NUI MAYNOOTH

Ollscoil na hÉireann Má Nuad

Improving Performance for CSMA/CA Based Wireless Networks

A dissertation
submitted for the degree of
Doctor of Philosophy

by

Tianji Li, BSc., MSc.

Supervisor: Prof. Douglas Leith

Hamilton Institute
National University of Ireland, Maynooth
Maynooth, Co. Kildare, Ireland

December 2007

Table of Contents

Notation

n	Number of Stations
M	Number of packets in a frame
m	Number of fragments in a frame
m'	Number of fragments in a packet
CW_{min}	Minimum contention window size
CW_{max}	Maximum contention window size
N_{BpS}	Number of encoded bits per OFDM symbol (NBpS)
T_{SIFS}	Duration of SIFS
T_{DIFS}	Duration of DIFS
T_{EIFS}	Duration of EIFS
T_{ack}	Time to transmit an ACK frame
T_{sym}	Interval of an OFDM symbol delay in 802.11a
T_{hdr}^{phy}	Time to transmit the PHY headers of a frame
T_{hdr}^{mac}	Time to transmit the MAC headers of a frame
T_{hdr}^{frag}	Time to transmit the fragment headers in a frame
T_p	Time to transmit a packet
T_f	Time to transmit a frame
T_{oh}^{p}	Overhead for transmitting a packet
T_{oh}^{f}	Overhead for transmitting payload of a frame
δ	Propagation delay
σ	PHY layer time slot
T_I	An idle slot duration in analysis

T_S	Average duration of a successful transmission in analysis
T_C	Average duration of a collision in analysis
L_{pld}	Payload size of a packet
L_{ACK}	MAC layer ACK frame size
L_{SER}	802.11a PHY layer SERVICE fields size ($16bits$)
L_{TAIL}	802.11a PHY layer TAIL fields size ($6bits$)
L_f	Payload size in a frame (bytes)
L_p	Packet size (bytes)
L_{frag}	Fragment size (bytes)
L_1	Fragment header size (bytes)
L_{hdr}^{mac}	Aggregate size of all MAC headers in a frame (bytes)
L_{hdr}^{frag}	Aggregate size of all fragment headers in a frame (bytes)
L_{FCS}	FCS size (bytes)
R	PHY layer data rate
p_c	Conditional collision probability seen by a packet transmitted on the channel
p_e	Packet error probability (PER) of a packet
p_b	Bit error rate (BER) of a STA in a stationary state
P_I	Probability of no transmissions in a slot
P_E	Probability of erroneous transmissions in a slot
P_C	Probability of collisions in a slot
P_S	Probability of successful transmissions in a slot

Table 1: Notation used in this thesis.

Abstract

Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) based wireless networks are becoming increasingly ubiquitous. With the aim of supporting rich multimedia applications such as high-definition television (HDTV, 20Mbps) and DVD (9.8Mbps), one of the technology trends is towards increasingly higher bandwidth. Some recent IEEE 802.11n proposals seek to provide PHY rates of up to 600 Mbps. In addition to increasing bandwidth, there is also strong interest in extending the coverage of CSMA/CA based wireless networks. One solution is to relay traffic via multiple intermediate stations if the sender and the receiver are far apart. The so called "mesh" networks based on this relay-based approach, if properly designed, may feature both "high speed" and "large coverage" at the same time. This thesis focusses on MAC layer performance enhancements in CSMA/CA based networks in this context.

Firstly, we observe that higher PHY rates do not necessarily translate into corresponding increases in MAC layer throughput due to the overhead of the CSMA/CA based MAC/PHY layers. To mitigate the overhead, we propose a novel MAC scheme whereby transported information is partially acknowledged and retransmitted. Theoretical analysis and extensive simulations show that the proposed MAC approach can achieve high efficiency (low MAC overhead) for a wide range of channel variations and realistic traffic types.

Secondly, we investigate the close interaction between the MAC layer and the buffer

above it to improve performance for real world traffic such as TCP. Surprisingly, the issue of buffer sizing in 802.11 wireless networks has received little attention in the literature yet it poses fundamentally new challenges compared to buffer sizing in wired networks. We propose a new adaptive buffer sizing approach for 802.11e WLANs that maintains a high level of link utilisation, while minimising queueing delay.

Thirdly, we highlight that gross unfairness can exist between competing flows in multi-hop mesh networks even if we assume that orthogonal channels are used in neighbouring hops. That is, even without inter-channel interference and hidden terminals, multi-hop mesh networks which aim to offer a both "high speed" and "large coverage" are not achieved. We propose the use of 802.11e's TXOP mechanism to restore/enfore fairness. The proposed approach is implementable using off-the-shelf devices and fully decentralised (requires no message passing).

Acknowledgements

I would like to thank most sincerely my advisor, Prof. Douglas Leith for his insightful guidance, constant encouragement and support. This thesis would not be possible without his rich expertise, excellent judgement, enthusiasm and dedication to top-quality research. I have learned a lot from him, not only how to write technical papers, most importantly how to be a good researcher. It is my fortune and honour to have an advisor like him.

I am indebted to my co-advisor David Malone (and his wife Sharon Murphy) and Qiang Ni (my previous co-advisor during the first year of my PhD) of Brunel University UK for their warm help on every aspect of work and life throughout all these years.

I am grateful to Yang Xiao of University of Alabama USA, Thierry Turletti of INRIA France, Ken Duffy, Peter Clifford, Rade Stanojevich, Ian Dangerfield, Domenico Giustiniano and Venkataramana Badarla for valuable discussions and various help.

I would like to thank Rosemary Hunt and Kate Moriarty for their great assistant. I thank Daniel Heffernan of the Department of Mathematical Physics for organising the basketball game on every Friday evening. Playing basketball with Rade Stanojevich, Dimitris Kalamatianos, Robert Shorten and Florian Knorn is a good fun.

.

CHAPTER 1

Introduction

In a computer or telecommunication network where participants communicate via a common physical medium, how we should coordinate their actions so that certain performance goals can be met? In the literature, this is known as the multiple access problem, with the corresponding protocols and mechanisms known as medium access control (MAC). The multiple access problem arises when the underlying medium is broadcast in nature, where messages from a station can be heard by all other stations that are in the listening area. With current physical layer techniques, if more than one station starts a transmission at the same time, all the transmitted frames will be lost. MAC layer protocols are therefore required to coordinate transmissions by competing stations to allow for sharing the common medium.

The coordination techniques can be classified into four subcategories: frequency division multiple access (FDMA), time division multiple access (TDMA), code division multiple access (CDMA) and carrier sense multiple access with collision avoidance (CSMA/CA). In this thesis, we consider how to improve performance for 802.11 family protocols which are based on CSMA/CA.

1.1 802.11 Family of Protocols

A series of CSMA/CA based standards have been specified by the IEEE. They are primarily designed for wireless local area networks (WLANs), but can also be used in wireless ad hoc or wireless multi-hop mesh networks. In this section, we provide a brief review of these protocols. Further details will be introduced in the following chapters when necessary.

1.1.1 802.11

In the first CSMA/CA based standard [1] which is known as 802.11, IEEE defines a MAC layer and three physical (PHY) layers at the 2.4 GHz band: Infrared, FHSS (frequency-hopping spread spectrum) and DSSS (direct-sequence spread spectrum). The maximum supported physical layer rate is 2 Mbps.

At the medium access layer, two MAC access methods are specified: a mandatory distributed coordination function (DCF) and an optional point coordination function (PCF). We only consider the DCF scheme in this thesis, as PCF is not typically implemented in real devices.

In the DCF, a station wishing to transmit first listens to the channel. If the channel is sensed "idle" for a time DIFS (distributed inter-frame space) then the station begins a random backoff. If the channel is sensed "busy" at any time during the backoff, the backoff is paused until the channel is sensed idle for time DIFS, and then resumed.

On transmitting a data frame, if this frame is received correctly, the receiver sends back an ACK frame after a short inter-frame space (SIFS) period. There are two requirements on the duration of SIFS. First, the duration should be long enough to ensure that the physical (PHY) layer of the receiver can turn itself from receiving to transmitting state. Second, SIFS should be shorter than DIFS so that an ACK frame can be sent before stations resume their backoff. In the DCF scheme, DIFS is equal to SIFS plus two idle slots[1].

[1]Slot time is physical layer dependent. For example, a slot time is 9 μs and 20 μs in IEEE 802.11a [3] and 802.11b [2] respectively.

The random backoff process before sending a data frame is used for mitigating collisions. Time is slotted and the random backoff is achieved by scheduling a random number uniformly from the range of $[0, CW - 1]$, where CW is the current contention window size. After each successful transmission, CW is reset to the minimum contention window size CW_{min}. If a transmission is lost (indicated by not receiving the corresponding ACK frame), the CW is doubled if the maximum allowable contention window size CW_{max} has not been reached yet (otherwise, CW_{max} is used as the new contention window size, i.e., $CW_{min} \leq CW \leq CW_{max}$).

1.1.2 802.11b/a/g — High Speed Extensions

The maximum PHY rate of 2 Mbps in 802.11 is typically not fast enough for modern applications. To support higher throughput, a series of extensions to the original 802.11 protocol are specified in which physical layer changes are specified but the MAC layer left unchanged.

In 802.11b [2], a physical layer standard for WLANs in 2.4 GHz radio band is specified. Three orthogonal channels are supported and the maximum link rate is 11 Mbps per channel.

In 802.11a [3] , another physical layer standard for WLANs is specified. But this time the radio band is restricted in the 5GHz band, which means that 802.11a is not compatible with 802.11b/g anymore. There are eight orthogonal channels and the maximum physical layer link rate is 54 Mbps per channel.

In 802.11g [4], a new physical layer standard for WLANs in the 2.4 GHz radio band is defined. There are three orthogonal channels available in 802.11g, with a maximum rate of 54 Mbps per channel. The 802.11g standard can support OFDM (orthogonal frequency-division multiplexing) modulation. For backward compatibility with 11b however, it also supports CCK (complementary code keying) modulation.

1.1.3 802.11e — The QoS Extension

IEEE 802.11e [5] is a MAC layer extension to the 802.11 standard for Quality of Service (QoS) provision. It can be used with the physical layers introduced above, i.e., 802.11a/b/g.

The QoS that is provided by 802.11e is known as soft QoS because it is not possible to guarantee absolute throughput/delay/jitter in CSMA/CA based networks (due to the randomness of the operations.). It is therefore more appropriate to refer to the QoS here as "differentiation", which is realised using four separate queues.

In more detail, at each station, 802.11e introduces four first-in first-out (FIFO) queues. Incoming packets from upper layers are mapped into different queue(s) according to their demands. A distinct CSMA/CA instance with its own parameters is used for each queue. Each queue thus behaves similarly to a single DCF station contending the channel. The differentiation between these queues is controlled by the associated CSMA/CA parameters $CW_{min[AC]}$, CW_{max}, $AIFS$ and $TXOP$. As noted above, CW_{min} and CW_{max} specify the minimum and maximum values of the contention window. The AIFS parameter is the name for DIFS in 802.11e. By adjusting this parameter the time that a CSMA/CA instance waits before resuming backoff can be controlled. The TXOP parameter specifies the duration that a queue is permitted to transmit once it has won a transmission opportunity. This duration can be made long enough to allow multiple frames to be transmitted back-to-back in a burst. Frames in the same TXOP burst are separated by SIFS. Note that when queues at the same station attempt transmissions in the same slot, a virtual collision is said to occur. In this case, the queue with the highest priority is allowed to transmit.

1.2 Challenges

CSMA/CA based 802.11 technology is becoming increasingly ubiquitous in last-mile access networks. With the aim of supporting rich multimedia applications such as high-definition television (HDTV, 20Mbps) and DVD (9.8Mbps), one of the technology trends is towards increasingly higher bandwidth. Some recent 802.11n proposals seek to support PHY rates

of up to 600 Mbps ([6, 8, 9, 124]). Meanwhile, in order to provide extended coverage range which is a key requirement for last-mile solutions, a second trend is towards greater use of multiple wireless hops in both office environments and the home. Multi-hop wireless is also of interest for broadband back-haul infill and for municipal and rural wireless coverage.

Higher PHY rates do not however necessarily translate into corresponding increases in upper layer throughput because overhead at the MAC layer means that efficiency typically decreases with increasing PHY rates [11, 122]. Improving both PHY and MAC efficiency is still not sufficient to ensure high system level throughput: consideration of interactions between the MAC/PHY and transport layer is also required. Moreover, increasing PHY rates typically come at the cost of reduced coverage. Although multi-hop networks can potentially be used to increase coverage while maintaining high rates, great unfairness can exist amongst competing traffic flows [42] due to the fact the 802.11 based MAC layer fails to provide a fair allocation capability. Our aim in this thesis is to address these key challenges.

1.3 Contributions

1.3.1 Mitigating Overhead For Very High-Speed WLANs

To understand the impact of MAC layer overhead on system performance, we first develop an analytical model for the current MAC layer scheme, i.e., the 802.11 DCF. Our results show that the MAC layer throughput can not exceed 50 Mbps when IEEE 802.11n[2] physical layer parameters are used. We then investigate the possibility of using existing schemes to achieve higher MAC layer throughput, with the particular aim of achieving at least 100 Mbps at the MAC layer. A promising option is the Block ACK (BTA) scheme of IEEE 802.11e [5]. With the BTA scheme, multiple data frames followed by only one ACK frame are transmitted in a block. We extend our 802.11 DCF model to evaluate the BTA performance. Our results show that the BTA scheme can achieve 100 Mbps at the MAC layer. However, it can only do so if the underlying physical rates used are higher than 500 Mbps, which means that

[2]IEEE 802.11n is a future standard that is being specified by the IEEE to support very high-speed in WLANs.

more than 80% of the capability provided by the physical layer is wasted by the MAC layer.

In the BTA scheme, the contention window backoff and MAC ACK frame transmission time are amortised over multiple frames, therefore improving efficiency. However, as the PHY rate is increased, the time to transmit a frame quickly becomes dominated by PHY headers, and the BTA throughput efficiency rapidly decreases.

We therefore develop a novel scheme called Aggregation with Fragment Retransmission (AFR). In the AFR scheme, multiple packets are aggregated and transmitted in a single large frame[3]. If errors occur during transmission, only the corrupted fragments of the frame are retransmitted. We propose a zero-waiting mechanism where frames are transmitted immediately once the MAC wins a transmission opportunity. An analytic model is developed to evaluate the throughput and delay of AFR over noisy channels, and to compare AFR with competing schemes. We implement the AFR scheme in the network simulator *NS-2* and present detailed results for TCP, VoIP and HDTV traffic. Results suggest that AFR is a promising MAC technique for very high-speed WLANs. Moreover, AFR is particularly effective for rich multimedia services with high data rates and large packet sizes, which are key applications in future WLANs.

1.3.2 Buffer Sizing for WLANs

We observed during the AFR work that there can be a close interaction between the MAC layer and the buffer above it for real world traffic such as TCP. Surprisingly, this buffering issue has received little attention in the 802.11 literature.

The classical rule of thumb is to provision buffers to be equal to the *bandwidth* of the link multiplied by the average *delay* (which is typically described by round trip time or RTT) of the flows utilising this link: the *Bandwidth-Delay Product* (BDP) [112]. While buffer sizing has been a major design issue for wired links (see for example [14] [97] [117]), a number of fundamental new issues arise in 802.11 WLANs. Firstly, the mean service rate

[3]We define a *packet* as what MAC receives from the upper layer, a *frame* as what MAC transfers to the PHY layer, and a *fragment* as part(s) of a frame.

at a wireless station is strongly dependent on the level of channel contention and thus on the number of active stations and their load. Secondly, even when the network load is fixed, the packet inter-service times at a station are not fixed but vary stochastically due to the random nature of the CSMA/CA operation. As a result, neither the bandwidth nor the delay in 802.11 WLANs are constant, in contrast to the wired links. We therefore do not have a fixed BDP value available to provide a basis for sizing buffers.

We first consider an adaptive sizing algorithm based on measurement of the current average service rate to emulate the BDP rule in WLANs. The effectiveness of this first algorithm is verified against simulations. To exploit statistical multiplexing gains in WLANs, we also propose a second adaptive buffer tuning algorithm. This involves feedback control of buffer sizes based on measurements of the buffer idle and busy time. The effectiveness of these algorithms for achieving high link utilisation while minimising queueing delay is demonstrated via extensive simulations.

1.3.3 Restoring Fairness in Multi-hop Networks

A relay-based multi-hop network, if properly designed, can potentially offer both "high speed" and "large coverage" simultaneously. However, we highlight that the 802.11 DCF scheme can lead to gross unfairness in multi-hop networks, even without interference and hidden terminals related issues. This MAC induced unfairness can be particularly problematic in the context of multi-hop as unfairness can become amplified over multiple hops. That is, a 802.11 multi-hop network offers both "high speed" and "large coverage" is still not practical. Using the TXOP functionality of 802.11e, we propose a new MAC scheme that restores/enforces per-flow fairness. The proposed scheme is simple, implementable using off-the-shelf devices and fully decentralised (requires no message passing). The effectiveness of the proposed scheme is demonstrated both via simulations and experimental measurements.

1.3.4 Publications

1. **Tianji Li**, Douglas Leith, David Malone, and Venkataramana Badarla, "Achieving End-to-end Fairness in 802.11e Based Wireless Multi-hop Mesh Networks", *ChinaCom 2008*.

2. **Tianji Li**, and Douglas Leith, "Adaptive Buffer Sizing for TCP Flows in 802.11e WLANs", *ChinaCom 2008*.

3. **Tianji Li**, and Douglas Leith, "Buffer Sizing TCP Flows in 802.11e WLANs", *IEEE Communications Letters*, vol. 12, no 3, Mar. 2008.

4. **Tianji Li**, Qiang Ni, David Malone, Douglas Leith, Yang Xiao and Thierry Turletti, "Aggregation with Fragment Retransmission for Very High-Speed WLANs", *IEEE/ACM Transactions on Networking*, to appear in Apr. 2009.

5. Ken Duffy, Douglas Leith, **Tianji Li** and David Malone, "Modelling 802.11 Mesh Networks," *IEEE Communication Letters*, vol. 10, no 8, Aug. 2006.

6. Ken Duffy, Douglas Leith, **Tianji Li** and David Malone, "Improving Fairness in Multi-Hop Mesh Networks Using 802.11e," *RAWNET* 2006 workshop (in conjunction with IEEE WiOpt 2006).

7. **Tianji Li**, Qiang Ni, David Malone, Douglas Leith, Yang Xiao and Thierry Turletti, "A New MAC Scheme for Very High-Speed WLANs," in *Proc. of IEEE WOWMOM* 2006.

8. **Tianji Li**, Qiang Ni and Yang Xiao, "Investigation of the Block ACK scheme in Wireless Ad-hoc Networks," *Wiley Journal of Wireless Communications and Mobile Computing (JWCMC)*, 2006.

9. **Tianji Li**, Qiang Ni, Thierry Turletti and Yang Xiao, "Performance Analysis of the IEEE 802.11e Block ACK Scheme in a Noisy Channel," in *Proc. of IEEE BroadNets* 2005.

10. Qiang Ni, **Tianji Li**, Thierry Turletti and Yang Xiao, "Saturation Throughput Analysis of Error-Prone 802.11 Wireless Networks," *Wiley Journal of Wireless Communications and Mobile Computing (JWCMC)*, 2005.

11. Qiang Ni, **Tianji Li**, Thierry Turletti and Yang Xiao, "AFR Partial MAC Proposal for IEEE 802.11n". *IEEE 802.11n Working Group Document*: IEEE 802.11-04-0950-00-000n, August 13, 2004.

12. Qiang Ni, David Malone, Peter Clifford, Ken Duffy, Douglas Leith and **Tianji Li**, "Modelling and Simulation Analysis of the 802.11/802.11e MAC Layers," *Modelling and Simulation of Wireless Networks (Book)*, Nova Science Publishers, New York, USA.

1.4 Outline

This thesis is organised as follows. In Chapter 2, we present the models for the DCF and BTA schemes. In Chapter 3, 4 and 5, respectively, the AFR scheme for very high-speed WLANs, the proposed buffer sizing scheme and the unfairness issues in multi-hop networks are introduced.

CHAPTER 2

Modelling Existing Protocols

The underlying MAC technique used by 802.11a/b/g/e is the same, i.e., the DCF scheme of 802.11. A first question we seek to address is whether it is possible to support 100 Mbps at the MAC layer if the DCF is used in future very high-speed WLANs. To answer this question, a detailed analysis of the DCF scheme is required. In this chapter, we develop an analytical model (called the DCF Model) for this purpose. A key novelty of the DCF Model is that we explicitly model the impact of channel errors on MAC layer throughput. This is important for two reasons. First, prior models normally assume that the wireless channel is error free, yielding results that tend to be optimistic. Second, as we will see in Chapter 3, in future 802.11n schemes, it is proposed that MAC layer ACK frames are sent back even when parts of the data frame are lost. This means that it is necessary to distinguish error events from frame losses due to collisions in order to gain a proper understanding of MAC layer operation. Using the DCF Model, we show that the DCF scheme is not fundamentally capable of supporting 100 Mbps at the MAC layer.

In the second part of this chapter, we then investigate the possibility of using existing proposals to achieve the desired aim, i.e., 100 Mbps at the MAC layer. The most promising option at the time of this work was the Block ACK (BTA[1]) scheme of IEEE 802.11e [5].

[1]We use BTA as the acronym for this scheme because in 802.11e [5] BA is used as the name of the block acknowledgement packet of this scheme.

With the BTA scheme, multiple data frames are transmitted in a block followed by a single ACK frame. We extend the DCF model to evaluate the performance of the BTA scheme. Using this model, we first show the advantages of BTA over DCF, and analyse how to select a proper block size so as to maximise performance. We then demonstrate that BTA can support 100 Mbps in future 802.11n only if the physical layer rate is faster than 500 Mbps.

2.1 802.11 DCF

2.1.1 Related work

The behaviour of the 802.11 DCF mechanism in error free channels has been analytically modelled in many previous papers, most of which are based the seminal paper of Bianchi [20]. Amongst these extensions, Wu et. al. in [120] consider the impact of a finite retransmission limit, which is assumed to be infinite in [20].

We give a detailed review of Bianchi's model with Wu's extension as our models for the DCF and BTA schemes in this chapter and our proposal for future 802.11n in the next chapter are all based on it.

In [20], Bianchi first introduced a 2-dimensional stochastic process $\{s(t), b(t)\}$ to model the backoff behavior of the DCF scheme with saturated stations[2]. Process $b(t)$ represents the backoff counter, and it is decremented at the beginning of each slot. The slots used in the model correspond either to idle slots, successful transmissions or colliding transmissions. For an idle slot, the duration corresponds to the physical slot time. In a successful or collision slot, however, $b(t)$ is frozen for the duration of a transmission. When $b(t)$ reaches zero, transmission is triggered and a new backoff starts. If the transmission is successful, the contention window is reset to CW_{min} and the backoff counter is assigned a value selected uniformly randomly from 0 to CW_{min}. If the transmission fails, the contention window is doubled if the maximum value CW_{max} has not been reached yet (otherwise, CW_{max} is used as the new contention window size). The backoff counter in this case is assigned a random

[2] A station is said to be saturated if it always has frames available to send.

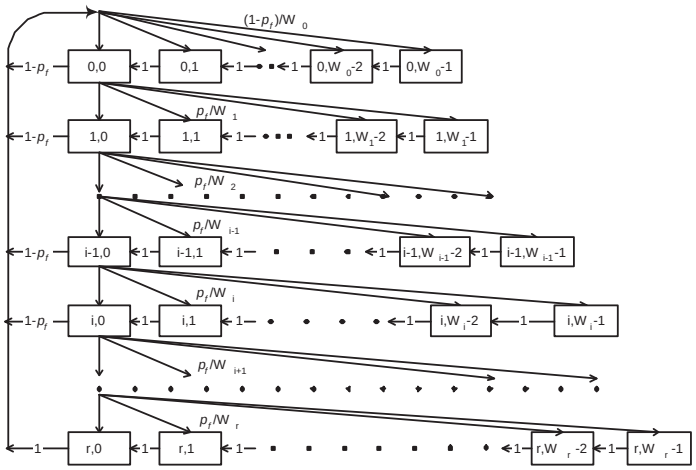


Figure 2.1: The Markov chain proposed in Wu's model. In this figure, $W_i = min(2^i CW_{min}, CW_{max})$ and p_f is the probability that a transmission fails.

value in the range from 0 to the new contention window.

This implies that $b(t)$ depends on the transmission history. To capture the history, another process $s(t)$ is defined for the backoff stage. In addition, the following two assumptions are made. First, the transmission probability τ in every slot time is constant and the same for all stations. Second, at each transmission attempt, regardless of the number of retransmissions, each frame is lost with an independent constant probability p_f. Under these assumptions, the 2-dimensional stochastic process $\{s(t), b(t)\}$ forms a Markov chain as shown in Fig. 2.1. In this chain, all states are aperiodic, recurrent and non-null, and thus the process is ergodic [58] and a stationary distribution exists. One can solve for τ and p_f for the stationary distribution of this Markov chain as follows.

In [20] and [120], it is assumed that there are no transmission errors, so

$$p_f = p_c = 1 - (1 - \tau)^{n-1}, \tag{2.1}$$

where p_c is the probability of a collision (conditioned on a transmission taking place) and n is the number of stations in the network.

Let r and z be the maximum retry limit and the maximum allowed backoff stage respectively. Notice that if $r \geq z$, the contention window size can be increased up to the maximum CW_{max} which will be used when a frame is re-sent more than z times, i.e.,

$CW_{max} = 2^z CW_{min}$. If $r \leq z$, a frame will be discarded before the current contention window reaches CW_{max}. We then have that the backoff window at the i-th backoff stage is

$$W_i = min(2^i CW_{min}, CW_{max}). \tag{2.2}$$

The transition probabilities of the Markov chain in Fig. 2.1 are given by

$$\begin{cases} P\{i,k|i,k+1\} = 1, & 0 \leq i \leq r \quad 0 \leq k \leq W_i - 2 \\ P\{0,k|i,0\} = (1-p_f)/W_0, & 0 \leq i \leq r-1 \quad 0 \leq k \leq W_0 - 1 \\ P\{i,k|i-1,0\} = p_f/W_i, & 1 \leq i \leq r \quad 0 \leq k \leq W_i - 1 \\ P\{0,k|r,0\} = 1/W_0, & 0 \leq k \leq W_0 - 1. \end{cases} \tag{2.3}$$

These transition probabilities account respectively for: 1) the decrements of the backoff counter when the channel is sensed idle for a slot; 2) after a successful transmission, the backoff for the new frame starts from stage 0; 3) a colliding transmission leads to an increase in backoff stage; 4) if the transmission is unsuccessful at the maximum backoff stage, then the backoff stage is reset to zero and the corresponding frame is dropped, or else in the case of successful transmission the backoff stage is also reset to zero but for sending a new frame.

Let $b_{i,k} = \lim_{t\to\infty} P\{s(t) = i, b(t) = k\}$ be the stationary distribution of the Markov chain, where $i \in [0,r]$, $k \in [0, W_i - 1]$. We have that

$$b_{i,0} = p_f^i b_{0,0}, \quad 0 \leq i \leq r. \tag{2.4}$$

Owing to the chain regularities, for each $k \in (0, W_i - 1)$, we have that

$$b_{i,k} = \frac{W_i - k}{W_i} \begin{cases} (1-p_f)\sum_{j=0}^{r-1} b_{j,0} + b_{r,0}, & i = 0, \\ p_f b_{i-1,0}, & 0 < i \leq r. \end{cases} \tag{2.5}$$

Using (2.4), Equation (2.5) can be simplified as

$$b_{i,k} = \frac{W_i - k}{W_i} b_{i,0}, \quad 0 \leq i \leq r. \tag{2.6}$$

Therefore, with (2.4), (2.6), and (2.2), $b_{0,0}$ is obtained using the normalization condition

$$1 = \sum_{i=0}^{r} \sum_{k=0}^{W_i - 1} b_{i,k} = \sum_{i=0}^{r} b_{i,0} \frac{W_i + 1}{2} = \sum_{i=0}^{r} p_f^i b_{0,0} \frac{W_i + 1}{2}, \tag{2.7}$$

from which we have that

$$b_{0,0} = \begin{cases} \frac{2(1-2p_f)(1-p_f)}{(1-p_f)W(1-(2p_f)^{r+1})+(1-2p_f)(1-p_f^{r+1})}, & r \leq z, \\ \\ \frac{2(1-2p_f)(1-p_f)}{(1-p_f)W(1-(2p_f)^{z+1})+(1-2p_f)(1-p_f^{r+1})+W2^z p_f^{z+1}(1-2p_f)(1-p_f^{r-z})}, & r > z. \end{cases} \tag{2.8}$$

We can now calculate the transmission probability τ that a station transmits in a randomly chosen time slot. Since a transmission occurs only when the backoff counter reaches zero, τ can be expressed as

$$\tau = \sum_{i=0}^{r} b_{i,0} = b_{0,0}\frac{1-p_f^{r+1}}{1-p_f} =$$
$$\begin{cases} \frac{2(1-2p_f)(1-p_f^{r+1})}{(1-p_f)W(1-(2p_f)^{r+1})+(1-2p_f)(1-p_f^{r+1})}, & r \leq z, \\ \\ \frac{2(1-2p_f)(1-p_f^{r+1})}{(1-p_f)W(1-(2p_f)^{z+1})+(1-2p_f)(1-p_f^{r+1})+W2^z p_f^{z+1}(1-2p_f)(1-p_f^{r-z})}, & r > z. \end{cases} \tag{2.9}$$

From Equations (2.1) and (2.9), a solution for p_f and τ can be found numerically.

2.1.2 The DCF Model

In this section we develop a new model that encompasses the impact of channel errors.

We define a *collision* as an event where at least two stations start transmissions in the same slot and assume that none of the receivers can decode the transmitted frames correctly[3]. We define an *error* as an event where a frame is corrupted due to reasons other than collisions.

In the case of collisions or errors, each sender waits for an ACK frame until its ACK timer expires, and then performs a new random backoff before retransmission. All other stations wait until the frame transmission is over and defer an EIFS duration before resuming their backoff process. The duration of EIFS is the sum of a SIFS, a DIFS and an ACK transmission interval, i.e., $T_{EIFS} = T_{SIFS} + T_{hdr}^{phy} + T_{ACK} + T_{DIFS}$ where the notation used is defined in Table 1.

[3]We assume that there is no capture effect and there are no hidden terminals.

Similarly to [20] and [120], we use a discrete time Markov chain model to study the random backoff behavior of a station. The key differences are i) we modify the probability p_f to include the probabilities of both collisions and errors, and ii) we introduce a new probability p_e which denotes the probability of an error event when a frame is being transmitted. We assume that at each transmission attempt, regardless of the number of retransmissions, each frame has a constant failure probability p_f which is expressed as

$$p_f = 1 - (1 - p_c)(1 - p_e) = p_c + p_e - p_e p_c. \tag{2.10}$$

where p_c is the probability of a collision (conditioned on a transmission taking place).

We can obtain p_e by assuming that the two events "data frame corrupted" and "ACK frame corrupted" are independent, i.e.,

$$p_e = p_e^{data} + p_e^{ack} - p_e^{data} p_e^{ack}, \tag{2.11}$$

where p_e^{data} and p_e^{ack} are the packet error rates for data frames and ACK frames respectively. Assuming that the wireless channel is Gaussian with uniformly distributed bit errors, the packet error probability p_e^{data} of a data frame is

$$p_e^{data} = \sum_{i=1}^{L_p} \binom{L_p}{i} p_b^i (1 - p_b)^{L_p - i} = 1 - (1 - p_b)^{L_p}, \tag{2.12}$$

where p_b and L_p are the bit error rate and the length (in bits) of the frame respectively.

Similarly, we obtain the frame error probability for an ACK frame p_e^{ack} as

$$p_e^{ack} = 1 - (1 - p_b)^{L_{ack}}, \tag{2.13}$$

where L_{ack} represents the length of the ACK frame.

The bit error rate (BER) p_b can be estimated by measuring the signal-to-noise ratio for different modulation schemes. For example, for BPSK and QPSK modulations it can be expressed as [92]

$$p_b = Q\left(\sqrt{2\frac{E_b}{N_0}}\right). \tag{2.14}$$

For M-ray QAM (M can be 16 or 64 in 802.11a), p_b can be expressed as

$$p_b \approx 4(1 - \frac{1}{\sqrt{M}})Q\left(\sqrt{\frac{3E_b}{(M-1)N_0}}\right), \tag{2.15}$$

where $\frac{E_b}{N_0}$ is the bit-energy-to-noise ratio of the received signal and Q is defined as

$$Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{\frac{-t^2}{2}}\, dt. \tag{2.16}$$

Assuming that each frame collides with a constant probability p_c, we have

$$p_c = 1 - (1 - \tau)^{n-1}. \tag{2.17}$$

Combining Equation (2.10) and Equation (2.17), we get the expression of p_f as

$$p_f = 1 - (1 - p_e)(1 - \tau)^{n-1}, \tag{2.18}$$

where p_e is from Equation (2.11).

Equation (2.9) and Equation (2.18) represent a nonlinear system of equations with two unknown variables (τ and p_f) which can be solved numerically.

The saturation throughput S is

$$S = \frac{P_S E[L_{pld}]}{E[T]}, \tag{2.19}$$

where P_S is the probability of a successful transmission, $E[L_{pld}]$ is the expected packet size, and $E[T]$ denotes the expected slot duration. To calculate the expected slot durations, we need to consider five possible types of slot (as shown Fig. 2.2) with durations:

- T_I, an idle slot;
- T_S, the average duration of a successful transmission;
- T_C, the average duration of a collision. Since hidden terminals are not considered, only the data frames can collide with each other, and there are no collisions between ACK frames;

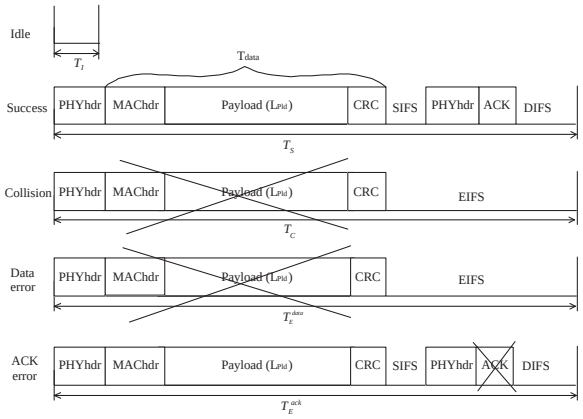


Figure 2.2: Slot durations used in the DCF Model.

- T_E^{data}, the average time that the channel is sensed busy when there is a transmission error in a data frame;

- T_E^{ack}, the average time that the channel is sensed busy when there is a transmission error in an ACK frame.

Note that the first three terms (T_I, T_S, T_C) have been used in the literature. However, the calculation of T_C in the literature (where $T_C = T_{hdr}^{phy} + T_{data} + \delta + T_{DIFS}$) is sometimes not accurate. According to the 802.11 standard [1] the channel has to be reserved for an EIFS interval (T_{EIFS}) after a data frame collision is detected. So,

$$T_C = T_{hdr}^{phy} + T_{data} + \delta + T_{EIFS} \tag{2.20}$$

and

$$T_{EIFS} = T_{SIFS} + T_{hdr}^{phy} + T_{ack} + \delta + T_{DIFS}. \tag{2.21}$$

where T_{data} and T_{ack} denote the transmission time of a data frame and an ACK frame respectively.

Moreover, because the 802.11 MAC protocol cannot differentiate collisions from error events, the protocol behaves the same in both cases, i.e., $T_E^{data} = T_C$. Other stations in these two cases use an EIFS interval as the waiting period, and the sender waits for

$T_{ACKOut} + T_{DIFS}$. According to the 802.11 standard [1]

$$T_{ACKOut} = T_{SIFS} + T_{hdr}^{phy} + T_{ack} + \delta, \tag{2.22}$$

so the sender waits for the same duration as other stations. If a data frame is successfully received but its ACK is lost due to errors, all other stations will treat this event as a successful transmission, i.e., reserve the channel for duration $T_E^{ack} = T_S$.

We therefore have that

$$\begin{cases} T_I = \sigma \\ T_S = T_{hdr}^{phy} + T_{data} + \delta + T_{SIFS} + T_{hdr}^{phy} + T_{ack} + \delta + T_{DIFS} \\ T_C = T_{hdr}^{phy} + T_{data} + \delta + T_{EIFS} \\ T_E^{data} = T_{hdr}^{phy} + T_{data} + \delta + T_{EIFS} \\ T_E^{ack} = T_S. \end{cases} \tag{2.23}$$

The values of T_{data} and T_{ack} are PHY layer dependent. In the case of an 802.11a PHY, we have that

$$T_{data} = T_{sym} \lceil \frac{L_{SER} + L_{TAIL} + L_p}{N_{BpS}} \rceil, \tag{2.24}$$

and

$$T_{ack} = T_{sym} \lceil \frac{L_{SER} + L_{TAIL} + L_{ack}}{N_{BpS}} \rceil, \tag{2.25}$$

where $\lceil . \rceil$ denotes the ceiling function that returns the smallest integer value greater than or equal to its argument value, T_{sym}, L_{SER}, L_{TAIL}, N_{BpS} denote the duration of a symbol, the size of the SERVICE field, the size of TAIL field, and the number of bit per symbol of 802.11a [3], respectively.

The expected slot duration $E[T]$ is then

$$E[T] = T_I P_I + T_S P_S + T_E^{data} P_E^{data} + T_E^{ack} P_E^{ack} + T_C P_C \tag{2.26}$$

where $P_I = (1-\tau)^n$ is the probability of an idle slot, P_S is the probability of a successful transmission, and P_C is the probability of a collision. P_E^{data} and P_E^{ack} are the probabilities that errors occur in a data or an ACK frame. We have that

$$P_E^{data} = n\tau(1-\tau)^{n-1} p_e^{data}, \tag{2.27}$$

and

$$P_E^{ack} = n\tau(1-\tau)^{n-1}(1-p_e^{data})p_e^{ack}. \tag{2.28}$$

The probability P_S for a successful transmission in a slot is

$$P_S = n\tau(1-\tau)^{n-1}(1-p_e) = n\tau(1-\tau)^{n-1}(1-p_e^{data})(1-p_e^{ack}). \tag{2.29}$$

and

$$P_C = 1-(1-\tau)^n - n\tau(1-\tau)^{n-1}. \tag{2.30}$$

Assuming a fixed frame size, the expected packet size $E[L_{pld}] = L_p$ where L_p is the packet size.

Finally, the system throughput can be computed as

$$S = \frac{P_S L_p}{T_I P_I + T_S P_S + T_E^{data} P_E^{data} + T_E^{ack} P_E^{ack} + T_C P_C}, \tag{2.31}$$

where all variables are known.

2.1.3 Validation

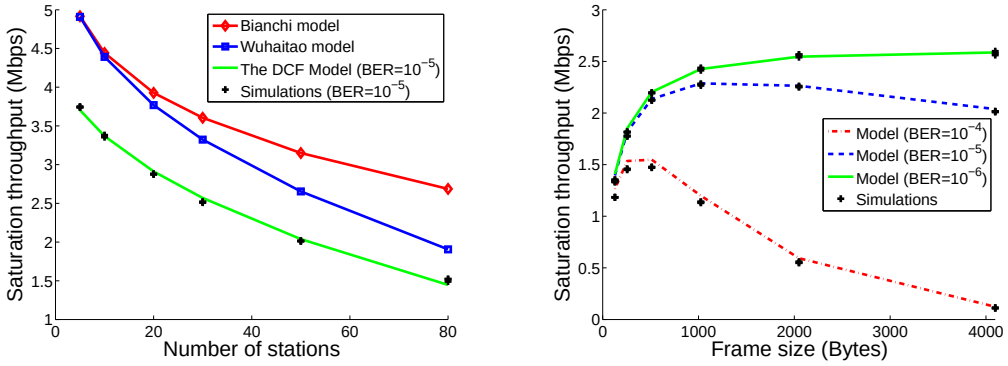
We validate the DCF Model using the network simulation[4] tool, *NS-2* (version 2.27) [12]. We modified the transmission time of the PLCP preamble and PLCP header in *NS-2* to be a constant 20 μs duration as required by the 802.11a specification (in the original *NS-2* code, the PLCP preamble and PLCP header are transmitted at the basic rate.). We also modified the simulator to generate transmission errors independently of collisions. The transmission power used by each station is set to be high enough so that stations are able to hear each other and there are no hidden terminals. Table 2.1 lists the 802.11a MAC/PHY layer parameters used.

We begin by comparing the simulation results with the predictions obtained from the Bianchi model [20], the Wu model [120] and the DCF Model. A BER of 10^{-5} is used and

[4]For all the simulations involved in this thesis, we run the tests for a long enough time to ensure stable results, i.e., without noticeable difference between each separate run. We also select a distinct random seed for each run to generate all the related random variables in the simulator, so that each run should behave differently if it is in a stable state.

T_{SIFS} (μs)	16
σ (μs)	9
T_{DIFS} (μs)	34
T_{hdr}^{phy} (μs)	20
CW_{min}	15
L_{FCS} $(bits)$	32
δ (μs)	1
T_{sym} (μs)	4
R (Mbps)	6

Table 2.1: MAC/PHY parameters used for the DCF scheme, corresponding to IEEE 802.11a



(a) (b)

Figure 2.3: (a): Comparison between the DCF Model, NS simulations and related models with $BER = 10^{-5}$. (b): The DCF Model vs. simulations as the frame size is varied. 50 stations and a variety of BER values. MAC/PHY Parameters as in Table 2.1.

the number of stations is varied from 5 to 80. As shown in Fig. 2.3(a), the DCF Model is in good agreement with the simulation results while the models in [20] and [120], where channel errors are ignored, yield less accurate predictions.

In Fig. 2.3(b), we illustrate the impact of channel errors on throughput as the frame size is varied. It can be seen that the DCF model is accurate for low and medium channel BERs (i.e. $BER = 10^{-6}$ and $BER = 10^{-5}$), and slightly overestimates the throughput on very noisy channels (i.e. when $BER = 10^{-4}$). Even for the latter case however, the difference between the model and the simulation is less than 1%, which means that the DCF Model is accurate enough to predict performance in error prone channels. An interesting observation from this figure is that a larger frame size results in a higher throughput when the channel

BER is low. However, when the channel BER is higher, increasing the frame size decreases the throughput, and thus the optimal frame size depends on the channel conditions. This issue will be further addressed in Chapter 3.

2.2 Throughput Limitations of DCF

Transmission of a frame inevitably carries an overhead, which we can consider as additional time T_{oh}^{p}. In the DCF the overhead includes the time T_{hdr}^{phy} required to transmit the PHY header, the time T_{hdr}^{mac} to transmit the MAC header, the CSMA/CA backoff time T_{CW}, and the time T_{ack} to transmit a MAC ACK.

In order to clarify the impact of the overhead, we define the MAC throughput efficiency as

$$\eta = \frac{T_p}{T_p + T_{oh}^{p}} \tag{2.32}$$

where T_p is the time required to physically transmit a packet (i.e., the frame payload), and $T_{oh}^{p} = T_{hdr}^{phy} + T_{hdr}^{mac} + T_{CW} + T_{ack}$ as explained above. As the PHY rate R increases, for a fixed packet size L_p the time $T_p = L_p/R$ to transmit the packet decreases. If T_{oh}^{p} does not also decrease then the efficiency $\eta \to 0$ as $R \to \infty$.

As the PHY rate increases, the contention time T_{CW} does not decrease towards zero due to the constraints placed on the minimum slot size by clock synchronisation requirements and on DIFS by the need for backward compatibility. Similarly, the duration of the PHY header is not expected to decrease with increasing PHY rate owing to backward compatibility and PHY-layer channel equalisation requirements [6]. Thus as the PHY rate is increased, the time to transmit a frame quickly becomes dominated by the fixed overhead associated with the PHY header, contention time etc.

In Fig. 2.4 it can be seen that even with the very high physical rates which are likely to be supported in future 802.11n networks, throughput never exceeds 50 Mbps with the DCF. In order to achieve 100 Mbps therefore the MAC layer has to be modified.

Some proposals for improving the DCF have been targeted at minimising the contention

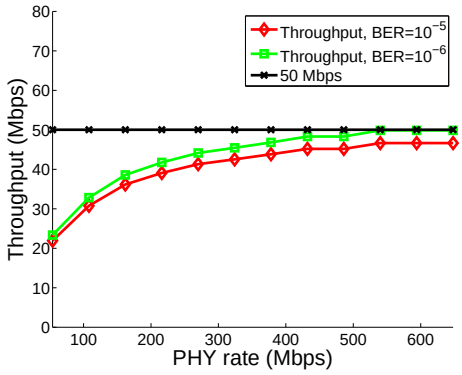


Figure 2.4: Throughput of the DCF scheme with IEEE 802.11n's physical rates [6]. The number of stations is 10, and the frame size used is 1024 bytes.

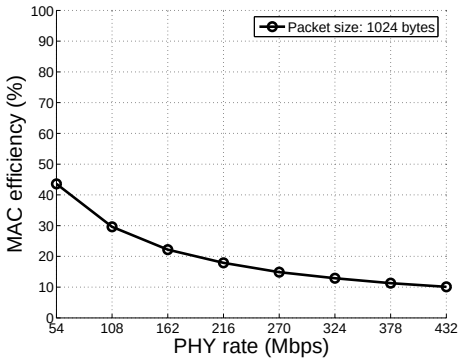


Figure 2.5: DCF efficiency vs PHY rate under ideal conditions (no collisions nor channel errors) with a 1024-byte frame size. The y-axis represents the ratio of the MAC throughput to the PHY rate.

time component T_{CW} of the overhead by regulating the random backoff process (see e.g., [30] [125] [80]) to reduce the number of collisions and idle slots. However, in very high-speed networks, the MAC throughput efficiency is still low even when the contention window is minimised. For example, we illustrate in Fig. 2.5 the efficiency in the ideal case where the channel is perfect with neither collisions nor errors [122], hence the overhead of the backoff process is minimised. It can be seen that the efficiency decreases dramatically as the PHY rate increases. In a 216 Mbps WLAN, the efficiency is only about 20%. When PHY data-rate increases to 432 Mbps, the efficiency decreases to around 10%.

The Burst ACK (e.g., [110] [93] and [115]) and Block ACK (e.g., [5], [122]) schemes have been proposed in the literature for improving the MAC efficiency. Burst ACK performs the

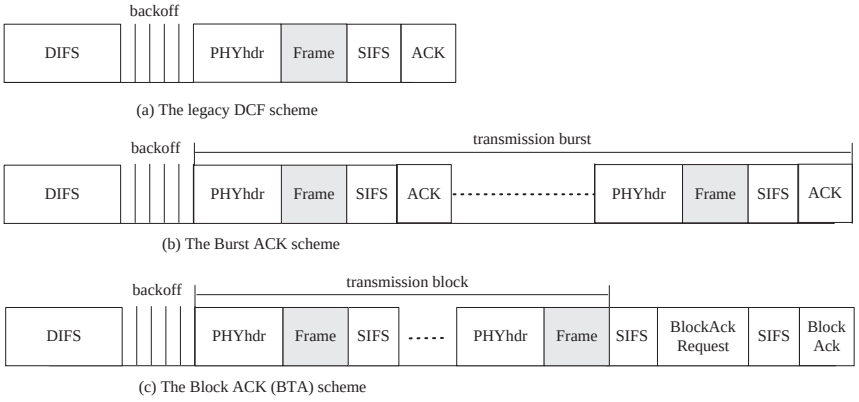


Figure 2.6: Schematic of the DCF, Burst ACK and Block ACK schemes.

backoff process once for a series of data and ACK frames (See Fig. 3.2 for details), while Block ACK goes one step further by using a single ACK frame for multiple data frames (Fig. 3.2), thus reducing the number of ACKs and SIFS.

In both schemes, the backoff time T_{CW} is incurred once for M frame transmissions, where M is the size of a burst/block. With Burst ACK, the per packet overhead is approximately $T_{oh}^{p} = T_{hdr}^{phy} + T_{hdr}^{mac} + T_{CW}/M + T_{ack}$, while for Block ACK it is $T_{oh}^{p} = T_{hdr}^{phy} + T_{hdr}^{mac} + T_{CW}/M + T_{ack}/M$. It can be seen that the contention overhead T_{CW} and MAC ACK overhead T_{ack} are amortised over multiple packets by these two schemes, thereby improving efficiency.

It can be seen that the Block ACK scheme is more promising in terms of reduced overhead. We thus focus on this scheme in the next section and develop an analytical model to investigate whether it can achieve 100 Mbps at the MAC layer as required by future 802.11n WLANs.

2.3 The Block ACK Scheme

2.3.1 Protocol Details

The Block ACK scheme is proposed in the IEEE 802.11e specification [5]. In the BTA scheme, on winning a transmission opportunity, multiple frames are transmitted to the same destination without being acknowledged, with frames sent back-to-back separated by

a SIFS period. After the last frame is sent, a block acknowledge request (BAR) frame is transmitted by the sender to enquire which frames have been received successfully, and then in response a block acknowledge (BA) frame is sent back by the receiver, see Fig. 3.2(c).

Upon receiving the BA frame correctly, the sender defers a DIFS interval and then starts a new backoff process. Other stations wait until the end of the BA transmission, and defer for DIFS before resuming backoff.

If two or more stations start transmissions in the same slot, a collision occurs. In this case, each station sends out a whole block and a BAR frame, and then waits for the BA frame. The receivers do not send back the BA frames if they detect the collision. When the senders fail to receive the BA frame, they infer that a collision has occurred, double their contention window sizes and start a new backoff process as usual.

In the case of frame errors occur due to channel noise, the sender sends out a whole block and a BAR frame. The receiver then sends back a BA frame to indicate which frames are successfully received. If the sender receives the BA frame successfully, the correctly received frames in the block are removed from the sending queue and a new block is constructed for the next round of transmission.

Here, we assume that stations are saturated so that there are always enough frames to be transmitted in a block. We note that in non-saturated situations, collisions may involve simultaneous transmissions with different block sizes and so some frames may still be received successfully. We do not consider this non-saturation scenario here.

Frame Formats

Fig. 2.7(a) shows the format of a BAR frame. There are two new fields in the BAR frame. The *BAR control* field is shown in Fig. 2.7(b). This field is used for quality-of-service negotiation between the MAC and its upper layer. The *Block ACK Starting Sequence Control* field is shown in Fig. 2.7(c). The last 12 bits of this field are used to record the first frame's sequence number in a block, the first 4 bits are reserved for future usage.

To inform the sender which frames have been lost in a block, a *Block ACK Bitmap* field

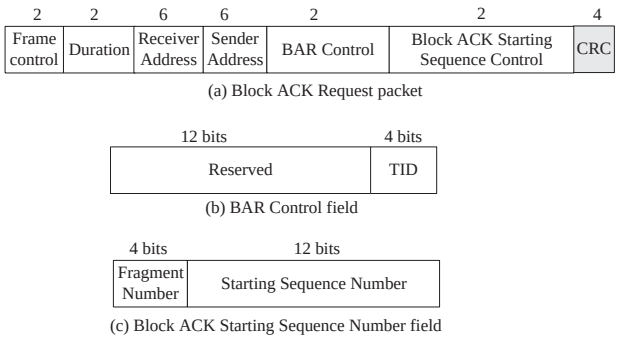


Figure 2.7: Format of the block acknowledge request (BAR) frame. (a) The Block ACK Request packet, (b) The BAR Control field, (c) The Block ACK Starting Sequence Number field.

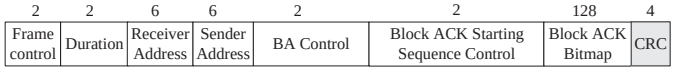


Figure 2.8: Format of the block acknowledge (BA) frame.

is designed in the BA frame as illustrated in Fig. 2.8. It is a 128-byte field, and thus it can support up to $128 \times 8 = 1024$ frames in a single block. The *Block ACK Starting Sequence Control* field is used to indicate to which BAR this BA frame responds.

Discussion

- BTA differ from previous schemes in the following ways. Firstly, the unit of transmission in the BTA scheme is a block, which consists of multiple data frames and one ACK. The unit of transmission in the DCF is a single data frame and an ACK frame, and in the Burst ACK scheme it is a burst which contains multiple data frames and multiple ACKs. Therefore, the BTA scheme is expected to be more efficient if collisions are rare. Secondly, in previous schemes, a data frame is acknowledged immediately by an ACK. In BTA, however, a modified sending queue and receiving queue are required to accommodate block transmissions.

- In the case of a collision in BTA, a whole block will be transmitted (and lost), thus reducing efficiency. To improve efficiency, IEEE 802.11e [5] proposes that a similar method to Request-To-Send (RTS)/Clear-To-Send(CTS) is used, i.e., before each

block transmission, the sender transmits an *Add Block ACK Request* frame to the receiver which should respond with an ACK. The receiver then sends an *Add Block ACK Response* frame to the sender. Note that use of RTS/CTS inevitably incurs an additional overhead which is not desired. The use of this technique is thus expected to be effective only if block sizes are large and at the same time collisions happen sufficiently frequently.

- BTA can be used to achieve so-called time based fairness [47] [98] in multi-rate networks. If the channel states are good for some stations and bad for others, however, BTA can be used by faster stations so that more frames are sent back-to-back in a given period of time, thereby improving system efficiency.

- Finally, it can be seen that from a modelling viewpoint the BTA scheme operates in a similar way to the DCF scheme. In particular, we may treat a block in the BTA scheme in a similar way to a frame in the DCF because both are considered as the unit of operation. This understanding suggests that it is possible to extend the previous analysis to model the BTA scheme.

2.3.2 The BTA Model

In this section, we present an analytical model to compute the saturation throughput for the BTA scheme in error channels.

We consider a network where all stations can hear each other. Collisions occur when at least two stations start transmissions at the same time. A transmission error occurs when frames are corrupted due to reasons other than collisions. We assume that the PHY headers are always transmitted successfully given the fact that they are usually transmitted at the basic, hence the safest, rate [1]. We also assume that the transmissions of the BAR and BA frames are always successful due to the same reason.

We first compute the expected slot duration $E[T]$. There are four types of durations in the BTA scheme as shown in Fig. 2.9.

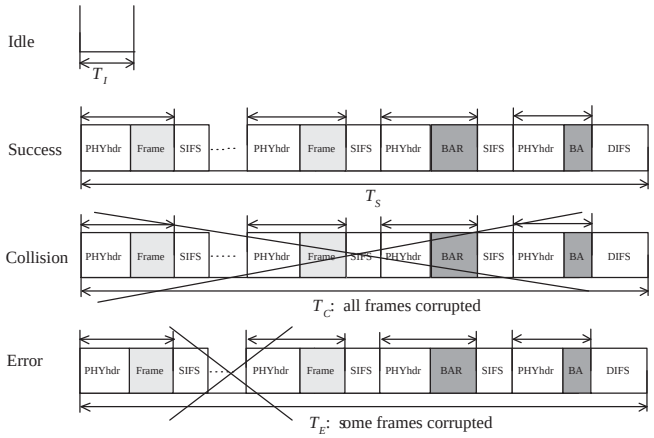


Figure 2.9: Slot durations in the BTA scheme

- If none of the stations transmit any frames, they all wait for a duration $T_I = \sigma$, where σ corresponds to a physical slot.

- T_S denotes the duration of a slot during which a whole block is transmitted successfully. In this case, only one station transmits frames and its transmissions are successful. The channel state is busy for the duration of a block of frame transmissions plus $(N_b - 1)$ SIFSs, a BAR and a BA transmission, where N_b denotes the block size.

- Let T_E be the duration of a slot in which at least one frame in a block is corrupted due to channel errors. The sender does not stop the transmission and the receiver responds with a BA frame. The other stations defer a block and a DIFS duration.

- Let T_C denote the duration of a slot in which at least two stations start transmissions simultaneously and so a collision occurs. In this case, no BA frames are sent by the receivers.

For brevity, we assume that channel errors do not occur on ACK frames. The slot

durations can then be expressed as

$$\begin{aligned} T_I &= \sigma \\ T_S &= N_b \cdot (T_f + T_{SIFS}) + T_{DIFS} + (T_{bar} + T_{SIFS} + T_{ba}) + (N_b + 2)(T_{hdr}^{phy} + \delta) \\ T_E &= T_S \\ T_C &= N_b \cdot (T_f + T_{SIFS}) + T_{EIFS} + (T_{bar} + T_{SIFS} + T_{ba}) + \\ &\quad (N_b + 1)(T_{hdr}^{phy} + \delta) \end{aligned} \tag{2.33}$$

where τ is calculated as per the DCF Model.

As before, the probabilities of each type of slot are

$$P_I = (1 - \tau)^n \tag{2.34}$$

$$P_S = n \cdot (\tau(1 - \tau)^{n-1}) \cdot (1 - p_e^{bta}) \tag{2.35}$$

$$P_E = n \cdot (\tau(1 - \tau)^{n-1}) \cdot p_e^{bta} \tag{2.36}$$

$$P_C = 1 - P_I - P_S - P_E. \tag{2.37}$$

Letting p_e be the error rate of a frame, the probability p_e^{bta} can be expressed as

$$p_e^{bta} = 1 - (1 - p_e)^{N_b}, \tag{2.38}$$

where the p_e can be computed if the bit error distribution is given.

The saturation throughput S_{BTA} can then be expressed as

$$S_{BTA} = \frac{P_S \cdot N_b \cdot L_p + P_E \cdot E[L]}{P_I T_I + P_S T_S + P_E T_E + P_C T_C}, \tag{2.39}$$

where $E[L]$ is the expected frame size successfully transmitted in an erroneous case. Let i denote the number of the corrupted frames. Assuming Gaussian channel errors, $E[L]$ is

$$E[L] = \sum_{i=1}^{N_b} \binom{N_b}{i} \cdot (p_e)^i \cdot (1 - p_e)^{N_b - i} \cdot (N_b - i) \cdot L_p. \tag{2.40}$$

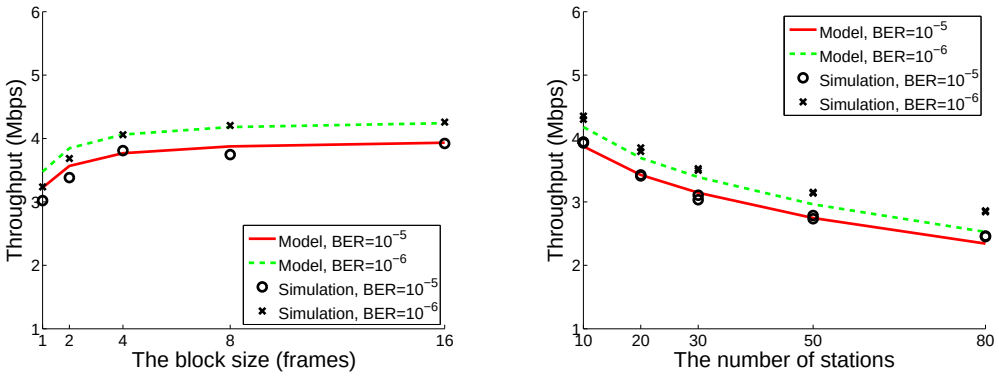
T_{SIFS} (μ s)	16
σ (μs)	9
T_{DIFS} (μs)	34
T_{hdr}^{phy} (μs)	20
CW_{min}	15
L_{FCS} (bits)	32
δ (μs)	1
T_{sym} (μs)	4
Retry limit	4
Frame size (bytes)	1024

Table 2.2: MAC/PHY parameters used for the Block ACK (BTA) scheme, corresponding to IEEE 802.11a.

2.3.3 Validation

We implemented the BTA scheme in the network simulator *NS-2* [12] to validate our analytical model. The simulation parameters are listed in Table 2.2.

In our implementation a *bitmap array*, a sending queue (Sq) and a receiving queue (Rq) are used. The *bitmap array* is for recording the number of frames that have been transmitted successfully. The Sq and the Rq are used to save frames temporarily at the MAC layer. For convenience, let h_{Sq}, t_{Sq}, h_{Rq}, and t_{Rq} denote the head of the Sq, the tail of the Sq, the head of the Rq, and the tail of the Rq, respectively. The sender stores a frame from the upper layer at the t_{Sq}, and checks whether N_b (the block size) frames have been transmitted. If so, it constructs a BAR frame at the MAC layer and transmits it. Otherwise, the first frame at the h_{Sq} is popped out and transmitted. On reception of a data frame f_j, the receiver checks its correctness and updates accordingly the *bitmap array* whose length is equal to N_b. Then f_j is appended at the t_{Rq} if it has not been received before. If f_j has been in the Rq but marked as 'corrupted', the receiver updates its flag. Upon receiving a BAR frame, the receiver responds with a BA frame containing the *bitmap array*. Then the *bitmap array* is reset for the next round of receiving, and all the correctly received in-order frames in the Rq are transferred to the upper layer. After receiving a BA frame, the sender removes all the frames that have been received successfully from the Sq. The contention window size is reset for both successful and erroneous transmissions. In the case of collisions, receivers



(a) (b)

Figure 2.10: Model validation. In (a), the number of stations is 10. In (b), the block size used is 8 frames. In both cases, the physical rate used is 6 Mbps. Other parameters are listed in Table 2.2.

do not transmit the BA frames. Then after a transmission block, a sender waits until the BAR timeout and retransmits the entire block.

We validate the BTA model in two ways. Firstly, a 10-station WLAN is simulated, in which each station has a PHY rate of 6Mbps and a UDP traffic rate of 6Mbps. BAR and BA frames are also transmitted at 6Mbps. Other parameters are listed in Table 2.2. We plot the results from the simulation and the model as the block size is increased from 1 to 16 frames. The results are plotted in Fig. 2.10(a), where it can be seen that the model predictions agree well with simulation results.

Secondly, we fix the block size and increase the number of stations. The corresponding results are plotted in Fig. 2.10(b). Again, it can be seen that the model remains accurate as the channel becomes highly loaded.

2.3.4 Choice of BTA Block Size

An important question for BTA is how to choose the block size to maximise the channel efficiency. In Fig. 2.11, we plot the performance of BTA for a range of PHY rates as the block size is increased from 1 to 64 frames. We observe that a block size of 16 frame is sufficient to achieve close to maximum efficiency over a wide range of PHY rates, with larger

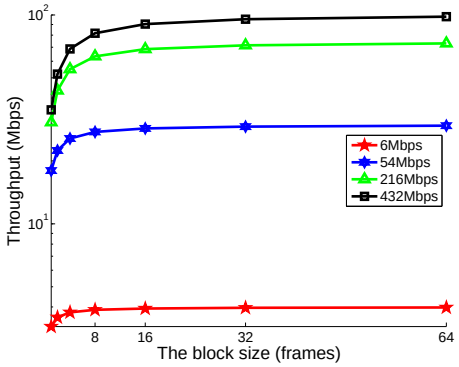


Figure 2.11: Impact of the block size on the performance achieved by BTA. 10 stations, BER is 10^{-5}, frame size used is 1024 bytes. Other parameters are listed in Table 2.2.

block sizes yielding only minor further improvements. We therefore use a block size of 16 frames hereafter.

2.3.5 Comparison with DCF

In Fig. 2.12(a) we compare the performance of the DCF and BTA schemes for increasing number of stations (and thus traffic load). It can be seen that the BTA scheme consistently achieves higher throughput than the DCF. However, the efficiency rapidly falls with increasing number of stations.

Fig. 2.12(b) shows the throughput of the BTA scheme over the range of PHY rates proposed in IEEE 802.11n [6]. It can be seen that the throughput increases with the PHY rate, but does not reach 100 Mbps until the PHY rate is faster than 500 Mbps.

2.4 Summary

In this chapter, we have developed a new analytical model for the DCF scheme that includes the effects of channel noise. Using this model, we demonstrate that the DCF scheme can not achieve a MAC layer throughput of 100 Mbps, even for very high PHY rates.

We then consider the BTA scheme of 802.11e. Our results show that the BTA scheme can achieve 100 Mbps at the MAC layer. However, it can only do so if the underlying physical

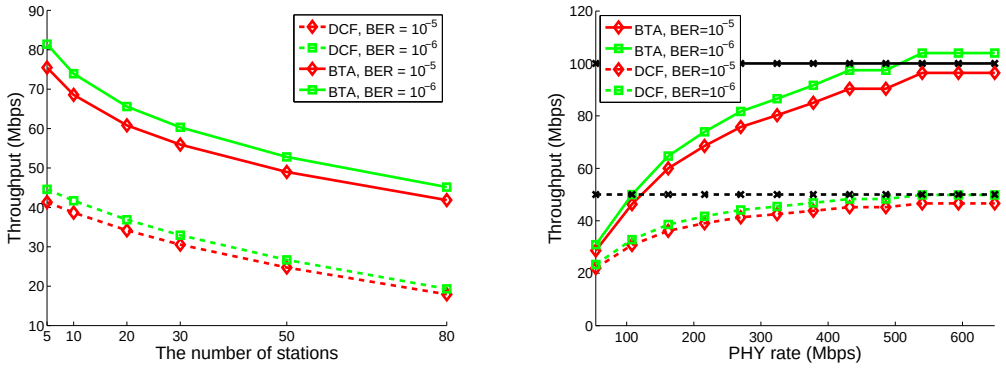


Figure 2.12: BTA vs DCF. (a) BTA vs DCF Model as the number of stations is increased. The block size is 16 frames, the physical rate is 216 Mbps. (b) Throughput of both schemes with IEEE 802.11n's physical rates [6]. The frame size used is 1024 bytes, the number of stations is 10, and the block size is 16 frames. Other parameters are listed in Table 2.2.

rates used are higher than 500 Mbps, and more than 80% of the capability provided by the physical layer is wasted by the MAC layer.

CHAPTER 3

A New MAC For Very High-Speed WLANs

3.1 Introduction

Wireless LANs based on 802.11 technology are becoming increasingly ubiquitous. With the aim of supporting rich multimedia applications such as HDTV (20Mbps) and DVD (9.8Mbps), the technology trend is towards increasingly higher bandwidths. Some recent 802.11n proposals seek to support PHY rates of up to 600 Mbps ([6, 8, 9, 124]). However, higher PHY rates do not necessarily translate into corresponding increases in MAC layer throughput as we have seen in the last chapter.

The problem here is a fundamental one for MAC design, namely that due to cross-layer interactions the throughput of the current 802.11 MAC does not scale well with increasing PHY rates. With continuing improvements in PHY technology and demand for higher throughput, the MAC scaling behaviour is of key importance.

While the current focus of 802.11n activity is on achieving 100Mbps throughput at the MAC layer, still higher target data rates can be expected in the future. To avoid repeated MAC redesigns, one basic question that we seek to answer is whether it is feasible to extend the 802.11 MAC to maintain high throughput efficiency regardless of PHY rates. We answer this in the affirmative. In particular, we identify fundamental properties that must be satisfied by any CSMA/CA based MAC layer and develop a novel scheme called

Aggregation with Fragment Retransmission (AFR) that exhibits these properties. In the AFR scheme, multiple packets are aggregated into and transmitted in a single large frame[1]. If errors occur during the transmission, only the corrupted fragments of the frame are retransmitted. In this scheme, a new delimitation mechanism allows for higher throughput with less overhead compared to previous designs. We study a fragmentation technique where packets longer than a threshold are divided into fragments before being aggregated. An analytic model is developed to evaluate the throughput and delay of AFR in noisy channels, and to compare AFR with competing schemes. Optimal frame and fragment sizes are calculated using this model, and an algorithm for dividing packets into near-optimal fragments is designed.

A second question we seek to answer is whether higher transmission delays are an unavoidable result of using aggregation to achieve high throughput. In particular, is additional delay necessarily introduced (i) by the need to wait until sufficient packets arrive to allow a large frame to be formed and (ii) for transmission of a large frame? We answer this question in the negative. Specifically, we propose a zero-waiting mechanism where frames are transmitted immediately once the MAC wins a transmission opportunity. In the zero-waiting aggregation scheme, the frame sizes adapt automatically to the PHY rate and channel state, thereby maximising the throughput efficiency while minimising the holding delay.

Thirdly, we investigate by simulations the impact of AFR on the performance of realistic applications with diverse demands – for this we followed the 802.11n usage model [10]. We implemented the AFR scheme in the network simulator *NS-2* and present detailed results for TCP, VoIP and HDTV traffic. Results suggest that AFR is a promising MAC technique for very high-speed WLANs. Moreover, AFR is particularly effective for rich multimedia services with high data rates and large packet sizes, which are key applications in future WLANs.

The remainder of this chapter is organized as follows. Section 3.2 details the motivation

[1]Recall that we define a *packet* as what MAC receives from the upper layer, a *frame* as what MAC transfers to the PHY layer, and a *fragment* as part(s) of a frame.

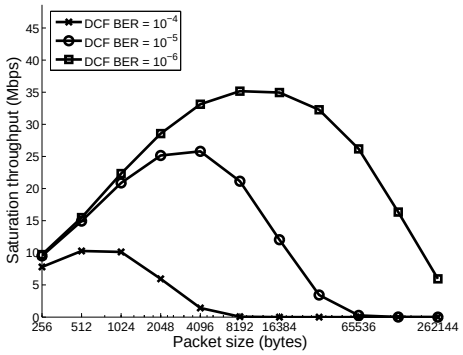
of this work. We identify in Section 3.3 the fundamental properties that must be satisfied by any aggregation scheme, and introduce in Section 3.4 the AFR scheme. A theoretical analysis is given in Section 3.5 while Section 3.6 presents detailed simulation results. Finally we summarise our conclusions in Section 3.8.

3.2 Motivation

As we have seen in the last chapter, the Burst ACK and the Block ACK schemes have been proposed to amortise the contention window overhead T_{CW} and MAC ACK overhead T_{ack} over multiple packets, therefore improving efficiency. However, the per packet PHY header overhead T_{hdr}^{phy} and the MAC header overhead T_{hdr}^{mac} are left untouched. According to the proposal 802.11n [6] for the future WLANs, it is likely to take at least $44\mu s$ to transmit a PHY header (and $48\mu s$ when two antenna radios are used [6]). For comparison, the transmission duration of a 1024-byte frame at a PHY rate of 216Mbs is $40\mu s$, and at 432Mbs is $20\mu s$. As the PHY rate is increased, the time to transmit a frame quickly becomes dominated by PHY headers, the throughput efficiency rapidly decreases and efforts to increase the system capacity purely by increasing the data rate are thus of limited effectiveness even when Burst ACK or Block ACK are employed.

Aggregation schemes seek to amortise the PHY header overhead across multiple packets. This is achieved by transmitting multiple packets in a single large frame. However, there is a traditional dislike for transmitting large frames in wireless networks since in a noisy channel (e.g., $BER \geq 10^{-5}$), the throughput can fall as larger frames are used [66]. We illustrate this in Fig. 3.1(a). However, we note that in traditional retransmission schemes a whole frame is retransmitted even if only one bit is lost. This raises the question of whether it is possible to retransmit only the erroneous part(s) of a frame – if properly designed, such partial retransmission could be expected to improve performance. This is a key motivation of the work presented here.

Although this idea seems simple at first glance, it is actually a radical challenge for



(a) DCF throughput

T_{SIFS} (μs)	16
σ (μs)	9
T_{DIFS} (μs)	34
T_{hdr}^{phy} (μs)	20
CW_{min}	15
δ (μs)	1
T_{sym} (μs)	4
Retry limit	4

(b) Parameters

Figure 3.1: (a) DCF throughput vs packet size and channel BER, 10 saturated stations and 54 Mbps PHY rate. (b) MAC and PHY parameters used.

PHY and MAC technology. From the PHY viewpoint, the traditional small-packet rule does not hold any more. The PHY layer has to transmit very large frames, and has to continue decoding even if the BER exceeds some previously unacceptable value. Under these conditions, the size of the largest practical frame is still unknown [6]. From the MAC viewpoint, any retransmission scheme carries an associated signalling overhead and hence a trade-off exists between system efficiency and the granularity of retransmission. Moreover, since real traffic is typically bursty/on-off in nature, this raises questions as to the optimal policy for aggregating packets into frames, for example how much time should the MAC wait for sufficient packets to arrive to form a large frame.

In parallel with our work, there have been other activities in the 802.11n standard working group on this topic (e.g., [6, 8, 9]). These support similar functionalities to our scheme, with a special *delimiter* for locating each fragment in a frame. Other related work includes that of [51] where an aggregation technique is used to solve an unfairness problem in WLANs by removing the DIFS, SIFS and backoffs before a series of packets, and transmitting the packets together in a large PHY layer frame. However, a small PHY header ($12\mu s$) is used to identify each packet within a frame. In [55], a two-level (one at MAC, another at PHY) aggregation scheme is proposed that uses a similar *delimiter* to that in the TGn Sync proposal [6].

Although aggregation is not a new idea, many fundamental questions remain open:

- How do we aggregate packets? The frames we want are larger than a typical packet. If the packets from the upper layer are large and arrive rapidly, then aggregation is simple. If not, should a timing mechanism be used to wait for sufficient packets to arrive to form a large frame? If so, how much time do we wait to maximise throughput while minimising delay?

- What is an appropriate (re)transmission unit? Should very large packets be divided for retransmission?

- A suitable analysis of aggregation throughput and delay performance is missing.

- How does packet aggregation impact real world traffic, e.g. voice, video and TCP traffic.

We address these open questions in this chapter.

3.3 Fundamental Considerations

We highlight in this section the basic requirements that must be respected by any aggregation scheme that seeks to maintain high MAC efficiency as PHY rates increase, and introduce the zero-waiting approach to aggregation.

3.3.1 MAC efficiency

The basic requirement for high efficiency is to aggregate packets into large frames so as to spread the cost of fixed overhead across multiple packets. To reduce the overhead associated with transmission errors, each frame is sub-divided into fragments, with packets that exceed the fragment size being divided. Fragments are the unit used in the retransmission logic, i.e., damaged fragments are retransmitted rather than the entire frame.

The time to transmit a packet is $T_p = L_p/R$, where L_p is the packet size and R is the PHY rate. Hence, the per packet MAC efficiency is

$$\eta_p = \frac{T_p}{T_p + T_{oh}^p} = \frac{L_p/R}{L_p/R + T_{oh}^p}, \tag{3.1}$$

where T_{oh}^p is the overhead of transmitting a packet.

We now show that under certain assumptions, it is indeed possible to maintain a constant MAC efficiency while R is increased. That is, we may decouple MAC efficiency from the PHY rate R.

We can see that $T_p = L_p/R$ scales with $1/R$. In order to maintain a constant η_p, we also require that the per packet overhead T_{oh}^p scales with $1/R$. Considering T_{oh}^p in more detail, we can typically decompose it into the following elements (where r denotes the average number of transmissions before all fragments from this packet are transmitted successfully and other notation is listed in Table 1):

$$T_{oh}^p = \frac{(T_{hdr}^{phy} + T_{hdr}^{mac} + T_{hdr}^{frag} + T_{CW} + T_{ack}) \cdot r}{M} \tag{3.2}$$

To ensure that T_{oh}^p scales with $1/R$, we require that:

- The number of packets M in a frame should be proportional to R, that is $M = bR$ for some constant b. This ensures that the overheads T_{hdr}^{phy}, T_{hdr}^{mac}, T_{ack} and T_{CW} translate into a per packet overhead that scales with R.

- Since there is only one MAC header and one ACK per frame, when M is proportional to R there is no fundamental constraint on the rate at which MAC headers and ACK frames are transmitted. The same is not true for fragment headers.

- For a given fragment size L_{frag}, the number of fragments in a frame m increases with the number of packets M in a frame. I.e., $m = m'M$ where m' is the number of fragments per packet, we thus have $m = m'bR$ when $M = bR$. Hence, for T_{hdr}^{frag}/M to scale with $1/R$ the rate at which fragment headers are transmitted must be chosen to be proportional (see *comment 1* below) to R, in which case $T_{hdr}^{frag}/M = mL_1/R = m'L_1/R$.

- The retransmission times r is constant. For a given packet size the number r is determined by the collision probability and the BER. In our analysis where the number of stations is fixed and the incoming traffic is known, the collision probability is constant. The BER itself depends upon the channel signal to noise ratio and the choice of coding. A rate controller is typically used to adjust the coding and rate R to maintain the BER below a target level [83] [84], reflecting application and transport layer requirements[2]. In the following we assume the use of a rate controller and thus that rate is adjusted to ensure that the average number of retransmissions r remains approximately constant. We also note that if the BER is not regulated via rate control, then provided r is bounded or is a known function of rate R then the scaling analysis can be extended to include this situation.

When the per packet overhead satisfies these conditions, the per packet MAC efficiency is

$$\eta_p = \frac{L_p}{L_p + r(a/b + m' \cdot L_1)} \tag{3.3}$$

where L_1 denotes the size of one fragment header, $a = T_{hdr}^{phy} + T_{hdr}^{mac} + T_{CW} + T_{ack}$ and $b = M/R$.

Firstly, observe that the efficiency is nicely decoupled from the PHY rate R, i.e., the throughput scales with R. Secondly, as we increase the factor b, we can see that the efficiency asymptotically tends to

$$\tilde{\eta_p} \quad = \quad \frac{L_p}{L_p + r \cdot m' \cdot L_1} = \frac{1}{1+d} \tag{3.4}$$

where $d = (rm'L_1)/L_p$.

That is, the efficiency is fundamentally limited by the number of fragments per packet m' and the number of retransmissions r. In particular, if we use a large fragment size, corresponding to a small m', such large fragments are more likely to be corrupted and we

[2]For example, since TCP congestion control views packet losses as an indicator of congestion, TCP throughput is strongly dependent on the link loss rate (e.g., [24] [25]) and too high a loss rate may then prevent high utilisation of the wireless channel.

have therefore small m' and large r. On the other hand, when a packet is divided into many small fragments, corresponding to use of a large m', the probability of a fragment being corrupted is low and we have large m' but small r. To achieve high efficiency, we study in Section 3.5.4 a fragmentation technique where packets with sizes exceeding a threshold are divided into fragments to deal with the tradeoff between m' and r.

Comment 1: At high rates in a noisy channel the question of the impact of errors in the received fragment headers arises. First, we only require that the rate used for sending the fragment headers is proportional to the data rate R. Thus, to protect the fragment headers they may be sent at a relatively low rate (e.g. at half, or less, of the data rate) and in this way we can ensure that the majority of bit errors affect the data payload only. Second, fragment header size (8 bytes in AFR, see Section 3.4.2) is minimised to ensure low error probabilities. Third, in the frame we collect the fragment headers together with the MAC header (details in Section 3.4.2) so that FEC can be more easily employed to enhance robustness.

3.3.2 Zero-waiting

In order to aggregate packets into large frames with minimum delays, we note that when the channel is lightly loaded to the extent that DCF can service the offered load, deliberate waiting only leads to higher delays. If the channel is in a heavily loaded condition where backlogged buffers mean that sufficient numbers of packets are always available, then all waiting schemes are equivalent. If the channel is in an intermediate situation between these two extremes, waiting for a certain amount of time for packets to accumulate seems reasonable at first glance. Nevertheless, we argue that fundamentally there is no need to wait for packets to accumulate at the MAC layer and it is sufficient instead to simply start a transmission whenever the MAC wins a transmission opportunity. This zero-waiting mechanism evidently performs well in both lightly and heavily loaded situations. At intermediate

traffic load[3], the frame size used adapts to the minimum required to service the offered load. Specifically, when the current level of efficiency is too low for the offered load, a queue backlog will develop which in turn induces larger frames and increased efficiency. If the incoming traffic subsides, smaller frame sizes will be automatically selected. Evidently, such a policy minimises the holding delay at the MAC layer.

We now show that this opportunistic aggregation policy can also lead to the desired efficiency $\tilde{\eta_p}$ where it is feasible to do so. Assuming that there are no collisions and errors in the network[4], corresponding to $r = 1$, we can write the per frame MAC efficiency as

$$\eta_f = \frac{T_f}{T_f + a + d \cdot T_f} = \frac{1}{1 + d + a/T_f}. \tag{3.5}$$

Write the desired throughput as $\tilde{S} = R/(1+d)$. Let the mean arrival rate of the offered load be $\nu = \alpha\tilde{S} = \alpha R/(1+d)$ bits per second where $0 \leq \alpha \leq 1$ is a real valued factor. During the time $T_f + a + d \cdot T_f$ to transmit a frame, on average we expect $\nu \cdot (T_f + a + d \cdot T_f)$ arrivals at the queue. Selecting the frame size to be the same as queue size $q(k)$, we have that,

$$\begin{aligned} E[q(k+1)] &= \nu \cdot [T_f + a + d \cdot T_f] \\ &= \nu \cdot [(1+d)E[q(k)]/R + a] \\ &= \alpha \cdot E[q(k)] + \frac{\alpha \cdot a \cdot R}{1+d}. \end{aligned} \tag{3.6}$$

These queue dynamics can be written as

$$E[q(k+t)] = \alpha^t E[q(k)] + \sum_{i=1}^{t} \alpha^{i-1} \cdot \frac{\alpha \cdot a \cdot R}{1+d}.$$

Hence, provided $\alpha < 1$ then as $t \to \infty$, we have that the queue dynamics are stable. Asymptotically, we have that,

$$E[L_f] = E[q] = \frac{\alpha \cdot a \cdot R}{(1-\alpha)(1+d)}. \tag{3.7}$$

[3]We note that this simple zero waiting scheme is also attractive from a practical point of view as it has been observed that real world traffic can exhibit complex bursty behaviour [82] [106] that is expected to make the effective design of a more complex waiting scheme difficult.

[4]Proof for more complicated cases is left as further work.

Combining Equation (3.5) and (3.7), we have that

$$\eta_f = \frac{\alpha}{1+d} = \alpha \cdot \tilde{\eta_p}.$$

As $\alpha \to 1$, we can see that the zero-waiting policy achieves the desired efficiency.

From equation (3.7), we can see two important features of zero-waiting. First, when the offered load is light (i.e., α is small) small frames will be used. As the load increases, larger frame sizes will be automatically selected. Thus, zero-waiting elegantly creates a feedback loop whereby MAC efficiency is regulated based on queue backlogs as expected. Second, for a given level of load α, the frame size L_f scales with R. Therefore, with a multi-rate enabled wireless card, the frame size also adapts automatically to the changing PHY rate R.

3.4 The AFR Scheme

In this section, we describe in detail the AFR scheme based on the insight provided by the foregoing analysis.

3.4.1 Scheme Description

The basic idea of the AFR scheme is to aggregate packets from the upper layer into large frames. Packets that exceed the fragmentation threshold are segmented into fragments. The MAC layer then transmits the large frame and only retransmits corrupted fragments, i.e., those that can not pass their Frame Check Sequence (FCS) checks by the receiver. An example of the AFR scheme is shown in Fig. 3.2.

At the sender, an outgoing packet is segmented if it is longer than the fragmentation threshold. Before transmission, all the fragments are marked as 'undelivered' and kept temporarily in a MAC layer sending-queue (Sq). MAC then constructs a frame in the following way: It searches the Sq from head to tail for fragments marked as 'undelivered' and aggregates them into the sending frame until either no 'undelivered' fragments available

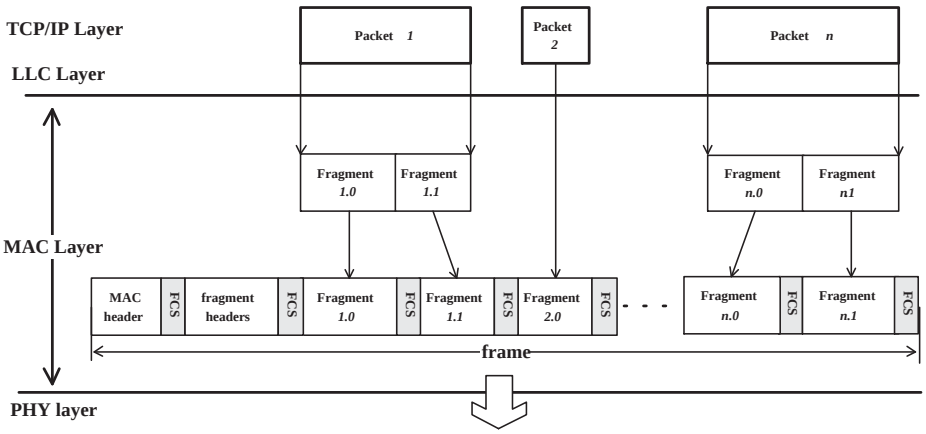


Figure 3.2: The Aggregation with Fragment Retransmission (AFR) scheme.

or the frame size is large enough. This frame (see Fig. 3.3) is then transmitted according to the DCF procedure.

Upon successfully receiving a frame, the receiver checks the FCS of each fragment, constructs and sends back the ACK frame (see Fig. 3.4) in which the lost fragments are indicated in a bitmap field. All the received fragments (no matter correct or corrupted) are stored in a receiving-queue (Rq), which is then checked by the MAC, and all the packets that have been received successfully are passed up to the upper layer and are removed from the Rq.

On receiving the ACK frame, the sender's MAC checks the ACK bitmap field and updates the Sq by marking correctly received fragments as 'delivered'. Then it removes the successfully received packets from the Sq. Afterwards, as long as the Sq is not null, MAC will construct and send out another frame immediately with the zero-waiting mechanism, i.e., without waiting for more packets even though they are not long enough.

In the cases when collisions happen, the AFR scheme runs in the same way as in the DCF scheme, i.e., the senders preform backoff.

There are two possibilities if transmission errors occur. First, the data frame may be corrupted while the ACK is successfully received. In contrast to the DCF scheme, the AFR scheme uses an ACK to notify the sender of which fragments have been lost. Therefore this

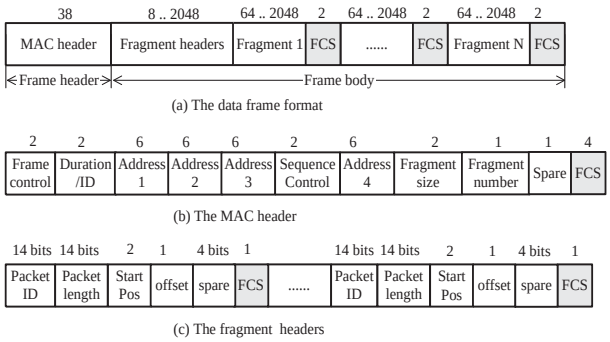


Figure 3.3: Data format in the AFR scheme.

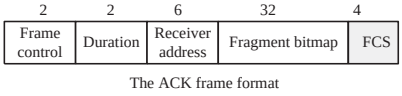


Figure 3.4: ACK format in the AFR scheme.

is treated by the AFR scheme as a successful transmission. Second, the ACK frame can be lost even if the data frame is correctly received. In this case the AFR scheme behaves in the same way as in the DCF scheme, i.e., the sender will double its contention window and try again.

3.4.2 AFR Implementation

Clearly, new data and ACK frame formats are a primary concern in developing a practical AFR scheme. Difficulties for new formats include (i) respecting the constraints on overhead noted previously and (ii) ensuring that in an erroneous transmission the receiver is able to retrieve the correctly transmitted fragments – this is not straightforward because the sizes of the corrupted fragments may be unknown to the receiver.

In our scheme, a MAC frame consists of a frame header and a frame body (Fig. 3.3(a)). In the new MAC header, the fields of the DCF MAC header remain unchanged, and we add three fields — *fragment size*, *fragment number* and a *spare* field. The *fragment size* represents the size of fragment used in the MAC frame. The *fragment number* represents the number of fragments in the current MAC frame. The *spare* field is left for future extensions

and maintaining alignment. The frame body consists of fragment-headers, fragment bodies and the corresponding FCS (See Fig. 3.3(b) and (c)).

The fragment-header section of the frame body has a variable size. It includes from 1 to 256 fragment headers, each of which is protected by a FCS. The length of each fragment header is constant (8 bytes) and known to both the sender and the receiver. Since the receiver knows where the first fragment header starts from and what the fragment header size is, it can locate all of the fragments in the frame even if some are corrupted during the transmission.

Each fragment header is composed of six fields: packet ID (*pID*), packet length (*pLEN*), *startPos*, *offset*, *spare* and *FCS*. *pID* and *pLEN* represent the corresponding ID and length of the packet P to which this fragment belongs. *startPos* is used to indicate the position of the fragment body in this frame and *offset* is used to record the position of this fragment in packet P.

The new ACK format is simple, we add a 32-byte bitmap in the legacy ACK format. Each bit of the bitmap is used to indicate the correctness of a received fragment (See Fig. 3.4).

To clarify the usage of the new formats, we give an example below. Suppose there are two packets (pkt_1 and pkt_2) with lengths of $L_{p1} = 1025$ bytes and $L_{p2} = 40$ bytes. The frame length is $L_f = 2048$ bytes and the fragment length is $L_{frag} = 512$ bytes[5]. Then AFR divides pkt_1 and pkt_2 into 3 and 1 fragments respectively and put them into the sending queue. A frame with *fragment size* of 512 bytes and *fragment number* of 4 is constructed. The corresponding fragment headers are shown in Table. 3.1. After receiving the frame, the receiver operates as shown in *Algorithm* 1 to recover the fragments.

[5]To show that AFR can support arbitrary sizes of fragmentation, we do not restrict ourselves to the fragmentation algorithm introduced in Section 3.5.4.

	packet ID	packet length	StartPos	offset
fragment 1	1	1025	0	0
fragment 2	1	1025	512	1
fragment 3	1	1025	1024	2
fragment 4	2	40	1025	0

Table 3.1: An example usage of the AFR frame formats.

Algorithm 1 : Pseudo Code of the receiver's running logic

```
1: if MAC header is correct then
2:     for i = 0 to fragment number - 1 do
3:         if Fragment i's header is correct then
4:             if packet length < fragment size then
5:                 fragment i's length = pLEN;
6:             else if offset = ⌊pLEN/fragment size⌋ then
7:                 fragment i's length = pLEN - offset * fragment size;
8:             else
9:                 fragment i's length = fragment size;
10:            end if
11:            fragment start position = startPos in the fragment header.
12:            check the correctness of the fragment body using the FCS of it.
13:        end if
14:        record correctness (including fragment header and fragment body) of the fragments in a data
           structure called the ACK bitmap.
15:    end for
16:    construct ACK frame using the ACK bitmap and send it back.
17:    update the receiving queue according to the ACK bitmap.
18:    check the receiving queue and transfer all correctly received packets upwards, and remove them from
       the receiving queue.
19: else
20:    discard this frame and defer an EIFS before next transmission.
21: end if
```

3.4.3 Comments

Frame/Fragment Size

Selection of the maximum frame size and near-optimal fragment size is discussed in detail in Section 3.5.3 and 3.5.4.

Fairness

AFR uses the same contention mechanism as the legacy DCF. Therefore the same fairness characteristics hold as in the legacy DCF. Techniques for improving the DCF's fairness may also be applied to the AFR scheme. Interested readers can refer, for example, to [26], [98] and [48].

Multiple destinations

Thus far, we have focussed only on aggregation between a single source-destination pair. This facilitates a clear understanding of the pros and cons of the aggregation itself. In order to support one-to-many aggregation, a broadcast/multicast MAC address should be used and all stations that hear the transmission then check a new receiver-list field in the MAC header which specifies the destination address for each fragment. That is, the only modification in terms of frame format is adding the receiver-list field. However, one-to-many aggregation introduces a number of new issues which we mention below. Resolving these issues is beyond the scope of the present work.

Firstly, one-to-many aggregation requires consideration of new ACKing techniques to avoid collisions between ACK transmissions by the multiple receivers. This might be achieved by sequential transmission of ACKs or perhaps by use of advanced physical layer techniques (coding, multiple antennas) to enable decoding of ACKs that are sent simultaneously (e.g., [45] [107]). The resulting performance requires detailed study and these techniques are not proposed for the future 802.11n standard [6].

Secondly, multiple antenna systems are widely considered to be of vital importance for achieving very high transmission rates [6]. The design of one-to-many aggregation for multiple antenna systems remains an open question that is likely to require tightly coupled cross-layer PHY/MAC design and operation.

Thirdly, the channel quality may differ between neighbours and it might therefore be necessary to use multiple sub-physical headers. These new headers clearly would cause extra overhead. Further, rate adaptation, which has become an indispensable functionality of 802.11 based networks, requires further work in the context of one-to-many aggregation.

Multi-rate

In current WLANs, a commonly used technique to resist channel noise is to lower the PHY rate when a high packet (or bit) error rate is measured. When the channel state improves,

the PHY layer increases its rate accordingly. There are two main issues to be addressed in order to support multi-rate operation in AFR: (i) should the frame size be adjusted with the PHY rate and (ii) should we support one-to-many aggregation when receivers have different channel states? Issue (i) is discussed in Section 3.3.2, and issue (II) is discussed above.

3.5 Theoretical Analysis

Building on previous modelling work [20], [120], [81], [68] and [73], in this section we develop a model and use it to analyse the saturation throughput and delay of the AFR scheme over a noisy channel.

3.5.1 Model

We assume stations are saturated, i.e., whenever the MAC layer needs a frame to transmit, it can always fill a long enough frame without waiting. The saturation throughput S is defined as the expected payload size of a successfully transmitted frame $E[L_f]$ in an expected slot duration $E[T]$, i.e., $S = \frac{E[L_f]}{E[T]}$. We first compute the expected state duration $E[T]$. Altogether, there are three kinds of events in the AFR scheme (notation is listed in Table 1):

- *Idle duration* T_I: In this case all stations are counting down so we have that $T_I = \sigma$.
- *Success/Error duration* T_3: In this case a frame is successfully transmitted or it is corrupted due to channel noise[6]. The slot duration is thus the sum of a frame, a SIFS and an ACK duration, i.e., $T_3 = T_{hdr}^{phy} + T_f + T_{ack}$.
- *Collision duration* T_C: Two or more stations transmit at the same time in this situation, and the duration is $T_C = T_{hdr}^{phy} + T_f + T_{EIFS}$.

The expected state duration is $E[T] = P_I T_I + P_3 T_3 + P_C T_C$, where P_I, P_3, P_C are the probabilities of *Idle*, *Success/Error* and *Collision* events respectively. Let τ denote the

[6]Recall that in the AFR scheme we consider frames that are partially corrupted by channel noise as successful transmissions.

station transmission probability and n the number of stations in the system. We have that

$$P_I = (1-\tau)^n, \tag{3.8}$$

$$P_3 = \binom{n}{1}\tau(1-\tau)^{n-1}, \tag{3.9}$$

and

$$P_C = 1 - P_I - P_3. \tag{3.10}$$

Solving for τ as in the DCF and BTA models (see Chapter 2), we can obtain the saturation throughput S_{AFR} of the AFR scheme from

$$S_{AFR} = \frac{P_3 \cdot E[L]}{P_I T_I + P_3 T_3 + P_C T_C} \tag{3.11}$$

Note that $E[L]$ is not the frame payload size, but rather the expected number of successfully transmitted bits – recall that the AFR scheme allows successfully transmitted fragments to be received even if some fragments within a frame are corrupted. We calculate $E[L]$ as follows. Let i denote the number of erroneous fragments, and m denote the number of fragments in a frame. Assuming independent and identically distributed errors, we have that

$$E[L] = \sum_{i=0}^{m} \binom{m}{i} \cdot (p_e^{frag})^i \cdot (1-p_e^{frag})^{m-i} \cdot (L_f - i \cdot L_{frag}), \tag{3.12}$$

and the fragment error rate p_e^{frag} is

$$p_e^{frag} = 1 - (1-p_b)^{L_{frag}+L_{FCS}}, \tag{3.13}$$

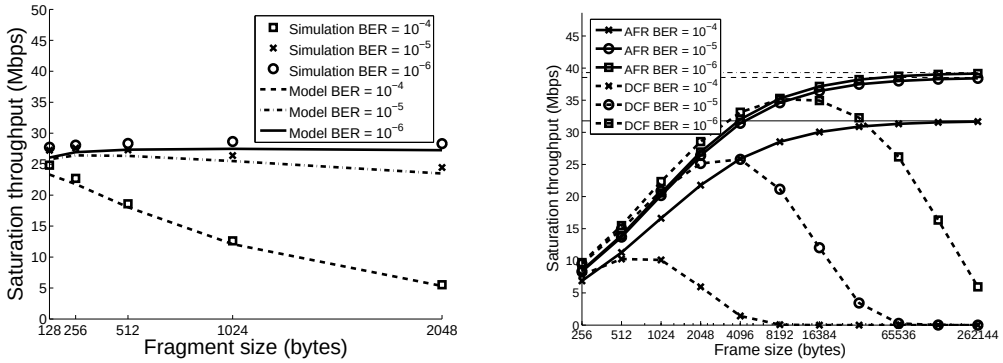
where L_{frag} and L_f are the length of a fragment and the length of frame payload respectively, and p_b is the BER.

Let $\Delta = \binom{m}{i} \cdot (p_e^{frag})^i \cdot (1-p_e^{frag})^{m-i}$. We have that

$$\begin{aligned} E[L] &= \sum_{i=0}^{m} [\Delta \cdot (L_f - i \cdot L_{frag})] \\ &= L_f \cdot (1 - p_e^{frag}). \end{aligned} \tag{3.14}$$

We thus have that

$$S_{AFR} = \frac{P_3 \cdot L_f \cdot (1-p_e^{frag})}{P_I T_I + P_3 T_3 + P_C T_C}. \tag{3.15}$$



(a) Model Validation

(b) AFR vs DCF with increasing frame sizes

Figure 3.5: (a) AFR: model vs. simulations. (b) The influence of frame sizes. Parameters are listed in Fig. 3.1(b) and Table 3.2.

	Fig. 3.5(a)	Fig. 3.5(b)	Fig. 3.6(a)	Fig. 3.7
Number of stations (n)	10	10	10	10
Application rate (Mbps)	54	54	$=R$	54
Data rate (Mbps) (R)	54	54	varied	54
Basic rate (Mbps)	6	6	$=R$	6
AFR sending queue (pkts)[a]	200	N/A	N/A	N/A
AFR IFQ (pkts)[b]	200	N/A	N/A	N/A
Packet (bytes)	2048	$= L_f$	1024	$= L_{frag}$
Frame (bytes) (L_f)	2048	256, $\cdots$, 262144	65536	8192
AFR fragment (bytes)(L_{frag})	128, $\cdots$, 2048	256	256	32, $\cdots$, 8192

Table 3.2: Parameters used for the AFR scheme, part one.

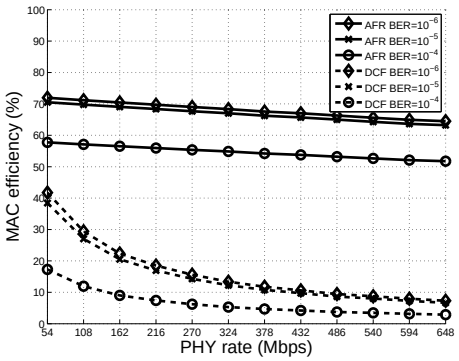
[a]AFR sending queue is the queue at MAC layer to temporarily store the packets from the AFR IFQ in AFR's simulations.

[b]AFR IFQ is the queue between MAC and its upper in AFR's simulations.

This model is validated against *NS-2* simulations. Both simulation and model results are shown in Fig. 3.5(a). As we can see from the results, the analysis and simulations match well.

3.5.2 Improvements over DCF

Fig. 3.5(b) compares the throughput of the DCF and AFR schemes. It can be seen that AFR fundamentally changes the throughput scaling behaviour in a noisy channel. Specifically, the DCF throughput exhibits a maximum value as the frame size is varied, with the maximum depending on the BER – this arises because while increasing frame payload size tends to



(a) AFR vs DCF with increasing PHY rates

Rates	frame=32768	frame=65536
54/6	2.5%	1.1%
108/24	4.2%	1.8%
216/24	8.3%	3.6%
432/54	**15.6%**	6.7%
648/216	**22.9%**	9.8%

(b) Loss in throughput

Figure 3.6: (a) AFR vs. DCF with increasing PHY rates. (b) In the first column, the PHY rates are on the left of the slash, the basic rates are on the right. The unit of the rates is Mbps. The values in the second and the third columns are differences between the throughput under the rates in the first column and the maximum throughput. Other parameters are listed in Fig. 3.1(b) and Table 3.2.

increase throughput, the probability of a frame being corrupted by noise also increases with frame size thereby tending to decrease throughput and the interaction of these two effects leads to the existence of an optimal size of frame that depends on the BER. In contrast, the AFR throughput *increases monotonically* with frame sizes even when the channel is noisy. The resulting gain in throughput compared to DCF is dramatic. For example, DCF achieves almost zero throughput for a frame size of 8192 bytes in a channel with BER of 10^{-4} while AFR achieves around 30 Mbps throughput under the same conditions.

Fig. 3.6(a) plots the MAC efficiency ($\frac{Throughput}{PHY\,Rate} \cdot 100\%$) of the DCF and AFR schemes as the PHY rate is increased and a frame size of 65536 bytes is used (see Section 3.5.3 for discussion of choosing frame sizes.). It can be seen that whereas the DCF efficiency rapidly decreases with increasing PHY rate (falling from 42% at 54Mbs to less than 10% at 432Mbs) the AFR efficiency is approximately constant with increasing PHY rate, in line with the discussion in Section 3.3. Observe that the efficiency falls with increasing BER as might be expected, but that the efficiency remains relatively high even under noisy conditions, e.g., achieving approximately 70% MAC efficiency for a BER of 10^{-5} and 60% efficiency for a BER of 10^{-4}.

3.5.3 Maximum frame size

It can be seen in Fig. 3.5(b) that the AFR throughput asymptotically approaches a maximum value as the frame size is increased. We can determine this asymptotic value analytically as follows. As the frame size $L_f \rightarrow \infty$, we have that (since T_3=T_C)

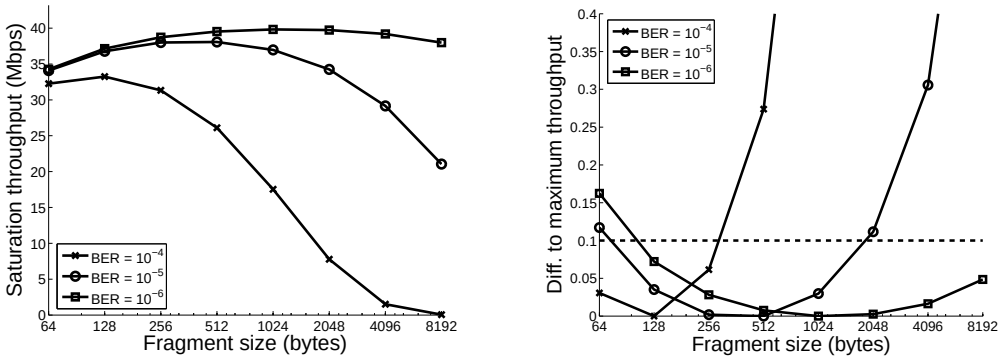
$$\begin{aligned} S_{AFR} &\rightarrow \frac{P_3 \cdot (1 - p_e^{frag})}{(P_3 + P_C) \cdot T_3/L_f} \approx \frac{P_3 \cdot (1 - p_e^{frag})}{(1 - P_I) \cdot T_f/L_f} \\ &= \frac{R \cdot P_3 \cdot (1 - p_e^{frag})}{(1 - P_I)}. \end{aligned} \tag{3.16}$$

Using this equation, the asymptotic values are $39.30, 38.55$ and 31.78 Mbps for $BER = 10^{-6}, BER = 10^{-5}$ and $BER = 10^{-4}$ respectively. These values are marked by horizontal lines on Fig. 3.5(b).

In practice, of course, arbitrarily large frame sizes are often not feasible. The upper limit on frame sizes depends on the PHY's abilities and is also constrained by interface memory and the size of the station's sending buffer. Fortunately, it can be seen in Fig. 3.5(b) that the gap between the maximum and actual throughput narrows rapidly with increasing frame sizes. Table 3.6(b) gives the loss in throughput (compared to the maximum achievable throughput) versus the frame size for a range of data rates. If we consider operation at 90% or higher of the maximum achievable throughput to be our target, it can be seen that a maximum frame size of 32768 bytes is acceptable for data rates up to 216 Mbps over a wide range of channel conditions while a maximum frame size of 65536 bytes is acceptable for data rates up to 648 Mbps. We note that 65536 bytes is also the maximum size proposed in TGn's 802.11n proposal [6].

3.5.4 Optimal fragment size

Fragmentation plays a central role in aggregation schemes such as AFR, with fragments being the unit used for retransmission. When a very small fragment size is used, only



(a) Throughput vs fragment size

(b) Diff. to maximum throughput

Figure 3.7: AFR throughput vs fragment size. The y-axis of (b) is the absolute (i.e., always positive) difference between the throughput using the fragment size marked on the x-axis and the throughput when using the optimal fragment size from (a). Other parameters are listed in Fig. 3.1(b) and Table 3.2.

Rates[a]	64[b]	128	256	512
54/6	2.5%, 10.4%,**14.5%**	0.0%, 2.9%, 6.2%	6.6%, 0.0%, 2.3%	**28.2%**, 0.0%, 0.0%
108/24	1.8%, 9.4%, **13.2%**	0.0%, 2.7%, 5.7%	6.9%, 0.0%, 0.2%	**28.4%**, 0.0%, 0.0%
216/24	0.1%, 8.3%, **11.6%**	0.0%, 2.6%, 5.2%	6.9%, 0.0%, 1.6%	**28.8%**, 0.0%, 0.0%
432/54	0.0%, 7.0%, 9.9%	0.0%, 1.9%, 4.1%	7.7%, 0.0%, 1.3%	**30.2%**, 0.1%, 0.0%
648/216	0.0%, 5.5%, 8.7%	0.0%, 0.1%, 3.3%	8.8%, 0.0%, 1.6%	**31.2%**, 0.0%, 0.0%

Table 3.3: AFR: Differences to maximum throughput with different PHY layers.

[a]The PHY rates are on the left of the slash, the basic rates are on the right. The unit of the rates is Mbps.

[b]The results are for frames with 64-byte fragments, under BER 10^{-4}, 10^{-5}, 10^{-6} respectively.

corrupted bits are retransmitted but since each fragment has a fixed size header the overhead is relatively large. When a large fragment size is used, the overhead created by the fragment header is small but many bits will be unnecessarily retransmitted since a single damaged bit in a fragment will lead to the entire fragment being retransmitted. For a given BER there therefore exists an optimal fragment size that balances the tradeoff between the fragment header overhead and excessive retransmissions. Fig. 3.7(a) plots throughput versus fragment sizes from which the existence of an optimal fragment size that maximises throughput is evident. Observe that the optimal fragment size depends on the BER, as is to be expected (128, 512 and 1024 bytes for BER=$10^{-4}, 10^{-5}, 10^{-6}$ respectively).

In practice, we are interested in determining a simple scheme that approximates the

optimal fragment sizes performance. It can be seen from Fig. 3.7(a) that the throughput peak is relatively flat and broad and thus we expect that the throughput reduction resulting from an approximate scheme can be kept relatively small. Fig. 3.7(b) plots the reduction in throughput, compared to that achieved with the optimal fragment sizes, of using a sub-optimal fragment size. From this plot we can see that if we can tolerate a throughput loss of up to 10%, then fragment sizes of 128 bytes and 256 bytes are near-optimal across a wide range of BERs. Corresponding data for a range of PHY rates are summarised in Table 3.3. It can be seen that fragment sizes of 128 and 256 bytes are always able to achieve within 10% of the maximum possible throughput. We have obtained similar results under a wide range of conditions including different numbers of stations, but these are not included here due to their similarity to the results in Table 3.3.

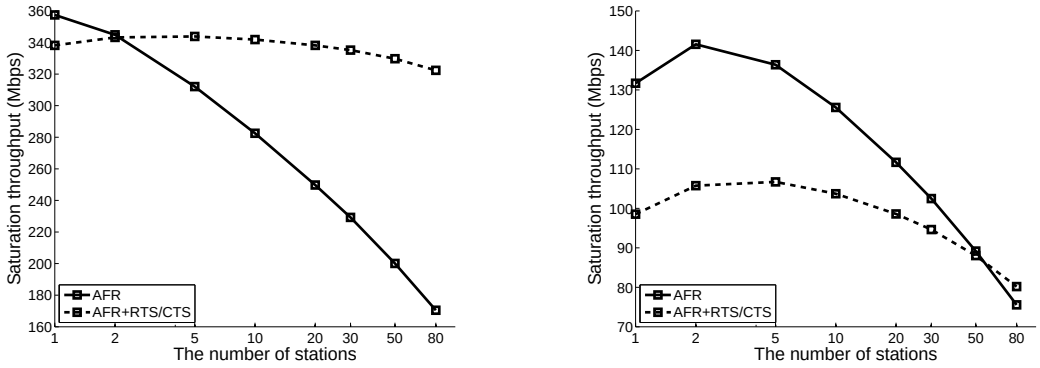
Based on these results, we propose a simple fragmentation algorithm: namely, for a packet P with a size of L_p, find the m' which satisfies

$$(m' - 1) \cdot 256 + 1 < L_p \leq m' \cdot 256,$$

where $m' = 1, 2, ..., 256$. We divide P into m' fragments, each of which has a size in the range of $(\frac{L_p}{m'}, \frac{L_p}{m'} + 1, ..., \frac{L_p}{m'} + (m' - 1))$. In this way, the sizes of all fragments fall between 128 and 256 bytes. Importantly, the resulting sizes are almost the same. For example, a 257 byte packet is divided into one 128-byte and one 129-byte fragment, rather than one 256-byte and one 1-byte fragment in which case the 1-byte fragment would result in high overhead.

3.5.5 RTS/CTS

Using large frames increases the duration of colliding transmissions, including collisions induced by hidden terminals. While consideration of hidden terminals is out of the scope of this work, we consider here the overhead associated with collisions. Collisions are of course a normal part of CSMA/CA operation. One technique for mitigating the duration of collisions is to probe the channel first using small packets so that collision losses only



(a) Large frames (BER=10^{-5}) (b) Small frames (BER=10^{-5})

Figure 3.8: Impact of RTS/CTS on AFR throughput. The frame sizes are 65536 and 4096 bytes in Fig. 3.8(a) and Fig. 3.8(b) respectively. In both figures, packet/fragment sizes are 2048/256 bytes, and the PHY data/basic rates are 432/54 Mbps. Other Parameters are listed in Fig. 3.1(b) and Table 3.4.

happen on small probing packets which can lead to improved channel utilisation. In Fig. 3.8 we use RTS/CTS as example probing packets to illustrate this trade-off for the AFR scheme. In Fig. 3.8(a), we vary the number of transmitting stations (the probability of a colliding transmission increasing with the number of active stations [20]) and use a fixed frame size of 65536 bytes and fragment size of 256 bytes in line with Sections 3.5.3 and 3.5.4 respectively. We observe that enabling RTS/CTS consistently results in significantly higher throughput when there is more than one station.

Nevertheless, using RTS/CTS adds a fixed extra overhead to each successful transmission which can have negative impact on performance. There is therefore a trade-off between reducing the duration of colliding slots and increasing the fixed overhead on successful transmissions. This can be seen in Fig. 3.8(b) where we compare AFR with and without RTS/CTS when the frame size is 4096 bytes. It can be seen that RTS/CTS starts to have positive impact only when there are more than 50 stations. In reality, an adaptive RTS/CTS mechanism is thus needed where RTS/CTS is enabled/disabled depending on frame sizes used and the channel load. We do not consider this in detail in the present work.

	Fig. 3.8(a)	Fig. 3.8(b)	Fig. 3.10(a)	Fig. 3.10(b) & 3.11
Number of stations (n)	10	10	10	10
Application rate (Mbps)	432	432	432	$=R$
Data rate (Mbps) (R)	432	432	432	varied
Basic rate (Mbps)	54	54	54	$=R$
AFR sending queue (pkts)[a]	N/A	N/A	N/A	N/A
AFR IFQ (pkts)[b]	N/A	N/A	N/A	N/A
Packet (bytes)	2048	2048	2048	1024
Frame (bytes) (L_f)	65536	4096	8192	varied
AFR fragment (bytes)(L_{frag})	256	256	256	256

Table 3.4: Parameters used for the AFR scheme, part two.

[a]AFR sending queue is the queue at MAC layer to temporarily store the packets from the AFR IFQ in AFR's simulations.

[b]AFR IFQ is the queue between MAC and its upper in AFR's simulations.

3.5.6 Comparison with Similar Schemes

In this section, we compare the throughput performance of AFR with four other schemes proposed in the literature: Burst ACK ([115] [93] [110]), Block ACK ([5] [122]), Packet Concatenation (PAC) [51] and *Aggregation* [55].

These schemes can be classified into two categories: 1) Burst ACK and Block ACK; 2) PAC, *Aggregation* and AFR. The schemes in the first category transmit multiple frames at each transmission opportunity. The schemes in the second category transmit only one frame and use packet aggregation. AFR is the only scheme to use both fragmentation and aggregation. In the Burst ACK and Block ACK schemes, collisions lead to the whole Burst/Block being lost while errors lead to retransmission only of the corrupted packet. The PAC scheme is similar to our AFR scheme, except that before each packet in a frame there is a sub-physical-header, which is of $12\mu s$ duration with an IEEE 802.11a PHY. The *Aggregation* scheme in [55] uses a special *delimiter* before each packet in a frame[7]. As shown in [70], delimitation techniques need support from the PHY layer. In particular, zeros should be inserted to ensure the uniqueness of the *delimiter*. The number of zeros

[7]Note that the *Aggregation* scheme in [55] is basically the pre-mature version of the TGn proposal for IEEE 802.11n. Details of the scheme such as frame format and design rationale, and simulation setup, are therefore business secret. The website of TGn, where their brief proposal could be downloaded, was even shutdown by the end of 2006. Due to this reason, we are unable to make further comparisons with their scheme.

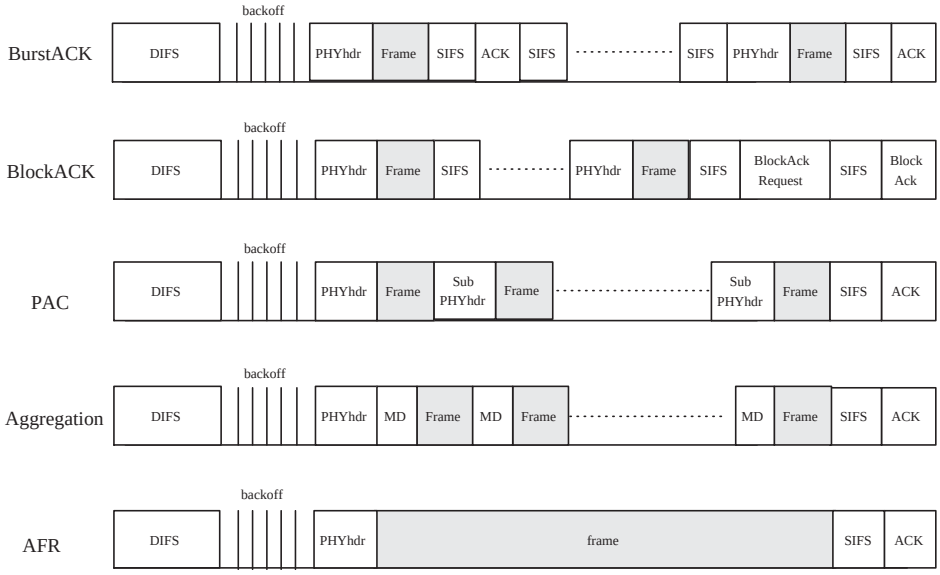


Figure 3.9: Five schemes compared in this chapter. 1) Burst ACK. 2) Block ACK. 3) Packet Concatenation from [51]. 4) Aggregation from [55]. 5) AFR.

inserted depends on the sizes of the *delimiter* and the packet. For an 8-bit *delimiter* as in [55], $L_p/(2^{\varsigma+1}-2)$ zeros are required, where L_p is the packet size, and $\varsigma = 5$ [70].

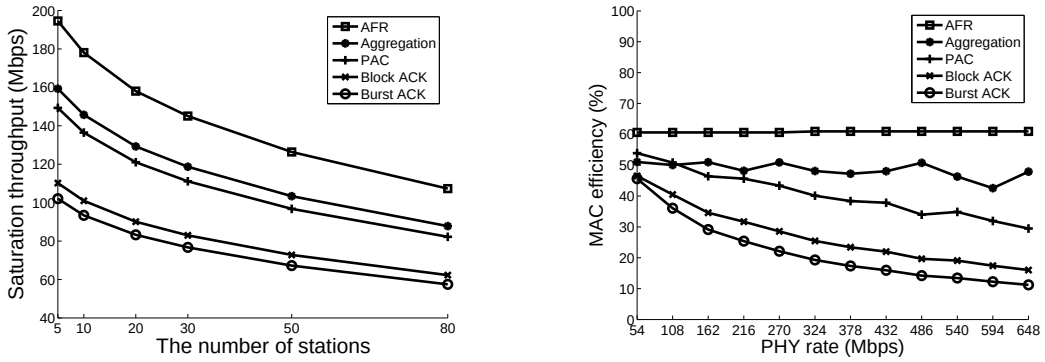
Note that apart from AFR, none of these schemes satisfy all of the scalability conditions derived in Section 3.3. Specifically,

- *Burst ACK and Block ACK.* A PHY header is transmitted before each packet. The PHY header duration has a minimum value as discussed previously, hence the per packet overhead does not decrease with increasing PHY rate.

- *PAC.* A sub-physical header is transmitted before each packet and similar comments apply.

- *Aggregation.* Fragmentation is not addressed in this scheme.

Results are shown in Fig. 3.10. It can be seen that the schemes employing aggregation (the second category) consistently outperform the Burst and Block ACK schemes. It can also be seen that the PAC scheme has the lowest throughput amongst schemes in the second



(a) Saturation throughput

(b) MAC efficiency

Figure 3.10: AFR vs alternative schemes. $BER = 10^{-5}$ in both figures. In Fig. 3.10(a), the PHY data rate is 432 Mbps and the basic rate is 54 Mbps. In Fig. 3.10(b), frames are selected so as to maintain a constant AFR efficiency. The other parameters are listed in Fig. 3.1(b) and Table 3.4.

category. This is due to the long duration of the sub-physical-header. AFR achieves the highest throughput regardless of the number of stations.

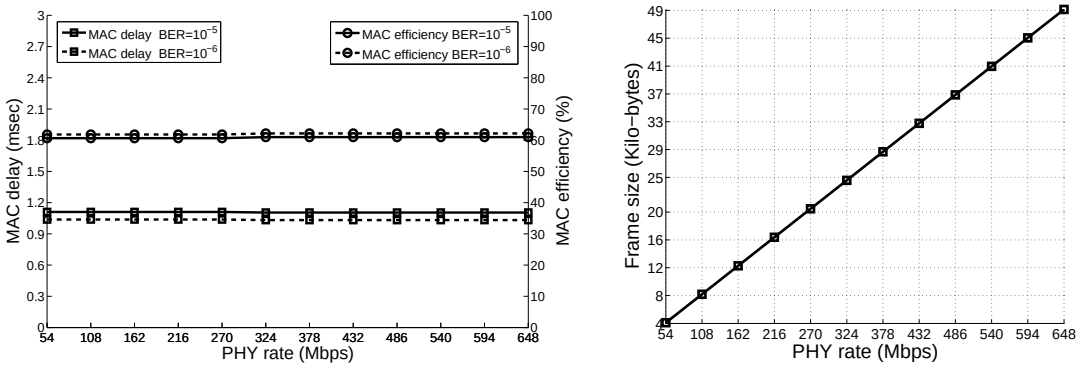
We further compare AFR to these alternative schemes with regard to MAC efficiency in Fig. 3.10(b). Here, all schemes use the same frame sizes which are selected to ensure a constant MAC efficiency for AFR. We can see that AFR consistently achieves the highest efficiency.

3.5.7 Delay Analysis

Our model can be extended to estimate the MAC layer delay, i.e., the mean time between a packet reaching the head of the MAC interface queue and being successfully transmitted. Let S^{frame} be the system throughput in frames-per-second rather than bits-per-second. That is, the MAC layer can transport S^{frame} frames in one second, thus the delay to successfully transmit one frame is $1/S^{frame}$, where

$$S^{frame} = \frac{E[number\ of\ frames]}{E[T]}. \tag{3.17}$$

In the AFR scheme, a packet is fragmented and may be only partially communicated in one transmission. Thus, we need to know the mean delay before all fragments of a packet are



(a) Efficiency/delay vs PHY rate (b) Frame sizes

Figure 3.11: Delay performance: In Fig. 3.11(a) we vary the frame sizes while increasing the PHY rates so that the MAC efficiency and MAC layer delay remain roughly constant, and the corresponding frame sizes are shown in Fig. 3.11(b). The other parameters are listed in Fig. 3.1(b) and Table 3.4.

successfully transmitted. Each fragment will be successfully transmitted in $\leq r'$ successful frame transmissions with probability

$$\begin{aligned}&(1-p_e^{frag})+(p_e^{frag})(1-p_e^{frag})+\ldots+(p_e^{frag})^{r'-1}(1-p_e^{frag})\\&=1-(p_e^{frag})^{r'}.\end{aligned} \tag{3.18}$$

Suppose that a packet arrives and is divided into m' fragments. The probability of successfully transmitting m' fragments in $\leq r'$ attempts is $(1-(p_e^{frag})^{r'})^{m'}$. Further, assuming that errors are independent, the probability of transmitting a packet in exactly r' attempts is $(1-(p_e^{frag})^{r'})^{m'}-(1-(p_e^{frag})^{r'-1})^{m'}$. So the expected number of retransmission attempts can be written as

$$r=\sum_{r'=1}^{\infty} r'\left[(1-(p_e^{frag})^{r'})^{m'}-(1-(p_e^{frag})^{r'-1})^{m'}\right]. \tag{3.19}$$

Here, the sum may be truncated to account for the finite number of retransmission attempts. Therefore we have that the per packet MAC delay D_{AFR}^{mac} is

$$D_{AFR}^{mac}=r\cdot\frac{P_IT_I+P_3T_3+P_CT_C}{P_3}. \tag{3.20}$$

For a fixed PHY rate, we expect the MAC delay to increase with frame size owing to the larger transmission time T_f for a frame. However, this is not the case when we

choose the frame size to be a function of the PHY rate. In particular, by scaling the frame size in proportion to the PHY rate not only do we maintain MAC efficiency but we also maintain an approximately constant frame transmission time in which case the MAC delay is invariant with PHY rate. This is illustrated in Fig. 3.11(a), which plots the MAC delay with increasing PHY rate. The corresponding frame size as a function of PHY rate is shown in Fig. 3.11(b). Note that while the MAC efficiency and MAC delay are constant, the actual throughput increases from $54 * 60\% = 32$ Mbps to $648 * 60\% = 388.8$ Mbps.

As noted previously, the level of MAC efficiency depends on the scaling factor b relating frame sizes to PHY rates. As we increase b, the efficiency rises. However, owing to the associated increase in frame transmission time, the MAC delay will also increase with b. A design decision therefore has to be made as to the desired trade-off between MAC efficiency and delay.

3.6 Simulations

As a complement to the theoretical analysis in Section 3.5, we have implemented the AFR scheme in the network simulator *NS-2* [12, 13]. The network topology is a peer-to-peer one where station i sends packets to station $i + 1$. We report here the simulation results for three types of traffic (TCP, HDTV and VoIP), all of which follow the requirements of the 802.11n usage model [10].

3.6.1 Metrics

We use the following metrics: Let c denote the number of packets (packet size is L_p bytes) successfully received by all of the stations and t denote the simulation duration. Let t_i^s be the time when the *i-th* packet is put in the interface queue (IFQ) lying between the MAC and the upper layer at the sender. Let t_i^e denote the time when the *i-th* packet is transferred to the upper layer at the receiver.

- Throughput ($= c * L_p * 8/t$ Mbps): Throughput is the maximum rate at which the

MAC layer can forward packets from sender to receiver. Since in a WLAN, all stations share a common medium, this throughput is that achieved by the whole system rather than by a single station.

- Peak delay ($= max\{d_1^{max}, d_2^{max}, \cdots, d_n^{max}\}$, where d_i^{max} denotes the maximum delay among all the packets successfully received by station i): Peak delay is the maximum delay experienced by a successfully transmitted packet. This metric is used for HDTV.

- Percentage delay: The metric we use for VoIP is the percentage delay at the application level. It is defined as the percentage of packets whose delay is greater than a delay upper limit (e.g, at the application layer, the system should have less than 1% of packets whose delays are greater than 30 ms. This is the criterion proposed in IEEE 802.11n's requirement [10]). At the MAC layer, we use a similar threshold, i.e., less than 1% of packets may have delay greater than 15 ms.

3.6.2 TCP traffic

TCP currently carries the great majority [127] of network traffic and it is therefore important to investigate the support of the AFR scheme for TCP traffic. Important features of TCP include the fact that traffic is (i) elastic and so achieved throughput is related to network capacity, and (ii) two-way involving TCP data and ACK packets. While TCP data packets are typically large, TCP ACKs are small packets so that it may be difficult to aggregate enough of them to form a large frame.

First, we evaluate AFR performance in a heavily-loaded WLAN with 50 stations. Each station performs a large FTP download, the data packet length is 984 bytes which yields an IP packet size of 1024 bytes when TCP and IP headers are added, TCP SACK functionality is used as this is prevalent in real networks. From Fig. 3.12(a) we can see that AFR achieves considerable throughput gains (by a factor of between 2 and 3 depending on channel conditions) over the DCF. As discussed previously, AFR performance is relatively insensitive to the choice of fragment size in the range 128-256 bytes, although as might be expected

	Fig. 3.12	Fig. 3.13	Table 3.6
Number of stations (n)	(a)50 (b)varied	varied	varied
Application rate (Mbps)	N/A	20	0.096
Data rate (Mbps) (R)	432	432	54
Basic rate (Mbps)	54	54	6
AFR sending queue (packets)[a]	10	10	10
AFR IFQ (packets)[b]	10	10	10
DCF IFQ (packets)[c]	20	20	20
Packet (bytes)	1024	1500	120
DCF frame (bytes)	1024	1500	120
AFR frame (bytes)	8192	9000	1200
AFR fragment (bytes)	(a)varied (b)512	750	120

Table 3.5: The parameters used in the *NS-2* simulations.

[a]AFR sending queue is the queue at MAC layer for temporarily storing the packets from the AFR IFQ.
[b]AFR IFQ is the queue between MAC and its upper layer.
[c]DCF IFQ is the queue between MAC and its upper layer.

	10	30	50	80	90
AFR ($BER = 10^{-4}$)	0.0%	0.0%	0.0%	**4.4%**	**15.4%**
AFR ($BER = 10^{-5}$)	0.0%	0.0%	0.0%	**1.1%**	**9.4%**
AFR ($BER = 10^{-6}$)	0.0%	0.0%	0.0%	0.9%	**3.9%**
DCF ($BER = 10^{-4}$)	0.0%	0.0%	0.0%	**24.9%**	**85.7%**
DCF ($BER = 10^{-5}$)	0.0%	0.0%	0.0%	**10.1%**	**75.2%**
DCF ($BER = 10^{-6}$)	0.0%	0.0%	0.0%	**9.2%**	**34.8%**

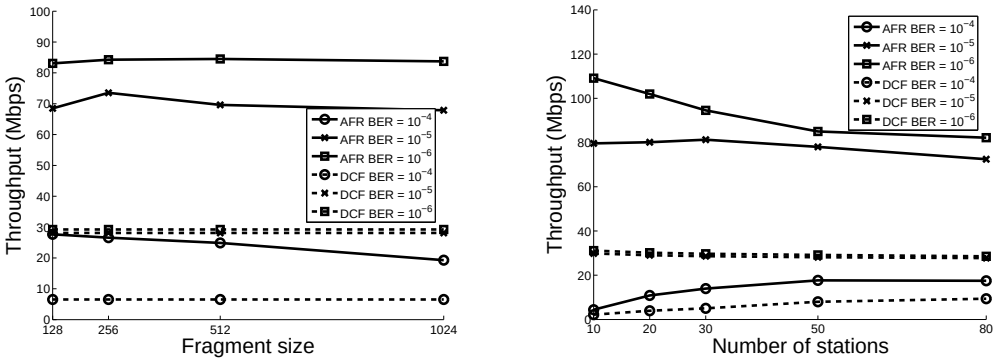
Table 3.6: AFR: Simulation results for VoIP traffic. The first row represents the number of stations. The other rows represent the percentage of packets with delay more than 15 ms with the bold figures show the percentage greater than 1%. The parameters are listed in Fig. 3.1(b) and Table 3.5.

the choice of fragment size becomes more important at higher BERs.

Second, we evaluate AFR performance as the number of stations is varied from 10 to 80. Fig. 3.12(b) shows both the AFR and DCF throughput. AFR achieves between 2.5 and 3 times the throughput of the DCF over this range of network conditions.

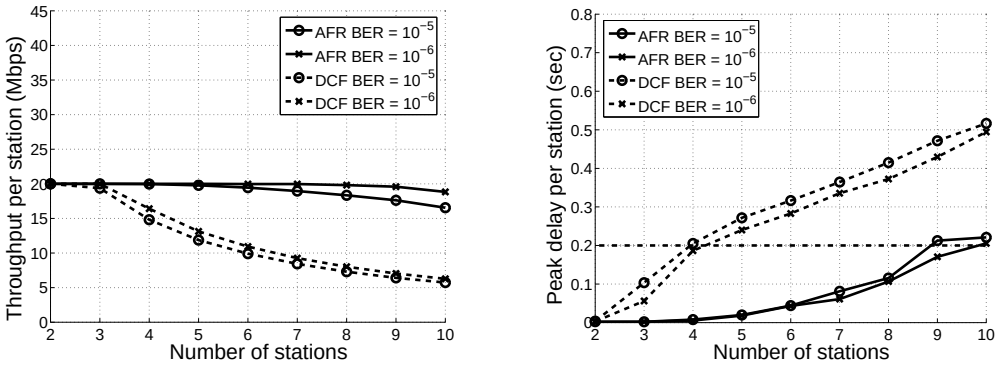
3.6.3 HDTV

According to the requirements of the IEEE 802.11n proposal [10], HDTV should be supported in future WLANs. HDTV has a constant packet size of 1500 bytes, a sending rate of 19.2-24 Mbps, and a 200ms peak delay requirement.



(a) Throughput vs fragment size (b) Throughput vs stations

Figure 3.12: Simulation results for TCP traffic. The parameters are listed in Fig. 3.1(b) and Table 3.5. Note that the DCF results for cases with BER of 10^{-5} and 10^{-6} almost overlap with each others due to similar values.



(a) Throughput (b) Peak delay

Figure 3.13: Simulation results for HDTV traffic. The parameters are listed in Fig. 3.1(b) and Table 3.5.

We investigate AFR HDTV performance with a 432 Mbps PHY data rate. Fig. 3.13 shows the throughput and delay performance of the AFR and DCF schemes as the number of stations (and so HDTV flows) is varied. The peak delay constraint of 200ms is marked on Fig. 3.13(b). It can be seen that the DCF can support only 2 simultaneous HDTV streams before the delay requirement is violated and the per flow throughput rapidly falls below the offered load. In contrast, AFR can support up to 9 and 10 streams for $BER = 10^{-5}$ and $BER = 10^{-6}$ respectively. That is, the HDTV capacity is increased by a factor of 5 over the DCF.

3.6.4 VoIP

The third application that we consider is VoIP, which is basically an on/off UDP stream with a constant on rate (96Kbps) and a small packet size (120 bytes) according to the IEEE 802.11n requirements [10]. VoIP is a challenging application for aggregation schemes because of its on/off nature and small packet sizes. Thus there may not be enough packets for AFR to aggregate and the DCF and AFR schemes might be expected to achieve more or less the same performance.

We consider a WLAN with pure VoIP traffic. We use Brady's model [118] of VoIP traffic in which the mean ON and OFF periods are 1500 ms. Our performance requirement is to have less than 1% of packets with delays larger than 15 ms. Table 3.6 shows the percentage of packets with delay exceeding 15 ms for a range of network conditions and numbers of voice calls. It can be seen that AFR's delay percentages are substantially less than the DCF's under all conditions, demonstrating the effectiveness of the AFR scheme even for traffic with very small packet sizes.

3.7 Scope of the Work

In this work, we restrict consideration to independent and identically distributed (i.i.d.) channel noise. Although we recognise that such a memory-less model is unable to capture fading characteristics in wireless channels, we comment that the PHY characteristics of IEEE 802.11n are still unknown at this time, making the selection of a more accurate channel model problematic. We note that provided the channel coherence time is long enough to support large frame transmissions, it is relatively straightforward to modify our analysis to encompass more complex channels. Moreover, it can be argued that i.i.d. noise is in fact a worst case for aggregation schemes since in fading environments the bit errors tend to cluster together into bursts [41] (see also the measurement of the bit error distribution from an IEEE 802.11a test-bed [77] [40]). An uneven error distribution typically benefits aggregation schemes since fewer retransmission are required compared to i.i.d. noise with

the same mean BER [11]. For instance, if there are ten corrupted bits in one frame which contains ten fragments, and each fragment has exactly one corrupted bit, then all of the fragments have to be retransmitted. If all the ten corrupted bits occur in burst and gather into say five fragments, it is obvious that fewer retransmissions are needed.

In this work we focus on the fundamental issues affecting the performance of aggregation schemes in 802.11 WLANs. Thus several other techniques for further optimising CSMA/CA performance are not addressed here. These include optimisation of the CSMA/CA contention window, which has been the subject of much attention in the literature, see [30, 125, 48, 80] and references therein for further details. Two-way aggregation is also possible, in which large frames are piggybacked in the ACK frames ([6], [67], and [123]).

3.8 Summary

To achieve high efficiency for next-generation very high-speed WLANs, we have developed a novel scheme called AFR after identifying fundamental properties that must be satisfied by any CSMA/CA based MAC layers. In the AFR scheme, multiple packets are aggregated into and transmitted in a single large frame. If errors happen during the transmission, only the corrupted fragments of the large frame are retransmitted. Transmission delays are minimised by using a zero-waiting mechanism where frames are transmitted immediately once the MAC wins a transmission opportunity.

An analytic model is developed to evaluate the throughput and delay of AFR over a noisy channel, and to compare AFR with competing schemes in the literature. If the application is saturated CBR traffic, analysis shows that AFR can achieve at the MAC layer more than 100 Mbps when the PHY rate is faster than 216 Mbps. As a complement to the theoretical analysis, we have investigated by simulations the impact of AFR on the performance of realistic applications including TCP, HDTV and VoIP.

The AFR scheme described has been developed as part of the 802.11n working group

work. The theoretical analysis presented here is general enough to be extended to the proposed scheme in the upcoming 802.11n standard. Trends indicated by our simulation results should extend to any well-designed aggregation schemes.

CHAPTER 4

Buffer Sizing for TCP Flows in 802.11e WLANs

We observed during the AFR work that there can be a close interaction between the MAC layer and the buffer above it for real world traffic such as TCP. Surprisingly, this buffering issue has received little attention in the literature even for the original DCF scheme of 802.11.

4.1 Introduction

We consider WLANs where the access point (AP) acts as a wireless router between the WLAN and the Internet. As TCP flows account for the vast majority (more than 90% [127]) of current Internet traffic and also of WLAN traffic [102], we consider buffer sizing for TCP flows.

Buffers are traditionally sized with two primary objectives in mind.

(i) *Accommodating short-term packet bursts.* Due to the nature of TCP, internet traffic tends to be bursty. Should too many packets arrive in a sufficiently short interval of time then a network device may lack the capacity to process all of the packets immediately. The first job of the buffer is to mitigate packet losses due to bursts by accommodating these packets in a buffer until they can be serviced.

(ii) *Ensuring AIMD throughput efficiency.* Most of the traffic on networks continues to be

carried by the TCP protocol. The AIMD congestion control algorithm used by TCP reduces the number of packets in flight by half on detecting network congestion. If buffers are too small, this backoff action will cause them to empty with a corresponding reduction in link utilisation.

The classical rule of thumb is to provision buffers to be equal to the *bandwidth* of the link multiplied by the average *delay* (which is typically described by round trip time or RTT) of the flows utilising this link: the *Bandwidth-Delay Product* (BDP) [112].

A number of fundamental new issues arise in 802.11 WLANs. Firstly, the mean service rate at a wireless station is strongly dependent on the level of channel contention and thus on the number of active stations and their load. Secondly, even when the network load is fixed, the packet inter-service times at a station are not fixed but vary stochastically due to the random nature of the CSMA/CA operation. As a result, neither the bandwidth nor the delay in 802.11 WLANs are constant, in contrast to the wired links. We therefore do not have a fixed BDP value available to provide a basis for sizing buffers.

In this chapter, we first consider an adaptive buffer sizing algorithm (called the aBDP algorithm) based on measurement of the current average service time. The current average bandwidth is then calculated as the reciprocal of the average service time. Rather than measure the average RTT of flows in the WLAN, we simply use a fixed RTT since we know that RTTs of flows in the Internet lie roughly within a known range. Given measured bandwidth and delay values, the BDP rule can be readily emulated in WLANs, yielding a buffer size that adapts to changes in the wireless link characteristics. The effectiveness of this first algorithm is verified against simulations.

Recent work on buffer sizes for wired links [14] shows that the BDP rule can be overly conservative, and suggests sizing buffers to $\frac{BDP}{\sqrt{n}}$ instead where n is the number of flows traversing a link. This exploits the statistical multiplexing when many flows share a link. Since real-world traffic patterns are often extremely complex, including a mix of connection sizes, RTTs, etc that change over time, adaptive buffer sizing is considered in [97] [117].

To exploit statistical multiplexing gains in WLANs, we consider a second adaptive buffer tuning algorithm (the ALT algorithm). This involves feedback control of buffer size based on measurements of the buffer idle and busy time. To accelerate the convergence rate of the ALT algorithm when traffic load increases suddenly, we also measure the current bandwidth. The effectiveness of the final algorithm (called the A* algorithm) is demonstrated with simulations.

The remainder of this chapter is organized as follows. Sections 4.2 and 4.3 introduce related work and the experimental setup used. In Section 4.4, performance with fixed-sized buffers is investigated. In Section 4.5, we introduce the aBDP algorithm based on the classical BDP rule. In Section 4.6, we develop the ALT algorithm to further exploit the possibility of statistical multiplexing to achieve lower delays while at the same time maintaining high efficiency. In Section 4.7, the A* algorithm is introduced.

4.2 Related Work

The classical approach to sizing Internet router buffers is the BDP rule proposed in [112]. Recently, in [14] it is argued that the BDP rule can be overly conservative on links shared by a large number of flows. In this case it is unlikely that TCP congestion window sizes (cwnd) evolve synchronously and due to statistical multiplexing of cwnd backoffs, the combined buffer requirement can be considerably less than the BDP. The analysis in [14] suggests that it may be sufficient to size buffers as the BDP divided by the square root of the number of active flows n. The potential may therefore exist to reduce buffer sizes by two to three orders of magnitude. This work is extended in [86], [38] and [119] to consider the performance of TCP congestion control with many connections under the assumption of small, medium and large buffer sizes. Several authors have pointed out that the value n can be difficult to determine for realistic traffic patterns, which not only include a mix of connections sizes and RTTs, but are also strongly time-varying [35], [117]. In [117], it is observed that in a production link, traffic patterns vary significantly, and may contain a

complex mix of flow connection lengths and RTTs. It is demonstrated in [35] that use of very small buffers can lead to an excessive loss rate. Motivated by these observations, in [97] [53] a measurement-based adaptive buffer size tuning method is therefore proposed.

The foregoing work is in the context of wired links, and to our knowledge the question of buffer sizing for 802.11 wireless links has received almost no attention in the literature. Notable exceptions include [74] [85] [104]. Sizing of buffers for voice traffic in WLANs is investigated in [74]. The impact of fixed buffer sizes on TCP flows is studied in [85]. In [104], TCP performance with a variety of AP buffer sizes and 802.11e parameter settings is investigated. With regard to the current state of the art, some vendors use small static buffers (e.g., in Proxim APs, up to 8 packets can be temporarily stored in the interface buffer), while others use large static buffers (e.g., in Atheros APs, up to 399 packets can be held in earlier versions and 50 packets for new versions.). As we will show, the former may lead to channel under utilisation if there are only a few active flows, while the latter tend to yield unnecessarily long delays. Adaptive buffer sizing for TCP traffic in WLANs has, to our knowledge, not previously been considered in the literature.

4.3 Setup

We consider the topology shown in Fig. 4.1 where the AP acts as a wireless router between the WLAN and the Internet. Upload flows originate from stations in the WLAN on the left and are destined to server(s) in the wired network on the right. Download flows are from the server(s) to stations in the WLAN. We ignore differences in wired bandwidth and delay from the AP to the servers which can cause TCP unfairness issues on the wired side (an orthogonal issue) by using the same wired-part RTT for all flows.

We note that in WLANs, TCP ACK packets can be easily queued/dropped due to the fact that the basic 802.11 DCF ensures that stations win a roughly equal number of transmission opportunities. For example consider n stations each carrying one TCP upload flow. The TCP ACKs are transmitted by the AP. While the data packets for the n flows

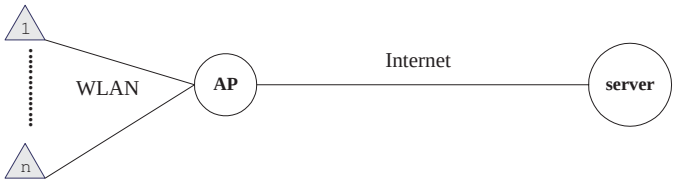


Figure 4.1: Topology used for buffer sizing in WLANs. Each station in WLAN is the source/destination of a single TCP flow. MAC parameters of the WLAN are listed in Table 4.1.

T_{SIFS} (μs)	10
Idle slot duration (σ) (μs)	9
Retry limit	11
Packet size (bytes)	1000
PHY data rate (Mbps)	54
PHY basic rate (Mbps)	6
PLCP rate (Mbps)	6

Table 4.1: MAC/PHY parameters used, corresponding to 802.11g.

have an aggregate $n/(n+1)$ share of the transmission opportunities the TCP ACKs for the n flows have only a $1/(n+1)$ share. Issues of this sort are known to degrade TCP performance significantly as queuing and dropping of TCP ACKs disrupt the TCP ACK clocking mechanism. Following [63], we address this problem using 802.11e. At the AP and each station we treat TCP ACKs as a separate traffic class, collecting them into a queue which is assigned high priority via $CW_{min} = 3$, $CW_{max} = 7$, $AIFS = 2$. TCP data packets are transmitted by another queue with parameters $CW_{min} = 31$, $CW_{max} = 1023$ and $AIFS = 6$. This makes use of 2 out of the 4 available queues in 802.11e.

We use IEEE 802.11g parameters as shown in Table 4.1. For TCP traffic, the widely deployed TCP Reno with SACK extension is used. The TCP slow start threshold is set to be 64 packets [100], the maximum congestion window size is set to be 4096 packets (each with a payload of 1000 bytes) as 4M bytes (4096*1000) is the default congestion window size staring from Linux kernels 2.6.17, and we are using 2.6.21.1.

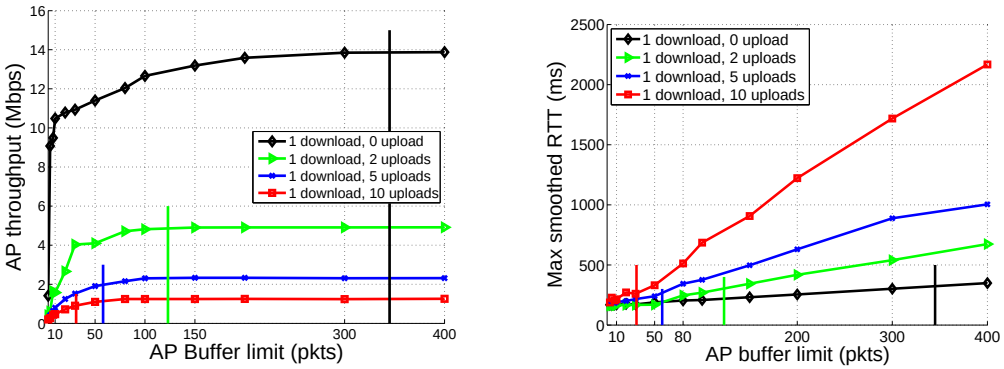


Figure 4.2: WLAN buffer sizing requirements. Data is shown for 1 download flow and 0, 2, 5, 10 competing uploads. Corresponding BDP values are marked by vertical lines. "Max smoothed RTT" denotes the maximum TCP *srtt* value observed. Wired backhaul link bandwidth 100Mbps, RTT 200ms.

4.4 Performance with Fixed Buffers

In contrast to wired networks, the mean service rate at a wireless station is not fixed but instead depends upon the level of channel contention and the network load. This is illustrated in Fig. 4.2 where the throughput and delay of a download flow are plotted as a function of AP buffer size when the number of competing upload flows (with one upload flow per wireless station) is varied. Similarly to wired networks, the throughput always increases monotonically with the buffer size, reaching a maximum above a threshold buffer size. However, it can also be seen that the download throughput falls as the number of competing uploads increases. The variation in throughput can be substantial, e.g., in this example the maximum throughput falls from 14Mbps to 1.25Mbps as the number of competing uploads increases from 0 to 10. As a result, the BDP – marked by vertical lines in Fig. 4.2 – also varies significantly and this is reflected in buffering requirements. For example, it can be seen from Fig. 4.2 that with no competing uploads the threshold buffer size above which the AP achieves maximum throughput is around 300 packets, while for 10 competing uploads this buffer size falls to approximately 50 packets.

In addition to variations in the mean service rate, the random nature of 802.11 CSMA/CA

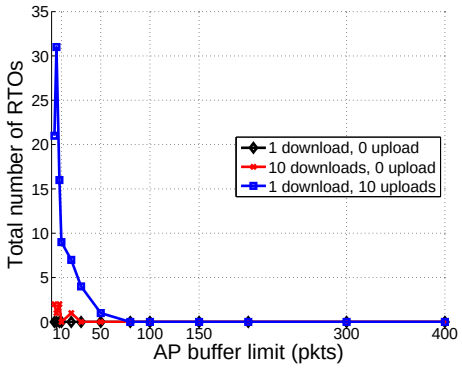


Figure 4.3: Number of TCP RTOs vs AP buffer size. Values collected in simulations run for 400 seconds. Note that the number of RTOs when there are 10 downloads and 1 download are nearly the same, so some of the points on the left side of this figure overlap with each other.

operations leads to short time-scale stochastic fluctuations in service rate. This is fundamentally different from wired networks and directly impacts buffering behaviour. For example, from Fig. 4.2 with 10 uploads the maximum download throughput is 1.25Mbps, yielding a BDP of 31 packets. However, it can be seen that at this buffer size the achieved download throughput is only about 60% of the maximum – a buffer size of at least 50 packets is required to achieve 100% throughput. Stochastic fluctuations in service rate can lead to early queue overflow or even TCP retransmission timeouts (RTOs) unless buffer provisioning over and above the BDP is used. For example, Fig. 4.3 illustrates the total number of TCP RTOs as the buffer size is varied. It can be seen that the number of RTOs decreases as the buffer size is increased. The buffer size required to minimise RTOs is dependent on network conditions.

One possible buffer sizing approach in WLANs is to provision buffers based on worst case conditions, i.e., based on the conditions requiring the largest buffering to achieve high throughput. However, while ensuring high throughput, this comes at the cost of high latency. For example, it can be seen from Fig. 4.2 that when a fixed buffer size of 338 packets is used (which in this example ensures maximum throughput regardless of the number of contending uploads), the round-trip latency experienced by the download flow is about 300ms with no uploads but rises to around 2s with 10 contending upload stations.

This occurs because TCP's congestion control algorithm probes for bandwidth until packet loss occurs and so flows will tend to fill buffers, regardless of their size.

Conversely, sizing the buffer to achieve lower latency across all network conditions comes at the cost of reduced throughput, e.g., a buffer size of 30 packets ensures latency of 200-300ms for up to 10 contending upload stations but when there are no contending uploads the throughput of a download flow is only about 75% of the maximum achievable.

Therefore, neither fixed long nor fixed short buffers can maintain both high throughput and low delays at the same time, motivating consideration of adaptive approaches to buffer sizing.

4.5 Emulating BDP: The First Algorithm

Motivated by the foregoing observations and the difficulty of selecting a fixed buffer size suited to a range of network conditions, we consider the use of an adaptive buffer sizing strategy. We note that a wireless station can readily measure its own service rate by observation of the packet inter-service time, i.e., the time between packets arriving at the head of the network interface queue t_s and being successfully transmitted t_e (which is indicated by receiving correctly the corresponding MAC ACK.). Note that this measurement can be readily implemented in real devices and incurs only a minor computational burden. Let T_{serv} be the mean inter-service time. We use exponential smoothing to calculate a running average, i.e., $T_{serv}(k+1) = (1-W)T_{serv}(k) + W(t_e - t_s)$ where $W = 0.001$. The choice of smoothing parameter W is considered in more detail later.

Using this measurement we propose the following adaptive BDP (called the aBDP algorithm) strategy. Let T' be the target queueing delay[1]. Noting that $1/T_{serv}$ is the mean service rate, we select buffer size Q according to $Q = min(T'/T_{serv}, Q_{max})$ where Q_{max} is the upper limit on buffer size (set to be 400 packets[2] here). This effectively regulates the

[1]We may select T' based on the average RTT of the flows sharing the queue, but here we simply use a fixed value of 200ms since this is an approximate upper bound on the RTT of the majority of the current Internet flows, and it is also an acceptable amount for non-interactive traffic.

[2]We select $Q_{max} = 400$ packets as it is the default buffer size used by the popular Atheros chip sets.

Algorithm 2 Drop tail operation of the aBDP algorithm.

Set the target queueing delay T'.
Set the over-provision parameter c.
for each incoming packet p **do**
  Calculate $Q = min(T'/T_{serv} + c, Q_{max})$ where T_{serv} is from MAC Algorithm 3.
  if current queue occupancy $< Q$ **then**
    Put p into queue
  else
    Drop p.
  end if
end for

Algorithm 3 MAC operation of the aBDP algorithm.

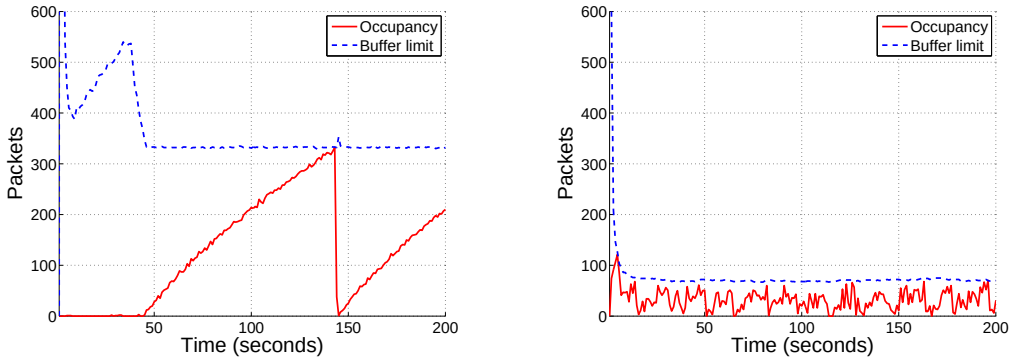
Set the averaging parameter W.
for each outgoing packet p **do**
  Record service start time t_s for p.
  Wait until receive MAC ACK for p, record service end time t_e.
  Calculate service time of p: $T_{serv} = (1 - W)T_{serv} + W(t_e - t_s)$.
end for

buffer size to equal the mean BDP. The buffer size decreases when the service rate falls and increases the buffer size when the service rate rises, so as to maintain an approximately constant queueing delay of T' seconds.

To account for the impact of the stochastic nature of the service rate on buffer size requirements, we modify this update rule to $Q = min(T'/T_{serv} + c, Q_{max})$ where c is an over-provisioning amount to accommodate short-term fluctuations in service rate. Based on the measurements in Fig. 4.2 and others, we have found that a value of $c = 40$ packets works well across a wide range of network conditions. Pseudo-code for the algorithm is shown in Algorithms 2 and 3.

The effectiveness of this simple adaptive algorithm is illustrated in Fig. 4.4. Recall that (see Fig. 4.2) 330 packets and 70 packets are the approximate threshold buffer sizes above which the AP achieves maximum throughput for respectively 1 download only and 1 download plus 10 contending uploads. It can be seen from Fig. 4.4 that the aBDP algorithm yields buffer sizes which are in good agreement with these thresholds. Observe also that

However, for future very high-speed WLANs such as 802.11n in which MAC layer throughput may reach 100 Mbps, a larger value of Q_{max} is likely to be required.



(a) 1 download, no uploads

(b) 1 download, 10 uploads

Figure 4.4: Histories of buffer limit and buffer occupancy with the aBDP algorithm. In (a) there in only one download flow. In (b) there are 1 download and 10 upload flows.

there are initial startup fluctuations in the adaptive buffer limit. This reflects the startup dynamics of the TCP congestion control. Once the buffer starts to fill, we observe that the aBDP algorithm maintains a steady buffer limit.

In Fig. 4.5 we plot the throughput percentage (ratio of achieved throughput to that with a 400-packet buffer) and max smoothed RTT as the number of downloads and uploads is varied. It can be seen that the adaptive algorithm maintains high throughput efficiency across the entire range of operating conditions. This is achieved while maintaining the latency approximately constant at around 400ms (200ms propagation delay plus $T' = 200$ms queueing delay) – the latency rises slightly with the number of uploads due to the over-provisioning parameter c to accommodate stochastic fluctuations in service rate.

While $T' = 200$ms is used as the target drain time in the aBDP algorithm, realistic traffic tends to consist of flows with a mix of RTTs. Fig. 4.6 plots the results as we vary the RTT of the wired backhaul link while keeping $T' = 200$ms. We observe that the throughput percentage is 100% for RTTs up to 200ms. For a RTT of 300ms, we observe a slight decrease in throughput when there is 1 download and 10 contending upload flows, which is to be expected since T' is less than the link delay and so the buffer is less than the BDP. We also observe that there is a difference between the max smoothed RTT with and without upload flows. The RTT in our setup consists of the wired link RTT, the queueing delays

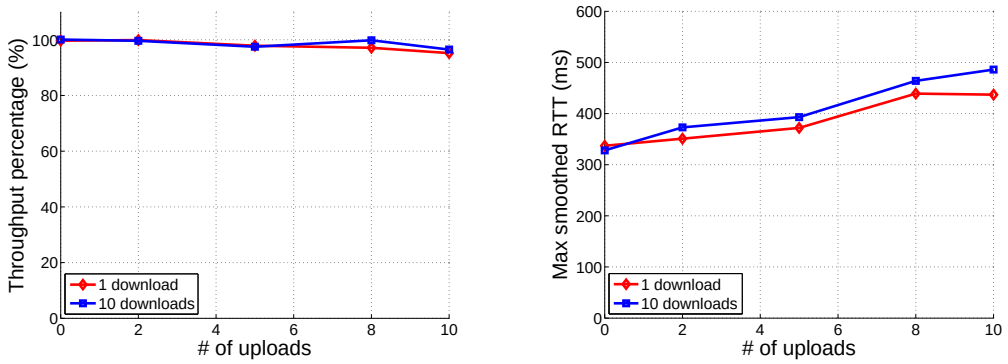


Figure 4.5: Performance of the aBDP algorithm as the number of upload flows is varied. Data is shown for 1, 10 download flows and 0, 2, 5, 10 uploads. Wired backhaul link bandwidth 100Mbps, RTT 200ms. Note that some of results when there is 1 and 10 downloads overlap with each other.

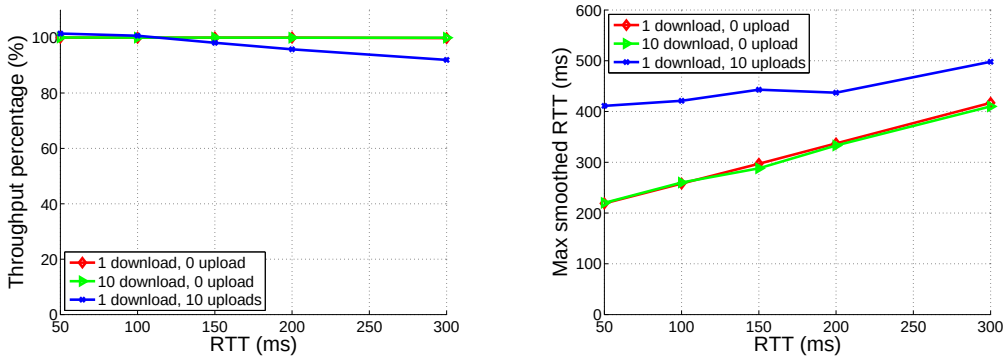


Figure 4.6: Performance of the aBDP algorithm as the RTT of wired backhaul is varied. Data is shown for 1, 10 downloads and 0, 10 uploads. Wired backhaul link bandwidth 100Mbps

for TCP data and ACK packets and the MAC layer transmission delays for TCP data and ACK packets. When there are no upload flows, TCP ACK packets can be transmitted with negligible queueing delays. This is because that they are assigned with high priority and so can be transmitted earlier when competing with the AP that is sending TCP data packets. When there are upload flows however, stations with TCP ACK packets have to contend with stations sending TCP data packets as well. TCP ACK packets therefore can be delayed accordingly, which causes the increase in RTT observed in Fig. 4.6.

Fig. 4.7 demonstrates the ability of the aBDP algorithm to respond to changing network conditions. At time 300s the number of uploads is increased from 0 to 10 flows. It can be seen

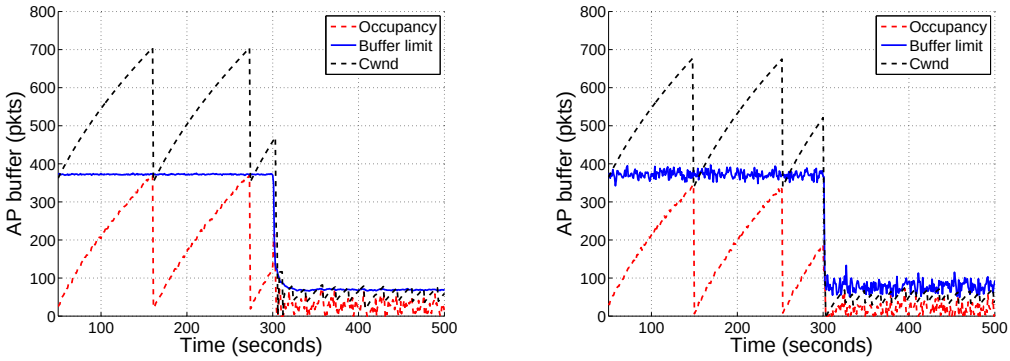
that the buffer size quickly adapts to the changed conditions when the weight $W = 0.001$. This roughly corresponds to averaging over the last 1000 packets[3]. When the number of uploads is increased at time 300s, it takes 0.6 seconds (current throughput is 13.5Mbps so $t = 1000 * 8000/13.5 * 10^6 = 0.6$) to send 1000 packets, i.e., the aBDP algorithm is able to react to network changes roughly on a timescale of 0.6 second. Also shown in Fig. 4.7 are the corresponding plots when $W = 0.1$, $W = 0.0001$ and $W = 0.00001$. While $W = 0.1$ potentially ensures a faster convergence rate, the averaging duration is too short to smooth the fluctuations in service rate and this is reflected in fluctuations in the buffer limit. For $W = 0.0001$ and 0.00001, the averaging intervals are rather long, leading to slow convergence rates.

4.6 Adaptive Buffer Limit Tuning: The Second Algorithm

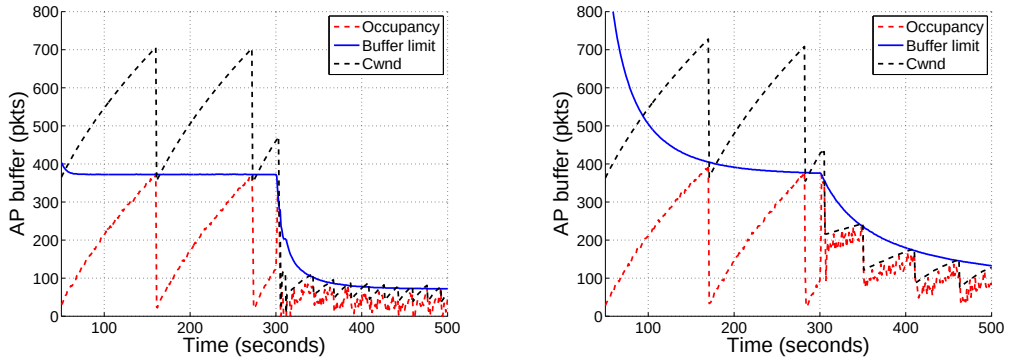
While the aBDP algorithm is simple and effective, it is unable to take advantage of the statistical multiplexing when multiple flows share the same link. For example, it can be seen from Fig. 4.8 that while a buffer size of 338 packets is needed to maximise throughput with a single download flow, this falls to around 100 packets when 10 download flows share the link. However, in both cases the aBDP algorithm selects a buffer size of approximately 350 packets, see Figs. 4.4(a) and 4.9.

It can be seen, however, from Fig. 4.9 that the buffer rarely empties when 10 flows share the link. That is, the potential exists to lower the buffer size (thereby reducing latency) without loss of throughput. In this section we consider the design of measurement-based algorithms that are capable of taking advantage of such statistical multiplexing opportunities.

[3]As per [28], the current value is averaged over the last t observations for x% percentage of accuracy where $x = 1 - (1 - W)^t$, t is the number of updates (which are packets in our case). When $W = 0.001$ and $t = 1000$ we have, for example, that $x = 0.64$.



(a) W=0.001 (b) W=0.1



(c) W=0.0001 (d) W=0.00001

Figure 4.7: Impact of weight W on convergence rate of the aBDP algorithm. One download flow, at time 300s the number of upload flows is increased from 0 to 10.

4.6.1 The ALT Algorithm

Our objective is to simultaneously achieve both high throughput efficiency and low delay. Intuitively, in order to ensure efficient link utilisation, the buffer should not lie empty for too long a time. Increasing the buffer size tends to reduce the link idle time. However, to ensure low delays, the buffer should be as short as possible and a trade-off therefore exists. This intuition suggests the following approach. We observe the buffer occupancy over an interval of time. If the buffer rarely empties, we decreases the buffer size to avoid high delay. Conversely, if the buffer is empty for too long a period, we increase the buffer size to maintain high throughput.

In more detail, we consider an Adaptive Limit Tuning (ALT) algorithm as follows. Let $t_i(k)$, $t_b(k)$ be the durations of idle and busy time in an observation interval t, i.e.,

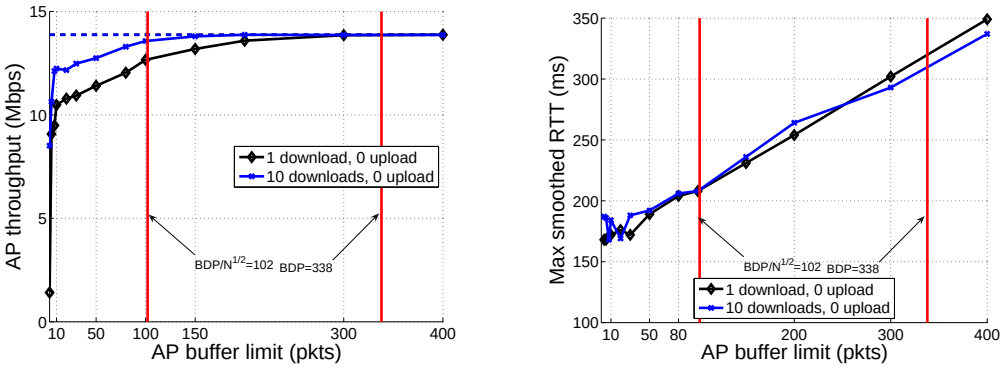


Figure 4.8: Impact of statistical multiplexing. There are 1/10 downloads and no uploads. Wired backhaul link bandwidth 100Mbps, RTT 200ms.

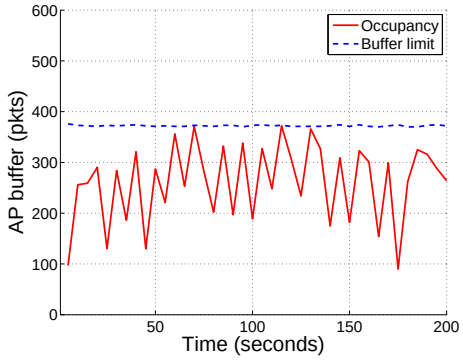


Figure 4.9: Histories of buffer limit and buffer occupancy with the aBDP algorithm when there are 10 downloads and no uploads.

$t = t_i(k) + t_b(k)$, and $q'(k)$ be the buffer limit during the k-th observation interval. The buffer limit is updated according to

$$q'(k+1) = q'(k) + a't_i(k) - b't_b(k), \tag{4.1}$$

where a' and b' are design parameters. Pseudo-code for the ALT algorithm is given in Algorithm 4.

Assuming q' converges (we discuss this in more detail later), then we have that $a't_i = b't_b$, i.e., $t_i = \frac{b'}{a'}t_b$ and the link utilisation is therefore lower bounded by

$$\frac{t_b}{t_i + t_b} = \frac{1}{1 + b'/a'}. \tag{4.2}$$

Choosing $\frac{b'}{a'}$ to be small then ensures high utilisation.

Algorithm 4 : The ALT algorithm.

1: Set the initial queue limit, the maximum buffer limit q_{max} and the minimum buffer limit q_{min}.
2: Set the increase step size a' and the decrease step size b'.
3: **for** Every t seconds **do**
4: Measure the idle time t_i.
5: $q_{new} = q + a't_i - b'(t - t_i)$.
6: **if** $q_{new} < q_{max}$ **then**
7: **if** $q_{new} < q_{min}$ **then**
8: $q \leftarrow q_{min}$
9: **else**
10: $q \leftarrow q_{new}$
11: **end if**
12: **else**
13: $q \leftarrow q_{max}$
14: **end if**
15: **end for**

It is prudent to constrain q' to lie between the minimum and the maximum values q_{min} and q_{max}. In the following, the maximum limit q_{max} and the minimum buffer limit q_{min} are set to be 400 and 30 packets, respectively. We use 400 packets as the maximum buffer limit to facilitate comparison with the fixed buffer scheme used in Atheros chip sets. In the next section, we provide a theoretical analysis which can assist in selecting values for a', b' and q_{min}.

The duration of the observation/update interval t should be so selected to reflect timely changes on buffer usage which are nonpredictable in reality (see for example [117] and the references therein). Too small t can therefore yields similar/repeated observations if traffic patterns are varying slowly, too large t will likely miss bursty changes. Here we choose the safer option, i.e., a short $t = 1$ second is used.

4.6.2 Theoretical Analysis

Let $Q(k)$ denote the buffer size at the k-th congestion event[4]. Then,

$$Q(k+1) = Q(k) + aT_I(k) - bT_B(k) \tag{4.3}$$

[4]A congestion event happens when the sum of all senders' cwnd decreases due to packet losses. This can be caused by a single packet loss, or multiple packet losses that are lumped together in one RTT.

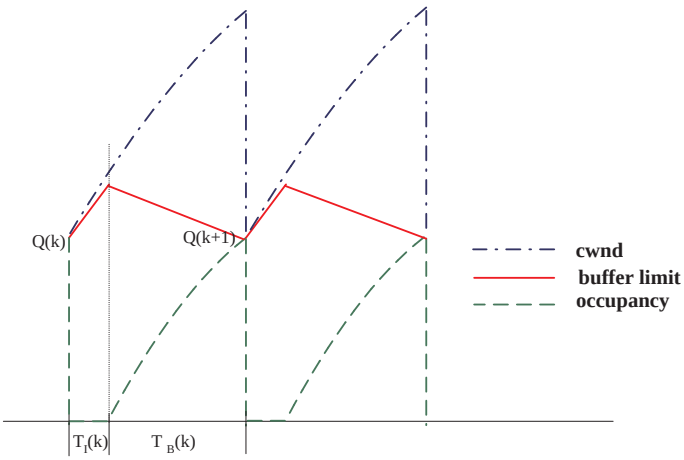


Figure 4.10: Evolution of the buffer limit.

where T_I is the "idle" time, i.e., the duration in seconds when the buffer is empty during the k-th congestion epoch[5], and T_B the "busy" time, i.e., the duration when the buffer is non-empty. This is illustrated in Fig. 4.10 for the case of a single TCP flow.

Notice that $a = a'$ and $b = b'$ where a' and b' are parameters used in the ALT algorithm[6]. Letting m be the number of observation/updating intervals in the k-th congestion epoch, it is straightforward that $mt_i(k) = T_I(k)$ and $mt_b(k) = T_B(k)$. That is, the per-interval updating strategy of the ALT algorithm directly maps to the per-epoch system dynamics in Equation 4.3. In the remainder of this section we investigate conditions to guarantee convergence and stability of the buffer dynamics with TCP traffic, which naturally lead to guideline for the selection of a' and b'. We first define some TCP related quantities before proceeding.

Consider the case where TCP flows may have different round-trip times and drops need not be synchronised. Let n be the number of TCP flows sharing a link, $w_i(k)$ be the cwnd of flow i at the k-th congestion event, T_i the round-trip propagation delay of flow i. To describe the cwnd additive increase we define the following quantities: (i) α_i is the rate in $packet/s$ at which flow i increases its congestion window[7], (ii) $\alpha_T = \sum_{i=1}^{m} \alpha_i$ (where

[5]A congestion epoch is the duration between two adjacent congestion events.

[6]The update interval in the ALT algorithm is $t = 1$ second, whilst in the analysis the update interval is congestion epoch.

[7]Standard TCP increases the flow congestion window by one packet per RTT, in which case $\alpha_i \approx 1/T_i$.

$m < n$ is the number of flows that do not backoff) is the aggregate rate at which flows increase their congestion windows, in packets/s, and (iii) $A_T = \sum_{i=1}^{m} \alpha_i/T_i$ approximates the aggregate rate, in $packets/s^2$, at which flows increase their sending rates. Following the k-th congestion event, flows backoff their cwnd to $\beta_i(k)w_i(k)$. Flows may be unsynchronised, i.e., not all flows need back off at a congestion event. We capture this with $\beta_i(k) = 1$ if flow i does not backoff at event k.

Now consider the idle time $T_I(k)$. On backoff after the k-th congestion event, if the buffer does not empty then $T_I(k) = 0$. Otherwise, immediately after backoff the send rate of flow i is $\beta_i(k)w_i(k)/T_i$. We then have that

$$T_I(k) = \frac{E[B] - \sum_{i=1}^{n} \beta_i(k)w_i(k)/T_i}{A_T}, \tag{4.4}$$

where $E[B]$ is the mean service rate of the considered buffer[8].

At congestion event k the aggregate flow throughput necessarily equals the link capacity, i.e.,

$$\sum_{i=1}^{n} \frac{w_i(k)}{T_i + Q(k)/E[B]} = E[B].$$

We then have that

$$\begin{aligned}
\sum_{i=1}^{n} \frac{w_i(k)}{T_i} &= \sum_{i=1}^{n} \frac{w_i(k)}{T_i} \frac{T_i + Q(k)/E[B]}{T_i + Q(k)/E[B]} \\
&= \sum_{i=1}^{n} \frac{w_i(k)}{T_i + Q(k)/E[B]} + Q(k)/E[B] \sum_{i=1}^{n} \frac{w_i(k)}{T_i + Q(k)/E[B]} \frac{1}{T_i} \\
&\approx E[B] + Q(k)/T_T,
\end{aligned}$$

where $T_T = \frac{n}{\sum_{i=1}^{n} \frac{1}{T_i}}$ is the harmonic mean of T_i. Hence,

$$\begin{aligned}
T_I(k) &= \frac{E[B] - \beta_T(k) \sum_{i=1}^{n} w_i(k)/T_i}{A_T} \\
&\approx \frac{E[B] - \beta_T(k)(E[B] + Q(k)/T_T)}{A_T}
\end{aligned}$$

That is,

$$T_I(k) \approx \frac{(1 - \beta_T(k))E[B] - \beta_T(k)Q(k)/T_T}{A_T} \tag{4.5}$$

[8]When rate adaptation is enabled and/or the number of competing flows vary frequently, the mean throughput is expected to change accordingly. Analysis for this case is left as future work. In implementation however, we do propose a technique to address this varying bandwidth issue. See Section 4.7 for details.

where $\beta_T(k) = \frac{\sum_{i=1}^{n} \beta_i w_i(k)/T_i}{\sum_{i=1}^{n} w_i(k)/T_i}$ is the effective aggregate backoff factor of the flows. When flows are synchronised, i.e., $\beta_i = \beta \ \forall i$, then $\beta_T = \beta$. When flows are unsynchronised[9] but have the same *average* backoff factor, i.e., $E[\beta_i] = \beta$, then $E[\beta_T] = \beta$.

If the queue empties after backoff, the queue busy time $T_B(k)$ is directly given by

$$T_B(k) = Q(k+1)/\alpha_T, \tag{4.6}$$

where α_T is the aggregate rate at which flows increase their congestion windows, in packets/s. Otherwise,

$$T_B(k) = (Q(k+1) - q(k))/\alpha_T, \tag{4.7}$$

where $q(k)$ is the buffer occupancy after backoff. It turns out that for the analysis of stability it is not necessary to calculate q_k explicitly. Instead, letting $\delta(k) = q(k)/Q(k)$, it is enough to note that $0 \leq \delta(k) < 1$.

Combining (4.3), (4.5), (4.6) and (4.7),

$$Q(k+1) = \begin{cases} Q(k) + a\frac{(1-\beta_T)E[B]-\beta_T(k)Q(k)/T_T}{A_T} - bQ(k+1)/\alpha_T & \text{if queue empties after backoff} \\ Q(k) - b(Q(k+1) - \delta(k)Q(k))/\alpha_T & \text{otherwise} \end{cases}$$

That is,

$$Q(k+1) = \begin{cases} \lambda_e(k)Q(k) + \gamma_e(k)E[B]T_T & \text{if queue empties after backoff} \\ \lambda_f(k)Q(k) & \text{otherwise} \end{cases} \tag{4.8}$$

where

$$\begin{aligned} \lambda_e(k) &= \frac{\alpha_T - a\beta_T(k)\alpha_T/(A_T T_T)}{\alpha_T + b} \\ \lambda_f(k) &= \frac{\alpha_T + b\delta(k)}{\alpha_T + b} \\ \gamma_e(k) &= a\frac{1-\beta_T(k)}{\alpha_T + b}\frac{\alpha_T}{A_T T_T} \end{aligned}$$

Taking expectations,

$$E[Q(k+1)] = \lambda(k)E[Q(k)] + \gamma(k)E[B]T_T \tag{4.9}$$

[9]Plus the probability of backoff independent of the flow congestion window. This appears to be a good approximation in many practical situations, see [94] for a more detailed discussion.

where

$$\begin{aligned}\lambda(k) &= p_e(k)E[\lambda_e(k)] + (1 - p_e(k))E[\lambda_f(k)] \\ \gamma(k) &= p_e(k)E[\gamma_e(k)]\end{aligned}$$

with $p_e(k)$ the probability that the queue empties following the k-th congestion event.

4.6.3 Stability

Provided $|\lambda(k)| < 1$ the queue dynamics in Equation 4.9 are exponentially stable. In more detail, $\lambda(k)$ is the convex combination of $E[\lambda_e(k)]$ and $E[\lambda_f(k)]$. Stability is therefore guaranteed provided $|E[\lambda_e(k)]| < 1$ and $|E[\lambda_f(k)]| < 1$. We have that $0 < E[\lambda_f(k)] < 1$ when $b > 0$ since α_T is non-negative and $0 \leq \delta(k) < 1$. The stability condition is therefore that $|E[\lambda_e(k)]| < 1$.

Under mild independence conditions,

$$E[\lambda_e(k)] = \frac{\alpha_T - aE[\beta_T(k)]\alpha_T/(A_T T_T)}{\alpha_T + b}.$$

Observe that,

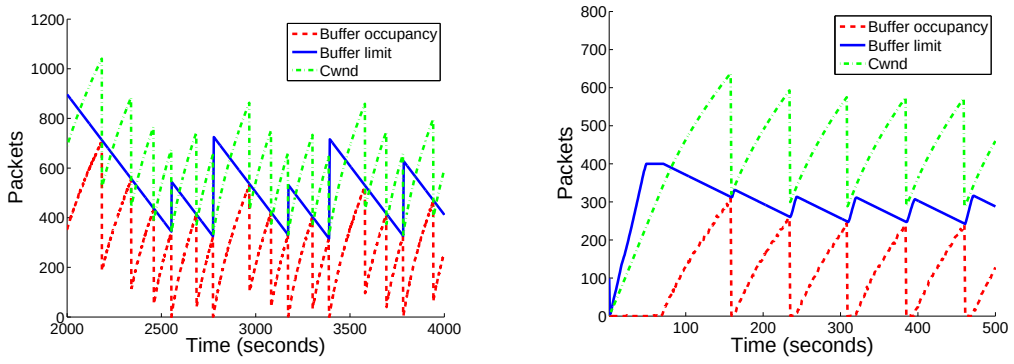
$$\frac{\alpha_T}{A_T T_T} = \frac{1}{n}\frac{(\sum_{i=1}^{n} 1/T_i)^2}{\sum_{i=1}^{n} 1/T_i^2}$$

when we use the standard TCP AIMD increase of one packet per RTT, in which case $\alpha_i \approx 1/T_i$. We therefore have that $1/n \leq \alpha_T/(A_T T_T) \leq 1$. Also, when the standard AIMD backoff factor of 0.5 is used, $0.5 < E[\beta_T(k)] < 1$. Thus, since $a > 0$, $b > 0$, $\alpha_T > 0$, we require

$$-1 < \frac{\alpha_T - a}{\alpha_T + b} \leq E[\lambda_e(k)] \leq \frac{\alpha_T}{\alpha_T + b} < 1$$

A sufficient condition (from the left inequality) for stability is then that $a < 4\alpha_T + 2b$. Using again (as in the aBDP algorithm) 200ms as the maximum RTT , a rough lower bound on α_T is 5 (corresponding to 1 flow with RTT 200ms). The stability constraint is then that

$$a < 10 + 2b. \tag{4.10}$$



(a) Instability (b) Stability

Figure 4.11: Instability and stability of the ALT algorithm. In (a), a=100, b=1, the maximum buffer limit is 50000 packets. In (b), a=10, b=1, the maximum buffer limit is 400 packets. In both figures, there is 1 download and no upload.

Fig. 4.11(a) demonstrates that the instability is indeed observed in simulations. Here, $a = 100$ and $b = 1$ are used as example values, i.e., the stability conditions are not satisfied. It can be seen that the buffer limit at congestion events oscillates around 400 packets rather than converging to a constant value.

Fig. 4.11(b) shows the corresponding results with $a = 10$ and $b = 1$, i.e., when the stability conditions are satisfied. It can be seen that the buffer limit at congestion events settles to a constant value, thus the buffer limit time history converges to a periodic cycle.

4.6.4 Fixed point

When the system dynamics are stable, from (4.9) we have that

$$\lim_{k\to\infty} E[Q(k)] = \frac{(1 - E[\beta_T])}{b/a + E[\beta_T]} E[B]T_T. \tag{4.11}$$

For synchronised flows with the standard TCP backoff factor of 0.5 (i.e., $E[\beta_T] = 0.5$) and the same RTT, $\frac{(1-E[\beta_T])}{b/a+E[\beta_T]}E[B]T_T$ reduces to the BDP when $b/a = 0$. This indicates that for high link utilisation we require the ratio b/a to be small. In more detail, using (4.5), (4.6) and (4.13) we have that in steady-state the expected link utilisation is lower bounded by

$$\frac{E[T_B]}{E[T_I] + E[T_B]} = \frac{1}{1 + \frac{b}{a}\frac{\alpha_T}{A_T T_T}} \geq \frac{1}{1 + \frac{b}{a}}. \tag{4.12}$$

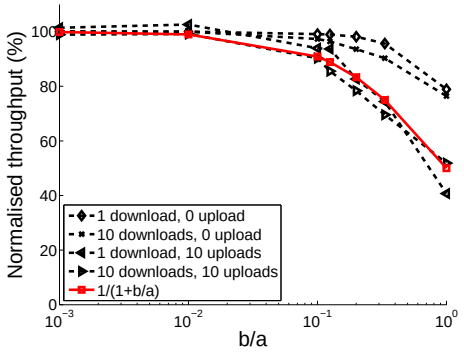


Figure 4.12: Impact of b/a on throughput efficiency. The maximum buffer limit is 400 packets, and the minimum buffer limit is 2 packets.

This is illustrated in Fig. 4.12 which plots the measured throughput efficiency vs b/a in a variety of traffic conditions. It can be seen that the efficiency decreases when the ratio of b/a increases. In order to ensure throughput efficiency $\geq 90\%$ it is required that

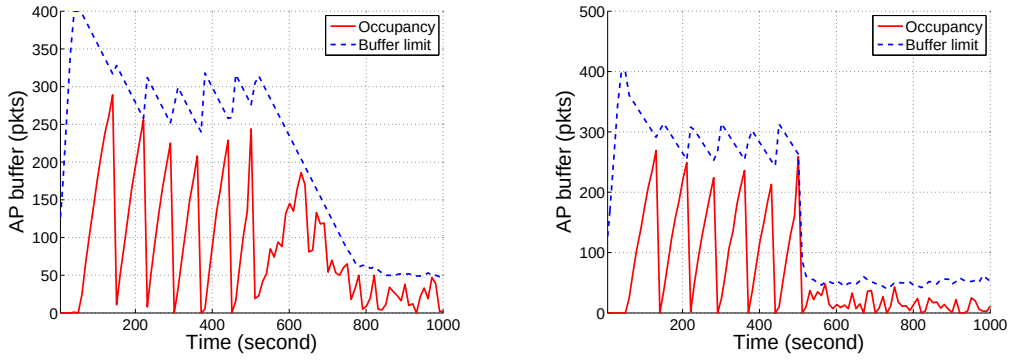
$$\frac{b}{a} \leq 0.1. \tag{4.13}$$

Combined with the stability condition in inequality (4.10), we have that $a = 10$, $b = 1$ are feasible integer values.

We note that when $0.1 \leq b/a \leq 1$ and there is 1 download and 10 uploads (or there are 10 downloads and 10 uploads), the desired buffer limits are less than 10 packets. With such short buffers TCP RTOs can occur frequently, and this leads to violation of the lower bound (4.12). This corresponds to an extreme operating regime however and for smaller values of b/a the lower bound is respected. Note also that we use 2 packets as the minimum buffer limit in Fig. 4.12 in order to exploit the behaviour of the ALT algorithm, but in practice large values, for example 30 packets is used here, would be used to avoid excessive RTOs.

4.6.5 Convergence rate

In Fig. 4.13(a) we illustrate the convergence rate of the ALT algorithm. There is one download, and at time 500s the number of upload flows is increased from 0 to 10. It can be seen that the buffer size limit converges to its new value in around 300 seconds or 5



(a) ALT (b) A*

Figure 4.13: Convergence rate of the ALT and A* algorithms. One download flow, $a = 10$, $b = 1$. At time 500s the number of upload flows is increased from 0 to 10.

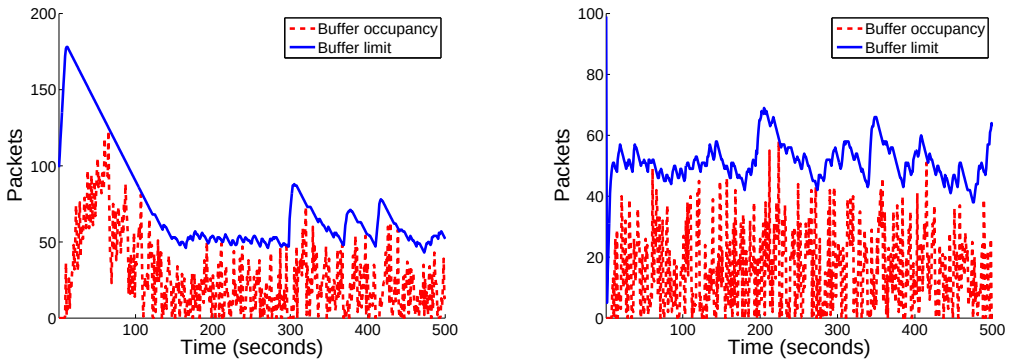
minutes. In general, the convergence rate is determined by the product $\lambda(0)\lambda(1)...\lambda(k)$. In this example, the buffer does not empty after backoff and the convergence rate is thus determined by $\lambda_f(k) = \frac{\alpha_T + b\gamma(k)}{\alpha_T + b}$. To achieve fast convergence, we require small $\lambda_f(k)$ so that $Q(k+1) = \lambda_f(k)Q(k)$ is decreased quickly to the desired value. We thus need large b to achieve fast convergence. However, $b = 1$ is used here in order to respect the stability condition in (4.10).

We note that slow convergence can also happen when rate adaptation is enabled, during initial startup stage, (see Fig. 4.14(a)), when channel states vary greatly due to channel noise, etc. In the next section, we address this issue by combining the ALT and the aBDP algorithms to create a new hybrid algorithm.

4.7 A*: The Final Algorithm

We can readily adapt quickly to changes in channel service rate by measuring the current bandwidth (see for example Fig. 4.7). We therefore provide a hybrid algorithm that combines the aBDP and the ALT algorithms. We call this hybrid the A* algorithm.

In the A* algorithm, we measure the mean inter-service time and calculate the instantaneous buffer limit Q_{aBDP} as before (see Section 4.5). This value is compared with the current buffer limit Q_{ALT} from the ALT algorithm. We use Q_{aBDP} as the new limit if



(a) ALT (b) A*

Figure 4.14: Convergence rate in the initial startup stage with the ALT and the A* algorithms. There are 1 download and 10 upload flows, and a=10, b=1.

$Q_{aBDP} < Q_{ALT}$, and Q_{ALT} otherwise.

When channel conditions change, the A* algorithm uses the measured bandwidth to adjust the buffer size promptly. The convergence rate depends on the updating interval (which is per-packet) and the smoothing weight W. As calculated in Section 4.5, it takes around 0.6 second for Q_{aBDP} to converge. Once Q_{aBDP} settles to a near constant value, the A* algorithm can further use the ALT procedure to fine tune the buffer size to exploit the potential reduction due to statistical multiplexing. The effectiveness of this hybrid approach when the traffic load is increased suddenly is shown in Fig. 4.13(b). In Fig. 4.14 we further illustrate the convergence rates of the ALT and the A* algorithms in the initial startup stage when there are 1 download and 10 upload flows. It can be seen that the ALT algorithm settles to the desired buffer size in 150 seconds, whilst it takes negligible time for the A* algorithm to do so.

4.7.1 Results

The basic impetus for the design of the ALT and the A* algorithms is to exploit the possibility of statistical multiplexing to reduce buffer sizes. Fig. 4.15 illustrates the performance of the A* algorithm when there are 10 downloads and no upload flows. Comparing with the

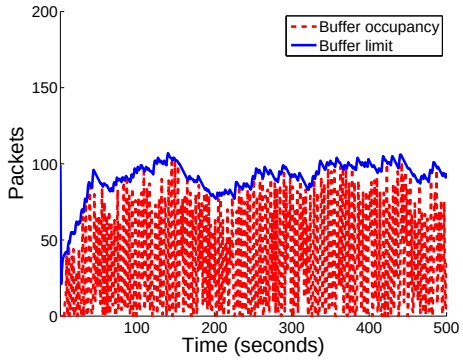


Figure 4.15: Buffer time histories with the A* algorithm, a=10, b=1, 10 downloads and no upload flows.

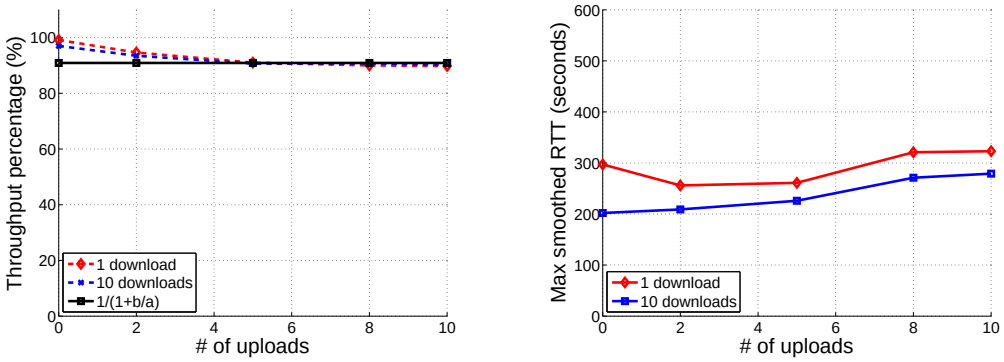


Figure 4.16: Performance of the A* algorithm as the number of upload flows is varied. RTT 200ms.

results in Fig. 4.8 using fixed buffers, we can see that the A* algorithm can achieve significantly smaller buffer sizes (i.e., a reduction from 350 packets to 100 packets approximately) when multiplexing exists.

As before, in Figs. 4.5 and 4.17, we further evaluate the A* algorithm when the RTTs are varied from 50-300ms and the number of uploads are varied from 0-10. Comparing these with the results of the aBDP algorithm (Fig. 4.5 and 4.6) we can see that the A* algorithm is capable of exploiting the statistical multiplexing where feasible. In particular, significant lower delays are achieved with 10 download flows whilst maintaining high throughput efficiency.

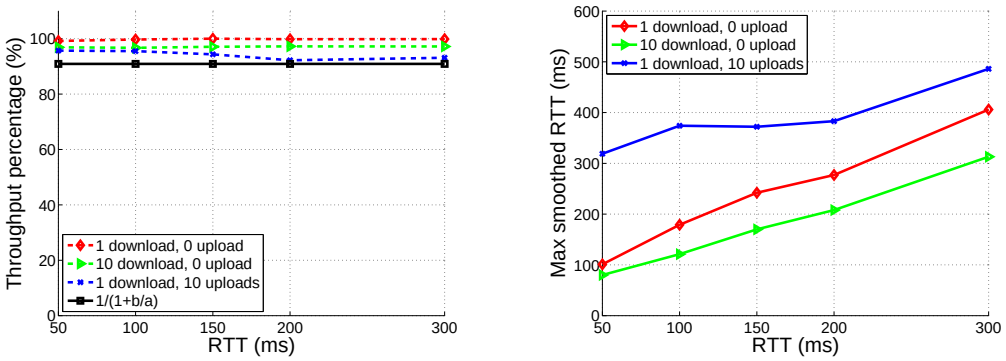


Figure 4.17: Performance of the A* algorithm as the wired RTT is varied.

4.8 Conclusions

While sizing router buffers in wired Internet is a key design issue, the size of buffers in WLANs has attracted little attention. We first show that the use of static buffers in WLANs leads to either undesirable channel under utilisation or unnecessary high delays. Although adaptive buffer sizing algorithms have been proposed for wired links, a number of fundamental new issues arise in WLANs. These new issues include the fact that the mean service rate is dependent on the level of channel contention, and packet inter-service times vary stochastically due to the random nature of CSMA/CA operation. As the classical rule of thumb is to provision buffers to be equal to the BDP, we propose an adaptive algorithm that emulates the BDP rule. To take advantage of the statistical multiplexing to decrease the buffer sizes where feasible, we then propose a second adaptive algorithm that achieves both high throughput efficiency and low delays simultaneously. As an enhancement to the second algorithm, we further design a third hybrid algorithm to adapt to channel variations promptly.

The physical layers of current WLANs can normally support multiple-rates [52] [31]. As discussed in earlier chapters, packet grouping (including the Burst ACK, the Block ACK and the aggregation schemes) is already a feature of 802.11e (via the TXOP mechanism) and is likely to play a key role in future high speed WLANs. However, consideration of the impact of rate adaptation and packet grouping on buffer sizing algorithms are left to future

work.

CHAPTER 5

Fairness at the MAC Layer

In addition to throughput and delay, fairness is another performance metric that should be considered for MAC layer design. Although the speed of WLANs is being improved dramatically, e.g., to 100 Mbps at the MAC layer in future 802.11n, the coverage remains confined to within several hundreds meters at best. With the aim of supporting more users or even to compete with cellular networks, larger coverage in future WLANs is a major design issue. One solution is to use the redundant bandwidth at local stations to relay traffic from nearby networks. Such a relay-based multi-hop network, if properly designed, may feature both "high speed" and "large coverage" simultaneously. However, using the 802.11 DCF scheme, gross unfairness can exist even without interference, hidden terminals and related issues.

In this chapter, we demonstrate that the TXOP mechanism of 802.11e, which has received little attention in the literature, can be readily exploited to restore/enforce fairness at the MAC layer in multi-hop networks. TXOP transmissions are essentially scheduled according to the Burst ACK scheme. As discussed in earlier chapters, other packet grouping schemes, e.g., Block ACK and aggregation schemes, share a similar nature in terms of bandwidth allocation. The trends indicated by the TXOP scheme extend therefore naturally to the other approaches which are key features for future 802.11n WLANs. For current devices, the proposed TXOP scheme is readily implementable in a simple and fully decentralised

way which requires no message passing.

5.1 Introduction

CSMA/CA based 802.11 technology is becoming increasing pervasive as the last-hop both in office environments and in the home. Looking ahead, the next step is likely to be towards greater use of multiple wireless hops. This not only includes the use of wireless for broadband backhaul infill, but also provision of municipal and rural wireless multi-hop networks. Within the home, moves towards Wifi-enabled multimedia distribution also lead almost inevitably to consideration of multiple wireless hops.

While there exists a considerable body of literature relating to 802.11 wireless multi-hop networks, much of this focusses on issues related to interference and routing which are well-known difficult problems in single channel 802.11 networks. For example, it has been observed that due to hidden terminal effects end-to-end traffic over more than around 3 hops tends to achieve rather limited throughput [42]. Recently, there has been great interest in the use of multi-radio multi-channel networks. This reflects technology road-maps, and also the fact that multi-radio architectures combined with appropriate channel allocations potentially offer practically effective solutions to interference management, see for example [89], [90], [121], [16], [76] [64] and references therein.

With this in mind, in this work we take as our starting point multi-radio multi-channel networks where the channel allocations have been chosen to avoid damaging interference[1]. We find that even when these issues are resolved in the aforementioned manner, gross unfairness can exist amongst competing flows. This unfairness is associated with the 802.11 MAC behaviour and can be particularly problematic in the context of multi-hop networks since unfairness can become amplified over multiple hops. That is, even without inter-channel interference and hidden terminals, a practical "high speed" and "large coverage" network is still not feasible.

[1]Note that interference and hidden terminals can also cause unfairness, which is a separate question to the one we consider here.

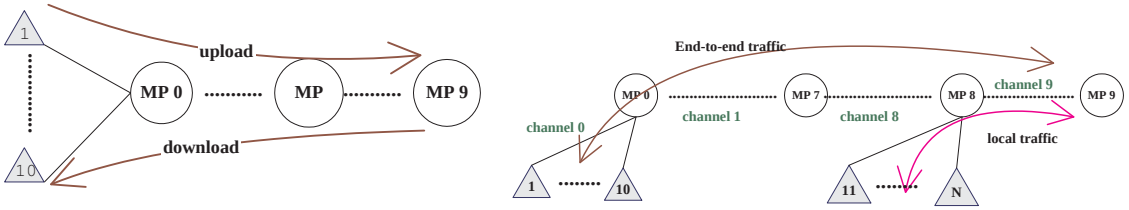
We propose the use of 802.11e's TXOP mechanism to restore/enforce fairness. We demonstrate the efficacy of this approach with both CBR and TCP traffic, and using experimental measurements we also demonstrate that it is capable of enforcing a variety of fairness requirements such as bandwidth reservation for local traffic. The proposed scheme is simple, implementable using off-the-shelf devices and fully decentralised (requires no message passing).

This chapter is organised as follows. Section 5.2 introduces prior work. In Section 5.3, we illustrate the unfairness behavior of 802.11 and identify the associated per-flow unfairness issue. In Section 5.4, we first provide a theoretical model to analyse the performance of the TXOP mechanism. Based on this model, we introduce a scheme to ensure per-flow fairness. Finally, Section 5.5 concludes this chapter.

5.2 Related Work

Most previous work in multi-hop networks has focussed on issues such as hidden terminals and interference (e.g., [89], [90], [121], [16], [76]). MAC-related unfairness has been studied in the context of single-hop 802.11 WLANs, e.g., see [63] [22] and references therein. However, fairness in multi-hop networks has received limited attention. In single-channel multi-hop networks, [42] illustrates that unfairness exists in parking lot deployments, and a congestion control algorithm is proposed to mitigate unfairness in [91]. The unfairness issue in [42] and [91] is caused by hidden terminals and interference. There has been even less work regarding the use of the TXOP mechanism. In [109], the authors evaluate the use of TXOP for stations with different physical rates.

To the best of our knowledge, there exists no prior work on enforcing/restoring per-flow fairness using 802.11e's TXOP in multi-hop networks.



(a) Scenario 1: end-to-end traffic only

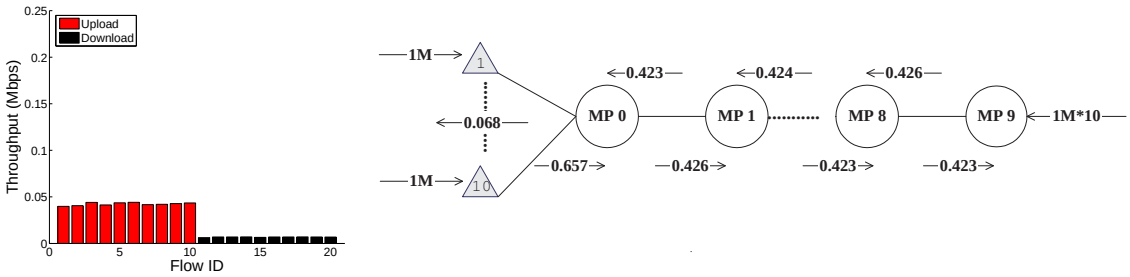
(b) Scenario 2: with local traffic

Figure 5.1: Illustrative wireless multi-hop scenarios.

5.3 Unfairness at Relay Stations

Before preceding we first describe the network setups used, see Fig. 5.1. Client stations are marked by shadowed triangles, and mesh points (MPs) by circles. MPs are stations that relay traffic for client stations. There are 10 MPs in both topologies. MP_9 acts as a gateway between the wireless multi-hop network and the wired Internet. Each MP has two radios that use channels in such a way that the channel in each hop is orthogonal to that in neighboring hops thereby avoiding interference between transmissions on different hops. Hence there are no hidden terminals. We assume that the set of routes from sources to destinations are already obtained by routing protocols such as those discussed in [34] and [36]. The routes are stable during the considered sessions' life time. We only consider single-path routing. Under these routing assumptions, all routes in the network form a tree topology. We use *station* to refer to any wireless devices (both client stations and MPs). We say *client station* when referring to wireless devices other than MPs.

Interestingly, even with such a simple network setup (no interference/hidden terminals, fixed routing, standard 802.11 parameters) significant unfairness can exist between traffic flows. This is illustrated, for example, in Fig. 5.2(a) which shows the measured throughput achieved by a mix of upload and download CBR flows (with one upload and one download flow per client station; sources for download flows and destinations for upload flows lie in the wired network). It can be seen that the throughput achieved by the upload flows is approximately an order of magnitude greater than that achieved by the download flows.



(a) Per flow throughput

(b) Per hop throughput (in Mbps)

Figure 5.2: CBR results for scenario in Fig. 5.1(a). One upload and one download flow per client station. Per flow throughput is shown in Fig. 5.2(a). Per hop aggregate throughput (in Mbps) is plotted in Fig. 5.2(b). Simulation parameters listed in Table 5.1.

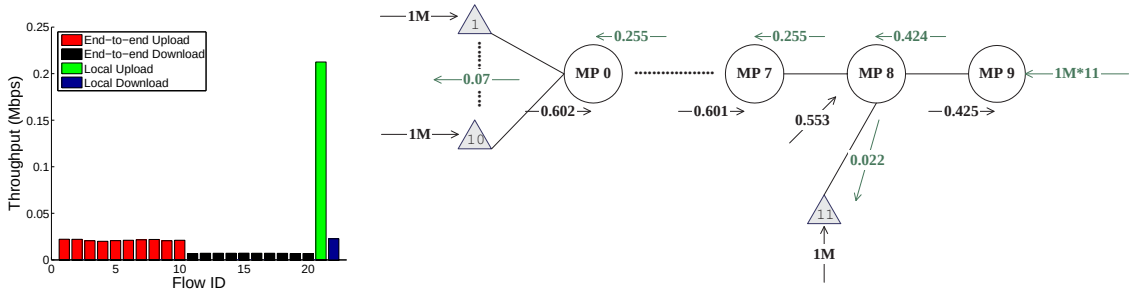
We can gain some insight into the source of this unfairness by looking at the corresponding per hop measurements shown in Fig. 5.2(b). It can be seen that on the relay hops, the aggregate throughput of the upload flows and of the download flows are approximately equal, as might be expected. However, at the left-hand hop, between the client stations and MP_0, the situation is very different. We can understand this behavior by noting that the 802.11 MAC ensures that roughly the same number of transmission opportunities are allocated to every station, including the MPs. Thus, if there are n_0 client stations, we expect each of them to obtain roughly a $1/(n_0 + 1)$ share of the bandwidth, and similarly for the MP to obtain a $1/(n_0 + 1)$ share. The n_0 upload flows therefore together obtain an $n_0/(n_0 + 1)$ share whereas since all of the download flows must be transmitted via the MP and so they can only obtain approximately a $1/(n_0 + 1)$ share altogether. We can confirm this approximate reasoning by noting that the aggregate upload throughput at the left-hand hop in this example is measured to be 0.657Mbps while the aggregate download throughput is 0.068Mbps. The ratio of upload to download throughput is thus 9.66, i.e. close to the value of $n_0 = 10$.

This type of unfairness is not new and has previously been observed in the context of single-hop WLANs (e.g., [63]). However, the impact of this unfairness can be far greater in a multi-hop context.

To see this, consider the multi-hop network in Fig. 5.1(b) with one local station at

T_{SIFS} (μs)	10
Idle slot duration (σ) (μs)	20
T_{DIFS} (μs)	50
CW_{min}	31
CW_{max}	1023
Retry limit	4
Packet size (bytes)	1000
PLCP rate (Mbps)	1

Table 5.1: MAC/PHY parameters used in multi-hop simulations.



(a) Per flow throughput

(b) Per hop throughput (in Mbps)

Figure 5.3: CBR results without TXOP for scenario in Fig. 5.1(b) with one client station at MP_8 (i.e. $N = 11$). Per flow throughput is shown in Fig. 5.3(a). Per hop aggregate throughput (in Mbps) is plotted in Fig. 5.3(b). Simulation parameters listed in Table 5.1. Note that 0.553 Mbps in Fig. 5.3(b) is the aggregate throughput of local upload originating from station 11 and relay uploads from MP_7 to MP_8.

MP_8. End-to-end traffic from the left-hand stations, numbered 1-10 in Fig. 5.1(b), now has to compete with the traffic from station 11 at the MP_8 hop. The foregoing unfairness effect now acts multiplicatively at hops MP_0 and MP_8, greatly amplifying the level of unfairness. This effect is illustrated in Fig. 5.3(a). Here, stations 1-11 each carries one upload and one download flow, yielding 11 upload and 11 download flows in total. It can be seen from Fig. 5.3(a) that the upload flow at station 11 gains much greater throughput than the other flows.

What is happening is that at MP_8, each local upload flow obtains roughly a $1/(n_8 + 2)$ share of the bandwidth, where $n_8 = 1$ is the number of client stations associated with MP_8 and the 2 on the denominator accounts for end-to-end upload traffic from MP_7 and download traffic from MP_8. The *aggregate* upload traffic from stations 1-10 also obtains a

$1/(n_8+2)$ share (corresponding the the share of upload transmission opportunities allocated to MP_7). Thus each individual upload flow from stations 1-10 obtains only a $1/10(n_8+2)$ share. In line with this analysis, Fig. 5.3(a) confirms that the upload flow from station 11 obtains roughly an order of magnitude greater throughput than the upload flows from stations 1-10.

The aggregate download traffic to stations 1-11 also obtains a $1/(n_8+2)$ share at the MP_8 hop. The download traffic to stations 1-10 then has to compete against the upload traffic from stations 1-10 for transmission opportunities at MP_0. This creates further unfairness. As discussed above, at the MP_0 hop there is approximately an order of magnitude unfairness between upload and download flows and this can been seen in Fig. 5.3(a).

The setup in Fig. 5.1(b), where download traffic must contend at two hops, is already sufficient to create a level of unfairness whereby download traffic to stations 1-10 is almost starved of throughput. By introducing contention at further relay hops, the unfairness can evidently be amplified still further. In effect, the potential exists for almost arbitrary levels of unfairness to exist between competing traffic flows in a multi-hop setting. Note that this effect is not associated with interference or other sources of unfairness. Rather it is a direct consequence of the properties of the 802.11 MAC.

5.4 Achieving Per-Flow Fairness

Since the unfairness behaviour noted above is associated with the MAC operation, it is natural to seek to improve fairness by investigating changes at the MAC layer. In this chapter, we propose the use of 802.11e's TXOP mechanism to restore/enforce fairness. We first model the functionality of TXOP, then discuss how to achieve fairness with it.

For ease of discussion, we specify the duration of a TXOP (denoted to be K) as the number of packets. That is, by saying $K_i = k$ we mean a duration during which a maximum of k packets can be transmitted by station i with a specific PHY date rate which does not change.

5.4.1 Modelling TXOP

We design a finite-load model to quantify TXOP's functionality. We use the approach proposed by Bianchi in [20] and extended in [73] to allow us to calculate the impact of TXOPs.

In multi-hop CSMA/CA based networks, modelling the relay traffic distribution from a previous hop is still an open problem. Following common practice (e.g., [43, 37]) we assume that the offered load at station i is an independent Poisson process with mean rate of λ_i bits/sec.

We therefore consider an intermediate hop between the source and the destination with relaying MP denoted as MP' and $n-1$ associated MPs/user stations. The quantity of interest is the throughput of station (recall that by station, we mean both MPs and user stations) i

$$x_i = \frac{P_{i,s}E[L_i]}{E[T]} \tag{5.1}$$

where $P_{i,s}$ is the probability that station i has a successful transmission, $E[L_i]$ is the expected number of bits transmitted in a transmission, and $E[T]$ is the expected slot duration (in seconds).

Let τ_i be the probability that station i attempts transmission, and p_i be the probability of station i collides with others in a real slot time. Following [73], we assume that for each station i there is a constant probability $1-q_i$ that the station's queue has no packets awaiting transmission in an expected slot. The probability q_i that one or more packets are available in $E[T]$ time is given by $q_i = 1 - e^{(-\lambda_i/K_i)E[T]}$ where K_i is the TXOP values, in packets.

Using a similar coupling technique as in [20], the probability τ_i can be modelled as a function of p_i and q_i using a Markov chain for the contention windows (see Equation (6) in [73]). A second relation relating τ_i and p_i is

$$1 - p_i = \prod_{j \neq i}(1 - \tau_j), \tag{5.2}$$

i.e., there is no collision for station i when all other stations are not transmitting. With n stations, $p_1, \ldots, p_n$ and $\tau_1, \ldots, \tau_n$ can be solved numerically.

Let P_{tr} be the probability that at least one station is transmitting, we then have that

$$P_{tr} = 1 - \prod_{i=1}^{n}(1 - \tau_i). \tag{5.3}$$

Let $P_{i,s}$ be the probability that station i successfully wins a transmission opportunity (which may involve transmitting one or multiple packets), then

$$P_{i,s} = \tau_i \prod_{j \neq i}(1 - \tau_j), \tag{5.4}$$

and combining with Equation (5.2), we have that

$$P_{i,s} = \tau_i(1 - p_i). \tag{5.5}$$

Let P_c be the probability that more than one station starts transmissions at the same time, we have that

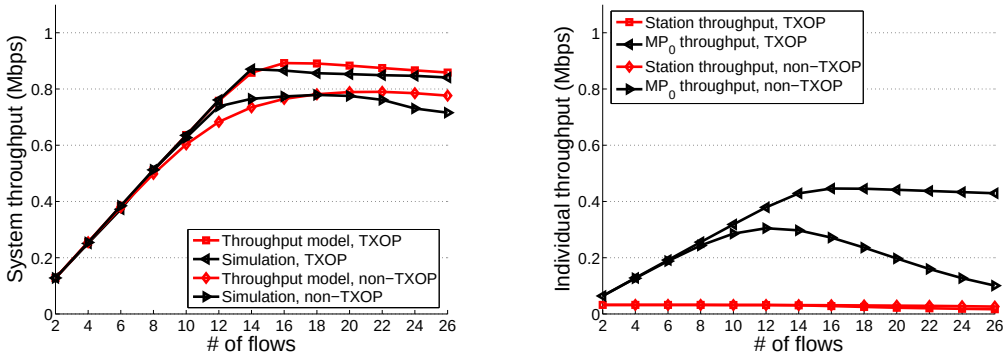
$$P_c = P_{tr} - \sum_{i=1}^{n} P_{i,s}. \tag{5.6}$$

Now we can represent the expected slot duration as

$$E[T] = (1 - P_{tr})\sigma + \sum_{i=1}^{n}(P_{i,s}T_{i,s}) + P_cT_c. \tag{5.7}$$

where σ is the idle slot duration, T_C is the collision duration, and $T_{i,s}$ is the successful duration. In the non-TXOP case, both T_C and $T_{i,s}$ correspond to a packet transmission and associated overhead, while in the TXOP case multiple packets can be transmitted.

There are two variables ($T_{i,s}$ and $E[L_i]$) in Equation (5.1) that are still unknown, with their relationship being that $T_{i,s} = E[L_i]/R + \Delta$ where R (bits/sec) denotes the physical rate, and Δ (in seconds) denotes the overhead including DIFS for 802.11 (AIFS for 802.11e), SIFS and ACKs. For calculating $E[L_i]$, we use an approximation that station i always waits until there are enough packets to transmit in one TXOP (as we will see that analysis with this assumption matches the simulations well), hence $E[L_i] = K_i * L$ where K_i is the TXOP duration in packets at station i and L is the packet size in bits. The aggregate overhead in



(a) System throughput (b) Individual throughput

Figure 5.4: Model vs. simulation for TXOP and 802.11 DCF.

one TXOP is thus $\Delta_i = DIFS/AIFS + K_i(2*SIFS + T_{ack} + 2*T_{phy,hdr} + T_{mac,hdr} + T_{other,hdr})$. The model is now complete.

This analysis is verified against simulations. We use a process with mean rate of 64 Kbps. A upload and a download flows are associated with user station and the MP'. The packet size, physical data rate and physical basic rate used is 80 bytes, 11Mbps and 1Mbps, respectively. The other parameters are listed in Table 5.1. In Fig. 5.4(a), we illustrate the results for both TXOP enabled 802.11e (labelled as TXOP) and 802.11 DCF (labelled as non-TXOP). It can be seen that (i) as the number of flows increases, in both cases the system throughput increases to a maximum level and remains thereafter, (ii) the use of TXOP allows higher throughput to be sustained compared with the 802.11 DCF. In Fig. 5.4(b), the individual throughput achieved by MP' and user stations is depicted. We can see that the throughput achieved by the 802.11 DCF drops rapidly when there are more than 12 pairs of traffic flows. With TXOP, however, MP' can maintain a near constant throughput after the channel becomes saturated. In both cases, user-stations throughput decrease slightly with the number of traffic flows. Here, MP' uses the number of flows as the TXOP value.

For stations which are backlogged, we have that the probability $q_i = 1$. According to Equations (5.4), (5.2), we know that these saturated stations have the same transmission success probability (represented as P_s^*) in a slot. The throughput ratio between these

stations is thus proportional to their TXOPs. i.e.,

$$\frac{x_i}{x_j} = \frac{P_s^* E[L_i]}{P_s^* E[L_j]} = \frac{K_i}{K_j}. \tag{5.8}$$

Recall that all stations are using the same parameters such as CW_{min}, CW_{max}, AIFS, etc.

In order to quantify the relationship between all stations that may or may not be saturated, we define the effective TXOP duration K_i' used by station i to be

$$K_i' = \frac{P_{i,s} E[L_i]/L}{P_s^*} \tag{5.9}$$

where $P_{i,s}$ is the actual successful transmission probability, L is the packet length. Observe that $K_i' = K_i$ for saturated stations, but $K_i' \leq K_i$ for stations which are not persistently saturated. That is, saturated stations can use up to the maximum assigned TXOP, but non-saturated stations can not. The advantage of working in terms of K_i' is that the throughput ratio between any stations can be written as

$$\frac{x_i}{x_j} = \frac{P_{i,s} E[L_i]}{P_{j,s} E[L_j]} = \frac{K_i'}{K_j'}, \tag{5.10}$$

i.e., this relationship holds for both saturated and non-saturated stations. This equation says that the ratio of throughput achieved by any two stations is equal to the ratio of their TXOPs. We can then control fairness between stations as long as proper TXOPs are chosen, which will be detailed in the following section.

5.4.2 The Proposed Scheme

Let the number of flows with packets queued at MP_i on channel l be n at a transmission opportunity. We select TXOP duration $K_{l,i} = n$ and use a modified queuing discipline (e.g., [96]) that serves one packet per flow at each transmission opportunity. Note that TXOP may change from transmission opportunity to transmission opportunity as the mix of queued packets varies and so the scheme automatically adapts to changes in the number of flows carried by a station.

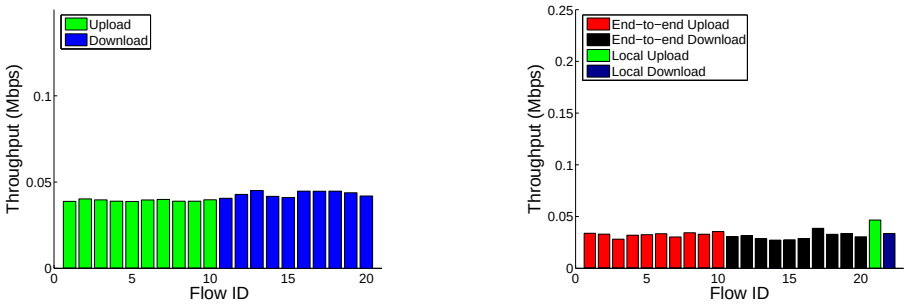
It follows immediately from (5.10) that the ratio of station throughput is approximately equal to the ratio of flows carried. In practice, this dynamic TXOP allocation scheme can

be simplified to select $K_{l,i}$ to equal the average number of flows carried by station i[2], and by employing FIFO queuing (rather than per-flow fair queueing) with little loss in performance – see the example below. There is no message passing required since each station is able to determine the number of flows it carries by inspection of its outgoing packet stream and thus the scheme is fully decentralised, greatly facilitating management and roll-out.

5.4.3 Remarks

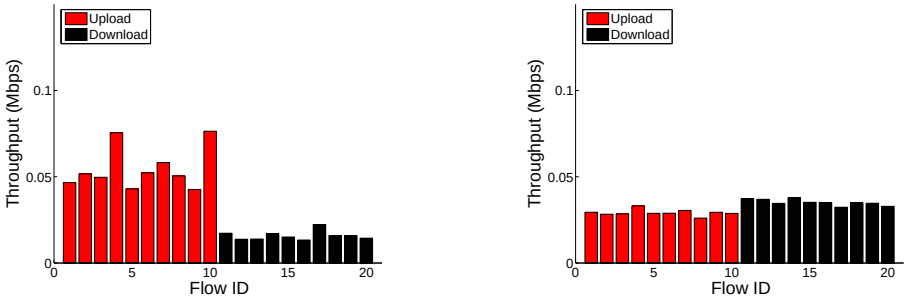
We comment that with this TXOP approach a station transmits n packets in a single burst. For n large, this can result in the station occupying the channel for a substantial consolidated period of time and this may, for example, negatively impact competing delay-sensitive traffic. We can address this issue in a straightforward manner by using multiple smaller TXOPs instead of a single one. When using smaller packet TXOPs, it is necessary to ensure a corresponding increase in the number of transmission opportunities won by the station. This can be achieved by using a smaller value of CW_{min} for the prioritised traffic class at the station. It is shown in [17] that competing traffic classes gain transmission opportunities approximately in inverse proportion to their values of CW_{min}. Let k denote the ratio of the stations CW_{min} value to the base value used in the network (e.g. 31 in 802.11b/g). 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 downstream flows. We then complement this with use of TXOP for fine grained adjustment of the packet burst lengths. For example, when $k = 2$ we halve the value of CW_{min} and also halve the value of TXOP to $n/2$. Hence, fine grained prioritisation can always be achieved while avoiding unduly large packet bursts.

[2]It is important to note that for a station that is assigned a long TXOP length, if during a transmission opportunity it has no packets to send (the network interface queue is empty) then that transmission opportunity is ended automatically. That is, if the offered load at a station is too low to make full use of its allocated TXOP share (or due to burstiness of the traffic, the interface queue is empty from time to time), the excess is not lost but rather becomes available on a best effort basis for use by other stations in the network.



(a) Scenario 1 (b) Scenario 2 (with N=11)

Figure 5.5: CBR results with TXOP per station fairness scheme. TXOP = 10 at MP_0, TXOP = 10 at MP_7, TXOP = 11 at MP_8.



(a) Without TXOP (b) With TXOP

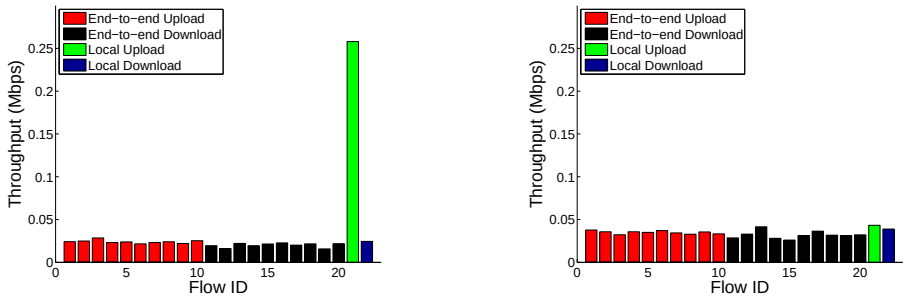
Figure 5.6: TCP results for topology in Fig. 5.1(a). Simulation parameters listed in Table 5.1.

5.4.4 CBR Results

We revisit the earlier multi-hop examples, and illustrate the impact of the proposed TXOP assignment scheme with CBR traffic. For the topology in Fig. 5.1(a), Fig. 5.5(a) demonstrates the impact of this change – it can be seen that fairness is restored between upload and download flows. For the second topology in Fig. 5.1(b), the corresponding results are shown in Fig. 5.5(b). Again, it is evident that fairness is restored.

5.4.5 TCP Results

Since TCP currently carries the vast majority of network traffic it is important to investigate the performance of the proposed scheme with TCP.



(a) Without TXOP (b) With TXOP

Figure 5.7: TCP results for topology in Fig. 5.1(b) (with N=11). Simulation parameters listed in Table 5.1.

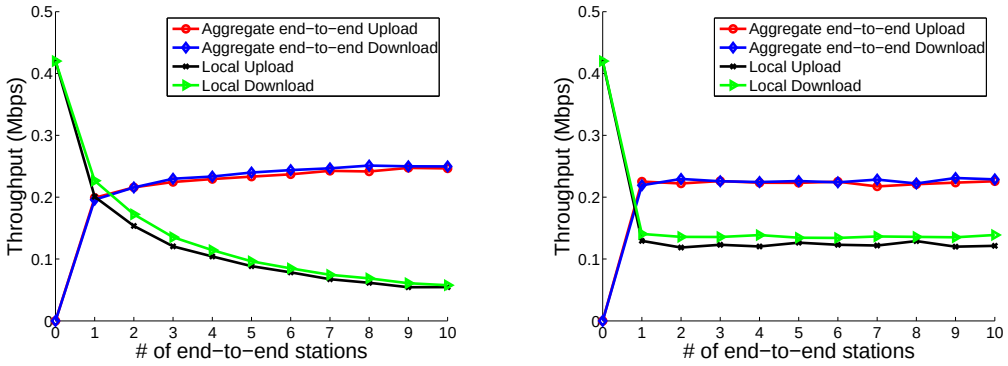
Fig. 5.6(a) and Fig. 5.7(a) shows the throughput of TCP upload and download flows for, respectively, the network topologies in Figs. 5.1(a) and 5.1(b). As expected, unfairness between upload and download flows is evident.

The performance with the proposed TXOP scheme is illustrated in Figs. 5.6(b) and 5.7(b). It can be seen that, as required, fairness is restored. Here, we use the method introduced in Chapter 4 to ensure reliable TCP ACK transmissions, i.e., TCP ACKs are stored in a queue with high priority by assigning it $CW_{min} = 3$, $CW_{max} = 7$ and $AIFS = 2$, and TCP data packets use a low priority queue with parameters $CW_{min} = 31$, $CW_{max} = 1023$ and $AIFS = 6$.

5.4.6 Prioritising Local Traffic

In a large wireless multi-hop network, per-flow fairness can lead to local access traffic being starved of bandwidth at MPs close to the wired gateway. For example, consider the network topology in Fig. 5.1(b) with one local station at MP_8 (i.e. N=11). Figure 5.8(a) plots the throughput of this local station as the number M of client stations at MP_0 is varied between 0 and 10. The per-flow scheme discussed above is used. The total number of destination stations is $2M+2$ and, as expected, the bandwidth share of the local station is proportional to $1/(2M+2)$ and so decreases towards zero as M increases.

Of course, this might be reasonable in some circumstances, e.g. where the network



(a) Per-flow fairness

(b) Protecting local traffic ($\alpha = 0.25$)

Figure 5.8: Protecting local stations for topology in Fig. 5.1(b). One local station at MP_8 (N=11) and number of client stations at MP_0 is varied. Simulation parameters listed in Table 5.1.

infrastructure is provided by a third party and resources are to be shared equally regardless of proximity to the gateway. However, it can also be undesirable, e.g., where the multi-hop network is formed by individual MP owners allowing shared access, subject to the proviso that each MP owner retains a minimum bandwidth share for his/her own traffic. In general, the choice of the most appropriate fairness requirement here is essentially a policy decision for the network operator. Our main point here is that TXOP does indeed provide the necessary flexibility to allow fairness to be controlled in a simple and practical manner. In particular, we show in this section that protecting local traffic can also be readily achieved using TXOP.

We consider two approaches for protecting local traffic. The first approach only requires adjustment of TXOP but makes use of two 802.11e traffic classes. This approach has the virtue of simplicity and requires only straightforward adjustment of the MAC TXOP parameters. It does, however, require the use of two traffic classes. The 802.11e standard provides a total of only four traffic classes. Hence, in mixed traffic environments with voice, video, data etc we can quickly run out of traffic classes. We therefore also present an alternative approach to protecting local traffic that uses only a single traffic class. This benefit comes at the cost of the need for changes (a software change only) to the queueing

discipline employed at the network interface queue.

Protecting local traffic using two traffic classes

As before, let n denote the number of neighbouring wireless stations (client stations and other MPs) on a given radio channel. At the i'th wireless station we assign local flows to one traffic class which is assigned a TXOP value of $N_{i,local}$ packets. Relay flows are assigned to a second traffic class with TXOP $N_{i,relay}$.

Summing over all of the neighbouring stations, the share of transmission opportunities allocated to local traffic is therefore $\frac{\sum_{i=1}^{n} N_{i,local}}{\sum_{i=1}^{n}[N_{i,local}+N_{i,relay}]}$. We require to allocate a minimum proportion α of transmission opportunities to local traffic. That is, we require $\frac{\sum_{i=1}^{n} N_{i,local}}{\sum_{i=1}^{n}[N_{i,local}+N_{i,relay}]} = \alpha$. Rearranging yields

$$\sum_{i=1}^{n} N_{i,local} = \frac{\alpha}{1-\alpha} \sum_{i=1}^{n} N_{i,relay}. \tag{5.11}$$

Any TXOP allocation that satisfies constraint (5.11) will ensure that local traffic receives an α share of the available transmission opportunities.

Note that freedom remains as to the selection of $N_{i,local}$ and $N_{i,relay}$, and this can be used to further control fairness. In particular, we consider selecting $N_{i,local} = k_l n_{local,i}$ and $N_{i,relay} = k_r n_{relay,i}$ where $n_{local,i}$ is the number of destination stations associated with the outgoing local traffic from node i and $n_{relay,i}$ is the corresponding relay traffic value. The scaling factors k_l and k_r are selected to satisfy constraint (5.11). By similar arguments to those in Section 5.4.1, this choice for $N_{i,local}$ ensures that the local traffic transmission opportunity share α is allocated among local traffic flows on a per station basis. Similarly, with the relay traffic share $1-\alpha$.

We illustrate the impact of this strategy in Fig. 5.8(b). In this example we select $\alpha = 0.25$, i.e., local traffic is allocated a minimum of 25% of the available transmission opportunities. It can be seen that the bandwidth share of the the local traffic is now lower bounded as required.

Protecting local traffic using one traffic class

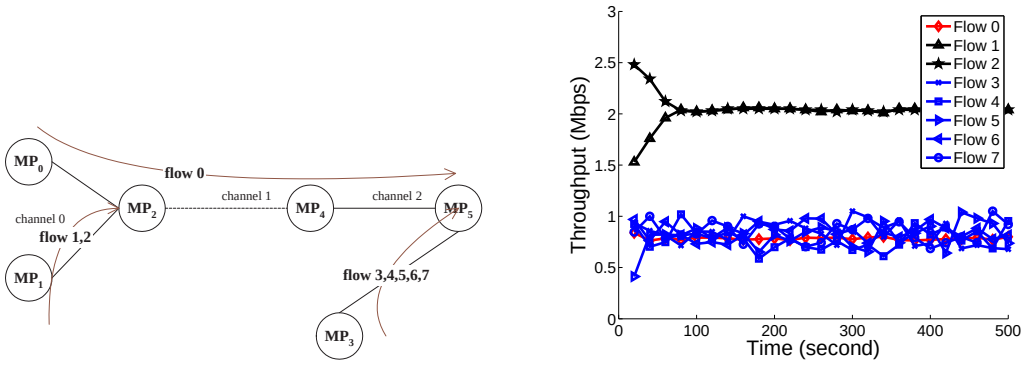
We can also use a single traffic class for both local and relay traffic. Let the TXOP value for this class at the i'th wireless station be T_i.

We now partition the total transmission time T_i into intervals $T_{i,local}$ and $T_{i,relay}$ with $T_{i,local} + T_{i,relay} = T_i$ and $T_{i,relay} = (1-\alpha)/\alpha\ T_{i,local}$. Then $T_{i,local}/T_i = \alpha$ and $T_{i,relay}/T_i = (1-\alpha)$. By partitioning TXOP in this way at every wireless station, we can protect the bandwidth share of local traffic as required. Specifically, on winning a transmission opportunity at station i, we use $T_{i,local}$ of the available transmit time to send local traffic and $T_{i,relay}$ to transmit relay traffic.

To implement this approach requires a software change that can be implemented in practice in a number of ways – for example, we can modify the wireless card device driver to perform a selective walk of the interface queue on each transmission opportunity. In practice, the interface queue is commonly divided into a device queue and a txqueue, with packets queued by the network stack in the txqueue before transferral to the device queue. While the device queue service discipline may be hardware dependent, the txqueue service discipline is generally implemented within the operating system kernel and can be readily modified. The proposed approach can thus be readily implemented by use of a TXOP-sized device queue combined with a selective walk of the txqueue when transferring packets from the txqueue to the device queue.

5.4.7 Tendency to Max-min Fairness

The proposed scheme in Section 5.4.2 considers providing per-flow fairness using the TXOP mechanism of 802.11e. The resulting allocation is close to max-min fair [18] in the considered topologies (as each flow achieves a same rate). This can also be seen for the parking lot topology in Fig. 5.9(a) which is often used to illustrate fairness of end-to-end traffic in general network setups (in both wired networks, e.g., [72] [78] [62]) and wireless networks



(a) Topology (b) Max-min fairness

Figure 5.9: Test-bed results. Parameters used is listed in Table 5.2.

T_{SIFS} (μs)	16
Idle slot duration (σ) (μs)	9
T_{DIFS} (μs)	34
CW_{min}	15
CW_{max}	1023
Retry limit	11
Packet size (bytes)	1500

Table 5.2: MAC and PHY parameters used in test-bed implementation.

(e.g., [42]). According to [72], vector

$$\{\frac{c_2}{6}, \frac{(c_0 - \frac{c_2}{6})}{2}, \frac{(c_0 - \frac{c_2}{6})}{2}, \frac{c_2}{6}, \frac{c_2}{6}, \frac{c_2}{6}, \frac{c_2}{6}, \frac{c_2}{6}\}$$

is the unique max-min allocation where c_i is the current capacity of channel i.

We have implemented the topology shown in Fig. 5.9(a) using a test-bed[3] constructed from Soekris net4801[4] stations with Atheros 802.11a/b/g miniPCI cards. All stations run the Linux 2.6.21.1 kernel with a version of the MADWiFi[5] wireless driver which is customised to allow the prioritisation described in this chapter. In order to ensure a non-interfering channel allocation at each MP and to avoid interference with neighboring WLANs, all of these tests are performed with 802.11a channels. We use channels 40, 48 and 56 of 802.11a for channels 0, 1 and 2, respectively. The channel rate is fixed at

[3]Main contributions of this test-bed implementation belong to Venkataramana Badarla and David Malone of our group.

[4]http://www.soekris.com/net4801.htm

[5]http://sourceforge.net/projects/madwifi/

6Mbps. To implement dual-radio MPs, we join two net4801 stations at 100 Mbps with a cross-over cable to form a single logical MP. Routing in the network is statically configured. We use iperf[6] to generate TCP traffic and data is collected from both iperf and tcpdump. All the control operations such as initializing flows, collecting statistics etc., are carried out using the wired Ethernet of net4801 stations. SACK enabled TCP NewReno with a large receiver buffers (16 MBytes) is used. The TCP data packet size is 1500 bytes. All other TCP parameters are taken as the default value of Linux Kernel 2.6.21.1. To prioritise TCP ACK packets, we put ACK packets into the highest priority queue (Queue 3) which is assigned with $CW_{min} = 3$, $CW_{max} = 7$ and $AIFS = 2$. TCP data packets are collected into lower priority queue (Queue 2) which is assigned with $CW_{min} = 31$, $CW_{max} = 1023$ and $AIFS = 6$.

Using the proposed scheme, the resulting allocation is shown in Fig. 5.9(b) where we use 0, 5000 and 12000 μs (which correspond to durations of transmitting 1, 2 and 5 packets) as TXOPs for MP_0, MP_1 and MP_3. It can be seen that an approximate max-min allocation is achieved[7].

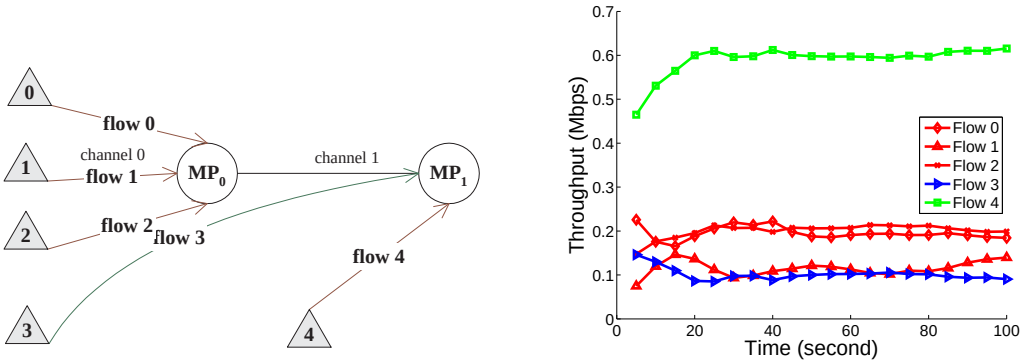
5.5 Conclusions and Future Work

We have shown in this chapter that gross unfairness can exist in multi-hop CSMA/CA based networks if the 802.11 DCF scheme is used at the MAC layer. We have demonstrated that the TXOP mechanism of 802.11e can be used to ensure/restore fair allocation. The proposed TXOP based scheme is implementable on standard hardware in a simple and fully distributed way without the needs of message passing.

The network setups considered are 802.11 based multi-radio multi-hop networks, where there are no packet losses due to MAC layer contention, channel noise and interference, etc.

[6] http://dast.nlanr.net/Projects/Iperf/

[7] Here, $c_2 = 4.5$ Mbps and $c_0 = 4.75$ Mbps – the capacity at each hop is not the same since 802.11 throughput is dependent on the number of contending stations, which differs at each hop. Flow 0 and flows 3, 4, 5, 6, 7 achieve the same throughput of 0.75 Mbps, while flow 1 and 2 achieve the same throughput of 2 Mbps.



(a) Topology (b) Results

Figure 5.10: Impact of excessive MAC layer contention on fairness.

When these factors are present however, tuning TXOP alone may not be sufficient. For example, in Fig. 5.10 we show the impact of MAC layer contention. There is only one end-to-end flow, i.e., flow 3, which traverses two orthogonal hops. A retry limit of 1, $AIFS = 2$, $TXOP = 1$, $CW_{min} = 7$ and $CW_{max} = 31$ are used for TCP data packets[8]. If the results were max-min fair, we would expect flows 0, 1, 2 and 3 to achieve the same throughput, namely about 1/4 of the capacity of channel 0. However, we observe that flows 2 and 3 obtain a substantially lower throughput than flows 0 and 1. This happens because frequent packet losses occur due to the small retry limits and contention window sizes. Even if the values used are unrealistically small in this example, we can expect that when there are large numbers of stations, similar behaviour would happen even if standard parameters are used.

Using static larger than standard contention windows and retry limits may mitigate the impact of excessive MAC layer contention. However, channel capacity in CSMA/CA based networks is load-dependent. When traffic load is varied, these values should be updated accordingly. That is, dynamic solutions (such as those proposed in [126] or [48]) may be useful to enhance the proposed TXOP scheme so as to tune related parameters to minimise contention losses. Further, if losses are caused by channel noise or hidden/exposed terminals,

[8]TCP ACK packets are still prioritised with retry limit of 2, $AIFS = 1$, $TXOP = 50$, $CW_{min} = 3$ and $CW_{max} = 7$.

tuning TXOP, contention window sizes and other parameters together may be necessary to ensure fairness. We leave the considerations for these cases to future work. In future work, we will also investigate the possibility of providing more general fairness criteria such as proportional fairness.

CHAPTER 6

Conclusions and Future Work

We started the thesis by developing an analytical model to explicitly model the impact of channel errors on MAC layer throughput. Using this model, we showed that the DCF scheme is not fundamentally capable of supporting 100 Mbps at the MAC layer which is the key aim of the future IEEE 802.11n standard. We then extended the DCF model to evaluate the performance of the BTA scheme – the most promising scheme in terms of improving throughput over the DCF scheme. Using the extended model, we demonstrated that BTA can support 100 Mbps in future 802.11n only if the physical layer rate is faster than 500 Mbps.

To further decrease overhead that causes the inefficiency of the DCF and the BTA scheme, we proposed a novel scheme whereby transported information is partially acknowledged and retransmitted. An analytic model is developed to evaluate the throughput and delay of AFR over a noisy channel and to compare AFR with competing schemes in the literature. If the application is saturated CBR traffic, analysis shows that AFR can achieve at the MAC layer more than 100 Mbps when the PHY rate is faster than 216 Mbps. As a complement to the theoretical analysis, we investigated by simulations the impact of AFR on the performance of realistic applications including TCP, HDTV and VoIP.

We next considered buffer sizing 802.11e WLANs. Although buffer sizing algorithms have been proposed for wired links, a number of fundamental new issues arise in WLANs.

These new issues include the fact that the mean service rate is dependent on the level of channel contention, and packet inter-service times vary stochastically due to the random nature of CSMA/CA operation. As the classical rule of thumb is to provision buffers to be equal to the BDP, we proposed an adaptive algorithm that emulates the BDP rule. To take advantage of the statistical multiplexing to decrease the buffer sizes where feasible, we then proposed a second adaptive sizing algorithm to achieve both high throughput efficiency and low delays simultaneously. As an enhancement to the second algorithm, we further designed a third algorithm to adapt to channel variations promptly.

In the last part of the thesis, we highlighted that gross unfairness can exist between competing flows in the relay based multi-hop networks even when orthogonal channels are used in neighbouring hops. We proposed the use of 802.11e's TXOP mechanism to enforce/restore fairness. We presented a readily implementable scheme which works in a simple and fully decentralised way without requiring any message passing.

There are interesting issues yet to be resolved at CSMA/CA based MAC layer.

First, great popularity means that in scenarios such as city centers, deployments of WLANs are becoming more and more dense. Interference related issues (notably hidden and exposed terminals) thus pose increasing challenges. While prior art regarding these issues mainly focusses on channel allocation, power control, carrier sensing range tuning, etc., great scope still exists to improve performance in such circumstances.

Second, rate adaptation has been one of the most important techniques for accommodating variations in wireless channel. However, real world measurements have shown that current implemented rate adaptation algorithms often perform poorly. We would like to consider improving rate adaptation in the future work. Indeed, the AFR scheme that we proposed for future very high-speed WLANs bears the similar purpose as rate adaptation algorithms, comparison (or better combination) of the AFR with properly designed rate adaptations is thus desired.

Third, while we considered improving CSMA/CA performance in very high-speed WLANs, it turned out that in the other extreme case where very low-speed is used at the physical

layer, the inefficiency suffered by CSMA/CA MAC layers might be quite similar, suggesting that similar techniques can be readily extended to that area.

Fourth, with activities of 802.11n standard are nearly close to the end and pre-11n wireless cards are available now, it is about the right time to test the real performance of it. Two issues deserve particular attention in this regard: one is compatibility with 802.11b/g/a, e.g., how fast a 802.11n card can be when 802.11b/g/a stations around? Given the huge efforts that have been made for 802.11n, is it possible that in the near future this kind of wireless cards will replace its predecessors? If not, what should be done?

Another issue is energy efficiency. Energy related considerations are gaining popularity quickly in wired networks and in cellular networks. As for 802.11n, enabling multiple sending and receiving antennas with the aim of providing the expected high rates naturally requires higher power. Can Power over Ethernet enabled switchs provide the power required by multiple antennas? If not, what wiil the performance be when 802.11n is running with low power?

Last, future wireless devices are likely to be capable of accessing multiple channels simultaneously. This means that design of spectrum agile MAC and PHY layers is an important issue, which inevitably required close interaction between these two layers, i.e., cross-layer design will become more and more important.

BIBLIOGRAPHY

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

[2] IEEE 802.11 WG, "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Higher-Speed Physical Layer Extension in the 2.4 GHz Band," IEEE Std. 802.11b, 1999.

[3] IEEE 802.11 WG, "Part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications: high-speed physical layer in the 5 GHz band," IEEE Std. 802.11a, Sep. 1999.

[4] IEEE 802.11 WG, "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 4: Further Higher Data Rate Extension in the 2.4 GHz Band," IEEE Std. 802.11g, Jun. 2003.

[5] IEEE 802.11 WG, "Part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications: medium access control (MAC) quality of service (QoS) enhancements," IEEE 802.11e D8.0, Feb. 2004.

[6] S. A. Mujtaba, et. al., "TGn Sync Proposal Technical Specification," www.tgnsync.org, IEEE 802.11-04/889r6, May 2005.

[7] Q. Ni, T. Li, T. Turletti and Y. Xiao, "AFR partial MAC proposal for IEEE 802.11n," IEEE 802.11-04-0950-00-000n, Aug. 2004.

[8] J. Ketchum, et. al., "System Description and Operating Principles for High Throughput Enhancements to 802.11," IEEE 802.11-04/0870r0, Aug. 2004.

[9] M. Singh, B. Edwards, et. al., "System Description and Operating Principles for High Throughput Enhancements to 802.11," IEEE 802.11-04-0886-00-000n, Aug. 2004.

[10] A. Stephens, et. al., "IEEE P802.11 Wireless LANs: Usage Models," IEEE 802.11-03/802r23, May 2004.

[11] Magis Networks White Paper, "IEEE 802.11 e/a Throughput Analysis," 2004, www.magisnetworks.com.

[12] NS, http://www.isi.edu/nsnam/ns/

[13] AFR Implementation, http://www.hamilton.ie/tianji_li/afr.html

[14] G. Appenzeller, I. Keslassy, and N. McKeown, "Sizing Router Buffers," in *Proc. of ACM SIGCOMM*, 2004. pp. 281–292.

[15] H. Balakrishnan, V. Padmanabhan, "How Network Asymmetry Affects TCP," *IEEE Communications Magazine*, Apr. 2001, pp. 60-67.

[16] P. Bahl, R. Chandra, and J. Dunagan, "SSCH: Slotted seeded channel hopping for capacity improvement in ieee 802.11 adhoc wireless networks," in *Proc. of ACM MOBICOM*, Sep. 2004, pp. 216-230.

[17] R. Battiti and B. Li, "Supporting service differentiation with enhancements of the IEEE 802.11 MAC protocol: models and analysis," University of Trento, technical report, DIT-03-024, May 2003.

[18] D. Bertsekas and R. Gallager, *Data Networks*, Prentice-Hall 1987.

[19] P. Bender, et. al, "CDMA/HDR: A bandwidth-efficient high-speed wireless data service for nomadic users," *IEEE Communications Magazine*, vol. 38, pp. 70-77, Jul. 2000.

[20] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function", *IEEE Journal on Selected Areas in Communnications*, vol. 18, no. 3, pp. 607-614, Mar. 2000.

[21] T. Bonald, and L. Massoulié, "Impact of Fairness on Internet Performance," in *Proc. of ACM SIGMETRICS*, Jun. 2001, pp. 82-91.

[22] M. Bottigliengo, C. Casetti, C. F. Chiasserini, and M. Meo, "Short-term Fairness for TCP Flows in 802.11b WLANs," in *Proc. of IEEE INFOCOM*, Mar. 2004, pp. 1383-1392.

[23] T. Bu, L. Li, and R. Ramjee, "Generalized Proportional Fair Scheduling in Third Generation Wireless Data Networks," in *Proc. of IEEE INFOCOM*, Apr. 2006, pp. 1-12.

[24] N. Celandroni, "Comparison of FEC types with regard to the efficiency of TCP connections over AWGN satellite channels," *IEEE Transactions on Wireless Communnications*, vol. 5, no. 7, pp. 1735-1745, Jul. 2006.

[25] N. Celandroni, F. Davoli, E. Ferro, and A. Gotta, "Long-Lived TCP Connections Via Satellite: Cross-Layer Bandwidth Allocation, Pricing, and Adaptive Control," *IEEE/ACM Transactions on Networking*, vol. 14, no. 5, pp. 1019-1030, Oct. 2006.

[26] G. Cantieni, Q. Ni, C. Barakat, and T. Turletti, "Performance Analysis under Finite Load and Improvements for Multirate 802.11," *Elsevier Computer Communications Journal*, vol. 28, no. 10, pp. 1095-1109, Jun. 2005.

[27] P. Chatzimisios, A. Boucouvalas, and V. Vitsas. "Performance analysis of IEEE 802.11 DCF in presence of transmission errors," *Proc. of IEEE ICC*, 2004, pp. 3854-3858.

[28] C. Chatfield, *The Analysis of Time Series, an Introduction*, CRC Press 2004.

[29] J. Choi, K. Park, and C. Kim, "Cross-layer Analysis of Rate Adaptation, DCF and TCP in Multi-rate WLANs," in *Proc. of IEEE INFOCOM*, May 2007, pp. 1055-1063.

[30] J. Choi, J. Yoo, S. Choi, and C. Kim, "EBA: An Enhancement of the IEEE 802.11 DCF via Distributed Reservation," *IEEE Transactions on Mobile Computing*, vol. 4, no. 4, pp. 378-390, Jul. 20054.

[31] J. Choi, K. Park, and C. Kim, "Cross-Layer Analysis of Rate Adaptation, DCF and TCP in Multi-Rate WLANs," in *Proc. of IEEE INFOCOM* 2007.

[32] P. Clifford, K. Duffy, D. Leith, and D. Malone, "On Improving Voice Capacity in 802.11 Infrastructure Networks," in *Proc. of IEEE WirelessCom*, Jun. 2005, pp. 214-219.

[33] T. Cover T and J. Thoma, *Elements of Information Theory*, John Wiley & Sons, 1991.

[34] D. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," in *Proc. of ACM MobiCom*, Sep. 2003, pp. 134-146.

[35] A. Dhamdher and C. Dovrolis, "Open Issues in Router Buffer Sizing," *Computer Communication Review*, Jan. 2006.

[36] R. Draves, J. Padhye, and B. Zill, "Comparison of routing metrics for static multi-hop wireless networks," in *Proc. ACM SIGCOMM*, Aug. 2004, pp. 133-144.

[37] K. Duffy, D. Leith, T. Li, and D. Malone, "Modeling 802.11 Mesh Networks," *IEEE Communication Letters*, vol. 10, no. 8, pp. 635-637, Aug. 2006.

[38] M. Enachescu, Y. Ganjali, A. Goel, N. McKeown, and T. Roughgarden, "Routers with Very Small Buffers," in *Proc. of IEEE INFOCOM*, Dec. 2006.

[39] S. Floyd and E. Kohler, "TCP Friendly Rate Control (TFRC): the Small-Packet (SP) Variant," RFC 4828, Apr. 2007.

[40] Paul Fuxjäger and Fabio Ricciato, "Collecting Broken Frames: Error Statistics in IEEE 802.11b/g Links," WiNMee 2008.

[41] R. Gallager, *Information Theory and Reliable Communication*, John Wiley & Sons, 1968.

[42] V. Gambiroza, B. Sadeghi, and E. W. Knightly, "End to End Performance and Fairness in Multihop Wireless Backhaul Networks," in *Proc. ACM MOBICOM*, Sep. 2004, pp. 287-301.

[43] M. Garetto, T. Salonidis, and E. W. Knightly, "Modeling Per-flow Throughput And Capturing Starvation In CSMA Multi-hop Wireless Networks," in *Proc. of IEEE INFOCOM*, Apr. 2006, pp. 1-13.

[44] Y. Ge, J. C. Hou, and S. Choi, "An Analytical Study of Tuning Systems Parameters in IEEE 802.11e Enhanced Distributed Channel Access," *Computer Networks*, vol. 51, no. 8, pp. 1955-1980, Jun. 2007.

[45] S. Ghez, S. Verdu, and S. C. Schwartz, "Stability Properties of Slotted Aloha with Multipacket Reception Capability," *IEEE Transcactions on Automatic Control*, vol. 33, no. 7, Jul. 1988, pp. 640-649.

[46] J. He, et. al., "Performance evaluation of distributed access scheme in error-prone channel," in *Proc. of IEEE TENCON* 2002, pp. 1142-1145.

[47] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance Anomaly of 802.11b," in *Proc. of IEEE INFOCOM*, 2003, pp. 836-843.

[48] M. Heusse, F. Rousseau, R. Guillier, and A. Duba, "Idle Sense: An Optimal Access Method for High Throughput and Fairness in Rate Diverse Wireless LANs," in *Proc. of ACM SIGCOMM*, 2005, pp. 121-132.

[49] K. Huang and K. Chen, "Interference analysis of nonpersistent CSMA with hidden terminals in multicell wireless data networks," in *Proc. of IEEE PIMRC* 1995, pp. 907-911.

[50] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experiment Design, Measurement, Simulation and Modeling*, John Wiley & Sons, 1991.

[51] Z. Ji, Y. Yang, J. Zhou, M. Takai, and R. Bagrodia, "Exploiting Medium Access Diversity in Rate Adaptive Wireless LANs," in *Proc. of ACM MOBICOM*, 2004, pp. 345-359.

[52] A. Kamerman and L. Monteban, "WaveLAN 2: A High-performance Wireless LAN for the Unlicensed Band," Bell Labs Tech. Journal, 1997.

[53] C. Kellett, R. Shorten, and D. Leith, "Sizing Internet Router Buffers, Active Queue Management, and the Lur'e Problem," in *Proc. of IEEE CDC*, 2006.

[54] J. Kim and J. Lee, "Capture effects of wireless CSMA/CA protocols in rayleigh and shadow fading channels," *IEEE Transactions on Vehicular Technology* 1999, pp. 1277-1286.

[55] S. Kim, Y. Kim, S. Choi, K. Jang, and J. Chang, "A High-Throughput MAC Strategy for Next-Generation WLANs," in *Proc. of IEEE WOWMOM*, 2005, pp. 220-230.

[56] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, pp. 237-252, 1998.

[57] S. Kunniyur and R. Srikant, "End-to-end congestion control: utility functions, random losses and ECN marks," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 689-702, Oct. 2003.

[58] L. Kleinrock, *Queueing Systems, Volume 1: Theory.* John Wiley & Sons, 1975.

[59] P. Karn, "MACA - A New Channel Access Method for Packet Radio," *ARRL/CRRL Amateur Radio 9th Computer Networking Conference*, 1990.

[60] S. Kumar, V. S. Raghavan, J. Deng, "Medium access control for ad hoc wireless networks: a survey," *Elsevier Ad Hoc Networks Journal*, 2004.

[61] Y. Kwon, Y. Fang, and H. Latchman, "A novel MAC protocol with fast collision resolution for wireless LANs," in *Proc. of IEEE INFOCOM*, 2003, pp. 853–862.

[62] J. Le Boudec, "Rate Adaptation, Congestion Control and Fairness: A Tutorial," on line.

[63] D. Leith, P. Clifford, D. Malone, and A. Ng, "TCP Fairness in 802.11e WLANs," *IEEE Communications Letters*, vol. 9, no. 11, pp. 964–966, Jun. 2005.

[64] D. J. Leith, and P. Clifford, "A Self-Managed Distributed Channel Selection Algorithm for WLANs," in *Workshop ACM/IEEE RAWNET*, Apr. 2006, pp. 1-9.

[65] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the Self-Similar Nature of Ethernet Traffic," *ACM SIGCOMM Computer Communication Review*, vol. 25, no. 1, Jan. 1995, pp. 202-213.

[66] P. Lettieri and M. B. Srivastava, "Adaptive Frame Length Control for Improving Wireless Link Throughput, Range, and Energy Efficiency," in *Proc. of IEEE INFOCOM*, 1998, pp. 564–571.

[67] C. Liu and A. Stephens, "An Analytic Model for Infrastructure WLAN Capacity with Bidirectional Frame Aggregation," in *Proc. of IEEE WCNC*, 2005, pp. 113–119.

[68] T. Li, Q. Ni, T. Turletti, and Y. Xiao, "Performance Analysis of the IEEE 802.11e Block ACK Scheme in a Noisy Channel," in *Proc. of IEEE Broadnets*, 2005, pp. 511-517.

[69] S. C. Liew and Y. J. Zhang, "Proportional Fairness in Multi-channel Multi-rate Wireless Networks," in *Proc. of IEEE GLOBECOM*, Nov. 2006, pp. 1-6.

[70] J. S. Ma, "On the Impact of HDLC Zero Insertion and Deletion on Link Utilization and Reliability," *IEEE Transactions Communnications*, vol. 30, no. 2, pp.375-381, Feb. 1982.

[71] B. Mah, "An empirical model of HTTP network traffic," in *Proc. of IEEE INFOCOM*, 1997.

[72] L. Massoulié and J. Roberts, "Bandwidth Sharing: Objectives and Algorithms," *IEEE/ACM Transcactions on Networking*, vol. 10, no. 3, pp. 320-328, Feb. 2002.

[73] D. Malone, K. Duffy, and D.J. Leith, "Modeling the 802.11 distributed coordination function in non-saturated heterogeneous conditions," *IEEE/ACM Transactions Networking*, vol. 15, no. 1, pp. 159-172, Feb. 2007.

[74] D. Malone, P. Clifford, and D. J. Leith, "On Buffer Sizing for Voice in 802.11 WLANs," *IEEE Communications Letters*, vol. 10, no. 10, pp 701–703, Oct. 2006.

[75] M. Mathis, J.y Semke, and J. Mahdavih, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm," *ACM SIGCOMM Computer Communication Review*, vol. 27, no. 3, pp. 67-82, Jul. 1997.

[76] R. Maheshwari, H. Gupta, S. R. Das, "Multichannel MAC Protocols for Wireless Networks," in *Proc. of IEEE SECON*, Sep. 2006, vol. 2, pp. 393-401.

[77] A. Miu, H. Balakrishnan, and C. E. Koksal, "Improving Loss Resilience with Multi-Radio Diversity in Wireless Networks," in *Proc. of ACM MobiCom*, 2005, pp. 16-30.

[78] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Transactions on Networking*, vol. 8, no. 5, pp. 556-567, Oct. 2000.

[79] T. Nandagopal, T. E. Kim, X. Gao, and V. Bharghavan, "Achieving MAC Layer Fairness in Wireless Packet Networks," in *Proc. of ACM MOBICOM*, Aug. 2000, pp. 87-98.

[80] Q. Ni, I. Aad, C. Barakat, and T. Turletti, "Modelling and Analysis of Slow CW Decrease for IEEE 802.11 WLAN," in *Proc. of IEEE PIMRC*, 2003, pp. 1717-1721.

[81] Q. Ni, T. Li, T. Turletti, and Y. Xiao, "Saturation Throughput Analysis of Error-Prone 802.11 Wireless Networks," *Wireless Communnications and Mobile Computing*, vol. 5, no. 8, pp. 945-956, Dec. 2005.

[82] V. Paxson and S. Floyd, "Wide-Area Traffic: The Failure of Poisson Modeling," *IEEE/ACM Transactions on Networking*, vol. 3, no. 3, pp. 226-244, Jun. 1995.

[83] Q. Pang, V. Leung, and S. C. Liew, "A Rate Adaptation Algorithm for IEEE 802.11 WLANs Based on MAC-Layer Loss Differentiation," in *Proc. of IEEE Broadnets*, 2005, pp. 659-667.

[84] F, Peng, J. Zhang, and W. E. Ryan, "Adaptive Modulation and Coding for IEEE 802.11n,"in *Proc. of IEEE WCNC*, 2007.

[85] S. Pilosof, et. al., "Understanding TCP fairness over Wireless LAN," in *Proc. of IEEE INFOCOM* 2003.

[86] G. Raina and D. Wischik, "Buffer Sizes for Large Multiplexers: TCP Queueing Theory and Instability Analysis," in *Proc. of EuroNGI*, Jul. 2005.

[87] B. Radunovič and J. Le Boudec, "Rate Performance Objectives of Multihop Wireless Networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 4, pp. 334-349, Oct. 2004.

[88] B. Radunovič and J. Le Boudec, "A Unified Framework for Max-min and Min-max Fairness with transportations," *IEEE/ACM Transactions on Networking*, vol. 15, no. 5, Oct. 2007.

[89] K. Ramachandran, E. Belding-Royer, K. Almeroth, and M. Buddhikot, "Interference-Aware Channel Assignment in Multi-Radio Wireless Mesh Networks," in *Proc. of IEEE INFOCOM*, Apr. 2006.

[90] B. Raman, "Channel Allocation in 802.11-based Mesh Networks," in *Proc. of IEEE INFOCOM*, Apr. 2006.

[91] A. Raniwala, P. De, S. Sharma, R. Krishnan, and Tzi-cker Chiueh, "End-to-End Flow Fairness over IEEE 802.11-based Wireless Mesh Networks," in *Proc. of IEEE INFOCOM*, Mini-Symposium, May 2007, pp. 2361-2365.

[92] Rappaport TS. *Wireless Communications: Principles and Practice*. Prentice Hall: New Jersey, USA, 1996.

[93] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knightly, "Opportunistic Media Access for Multirate Ad hoc networks," in *Proc. of ACM MOBICOM*, 2002, pp. 24-35.

[94] R. Shorten, F. Wirth, and D. Leith, "A Positive Systems Model of TCP-Like Congestion Control: Asymptotic Results," *IEEE/ACM Transactions on Networking*, vol. 14, no. 3, pp. 616–629, Jun. 2006.

[95] W. Stevens, "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms," RFC 2001, Jan. 1997.

[96] R. Stanojevic, and R. Shorten, "Beyond CHOKe: Stateless fair queueing," in *Proc. of EuroFGI NET-COOP* 2007.

[97] R. Stanojevic, C. Kellett, and R. Shorten, "Adaptive Tuning of Drop-Tail Buffers for Reducing Queueing Delays," *IEEE Communications Letters*, vol. 10, no. 7, pp 570–572, Jul. 2006.

[98] G. Tan and J. Guttag, "Time-based Fairness Improves Performance in Multi-rate Wireless LANs," in *Proc. of USENIX*, 2004.

[99] G. Tan and J. Guttag, "The 802.11 MAC Protocol Leads to Inefficient Equilibra," in *Proc. of IEEE INFOCOM*, 2005, pp. 1-11.

[100] A. Tanenbaum, *Computer Networks*, Fourth Edition, New Jersey, Prentice Hall PTR, 2003.

[101] A. Tang, J. Wang, and S. Low, "Is Fair Allocation Always Inefficient," in *Proc. of IEEE INFOCOM*, Mar. 2004.

[102] D. Tang and M. Baker, "Analysis of A Local-Area Wireless Network," in *Proc. of ACM MobiCom* 2000.

[103] Y. Tay and K. Chua, "A capacity analysis for the IEEE 802.11 MAC protocol. *Wireless Networks*, 2001, pp. 159-171.

[104] M. Thottan and M. C. Weigle, "Impact of 802.11e EDCA on mixed TCP-based applications," in *Proc. of IEEE WICON* 2006.

[105] K. Thompson, G. Miller, and R. Wilder, "Wide-area Internet traffic patters and characteristics," *IEEE Network*, vol. 6, no. 11, pp. 10–23, Nov. 1997.

[106] O. Tickoo and B. Sikdar, "On the Impact of IEEE 802.11 MAC on Traffic Characteristics," *IEEE Journal on Selected Areas in Communnications*, vol. 21, no. 2, Feb. 2003, pp. 189-203.

[107] L. Tong, Q. Zhao, and G. Mergen, "Multipacket Reception in Random Access Wireless Networks: From Signal Processing to Optimal Medium Access Control," *IEEE Communnications Magazine*, vol. 39, no. 11, Nov. 2001, pp. 108-112.

[108] J. Tourrilhes, "Packet Frame Grouping: Improving IP multimedia performance over CSMA/CA," in *Proc. of ICUPC*, 1998, pp. 1345-1349.

[109] I. Tinnirello and S. Choi, "Temporal Fairness Provisioning in Multi-Rate Contention-Based 802.11e WLANs," in *Proc. of IEEE WOWMOM*, Jun. 2005, pp. 220-230.

[110] J. Tourrilhes, "Packet Frame Grouping: Improving IP multimedia performance over CSMA/CA," in *Proc. of ICUPC*, 1998.

[111] Z. Velkov, B. Spasenovski, "Saturation throughput-delay analysis of IEEE 802.11 DCF in fading channel," in *Proc. of IEEE ICC*, 2003, pp. 121-126.

[112] C. Villamizar and C. Song, "High Performance TCP in ANSNET," *ACM Computer Communication Review*, vol. 24, no. 5, pp. 45-60, 1994.

[113] V. Vishnevsky and A. Lyakhov, "IEEE 802.11 LANs: saturation throughput analysis with seizing effect consideration," *Journal of Cluster Computing*, 2002, pp. 133-144.

[114] V. Vishnevsky and A. Lyakhov, "802.11 LANs: saturation throughput in the presence of noise," in *Proc. of IFIP Networking*, 2002, pp. 1008-1019.

[115] V. Vitsas, et. al., "Enhancing performance of the IEEE 802.11 Distributed Coordination Function via Packet Bursting," in *Proc. of GLOBECOM*, 2004, pp. 245-252.

[116] M. Vojnović, J. Le Boudec, and C. Boutremans, "Global Fairness of Additive-Increase and Multiplicative-Decrease With Heterogeneous Round-Trip Times," in *Proc. of IEEE INFOCOM*, Mar. 2000, pp. 1303-1312.

[117] G. Vu-Brugier, R. Stanojevic, D. Leith, and R. Shorten, "A Critique of Recently Proposed Buffer-Sizing Strategies," *ACM Computer Communication Review*, vol. 37. no. 1, Jan. 2007.

[118] W. Wang, S. Liew, and V. Li, "Solutions to Performance Problems in VoIP over a 802.11 Wireless LAN," *IEEE Transactions On Vehicular Technology*, vol. 54, no. 1, pp. 366-384, Jan. 2005.

[119] D. Wischik and N. McKeown, "Part I: buffer sizes for core router," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 3, Jul. 2005.

[120] H. Wu, Y. Peng, K. Long, S. Cheng, and J. Ma, "Performance of Reliable Transport Protocol over IEEE 802.11 Wireless LAN: Analysis and Enhancement," in *Proc. of IEEE INFOCOM*, 2002, pp. 599-607.

[121] H. Wu, F. Yang, K. Tan, J. Chen, Q. Zhang, and Z. Zhang, "Distributed Channel Assignment and Routing in Multi-radio Multi-channel Multi-hop Wireless Networks," *Journal on Selected Areas in Communications*, vol.24, pp. 1972-1983, Nov. 2006.

[122] Y. Xiao and J. Rosdahl, "Performance analysis and enhancement for the current and future IEEE 802.11 MAC protocols," *ACM SIGMOBILE Mobile Computing and Communications Review (MC2R)*, Vol. 7, No. 2, Apr. 2003, pp. 6-19.

[123] Y. Xiao, "IEEE 802.11 Performance Enhancement via Concatenation and Piggyback Mechanisms," *IEEE Transactions on Wireless Communnications*, vol. 4, no. 5, Sep. 2005, pp. 2182-2192.

[124] Y. Xiao, "IEEE 802.11n: Enhancements for Higher Throughput in Wireless LANs," *IEEE Wireless Communnications*, vol 12, no. 6, Dec. 2005, pp. 82-91.

[125] X. Yang and N. Vaidya, "A Wireless MAC Protocol Using Implicit Pipelining," *IEEE Transactions on Mobile Computing*, vol 5, no. 3, Mar. 2006, pp. 258-273.

[126] Y. Yang, J. Wang and R. Kravets, "Distributed Optimal Contention Window Control for Elastic Traffic in Wireless LANs," in *Proc. of IEEE INFOCOM*, Mar. 2005, pp. 35-46.

[127] Z. Zhao, S. Darbha, and A. L. N. Reddy, "A method for estimating the proportion of nonresponsive traffic at a router," *IEEE/ACM Transactions Networking*, vol. 12, no. 4, pp. 708-718, Aug. 2004.