.

Nevertheless, these two cases can be simply generalized to arbitrary traffic mix.

For Case 1, it is natural to set $qa.ref = qd.ref$. The fairness performance is better with $\gamma \approx 3.64$ than a previous value of $\gamma \approx 41.34$ for a medium $N = 20$, but is still not so good. This result confirms that reducing data packet losses by controlling the queue length is insufficient for achieving satisfying uplink/downlink fairness. In the next section, we will introduce a way that informs the AP to adjust the access probability by modifying its contention window size. We show that this combined approach leads to a throughput ratio $\gamma \approx 0.92$.

For Case 2, we need to dynamically set $qa/qd.ref$. For this purpose, the AP monitors traffic intensity of uplink and downlink. Since all traffic must be forwarded by the AP, and it is aware of both N_{up} and N_{dn}, this approach is affordable. By investigating in simulations the impact of different $qa/qd.ref$ settings on fairness performance, we get the conclusion that it is not necessary to set the reference queue length accurately, as long as it does not lead to a large discrepancy. For example, we can set $\frac{qa.ref}{qd.ref} = \kappa \cdot \frac{D_{uptotal}}{D_{dntotal}}$, where $D_{[.]total}$ is the estimated traffic intensity in uplink/downlink direction, and $0.5 \leq \kappa \leq 3$.

2) AP's Contention Window Adaptation: The mechanism described above can suppress TCP's congestion control with respect to data losses of downlink flows, which improves the uplink/downlink fairness. However, it is unable to equally distribute the available bandwidth between uplink and downlink flows. This inequality is largely due to the same transmission opportunities gained by the AP, compared with the wireless stations. Therefore, we should provide a mechanism to differentiate the contention window sizes between the AP and the wireless stations. To avoid handling all cw's of the AP and stations, we choose to adapt the AP's cw only. Specifically, the proposed scheme assumes that the AP's MAC is aware of the dynamics of its downlink buffer. Whenever the AP finds that the utilization of the data queue is very low (while there is a large amount of active downlink traffic), compared to the utilization of the ACK queue, the next enqueued packet is marked in the flag bits in its header (some reserved bits can be utilized for this purpose); when the MAC layer of the AP receives the marked packet, it updates the credits counter for increasing its access weight; and when enough credits are accumulated, its cw is multiplicatively decreased. On the other hand, the AP should be able to decrease its access probability, in the case that its access weight is assigned to a too high value. This is achieved in a similar way as presented above by indicating in the flag bits with a different value. The details are illustrated in Algorithm 2, where cw is AIMD adapted with respect to currently accumulated credits.

TABLE II
PARAMETERS FOR DUAL VIRTUAL PI QUEUES

ω	sampling frequency
qa, qd	virtual ACK/data PI queue
($qa.a$, $qa.b$), ($qd.a$, $qd.b$)	PI parameters for virtual ACK/data queue
$qa.ref$, $qd.ref$	reference virtual ACK/data queue size
$qa.len$, $qd.len$	current virtual ACK/data queue size
$qa.old$, $qd.old$	old virtual ACK/data queue size
$qa.prob$, $qd.prob$	probability to drop a ACK/data packet
$qa.count$, $qd.count$	$\#packets$ since qa/qd.ref was last updated

TABLE III
SIMULATION PARAMETERS

$qa/qd.a$	1.822e-5	TCP Packet size	1500 Bytes
$qa/qd.b$	1.816e-5	AP's downlink buffer size	50 packets
ω	160 Hz	Initial TCP congestion window	2 packets
δ	0.01	AIMD parameters (α, β)	(8, 1.5)

IV. PERFORMANCE EVALUATION

A. Simulation Settings

In this section, we consider the network topology shown in Fig. 1. Simulations have been carried out in the ns-2 simulator (version 2.29.3) [11]. Within this environment, we suitably defined the simulation scenario (in *.tcl* files) and wrote the necessary additional codes. TABLE III summarizes the parameters used in the simulations. We assume that TCP sources always have data to send.

B. Simulation Results

In this section, we present the simulation results without/with our AQM, referred to as the BASE method and the V2PI method respectively, subject to different numbers of wireless stations. We still consider the two simple cases listed in Section III-B.

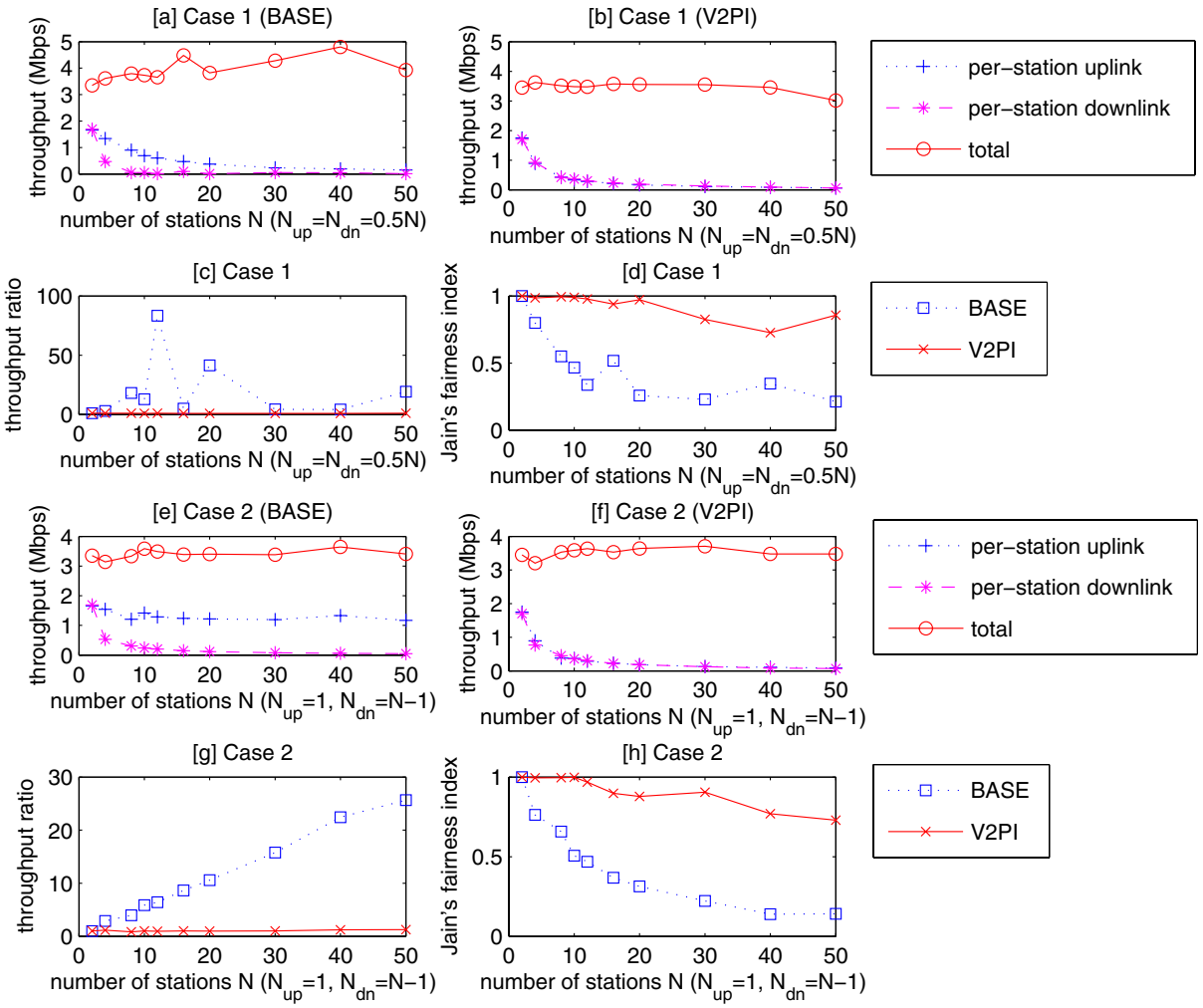

Fig. 3. Throughput/fairness performance for Case 1 and Case 2.

1) TCP Uplink/Downlink Fairness: We first study TCP's uplink/downlink fairness performance with the BASE method and the V2PI method. For Case 1, Fig. 3[a] shows the average per-station throughput in the uplink and downlink direction, together with the system aggregate throughput, with the BASE method; throughput result in Fig. 3[b] is with the V2PI method. As N increases, the average per-station throughput ratio between uplink and downlink varies a lot with BASE, and occasionally, the downlink flows are almost throttled, leading to an extremely high throughput ratio (Fig. 3[c]). However, with V2PI, the throughput performance is much more stable even when N gets larger. The bandwidth is fairly distributed between the uplink and downlink stations: the throughput ratio is always close to the ideal value 1. Similar behavior is observed for Case 2 in Fig. 3[e]-[g].

2) Jain's Fairness Index: Fig. 3[d] and [h] study the global fairness performance by Jain's fairness index [12] for Case 1 and Case 2, respectively. What we find is that, although we do not consider the fairness among the flows in the same direction in our V2PI scheme, it can still achieve better global fairness performance (above 0.7) compared to the BASE case, which drops significantly (below 0.3) as N increases. We note that this result is due to the reduced bursty data losses and the higher access priority of the AP. We believe that by simple scheduling at the AP, more desirable fairness performance in the same direction can be achieved.

3) Discussion: Comparing the results of Case 1 and Case 2 with the BASE method, we find that the figures in Case 1 fluctuates a lot and are hardly predictable, while the ones in Case 2 are more stable. This is mainly caused by the increased contention level for medium access, coupled with bursty data losses due to buffer overflow in Case 1, where N_{up} increases with N. Again, the V2PI method used for Case 1 can substantially reduce this effect, by limiting losses under contention and buffer overflow.

Algorithm 1 Dual Virtual PI AQM

enqueue(pkt)
if $isTcpAck(pkt)$ **then**
  $curq = qa$;
else
  $curq = qd$;
end if
$curq.count++$;
if $qlen >= qlim$ **then**
  $drop(pkt)$; {drop pkt due to overflow}
  $curq.count = 0$;
else if $drop\_early(pkt, curq)$ **then**
  $drop(pkt)$;
else
  $pktq.enqueue(pkt)$;
  $curq.len++$;
  <check wAP and set properly the weight related flags in the IP header, if necessary;>
end if

dequeue()
$pkt = pktq.dequeue()$;
if $isTcpAck(pkt)$ **then**
  $curq = qa$;
else
  $curq = qd$;
end if
$curq.len--$;
return pkt;

calculate_p()
$qa.prob = qa.a*(qa.len - qa.ref) - qa.b*(qa.old - qa.ref) + qa.prob$;
$qa.old = qa.len$;
$qd.prob = qd.a*(qd.len - qd.ref) - qd.b*(qd.old - qd.ref) + qd.prob$;
$qd.old = qd.len$;
<monitor the traffic intensity and properly increase/decrease the AP's access weight reflected in wAP;>
<set an interrupt with frequency ω to invoke this procedure.>

drop_early(pkt, qlen)
$u = uniform()$;
if $u \leq curq.prob$ **then**
  $curq.cout = 0$;
  return 1; {drop}
else
  return 0; {no drop}
end if

Algorithm 2 AP's cw Adaptation

MAC receives a pkt dequeued from the AP's buffer
if "inc weight" flags set in the IP header **then**
  $incAW += \delta$;
  clear flags in the IP header;
else if "dec weight" flags set in the IP header **then**
  $incAW -= \delta$;
  clear flags in the IP header;
end if
AIMD compensation for the AP's cw size
if $incAW > 0$ **then**
  $cw *= pow(1/\beta, int(incAW))$;
  $cw = max(cw, cw_{MIN})$;
else if $incAW < 0$ **then**
  $cw += \alpha * int(incAW)$;
  $cw = min(cw, cw_{MAX})$;
end if

V. Conclusion

Most of the Internet applications over WLANs are based on TCP. Due to the asymmetric behavior of TCP and MAC towards uplink/downlink flows, unfairness arises where service is biased to uplink stations while the downlink stations starve and may even undergo service outage. With the stringent requirements to fairly allocate the WLAN bandwidth between uplink and downlink flows, in this paper, we propose an approach utilizing the dual virtual PI AQM with AP's cw adaptation. The dual virtual PI queues are maintained to constrain the enqueued number of ACK packets and prevent them from overwhelming the overall buffer space; hence, the bursty data losses due to buffer overflow are largely reduced. We also introduce a way of dynamically setting the reference virtual queue sizes that can reflect the traffic mix/intensity. In addition, to resolve the MAC induced asymmetry, we intentionally raise the AP's priority to access the medium more. This is achieved by monitoring the traffic loads and queue dynamics, and informing the AP through sending a marked packet. The simulation results demonstrate that the proposed approach can provide quite good uplink/downlink fairness. Since it is only implemented at the AP and does not require per-flow/station queue, it can be implemented in practice with few modifications.

References

[1] IEEE Std 802.11-1999, *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, Std., Aug. 1999.

[2] S. Pilosof, R. Ramjee, D. Raz, Y. Shavitt, and P. Sinha, "Understanding TCP fairness over Wireless LAN," in *Proc. Infocom'03*, San Francisco, California, USA, apr 2003.

[3] N. Blefari-Melazzi, A. Detti, A. Ordine, and S. Salsano, "Controlling TCP Fairness in WLAN access networks using a Rate Limiter approach," in *2nd International Symposium on Wireless Communication Systems 2005 (ISWCS2005)*, Siena, Italy, sep 2005, pp. 375–379.

[4] E.-C. Park and D.-Y. Kim, "Asymmetric Behavior of TCP Flows and Unfairness of Channel Sharing in Wi-Fi Hot Spots," Tech. Rep., 2006. [Online]. Available: www.samsung.com/.../WinningPapers/downloads/12th/s6.pdf

[5] M. Bottigliengo, C. Casetti, C.-F. Chiasserini, and M. Meo, "Smart traffic scheduling in 802.11 wlans with access point," in *Vehicular Technology Conference, 2003 (VTC 2003-Fall)*, vol. 4, oct 2003, pp. 2227–2231.

[6] F. Vacirca and F. Cuomo, "Experimental results on the support of TCP over 802.11b: an insight into fairness issues," in *Proc. WONS'06*, Les Menuires, France, jan 2006.

[7] F. Liang, X. Wang, and L. Xu, "TFBR: A New Queue Management Guaranteeing TCP Fairness Based on TCP Advertised Receiving Windows Over WLAN," in *2nd International Conference on Mobile Technology, Applications and Systems 2005*, 2005, pp. 1–7.

[8] A. Banchs, A. Azcorra, C. Garcia, and R. Cuevas, "Applications and challenges of the 802.11e EDCAmechanism: An experimental study," in *IEEE Network*, jul 2005, pp. 52–58.

[9] M. Thottan and M. C. Weigle, "Impact of 802.11e edca on mixed tcpbased applications," in *Proc. Wicon'06*, Boston, MA, USA, aug 2006.

[10] H. Xu, Q. Xue, and A. Ganz, "Adaptive Congestion Control in Infrastructure Wireless LANs with Bounded Medium Access Delay," in *Proc. MobiWac'02*, Fort Worth, Texas, USA, Oct 2002, pp. 44–49.

[11] NS2, "The Network Simulator2," 2003. [Online]. Available: http://www.isi.edu/nsnam/ns

[12] D. Chiu and R. J. June, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Journal of Computer Networks and ISDN*, vol. 17(1), 1989.