



NUI MAYNOOTH

Ollscoil na hÉireann Má Nuad

**Wireless Network Measurement:
VoIP and 802.11e**

by

Ian Dangerfield, B.A

Masters Thesis

Hamilton Institute

National University of Ireland Maynooth

Maynooth

Co. Kildare

December 2007

Research Supervisor: Dr. David Malone

Head of Department: Prof. Douglas Leith

Contents

1	An introduction to 802.11	1
1.1	Overview of 802.11	1
1.1.1	Contention mechanisms	2
1.1.2	DCF contention mechanism	3
1.1.3	Main issues with DCF	4
1.1.4	PCF contention mechanism	5
1.1.5	Main issues with PCF	6
1.2	802.11e: extension to 802.11 MAC	6
1.2.1	Hybrid Coordination Function - HCF	8
1.3	Other issues	9
1.4	Voice over 802.11 networks	9
1.5	Summary	10
2	A review of some Analytical models of 802.11	11
2.1	Models of 802.11 MAC	11
2.2	Bianchi's model of DCF	13
2.2.1	Throughput estimation	16
2.2.2	RTS/CTS mechanism	18
2.3	Extensions to the model	18
2.3.1	Unsaturated model	18
2.3.2	EDCA extensions	21
2.3.3	Other variations	24
2.4	Summary	24

3	802.11 testbed techniques and validation	25
3.1	The Testbed	25
3.1.1	Physical Setup	26
3.1.2	Driver modifications	26
3.1.3	Other software	28
3.2	Measurement	29
3.2.1	Delay	29
3.2.2	Standard Behaviour	32
3.2.3	Throughput	35
3.3	Studying Model Assumptions	38
3.4	Issues and Improvements	43
3.5	Summary	44
4	Experimental VoIP measurements	45
4.1	Introduction	45
4.2	Performance of VoIP over 802.11	46
4.3	Single VoIP vs Data behaviour	50
4.4	VoIP versus saturated traffic	54
4.4.1	VoIP throughput	55
4.4.2	Delays for VoIP	60
4.4.3	Queue occupancy	68
4.4.4	Collision probabilities	71
4.5	Conclusions	75
5	Future work	77

List of Figures

1.1	Basic DCF Operation	3
1.2	Collision during DCF operation	4
1.3	AIFS difference between two stations	7
1.4	TXOP in operation	8
2.1	Markov Example	12
2.2	Bianchi model for DCF	13
2.3	Finite load section of model	19
2.4	AIFS extension to the model	23
3.1	Measurement Scheme	30
3.2	NTP timing	31
3.3	Packet size versus delay for long and short preambles	32
3.4	Delays when CW is 0 to 31	33
3.5	Delays when CW is 0 to 3	34
3.6	Delays when CW is 0 to 31 and AIFS is 8	34
3.7	Delays when CW is 0 to 3 and TXOP is set for 2 packets	35
3.8	Variation in throughputs for two stations as one station varies CWmin	36
3.9	Variation in throughputs for two stations as one station varies TXOP	37
3.10	Variation in throughputs for two stations as one station varies AIFS	37
3.11	2 STA P(collision) versus offered load	39
3.12	10 STA P(collision) versus offered load	40
3.13	Measured versus expected collision probability	41
3.14	Measured versus expected throughput	42
4.1	Mean MAC delays at AP for variable buffer sizes	47

4.2	Mean MAC delays at a STA for variable buffer sizes	47
4.3	Comparison between model predictions and throughput at an unprioritised AP	48
4.4	Throughput at prioritised AP, TXOP=N, CW=31	49
4.5	Throughput at prioritised AP, TXOP=N/2, CW=15	49
4.6	Throughput of a single VoIP station versus N saturated stations	50
4.7	Delay experienced by a single VoIP station versus N saturated stations	51
4.8	CDF for an unprioritised VoIP station	52
4.9	CDF for a prioritised VoIP station, AIFS=6	52
4.10	CDF for a prioritised VoIP station, AIFS=8	53
4.11	Autocorrelation of packets over a number of schemes	53
4.12	AP throughput per voice sta versus N voice conversations and 4 saturated stations	55
4.13	Voice throughput versus N voice conversations and 4 saturated stations	56
4.14	AP throughput per voice sta versus N voice conversations and 10 saturated stations	57
4.15	Voice throughput versus N voice conversations and 10 saturated stations	57
4.16	Throughput for 4 saturated stations	58
4.17	Throughput for 10 saturated stations	59
4.18	AP MAC delay versus N voice conversations and 4 saturated stations	61
4.19	STA MAC delay versus N voice conversations and 4 saturated stations	61
4.20	AP MAC delay versus N voice conversations and 10 saturated stations	62
4.21	STA MAC delay versus N voice conversations and 10 saturated stations	62
4.22	AP queuing delay versus N voice conversations and 4 saturated stations	63
4.23	STA queuing delay versus N voice conversations and 4 saturated stations	64
4.24	AP queuing delay versus N voice conversations and 10 saturated stations	64
4.25	STA queuing delay versus N voice conversations and 10 saturated stations	65
4.26	Close up of AP queuing delay (N voice conversations and 4 saturated stations)	65
4.27	Close up of STA queuing delay (N voice conversations and 4 saturated stations)	66
4.28	Close up of AP queuing delay (N voice conversations and 10 saturated stations)	67
4.29	Close up of STA queuing delay (N voice conversations and 10 saturated stations)	67
4.30	Average MAC queue occupancy at AP versus 4 saturated stations	68
4.31	Average MAC queue occupancy at STA versus 4 saturated stations	69
4.32	Average MAC queue occupancy at AP versus 10 saturated stations	70
4.33	Average MAC queue occupancy at STA versus 10 saturated stations	70
4.34	Collision Probability at AP versus 4 saturated stations	73

4.35 Collision Probability at VoIP STA versus 4 saturated stations	73
4.36 Collision Probability at AP versus 10 saturated stations	74
4.37 Collision Probability at VoIP STA versus 10 saturated stations	75

Abstract

802.11 wireless networks are a convenient way to provide ‘last-hop’ network connections in many environments. However the distributed nature of their operation poses problems in providing quality of service guarantees, which are vital for providing interactive services such as Voice over IP. In this thesis we will review the operation of 802.11 and its extended MAC layer, 802.11e. We will then review some analytical models used to explore the behaviour of the 802.11 MAC and test some of the assumptions underlying these models. We confirm that our testbed operates according to the standards before using it to explore the behaviour of VoIP traffic, using it as a test case to examine the provision of quality of service guarantees via the extended 802.11e MAC. This work is based on several papers [9] [19] [10].

Acknowledgements

I would like to start by thanking my supervisor David Malone for all his inspiration and patient help during the last two years. I would also like to thank Doug Leith and Ken Duffy for their important input and support. Thanks are due as well to the rest of the Hamilton staff, Rosemary Hunt and Kate Moriarty in particular, for making the Hamilton institute such a good place to work and learn.

Thanks also go to my parents, Anna and Michael, and my sister, Marie-Claire for all their understanding and support. Finally, I would like to thank Serena Corr, whose continuous support and encouragement made all of this possible.

Chapter 1

An introduction to 802.11

1.1 Overview of 802.11

Wireless networks are becoming increasingly popular due to the availability of cheap hardware and the intrinsic advantage it has for the user over wired alternatives, namely no wires! The availability of hardware is in part due to the standardisation efforts of the IEEE, in particular ‘The Working Group for WLAN Standards’, referred to as 802.11. This group provides the underlying standards for initiatives such as WiFi, who seek to ensure products from different manufactures inter-operate. It is part of the larger ‘802 LAN/MAN Standards Committee’ a body which has been responsible for, among other things, standardising ethernet for wired networking.

The particular standards that we are interested in are 802.11a/b/g [1], which deal with the MAC or ‘Medium Access Control’ and PHY or ‘Physical’ layers, as well as 802.11e which extends the MAC for the above standards. ¹ The MAC layer is responsible for determining when a packet is to be sent and defines the basic rules for doing so. The PHY layer is responsible for changing the bits into whatever physical means the network uses to transmit data, radio waves in the case of 802.11 networks. In this chapter we will describe the operation of these 802.11 standards from the perspective of the MAC layer, showing both the basic operation from the earlier standards and the extended operation defined in 802.11e. It should be noted that the WiFi Alliance have certification for WME/WMM,

¹When referring to ‘layers’, we are referring to layers from the OSI Model [30].

‘Wireless Multimedia Extensions’ and ‘WiFi Multimedia’ respectively, which provides for a subset of the features in the 802.11e standard [2].

1.1.1 Contention mechanisms

As we are mostly concerned with the MAC portion of 802.11, we need to first describe the medium access functions that it describes. Wireless is of its nature a broadcast medium, with transmissions received by everybody on the network. In 802.11 wireless networks with current hardware, it is not possible to both transmit and listen to the medium at the same time. This is due to the huge difference in power between that of a received packet and one being transmitted at the node. Even with multiple aerials, if the distance between aerials on a node is small in comparison with the distance to another node, the power difference will still be far too large. Because of this, a node on a wireless network must share time on the medium with other nodes, a similar state of affairs with old Ethernet [23]. These schemes are referred to under the general title of ‘Carrier Sense Multiple Access’, or CSMA. The setup involves each station having to sense when the medium is idle and then transmitting. With old wired Ethernet however, the difference in transmission versus reception power was not such a significant issue and as a result a node on the network could sense when a collision had occurred. This is referred to as ‘CSMA/CD’, CD for ‘Collision Detection’.

The disparity in power levels in 802.11 precludes a node from sensing directly if a collision is taking place. This is because it is incapable of sensing the medium when it is transmitting. This presents two problems. First, the node which has attempted to send a packet must be able to indirectly sense if a collision has occurred, so that it can attempt a retransmission. Secondly, a scheme must be employed to minimise the number of collisions, as the efficiency will be reduced as the number increases. The first problem is dealt with by requiring that frames are ‘acknowledged’ by the receiving node. The receiving station is required to send an acknowledgement frame within a short space of time to the original transmitting node. If this is not successfully received then the first node assumes a collision has taken place. The second issue is dealt with by a ‘contention mechanism’ which we will discuss in detail. This approach is referred to as ‘CSMA/CA’, with CA for ‘Collision Avoidance’.

There are two contention mechanisms available for use in 802.11 wireless networks. First there is the ‘Distributed Coordination Function’ or DCF as it is referred to. Then there is the ‘Point Coordination Function’ or PCF. As the names imply, DCF is a totally distributed system with no central control.

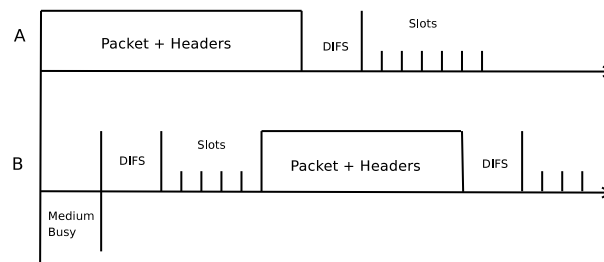


Figure 1.1: Basic DCF Operation

On the other hand, PCF, while compatible with the DCF, has a centralised control mechanism. We will discuss these mechanisms in detail and try to outline some of their respective advantages and problems.

1.1.2 DCF contention mechanism

This is by far the most commonly used contention mechanism in 802.11 networks, because it is a required part of the standard to implement. Therefore, other mechanisms must inter-operate with nodes using this scheme.

Figure 1.1 shows two scenarios. In scenario 'A', a node which has been idle for some time gets a packet to send. First the node senses the medium for a period of time to determine if another node is currently transmitting. This period is the 'DCF InterFrame Space' or DIFS. If no transmissions are sensed, then the node transmits its packet. It then waits for a period after the transmission. This is the 'Short InterFrame Space', or SIFS, which is shorter than DIFS. The receiving node then transmits an acknowledgement packet (or 'ACK') to indicate that the initial packet has been received. If both of these packets are successfully received by the two nodes, the transmission is deemed successful. Now, if the first node has further packets to transmit it does not start transmission straight away. It must first 'back-off' in order to allow other stations the opportunity to transmit their packets. It does this by randomly picking a number between 0 and CW_{min} , where CW is the 'Contention Window'. In 802.11, the value of CW is to the power of 2 minus 1, with a minimum value of 31 and a maximum value of 1023, referred to as ' CW_{max} '. The counters go from 0, hence the -1. The node then counts down CW 'slots', each of time 20us for 11b- or 9us for 11a/11g-only networks. It then attempts to transmit its packet. This type of back-off is known as the post back-off and ensures that one station

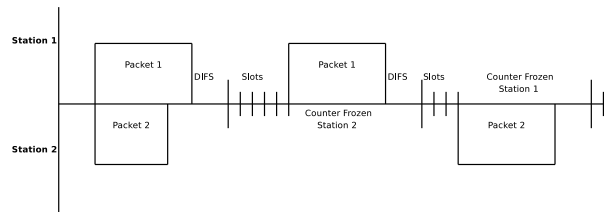


Figure 1.2: Collision during DCF operation

does not hog the medium.

If during this time, another station begins transmission then this node freezes its back-off timer and does not restart it until it has sensed that the medium has been idle for DIFS. It then restarts its counter and attempts to transmit when the counter reaches 0, as shown in scenario ‘B’.

If two or more stations attempt to transmit at the same time, a collision occurs. In this case the stations both increase their CW value (for example, from 31 to 63), randomly pick new counter values and count down as before. This situation is shown in Figure 1.2. Note that if a station transmitted straight away after being idle it would now pick the second CW value 63, and not 31, since it is deemed to have backed-off already just like the other station. This type of back-off is designed to avoid collisions by making it increasingly less likely for two stations to pick the same slot to transmit in. If more collisions occur, the CW value is increased by powers of 2 as before up to the maximum value CW_{max}. If collisions occur when the CW value is already at CW_{max}, a new counter value is chosen from CW_{max} and the count down is repeated using the new counter.

1.1.3 Main issues with DCF

From the above description, it can be seen that no special information is required to operate this system since it is completely distributed. However this also means there are some fundamental issues with this scheme, particularly with providing ‘Quality of Service’, or QoS, guarantees. In fact, there is no way to provide them at all!

In an infrastructure network the access point has no special status when using this scheme and so shares access to the medium evenly, just like a station. This means that the access point will only get $1/(N+1)^{th}$ of the transmission opportunities on a busy network, where N is the number of other

stations on the network. This can present a serious problem. Firstly, if a frame is being transmitted from one station to another, it has to be retransmitted by the access point. Secondly TCP, the most common protocol used, requires that its packets are acknowledged. This acknowledgement is different to the 802.11 ACK since this is sent as a regular data packet. So even if TCP traffic is leaving the network via the access point, and does not need to be retransmitted, the TCP acknowledgements still need to be sent by the access point. This results in the access point requiring more transmission opportunities than a regular station.

1.1.4 PCF contention mechanism

The other scheme available is the ‘Point Coordination Function’ or PCF. This scheme combines a contention period, using the DCF model described before, with a contention free period controlled by a ‘point coordinator’. In this case, the point coordinator is also the access point. The contention free period is of limited duration so that the medium is accessible to stations using only the DCF scheme; otherwise, they would be completely unable to transmit on the medium.

A contention free period is started by the point coordinator sending a beacon frame. This frame contains the maximum duration of the contention free period, so all stations listening can refrain from transmitting during this period. The point coordinator then polls individual stations to allow them to transmit a frame. The station waits only a SIFS before transmitting a frame and to make even more efficient use of the medium, different types of frame can be combined in a single transmission attempt. For example, a new frame and an ACK can be sent together when a station, or the point coordinator, is given a transmission opportunity. This makes more efficient use of the medium, since more time is spent transmitting and less backing-off. Also collisions due to two stations picking the same back-off CW value are eliminated.

It is possible that a station may not respond to a poll from the point coordinator. If this occurs then the point coordinator polls another station within PIFS, ‘PCF InterFrame Space’. This is shorter than a DIFS, but longer than SIFS, and so allows the point coordinator to keep the medium from stations only using DCF, since they will not have restarted their back-off counters. The point coordinator can also send a beacon to end the contention free period. Stations receiving this will then revert to using the DCF scheme. For more details on PCF, as well as 802.11 in general see OReillys ‘802.11 Wireless Networks’ [13].

1.1.5 Main issues with PCF

The single major issue with this scheme is a lack of implementation. This is an optional component of the standard and is not implemented on many platforms. Also, this scheme requires that the network is an infrastructure network, with an access point and stations and so cannot be used in an ad-hoc network.

This scheme would seem to be a logical choice for providing ‘Quality of Service’ (QoS), since the point coordinator has control over which station (including itself) can transmit during the contention free period and it makes efficient use of the medium during these periods. However the point coordinator needs to know which stations require prioritisation, since this is not built into the scheme. The scheme could implement QoS but only if the point coordinator had some other way of identifying the requirements, or knowing them in advance. This is not ideal especially in a situation where stations join and leave the network on a regular basis. Also, the standard mandates a ‘Round Robin’ scheduling algorithm and more flexible scheduling may be required to provide QoS in certain cases.

In addition, the PCF scheme depends on the DCF scheme for the simple reason that new stations need to avail of the DCF scheme in order to join the network and register with the access point.

If there are several networks located close together, so that they are interfering with each other, the PCF scheme can not be expected to behave well. It assumes that the access point has complete control over access to the medium and does not compensate for losses caused by interference.

1.2 802.11e: extension to 802.11 MAC

The IEEE 802.11e standard [2] attempts to provide quality of service guarantees over 802.11 networks by making changes to the standard 802.11 MAC. At the most basic level, the 802.11e MAC allows for the changing of parameters which were previously fixed in the standard MAC, with additional added parameters.

The first change is to the contention window, CW. The CW range is still doubled after each collision and increases from an initial value CW_{min} to a maximum value CW_{max} , as in the original standard. 802.11e allows the values of CW_{min} and CW_{max} to be set as powers of two, instead of being fixed values. Previously, the window sizes were set and not changeable.

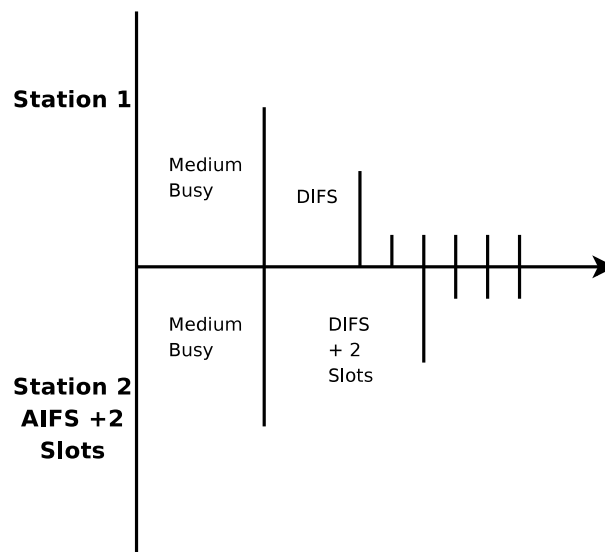


Figure 1.3: AIFS difference between two stations

The next difference is with the DIFS value. In 802.11e, this is referred to as the ‘Arbitration Inter Frame Space’, or AIFS, and can take different values. These values are based on the slot length and are related to an integer number of slots $N * S$. The operation of AIFS can be seen in figure 1.3.

The new standard also introduces a new parameter, TXOP, for ‘Transmission Opportunity’. This is a time during which a station may transmit multiple packets. When a station wins a transmission opportunity, either through contention or polling, it is permitted to keep transmitting frames until the TXOP period expires. This is checked just before a frame is transmitted, so the station may well hold on to the medium for longer than the TXOP period. The station holds on to the medium by only waiting SIFS between transmissions, thus other stations do not restart their counters until the station has finished, as can be seen in figure 1.4. Having a large TXOP value and only using a small portion or none of it is not an issue, since after AIFS (or DIFS) other stations will start to contend again. Note that a TXOP value of 0 has special meaning, in that it gives the same behaviour as in the original 802.11 standard.

Another significant change is the addition of four different ‘access categories’ to the MAC. Each access category can have a different set of the above parameters, CW_{min}/max, AIFS and TXOP. An individual station is therefore free to apply some form of prioritisation to different traffic types itself. This leads to the possibility that two packets may arrive at the head of their respective queues at the

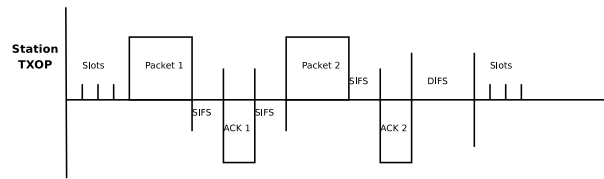


Figure 1.4: TXOP in operation

same time. This is referred to as a ‘virtual collision’. In this case the higher priority packet, from the higher valued queue, is transmitted and the packet from the lower priority queue is treated as if a collision occurred. It increases its CW and chooses a new counter. However it is not clear how strictly this is actually implemented.

The 802.11e standard provides for contention-based and polling-based access to the medium as the old standard did. It implements these two schemes through what is known as the ‘Hybrid Coordination Function’, or HCF. This operates in two ways, the ‘Enhanced Distributed Channel Access’ or EDCA mode which is contention-based and the ‘HCF Controlled Channel Access’ or HCCA mode which is polling-based. When operating in solely contention based mode, this is referred to as the ‘Enhanced Distributed Coordination Function’, EDCF.

1.2.1 Hybrid Coordination Function - HCF

The contention-based part of HCF, EDCA, is very similar in operation to the DCF scheme used in the original MAC standard. The main difference is that the parameters are changeable and dependent on the access category, and perhaps the station, trying to send the frame. This is the part of the 802.11e standard that we will be most concerned with in this work.

There is also the HCCA polling-based scheme that 802.11e implements. However, this suffers from the same major issue as the PCF scheme, namely lack of implementation.

The HCCA polling scheme works in the same way as the PCF scheme described earlier. However, there are two improvements. First the ‘hybrid coordinator’, or the access point, is free to use whatever scheduling algorithm it wishes in order to decide who to poll next. Also, due to the presence of the new queues, the scheduling can be done in a per class way instead of purely by station. For example a station could be polled to send a packet from a certain AC. If it does not have a packet from that

AC to send, it simply sends nothing and another station can be polled instead.

1.3 Other issues

There are some other issues with 802.11 networks that we will not be considering here but which are worth mentioning.

Firstly ‘hidden nodes’, stations which cannot hear each other but which can interfere with transmissions at a third intermediate station, are a problem for 802.11 networks. The standard implements a hand-shaking scheme, RTS (Request to Send)/CTS (Clear to Send), to counteract the effect of hidden nodes but at a cost to efficiency [6] [29].

Rate control schemes also have an important impact. If stations are physically close and channel conditions are good, then higher rates may be used. However it is important to distinguish losses due to a poor channel from those due to a collision, as the rate should not be altered as a result of collisions. Similarly picking a channel to transmit on is a problem when several networks are close together, and poor choice of channel can significantly affect the networks [18].

These are all real issues for 802.11 networks, however they are not directly QoS issues and as such we will not consider them here.

1.4 Voice over 802.11 networks

Voice over IP, ‘VoIP’ from here on, over 802.11 networks is interesting for a number of reasons. Unlike data traffic, voice is quite QoS sensitive, and is therefore a good test case for any scheme that aims to provide QoS, like the 802.11e standard. Secondly, the growing use of WiFi and of VoIP applications such as ‘Skype’ [4], have occurred at roughly the same time, creating a demand for VoIP to operate over WiFi networks. Currently only a few voice conversations (2-way traffic) coupled with data (up or down) will render the quality unacceptable. QoS is therefore a much needed addition to 802.11 networks, as the behaviour of their MAC layer portion introduces difficulties for time sensitive traffic.

We should also note that streamed content such as video or music are also QoS sensitive, but problems

can often be avoided using buffering, which is not an option for interactive content such as voice calls.

1.5 Summary

Off-the-shelf hardware supporting 802.11e is now available which makes it practical to implement a test-bed to examine the behaviour of VoIP over a wireless network. In chapter 2 we will review some of the analytical models which can be used to make predictions about the behaviour of the traffic in wireless networks. In chapter 3 we will describe the experimental setup and the methods of collecting data. We will also look to confirm the hardware's adherence to the 802.11e standard as well as examine some of the basic assumptions of the analytical models. In chapter 4 we will see how VoIP traffic actually performs in the test-bed, and investigate schemes using the new 802.11e enhancements to prioritise VoIP traffic in the network.

Chapter 2

A review of some Analytical models of 802.11

2.1 Models of 802.11 MAC

The basic model we will consider for the 802.11 DCF mechanism, and the one upon which the 802.11e models are based on, was developed by Bianchi [6]. This model describes the behaviour of an 802.11 MAC as a discrete time Markov chain and allows various important properties to be predicted, such as each stations throughput and their collision probabilities (the likelihood that a packet experiences a collision given that the station is attempting transmission). It is described in detail in [6], however here we will go through the core elements of the model and describe its construction. We then describe some of the extensions to this model developed by others, [20] [28] [5], concentrating on the way the model was enhanced in each case. The reason we are interested in these models is because they were used to inform real test-bed experiments and, in turn, the test-bed was used to validate some of the fundamental assumptions of the models.

A Markov Chain is a system which evolves in discrete time and in which the probability of it choosing any given state depends only on the current state. The Stationary distribution of a Markov chain is the chance of finding the system in a particular state if it were stopped randomly after the system has been running for a infinite number of iterations.

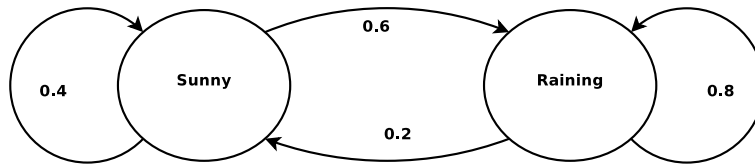


Figure 2.1: Markov Example

Figure 2.1 shows a simple example of a Markov Chain. In this example the transition matrix, which represents the probability that is straight forward to write down

$$\begin{pmatrix} 0.4 & 0.6 \\ 0.2 & 0.8 \end{pmatrix}.$$

This matrix represents the probability of moving from one state in the Markov Chain to another. Expressing our current state as a vector, say $p_0 = (1 \ 0)$ we can get the probability of being in the different states of the Markov Chain by multiplying this vector by the transition matrix A . In other words

$$Ap_0 = p_1,$$

where p_1 is vector representing the probabilities of being in the states in the next time-step. Although Markov Chains only depend directly on the previous state, we can still ask what the probability is of being in a particular state n time-steps in the future. This is because the probability of being in the previous state depended on the state before that, and so on. In order to get this we can simply multiply the matrix by itself n times.

$$A^n p_0 = p_n.$$

Taking the limit as n tends to infinity gives

$$\lim_{n \rightarrow \infty} A^n p_0 = p_\infty,$$

will give us the probability that we find ourselves in any particular state. To see this multiply across by A

$$Ap_\infty = A \lim_{n \rightarrow \infty} A^n p_0 = \lim_{n \rightarrow \infty} A^{n+1} p_0$$

If the limit exists for A^n then it will be the same for A^{n+1} . Therefore we can say

$$\lim_{n \rightarrow \infty} A^{n+1} p_0 = p_\infty$$

This set of probabilities constitute the Stationary Distribution for the Markov Chain. For a more detailed introduction to Markov chains see [12].

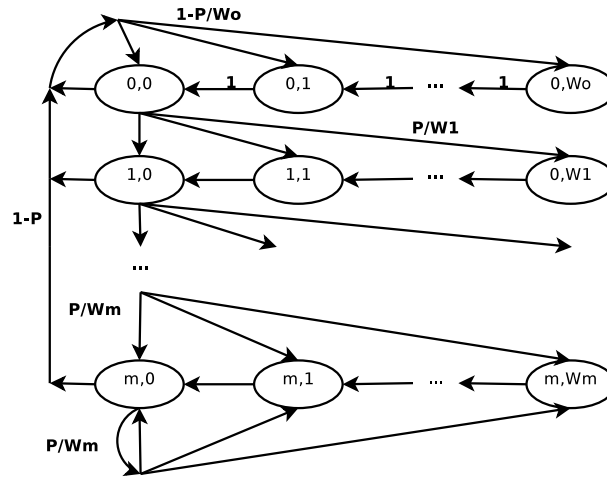


Figure 2.2: Bianchi model for DCF

2.2 Bianchi's model of DCF

Bianchi's model [6] describes each possible state of the 802.11 MAC in terms of a state in a discrete time Markov chain. This is illustrated in Figure 2.1. It turns out that this approach matches the operation of the DCF quite well. In the basic model, each state the system can be in is described by two variables, a back-off stage and a back-off counter. The first variable, the back-off stage, refers to how many collisions the 802.11 MAC has experienced attempting to transmit the packet. The second variable, the back-off counter, refers to the window size that is chosen, or arrived at as the station counts down. Recall that this counter is decremented after the medium is sensed idle and then after every MAC level slot, σ , again once the medium remains idle. The range of values which the window is chosen from is determined by the back-off stage, and doubles in size up to a maximum, reached when $i = m$. The MAC may still experience collisions here, however the back-off stage no longer increases. Instead, a new window is chosen and the count down begins again. In Figure 2.2 each state is labeled i, k , where i is the back-off stage and k is the back-off counter. This state i, k corresponds to the state of the Markov Chain.

It should be noted that there are several assumptions being made here. Firstly, each station is assumed to always have a packet to send. In other words they are assumed to be saturated. Secondly, the probability of having a collision, p , is assumed to be the same for all stations and for each back-off stage. Later models will expand on the first assumption by introducing an unsaturated model, where

stations have random packet arrivals. In addition, although a maximum back-off stage is defined there is no maximum retry limit imposed in this model. This was considered in [28].

Here, it is important to note that although this is a discrete time model, the time taken to move from one state to another is not going to be constant. The time that it takes to move from one state to another is referred to as a time slot, to fit in with the discrete nature of the Markov chain. This has an important implication for collisions. We assume that a collision occurs if and only if two or more stations transmit in the same slot and that both of the stations then experience a collision¹. This enables us to state the following relationship

$$1 - p = (1 - \tau)^{n-1}$$

The interpretation of this is that the chance of not having a collision, $1 - p$, is equal to the chance that nobody else transmits.

Given these assumptions we can write down the probabilities of moving from one state to another. The first thing to note is that if we are in any state where the back-off counter is not zero, we will end up in the state with the same back-off stage but with the counter decremented by one with probability one. In other words

$$P((i, k - 1)|(i, k)) = 1, \quad k \in (1, W_i - 1)$$

There are three transitions, however, that have different probabilities. In these equations p is the conditional collision probability and W_i is the window size at back-off stage i :

$$P((0, k)|(i, 0)) = \frac{1 - p}{W_0}, \quad k \in (0, W_0 - 1), i \in (0, m); \quad (2.1)$$

$$P((i, k)|(i - 1, 0)) = \frac{p}{W_i}, \quad k \in (0, W_i - 1), i \in (1, m); \quad (2.2)$$

$$P((m, k)|(m, 0)) = \frac{p}{W_m}, \quad k \in (0, W_m - 1). \quad (2.3)$$

Equation (1) arises since if we do not have a collision when we are attempting to transmit, then we will pick another back-off counter from CW_{min} , corresponding to back-off stage 0 regardless of which back-off stage we were in. Equation (2) shows that if we do have a collision we increase our back-off stage and choose a new counter from the new window. If we are in the final back-off stage $i = m$, then we do not increase our back-off stage, we just pick a new counter from the window as shown in equation (3).

¹This is not always true, one station can ‘capture’ the transmission and get its packet through [14].

We now want to determine the probability that a station transmits in a randomly chosen slot, τ . Following the notation from [6], we will denote the probability of being in a state defined by i and k as $b_{i,k}$. This is the stationary distribution for our Markov chain. Since stations only attempt to transmit when the back-off counter has reached 0, this can be expressed as

$$\tau = \sum_{i=0}^m b_{i,0}$$

This relationship is straight forward to understand. We have simply said that τ is the chance we are in a state where we attempt transmission. We now can derive a relationship between $b_{0,0}$ and p . Using the stationary distribution of our Markov Chain, we can write

$$b_{i,0} = b_{i,1} + b_{i-1,0} \frac{p}{W_i} \quad 0 < i < m$$

$$b_{i,k} = b_{i,k+1} + b_{i-1,0} \frac{p}{W_i}, \quad 0 < i < m, k \in 0, W_{i-2}.$$

In the above relationships we are simply summing up the possible ways of arriving into a particular state: either decrement our counter or we have a collision in a lower state. There is only one way of getting into the final state however, that is having a collision.

$$b_{i,W_{j-1}} = b_{i-1,0} \frac{p}{W_j}, \quad i \in 0, m-1.$$

We can substitute this into the equation before it and we obtain

$$b_{i,0} = b_{i-1,0} \frac{p}{W_i} + (b_{i-1,0} \frac{p}{W_i} + (\dots) \dots).$$

Since there are W_i states, this leads us to the relations

$$b_{i,0} = p b_{i-1,0}$$

$$b_{i,0} = p^i b_{0,0}.$$

However we need to be careful about the transitions to the final back-off stage m, k since we can reach these states by having a collision in the $m-1, 0$ state or by having another collision in $m, 0$. Specifically,

$$b_{m,0} = b_{m-1,0} \cdot p + b_{m,0} \cdot p$$

$$b_{m,0} - b_{m,0} \cdot p = b_{m-1,0} \cdot p$$

$$b_{m,0} = \frac{p}{(1-p)} b_{m-1,0}$$

$$b_{m,0} = \frac{p^m}{(1-p)} b_{0,0}.$$

Combining these two equations by summing over $0 \leq i \leq m$, in order to get an expression for τ in terms of $b_{0,0}$ and p

$$\tau = \sum_{i=0}^m b_{i,0} = \frac{b_{0,0}}{(1-p)}.$$

If we consider that we pick a new value k , between W_i and 0 every time we change back-off stage, and that we proceed to state $i, 0$, passing through all the other states i, j where $j < k$ with probability 1, then we can simply state the relationship as

$$b_{i,k} = \frac{W_i - k}{W_i} b_{i,0}$$

It is worth noting that $b_{i,k} \leq b_{i,0}$ and increasing, as k goes to 0. This is because we always count down and never up. As the $b_{i,k}$ values are probabilities we have one last relationship, a normalisation

$$\sum_{i=0}^m \sum_{k=0}^{W_i-1} b_{i,k} = 1.$$

Using the relationships described above we can get a relationship for $b_{i,k}$ generally and then express τ as a function of W_0 , m and p

$$b_{0,0} = \frac{2(1-2p)(1-p)}{(1-2p)(W_0+1) + pW_0(1-(2p)^m)}.$$

Since we can relate τ to $b_{0,0}$ and p from before we can write an expression for τ

$$\tau = \frac{2(1-2p)}{(1-2p)(W_0+1) + pW_0(1-(2p)^m)}. \quad (2.4)$$

2.2.1 Throughput estimation

Now that we have an expression for the chance of transmitting, we can estimate the throughput by comparing how much data we send in an average slot to how long an average slot is. Here we'll consider the simple case where all the packets being sent are the same size. Variable sized packets are considered in [6]. Keeping the notation from [6] and letting S be the throughput we can say

$$S = \frac{E[\text{payload tx in slot time}]}{E[\text{length of slot}]},$$

where $E[X]$ is the expected value of a random variable, X .

Two probabilities that we need to know are the chance that a transmission has taken place in a slot, p_{tx} and the chance that it was successful, p_s . The chance that a station attempts transmission is

τ , but this is not the same as p_{tx} since there may be several stations. We are just interested in the chance that at least one transmission was attempted, and this is simply

$$p_{tx} = 1 - (1 - \tau)^n.$$

Here $(1 - \tau)^n$ is the chance that all n stations do not transmit.

Next we need to get the chance that the transmission was successful. Since we have assumed that packets are only lost to collisions, this is the probability that only one transmission was attempted in the slot. The chance of a successful transmission given that a station transmits is

$$p_s = \frac{n\tau(1 - \tau)^{n-1}}{1 - (1 - \tau)^n}.$$

Here $n\tau(1 - \tau)^{n-1}$ is the chance that any one of n stations transmits, but we need to condition this on the probability that at least one station did transmit. Now, the chance of a successful transmission is simply $p_s p_{tx}$ and so

$$E[\text{payload tx in slot time}] = p_s p_{tx} E[\text{payload}].$$

This is straight forward, but in order to estimate the length of a slot in real time we need to know how long a transmission, a collision and an empty MAC slot are. Let σ be a MAC slot length (for 802.11b = $20\mu s$) and let T_s and T_c be the time for a successful transmission and a collision respectively. We can now say

$$T_s = \text{PHY/MAC header} + \text{PAYLOAD} + \text{SIFS} + \text{ACK} + \text{DIFS},$$

$$T_c = \text{PHY/MAC header} + \text{PAYLOAD} + \text{ACK TIMEOUT}.$$

Note here that the PAYLOAD is a time, determined by the bit rate and the size of the packet. A slot will be of length σ if no one transmits, of length T_s if one station transmits and of length T_c if more than one station transmits. Now we can express the expected length of a slot as

$$E[\text{length of slot}] = (1 - p_{tx})\sigma + p_{tx}p_s T_s + p_{tx}(1 - p_s)T_c.$$

The throughput can now be expressed as

$$S = \frac{p_s p_{tx} E[\text{payload}]}{(1 - p_{tx})\sigma + p_{tx}p_s T_s + p_{tx}(1 - p_s)T_c}. \quad (2.5)$$

Since both p_{tx} and p_s are functions of τ and n we can calculate the throughput for a given number of stations.

2.2.2 RTS/CTS mechanism

In order to consider the RTS/CTS mechanism, the only changes that need to be made are to the values of T_s and T_c . Specifically

$$T_s = \text{RTS} + \text{SIFS} + \text{CTS} + \text{SIFS} + \text{PHY/MAC header} + \text{PAYLOAD} + \text{SIFS} + \text{ACK} + \text{DIFS}$$

$$T_c = \text{RTS} + \text{CTS TIMEOUT}.$$

These are the only changes required to the model in order to accommodate this mechanism and shows that the model is flexible enough to accommodate extensions and improvements without having to be fundamentally altered.

2.3 Extensions to the model

We will now look at how this model can be extended in order to consider more situations, such as varying parameters and load. It is important to note that the approach taken does not change, instead new states are introduced which change the expression for τ and so also change the expressions for p_{tx} and p_s . Once these expressions are known they can be solved as before. In this section we will go through the changes made to the basic model in terms of any new states and transitions that are introduced to the model and then present the resulting expressions for τ . The derivations of these relationships are quite long and are dealt with in full in published literature [6][5][20]

2.3.1 Unsaturated model

The first extension to the model that we will consider is the unsaturated model, proposed in [20]. This is an important extension to Bianchis' model since traffic is not always saturated and it is therefore important to understand the behaviour in unsaturated conditions.

In order to deal with unsaturated conditions, we need to drop the assumption that there will always be a packet to send. We thus need to introduce a new probability q that packet will arrive in a given amount of time. This presents some problems in that we need to think about how to deal with a queue of packets. Also q may become very difficult to estimate if we have to track how long we have waited since the last packet arrived, since we may move through many different states. The first

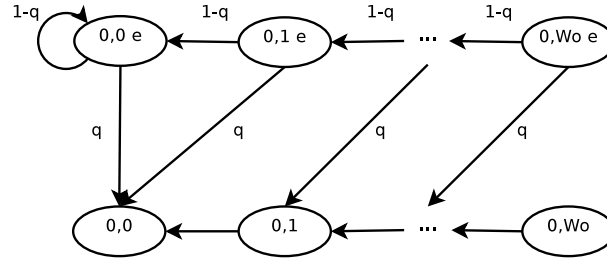


Figure 2.3: Finite load section of model

simplification that is made is to consider that each station has only a single packet buffer, it either has a packet to send or it does not. The second one is that packets arrive in a ‘Poisson’ fashion, and so inter-arrival times are exponentially distributed. This simplifies the task greatly. First if we have a packet to send we can ignore the arrival probability q completely and proceed as before. Second, if no packet is available we can independently check to see if a packet has arrived using q , without taking into account how long we’ve been without a packet, since Poisson arrivals are memory-less.

In order to adapt the model in this fashion, we need to add a new set of states, $(0, k)_e$, which are used when the station has no packet to send. New transitions between $(0, k)$ and $(0, k)_e$ states are introduced, however the later back-off stages are not affected because if a collision has occurred then a packet must be present.

Again we need to look at the transition probabilities, first the transitions not involving the new states.

$$P((i, k-1)|(i, k)) = 1, \quad k \in (1, W_m - 1)$$

This is straight forward, however now it excludes the $0, k$ as there will be new transitions from the new states.

$$P((i, k)|(i-1, 0)) = \frac{p}{W_i}, \quad k \in (0, W_i - 1), i \in (1, m-1)$$

If a collision occurs then we increase the back-off stage by one, to a maximum of m , and choose a new counter exactly as before.

$$P((m, k)|(m, 0)) = \frac{p}{W_m}, \quad k \in (0, W_m - 1)$$

If we are in the final back-off stage, again we choose a new counter from W_m exactly as before.

$$P((0, k)|(i, 0)) = \frac{(1-p)q}{W_0}, \quad k \in (0, W_0 - 1), i \in (0, m)$$

This relationship has also changed slightly. Now when we reach an end state and do not have a collision we end up in one of the $0, k$ states with probability q , since we need a new packet to arrive.

$$P((0, k)_e | (i, 0)) = \frac{(1-p)(1-q)}{W_0}, \quad k \in (0, W_0 - 1), i \in (0, m)$$

This is the first new transition to consider. After transmitting a packet we will enter the $0, k_e$ states if there is no packet to send, and this happens with probability $1 - q$.

Now we need to consider the transitions from the $0, k_e$ states into the $0, k$ states.

$$P((0, k - 1)_e | (0, k)_e) = 1 - q$$

$$P((0, k - 1) | (0, k)_e) = q.$$

These are straight forward relationships, we stay in the $0, k_e$ state if we do not receive a new packet and we move to $0, k$ if we do. So far we have only made minor changes to the scheme. However we have not considered transitions to or from the $0, 0_e$ state. These transitions prove to be complex since in this state we have completed the post back-off and so when a packet arrives we need to test to see if the medium is idle. If it is, we can attempt transmission straight away. If the medium is busy then we need to being another back-off before transmitting. In either case it is possible for the transmission to be successful or result in a collision, a further complication.

In order to proceed we need an expression for the chance that the medium is idle. Given the assumptions we have already made about the model, a good measure for this is to say that it is the chance that if a transmission was attempted in this slot it would be successful. The chance that this happens can be expressed as the chance that one and only one station attempts a transmission, or $(1 - \tau)^{n-1}$ for n stations. However we assume that the only losses occur due to collisions, so this is simply equal to $1 - p$ the chance that we don't have a collision. Note that although the transition probabilities are the same, the real time taken is very different, we are not actually transmitting a packet.

$$P((0, 0)_e | (0, 0)_e) = (1 - q) + \frac{q(1 - p)^2}{W_0}.$$

There are two ways that we can return to state $0, 0_e$ given that we are already in it. First no packet can arrive to be sent and this occurs with probability $1 - q$. Alternatively a packet may arrive, q , the medium is sensed idle, $1 - p$, the packet is transmitted successfully, $1 - p$, and 0 is chosen from W_0 , $\frac{1}{W_0}$. Combining these yields the relationship above, and following similar logic we can describe the remaining transitions.

$$P((0, k)_e | (0, 0)_e) = \frac{q(1 - p)^2}{W_0}, \quad k > 0$$

$$P((1, k)|(0, 0)_e) = \frac{q(1-p)p}{W_0}, \quad k \geq 0.$$

The first relationship is slightly different from the one before as if a packet does not arrive this transition will not take place. Only the second term from the above expression is used. The second relationship describes the situation where a collision occurs, with probability p .

$$P((0, k)|(0, 0)_e) = \frac{q(1 - (1-p))}{W_0} = \frac{qp}{W_0}.$$

Lastly if a packet arrives and the medium is not idle, a new back-off counter is chosen.

The derivation of the $b_{i,k}$ is lengthy and is found in [20]. However it is not materially different to Bianchis' original model as described earlier, in that τ is found in the same manner except with more states to consider. The expression for the throughput does not change at all. This is because p_{tx} and p_s are functions of τ . τ is now a function of q , as is to be expected. Also, the T_s and T_c values are not affected by the introduction of new states. The expressions for p_{tx} , p_s and q are used in the same way, with $1 - p = (1 - \tau)^{n-1}$, to solve the model.

2.3.2 EDCA extensions

The next important extension to the model is to take into account the MAC extensions offered by 802.11e, as discussed in Chapter 1. We will consider three of these extensions, CW_{min} , $TXOP$ and $AIFS$ and show how the model is extended to accommodate these variables. Including CW_{min} and $TXOP$ proves to be very straight forward, however including $AIFS$ is more involved. These extensions are described in detail in [5], including the full derivations of the relationships and the value of τ . This model has been combined with a non-saturated model described in [8].

CWmin and TXOP

Expanding the model to include CW_{min} and $TXOP$ proves to be straight forward in that none of the previous analysis needs to be altered. It is worth noting that in the previous analysis all the stations involved had the same set of parameters, so one set of equations was all that was required. As we introduce stations with different sets of parameters, we refer to these as 'classes' of stations, we need to have set of equations for each. Each class of station will have a different value of τ , so we need to calculate the τ and p values for each class separately. The result of this is the expression

$1 - p = (1 - \tau)^{n-1}$, becomes

$$1 - p_j = \prod_{i \neq j}^n (1 - \tau_i),$$

in order to allow for the different classes.

CW_{min} can be accommodated by changing the value of W_0 for a class of the stations. It is worth noting that CW_{max} can be similarly modeled by changing the value of m .

$TXOP$ can be modeled by changing the value of T_s for a class of stations. This is because there is only contention for the first packet to be sent, as normal, and the remaining packets are sent before any contention may take place. This is another place where we use the assumption that the only losses are due to collisions. If during a TXOP period an *ACK* is not received for any one of the packets, the transmitting station backs off as normal and waits for another transmission opportunity to transmit the remaining packets.

AIFS

Replacing the idle sense time *DIFS* with an adjustable time *AIFS* proves to be quite complex. Unlike the other parameters, *AIFS* can not be modeled by just solving the equations with different parameters for each class of station. Now, a class of stations with a larger value of *AIFS* will wait longer before starting to decrement their back-off counters. If nobody transmits while the station with the largest *AIFS* waits its *AIFS*, then the situation is unchanged. However the model can no longer deal with the situation if another station starts transmission. We need to add more states to the model to deal with this. Quite a large number of states are added or modified in order to accommodate this, here we will only consider the important changes to the model. For a full description see Battiti[5].

As stated above, the key problem here is that with variable *AIFS* a station may sense the medium idle, count down some number of slots and transmit a packet before another station has waited its *AIFS*. To consider this we take the situation where we have two classes of stations with a difference in their *AIFS* values of D . The first class, with the shorter value, is not affected directly by this so we can use the model from before. The second class now has a three dimensional state describing its Markov chain, as can be seen in figure 2.4.

As before, we make the assumption that a station finds the medium idle if it would not have a collision if it attempted transmission. However now we need to distinguish between everybody being silent and

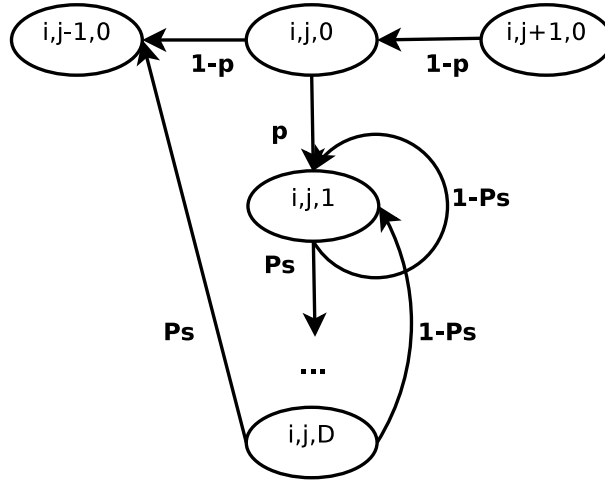


Figure 2.4: AIFS extension to the model

the class 1 stations being silent. The following relationships are for class 2 stations with a larger *AIFS*. If we have n_1 class 1 and n_2 class 2 stations we can define two different probabilities,

$$1 - p = \prod_{i=1}^{n_1} (1 - \tau_i^1) \prod_{j=2}^{n_2} (1 - \tau_j^2)$$

$$P_{s1} = \prod_{i=1}^{n_1} (1 - \tau_i^1),$$

where p is the collision probability as before, but now with two classes of station, and P_{s1} as the probability that the class 1 stations are silent. In the first relationship $j = 2$ since we do not count ourselves.

Figure 2.4 shows part of the Markov chain for a class 2 station counting down. As before we need to remember that we are operating in ‘slotted’ time and a class 2 station can only decrement its counter after waiting an extra D slots. This is because the class 1 stations AIFS values allow them to count down and possibly begin transmission before a class 2 station can begin. So, if a class 2 station would not have had a collision if it had attempted transmission then it decrements its counter as normal. This is because, for this particular slot, nobody was transmitting. However if it would have had a collision, then it must conduct an extra back-off to compensate for its longer AIFS versus the class 1 stations. Only when this is successfully completed does the station decrement its counter.

For a full description of the model, particularly for combining AIFS and the unsaturated models, see [5][8][20].

2.3.3 Other variations

Variations on the basic model are possible as demonstrated above, and also in [28] and [11].

Duffy and Ganesh in [11] expand on the models described above to incorporate an infinite buffer. Their version of the model is effectively the same as the ones described. Certain assumptions about the arrival mechanism have to be made to accommodate the infinite buffer model, it's a Poisson arrival process. However similar assumptions are made about the models described here and the model seems relatively insensitive to the difference.

Xiao, in [28], has also modify the model to incorporate additional features. A three state model is used (i, j, k) , with the i index corresponding to the priority class of the packet to be sent. j is the back-off counter, as in the original model. k describes the back-off stage, however this concept has been slightly changed. The maximum back-off stage corresponds to the maximum allowed number of retries, instead of the largest allowed contention window, CW_{max} in the original model. Xiaos' model is a saturated model and he does not appear to have applied it to *AIFS* although he does mention this in [28]

2.4 Summary

In this chapter we have seen that a discrete time Markov Chain model, originally developed by Bianchi, allows us to investigate the operation of the 802.11 MAC. This approach has the advantage that it can be numerically solved relatively easily using computers. Although the model has had to be extended to include the 802.11e MAC, the basic operation of the model is unchanged. It has proved to be a valuable tool for investigating the operation of 802.11 networks, and has been used here to inform our choice of parameters.

Chapter 3

802.11 testbed techniques and validation

3.1 The Testbed

The testbed was setup to investigate the effects of the 802.11e parameters, as have been discussed earlier, particularly with respect to providing QoS in a distributed manner over an 802.11 network. There were several elements which were vital to the testbed. Firstly, ‘off-the-shelf’ hardware which supported the 802.11e extensions was required and found with Atheros AR5212 based PCI cards. Next, and equally importantly, was access to the driver source code, so modifications could be made and data collected. Finally, small low-powered devices provided a cheap and convenient platform for the test bed.

In the next sections, we will describe the test-bed in detail and explain what modifications were made in order to test the cards and collect data. Then we will investigate the accuracy of the measurements and the cards’ adherence to the 802.11e standard. Finally we will test some of the basic assumptions underlying the analytical models described in Chapter 2.

3.1.1 Physical Setup

The testbed consists of 18 Soekris net4801 embedded systems running the Linux 2.6.8.1 kernel. There are also two desktop PCs running the same Linux kernel version. All of these machines are using the same Atheros based PCI 802.11b/g card.

The reason for having two Desktop PCs was to ensure that sufficient resources were available to record data as well as generate and send packets over the data. One of the PCs acts as a client (STA) and the other is the networks access point (AP), both of which collect data. The Soekris boxes can then act as nodes on the network and, as they do not collect per-packet statistics, they are not a bottleneck to the performance of the network. All the Soekris boxes and the two PCs are connected via wired ethernet to facilitate control of the testbed.

As mentioned before, the primary reason for choosing the Atheros based WLAN cards was the open nature of their drivers. It is vital for this work that we have control over the 802.11e parameters and the open source driver allowed this. Initially the parameters were hard-coded into the driver; however, this proved to be inefficient since the drivers need to be recompiled each time a parameter is changed. Later revisions of the MadWiFi driver allowed these parameters to be adjusted via the `iwpriv` command. It should be noted that in version 2133 and its predecessors, a bug exists that stops the parameters being set on the best-effort queue when the card is in 'Master' mode, i.e., it is acting as an access-point. However this did not impact on the experiments since a different queue can be used.

3.1.2 Driver modifications

Since the 802.11e EDCA parameters can be set via `iwpriv`, modifications to allow parameters to be set eventually proved redundant. Three other modifications were required however, to time-stamp the packets as they were added to the queue, to print out statistics for each packet and to stop the driver attempting multi-rate retries.

Time stamping was accomplished by modifying the `ath_buf` structure found in `ath/if_athvar.h`. An extra variable was added to record the time at which the packet was added to the drivers queue. This structure is created by the driver and contains a pointer to the packet along with all the details the card needs to send the packet. It is then passed to the card and queued. When the packet is

queued the time can then be saved to this variable and then compared with the time at the point when the packet has been sent, or dropped due to excessive retries, an event that we will see later is very rare. We set a flag in the `ath_buf` structure to request an interrupt immediately after the packet is successfully sent (or dropped). Initially a system call was used to get the time-stamp however the driver exposes a time stamping function on the Atheros cards themselves, `ath_hal_gettsf32()`, which was used in preference. This was because the hardware actually timestamps the success (or drop) automatically, and using the hardware time-stamp would make the immediate interrupt unnecessary. However, the hardware time-stamp was provided at a millisecond, rather than microsecond, resolution, so the interrupt based scheme was found to be satisfactory. There seemed to be no difference between either approach other than the format of the time-stamp, although using the Atheros cards time-stamp function had the potential to avoid extra interrupts, this feature was not used.

The driver was modified to print out statistics for each packet that the driver sends. This excluded special packets such as ACK packets, since they are handled specially by the driver. This was done by simply adding a `printk()` statement to the end of the `ath_tx_processq()` function. At this point a packet has either be successfully sent or dropped due to excessive retries. The driver also records statistics for the packet, such as the number of retries and the rate at which it attempts to send the packet. The current time is printed out here as well, again using the same function. We know this time to correspond closely with the completion time of the packet, because we have requested an immediate interrupt. The timestamps and the other statistics are then collected using `dmesg` running in a loop, although they can also be recorded using `syslogd`, and processed offline. The relevant code is shown below, with `ath_hal_gettsf32(ah)` being the function call to get the current time, and `bf->time_stamp` being the time-stamp recorded when the packet was queued.

```
printk("%d\t%u\t%u\t%d\n",
ds->ds_txstat.ts_seqnum,
ath_hal_gettsf32(ah), //needs %u ...
bf->time_stamp, //needs %u ...
bf->bf_skb->len,
);
```

By default the driver will attempt to send the packet at different rates depending on the number of the retry that is currently being attempted. This would obviously have an effect on results so this behaviour was disabled. This was accomplished by changing the rate control section in `ath_tx_start()`

so that the transmission rate was hard-coded into the driver. All packets were sent at the desired rate of 11Mbps.

```
ath_rate_findrate(sc, an, shortPreamble, skb->len,
                  &rix, &try0, &txrate);

//ian
//added this, for 11Mbps
rix = ath_tx_findindex(rt, 11000);
txrate = rt->info[rix].rateCode;
try0 = ATH_TXMAXTRY;
```

Here setting the default number of retries to `ATH_TXMAXTRY` disables multi-rate retries, where a packet could be sent at a lower rate depending on the retry number. It should be noted here that 802.11 ACK packets can be set to be transmitted at different rates. By default the driver sends the ACK packets at 2Mbps, if it is transmitting packets at 2Mbps or higher. These packets can be sent at the same rate as other packets using a `sysctl dev.wifi0.ackrate=1` command.

3.1.3 Other software

In order to study voice over IP (VoIP) in a real testbed a source of VoIP traffic is required. Using recorded conversations is not particularly practical since it is difficult to playback two way conversations over the network. Also the conversations must be representative of ‘typical’ use and long in duration so that they are not looped over excessively. We chose to use a G.711 [15] style encoding of speech. This scheme requires 64 Kbps of bandwidth, 32 Kbps in each direction, and with a 10 millisecond packetisation interval. Silence suppression, where no packets are transmitted if no sound is being made, is simulated by a talk-and-listen behaviour of the simulator described below. Both stations talking and both stations speaking are situations not simulated. These values still represent a relatively pessimistic estimate for VoIP traffic. Voice codecs can operate with smaller packets and with a larger packetisation interval, however the values used correspond to G.711 codec so they are ‘real’. It should be noted that it has been observed [] that the packetisation interval is important for 802.11 VoIP due to large packet overheads.

A simulation of voice traffic, as suggested in [21], was used in place of real recorded speech. The traffic

generated is two-way, with the 'on' period of one station corresponding to the 'off' period of the other station. This is equivalent to silence suppression used in many voice codecs. Specifically, the simulator generates 80 byte packets, one packet every 10 milli-seconds, for an exponentially distributed period with a mean of 1.5 seconds. The other station is silent until the first station finishes then generates another period according to the same scheme and transmits for that period. A minimum period of 0.25 seconds is applied to better simulate the minimum talking period. The stations communicated their generated period to the other station via wired ethernet, in addition to the wireless link, so that the other station always starts at the correct time even if the packets are lost. By using a pessimistic estimate we can say that real VoIP traffic will always have 'better' properties than the traffic generated by our simulator. Here by 'better' we mean lower delay and higher (or more consistent) throughput.

In addition standard tools such as 'tcpdump' [16] and 'iperf' [27] were used in the testbed.

3.2 Measurement

The changes to the driver allow us to collect data and make measurements in several ways. The next sections will describe the measurement of four important variables, the delay, throughput, collision probability and the driver queue occupancy. It also allows us to test whether the driver and hardware are behaving in a standard compliant manner. This work is based in part on the results presented in [19].

3.2.1 Delay

Delay is the single most difficult measure that we make in the testbed, simply because the timescales are so short, on the order of 10^{-6} seconds, and that we are really interested in the time between a packet reaching the front of the senders queue and being successfully received.

As discussed above each packet is timestamped twice, T_{start} and T_{end} , once when the packet is put into the drivers MAC layer queue, and again after the packet has been sent or dropped. There are two separate delay statistics that are of interest, and which can be worked out from the timestamps, the queuing delay and the MAC delay. The MAC delay is the delay that the time that the card spends contending for access to the medium and the time spent transmitting the packet. It also contains the

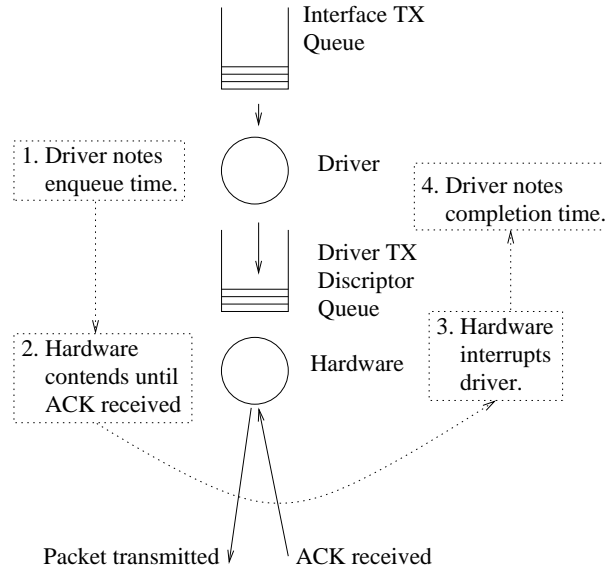


Figure 3.1: Measurement Scheme

delay for any retransmissions of the packet and repeated contention attempts. The queuing delay is the time spent by the packet in the MAC queue waiting to be transmitted.

Figure 3.1 shows the scheme that is employed to calculate these delays. The sum of the queuing and MAC delays is simple to work out, it is the difference between the two timestamps for a give packet. The MAC delay is found from comparing T_{end} of the previous packet to the T_{start} of the current packet. If T_{end} is more recent then the packet has spent some time in the queue. The MAC delay is worked out according to

$$\text{MAC delay} = T_{end} - \max(T_{start}, T_{end}(\text{lastpacket}))$$

and the queuing delay is simply

$$\text{Queuing delay} = T_{end} - T_{start}$$

In this way we can get the two delay values for each packet. A nice property of this scheme is that only one clock is required to get this information. This is possible since for a successful transmission an explicit acknowledgement must be received by the sender.

The sum of these delays may seem to be the most important statistic since VoIP is very time sensitive and the sum corresponds to the time that a packet takes to be sent including queuing by the driver.

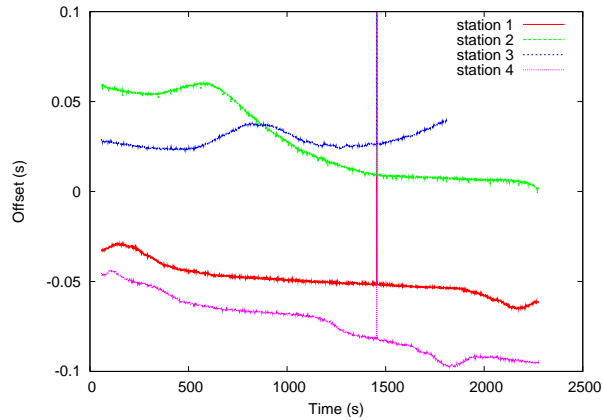


Figure 3.2: NTP timing

However, the MAC delay turns out to be very useful since it is related to the service rate for the drivers' MAC queue.

An alternative way to measure delay in this setup is to try to synchronise the clocks on both the sender and the receiver. This is commonly attempted using 'NTP' the network time protocol which communicates the differences in time between two clocks over the network. This is not ideal, however since the latency even over wired ethernet can be tens of micro-seconds. This is an insignificant amount of time for normal purposes but here we really want to record times with an accuracy of around a slot length, which is 20 micro-seconds for our experiments. This is so we can see the back-off behaviour of the wireless cards. Figure 3.2 illustrates some of the timing difficulties we experienced using the NTP approach. Other researchers have investigated this area, for example Melvin and Murphy [22] and Y. Sun et al. [25].

The first test of this scheme is to ensure that reasonable times are indeed being calculated for packets. To test this a single station sent data packets of variable size, at a low rate to the access point. This is to ensure that the back-off is not included in the measurement. The delays are shown in Figure 3.3. The most common values are plotted for each of the data sizes. We don't expect to see any delay due to contention since there are no competing stations and packets are being sent at a low enough rate that post-back-off will have completed by the time a new packet arrives to be sent.

As expected, the slope corresponds to a data rate of $11Mbps$. Also, the experiment was repeated using a long preamble which should add $192\mu s$ to the delay, due to the differences in the PLCP header.

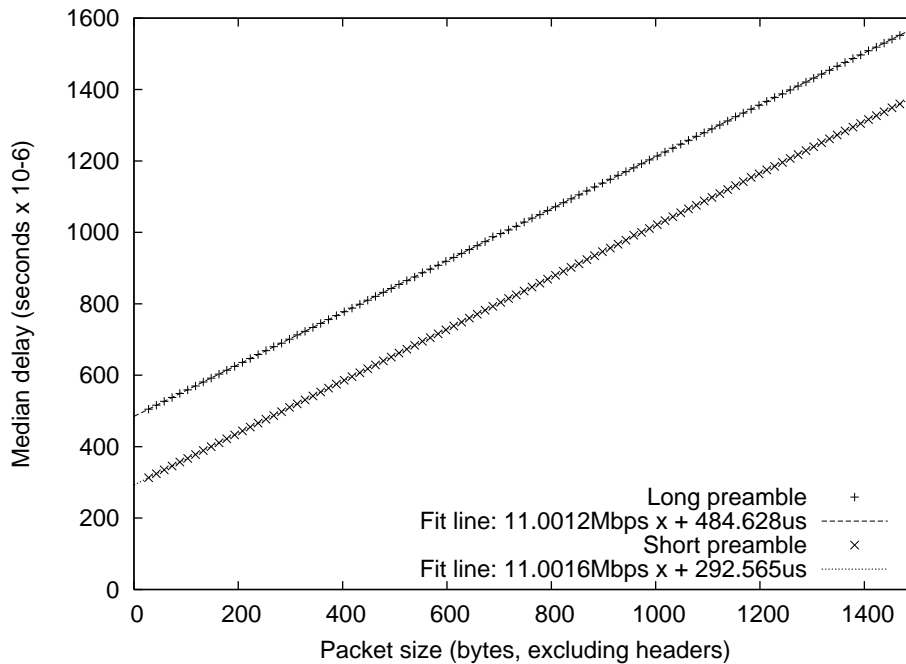


Figure 3.3: Packet size versus delay for long and short preambles

When a ‘long’ preamble is being used, all of the header is transmitted at 1 Mbps and the first section of the header is 144 bits in length. When a ‘short’ preamble is being used the first section is sent at 1 Mbps but is only 72 bits long, and the rest of the header is sent at 2 Mbps. It is clear that a short preamble takes half as long as a long preamble, and since it is sent twice, once for the data packet and once for the ACK, the duration is reduced by $192\mu s$.

3.2.2 Standard Behaviour

Before we can start to investigate VoIP and other traffic types we need to be certain that the cards are adhering to the 802.11 standard and that we can control the 802.11e EDCA parameters. Some cards have been observed to exhibit non-standard behaviour that could invalidate our measurements [7]. The three parameters we are concerned with are CW_{min} , the initial maximum contention window size, $AIFS$ the variable inter-frame and $TXOP$ the time, in μs that a station can hold the medium.

Figure 3.4 shows the distribution of MAC delays recorded on one station sending fixed size packets at a high rate. Here the station always has packets to send so we should see a distribution of delays

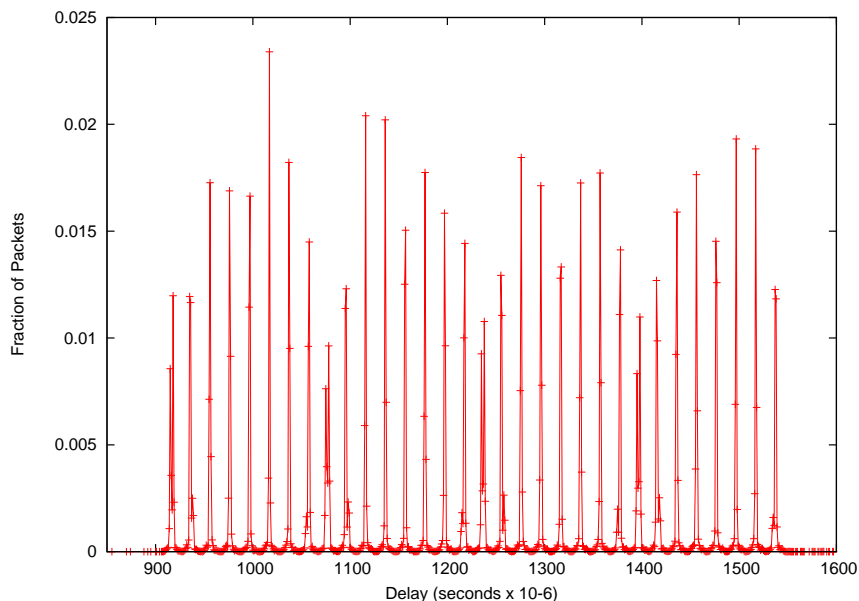


Figure 3.4: Delays when CW is 0 to 31

corresponding to a varying post-back-off which depends on the CW_{min} value chosen. Here the values correspond to the original 802.11 standard, $CW_{min} = 31$, $AIFS = DIFS = 2$ and $TXOP = 0$. We see 32 peaks which corresponds to the MAC choosing values from 0 to CW_{min} as expected.

Next Figure 3.5 show the same experiment but with $CW_{min} = 2^2$. Now we see only four peaks again as expected. These results indicate that the cards are indeed behaving in accordance with the standard regarding CW_{min} and that we can control it's value. The first peak is also in the same place as before.

Figure 3.6 shows the same experiment as in figure 3.4 but this time the value of $AIFS$ was increased to 8. We still see 32 peaks but the value of the first peak is now shifted out $120us$ as we expect.

$TXOP$ is last to be tested. Using the results obtained in figure 3.3 we can estimate the (minimum) time for a packet of a given size to be transmitted. By setting $TXOP$ to be larger than this time plus $AIFS$ and $CW_{min} * slotlength$ (here $CW_{min} = 2^2$) but smaller than twice this value, we know the station will be able to send 2 packets every time it wins a contention opportunity. Further more only the first of these two packets will actually contend for access as the second is sent after $SIFS$. In Figure3.7 we can see 5 peaks. Comparing with figure3.5 we can see that the last 4 peaks are in the same place and correspond to the normal post back-off. The new large first peak corresponds to the

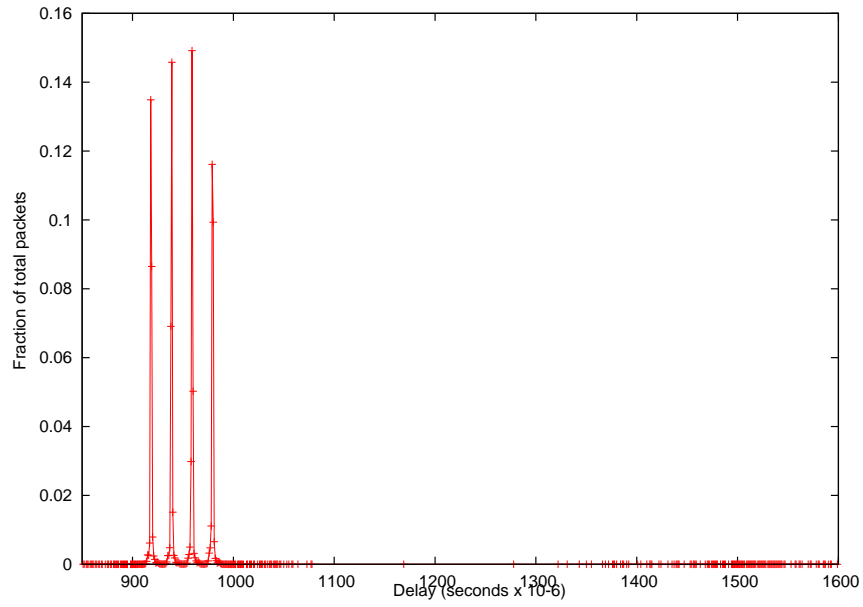


Figure 3.5: Delays when CW is 0 to 3

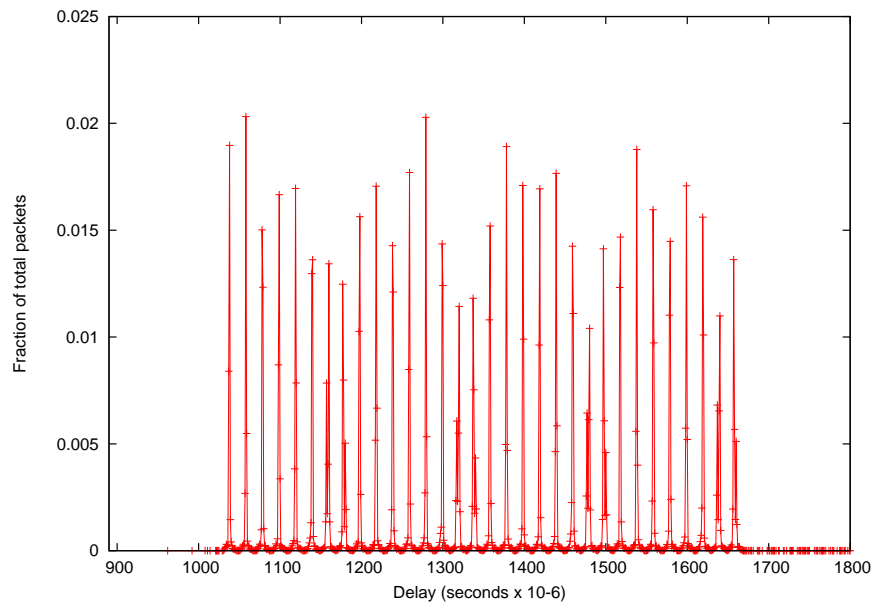


Figure 3.6: Delays when CW is 0 to 31 and AIFS is 8

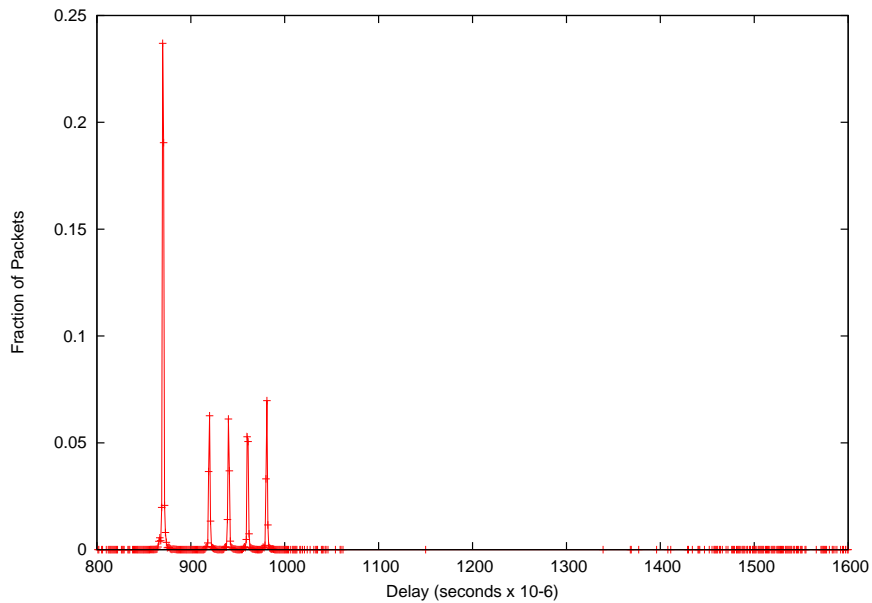


Figure 3.7: Delays when CW is 0 to 3 and TXOP is set for 2 packets

‘second’ packets being sent and constitute roughly half of the total number of packets.

3.2.3 Throughput

Throughput is comparatively easy to measure in comparison to the delay. The driver prints out the final state of the packet (sent or dropped) and so knowing this we can simply add up the number of packets, multiply by their size and divide by the time of the experiment to get the average throughput. Since timing using shell scripts is not very exact we use the last time-stamp minus the first time-stamp recorded to determine the length of the experiment. We can also use tcpdump to record packets as they arrive at the access-point and work out the throughput from it in a similar manner.

We have seen that we can control the 802.11e EDCA parameters, but to make sure that this approach gives the expected throughputs, we test again. This time a pair of stations are used, both of which always have a packet to send. One station has the default 802.11 parameter (as before) while we vary the other stations parameters and compare the throughputs that each station gets.

Figure 3.8 shows a varying CW_{min} , in powers of two. The station that increases its CW_{min} shows a dramatically dropping throughput. This is because, at each step its chance of winning the next

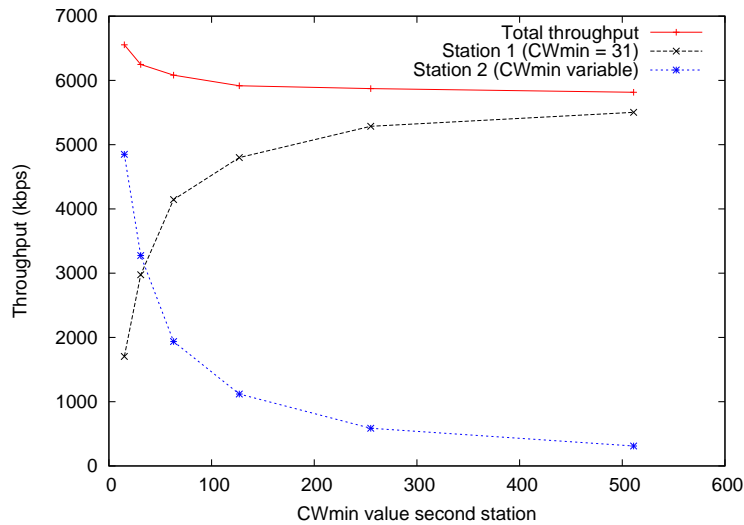


Figure 3.8: Variation in throughputs for two stations as one station varies $CWmin$

transmission opportunity is halved. The blue line indicates the sum of the throughputs and decreases slight as the $CWmin$ value is increased. This is because as $CWmin$ goes up more time is spent counting down and so the total throughput of the network will drop.

Next we look at the impact $TXOP$ has on the throughput. Figure 3.9 is quite straight forward, the two stations throughput remains the same until the first stations $TXOP$ is long enough for it to send a second packet. Then its throughput jumps up, while the second stations throughput drops dramatically. It's worth noting that the first stations throughput doesn't double, but rather increases by half at the expense of the other station. This is because both stations are still contending for access to the medium but the first station gets to send two packets when it wins. The blue line, again indicating the total throughput, increases as one stations $TXOP$ value goes up. This is because less time is being spent contending for access and more time is being spent actually transmitting packets. Although this makes more efficient use of the available bandwidth it is obvious that this is not fair to the second station.

Lastly in figure 3.10 we look at the impact varying $AIFS$ has on the throughput. The impact is harder to understand quantitatively. $AIFS$ controls how long a station needs to hear the medium idle before restarting its back-off timer. This impacts the chance of winning a transmission opportunity since after a transmission the first station can count down x slots before the second station can restart its back-off counter (where x is the difference) in $AIFS$ values. So as the difference in $AIFS$ values

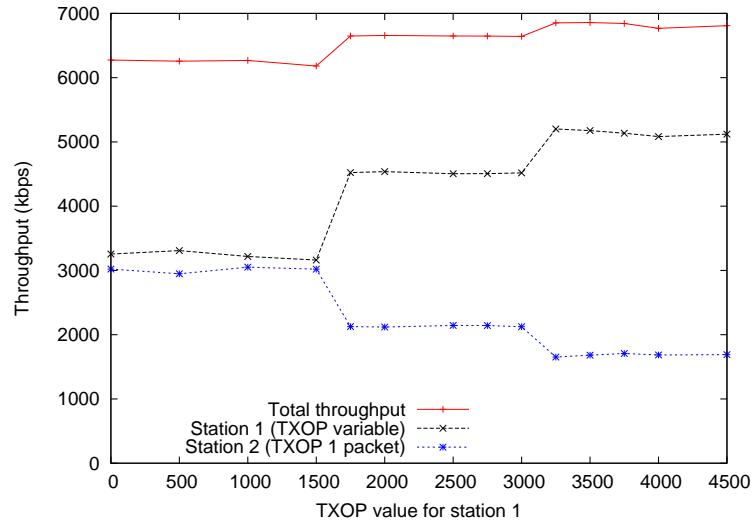


Figure 3.9: Variation in throughputs for two stations as one station varies TXOP

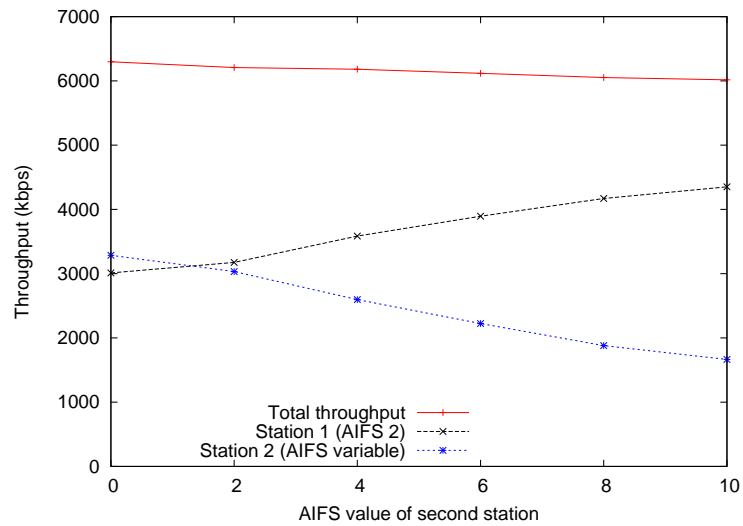


Figure 3.10: Variation in throughputs for two stations as one station varies AIFS

increases the second stations chance of winning a transmission opportunity decreases, it sends fewer packets and so its throughput drops. Also, the total throughput slowly drops since more time is being spent contending for access and less time is spent sending packets.

It's worth noting that *AIFS*, in this respect, is linked to the *CW* value of the *other* station. If the first station decreased its *CW_{min}* value, the second station would be even more severely penalised. This is because the first station is gaining in proportion to the chance of it picking a *CW* value that is less than or equal to the difference in *AIFS* values between the two stations. So we expect a roughly linear decrease in throughput as *AIFS* is increased on one of the stations. It is worth noting that if $CW_{min1} < AIFS_1 - AIFS_2$ then station 1 can effectively lock out station 2, (for the first back-off stage).

3.3 Studying Model Assumptions

One of the key assumptions made in Bianchi's model is that the collision probability is independent of back-off stage when attempting to transmit a packet. The modifications made to the driver print out the number of retransmission attempts that are made for each packet before being successfully transmitted or dropped although dropping a packet in this way is a very rare event. This allows us to calculate the collision probability and attempt to test the validity of the constant collision probability assumption. This work is discussed in detail in [19]. We also assume that losses are only caused by collisions with other packets. The error rate, the packet loss rate not due to collisions, can be estimated by running traffic from a single station to the access point without any other stations transmitting and looking at the number of retries experienced. It is worth noting that in our testbed the error rate, losses not caused by collisions, was consistently $\leq 0.1\%$ when checked.

The collision probability was calculated by counting the total number of retransmissions experienced during an experiment. This should give the total number of collisions. The probability is worked out according to

$$p(\text{collision}) = \frac{R_{tot}}{P + R_{tot}}$$

Where R_{tot} is the total number of retransmissions and P is the number of successful transmissions. This expression gives us the average collision probability including all the back-off stages. In order for the assumption to be true the collision probability should be the same regardless of the back-off stage. Since we know how many retransmissions, and therefore collisions, each packet experienced we can

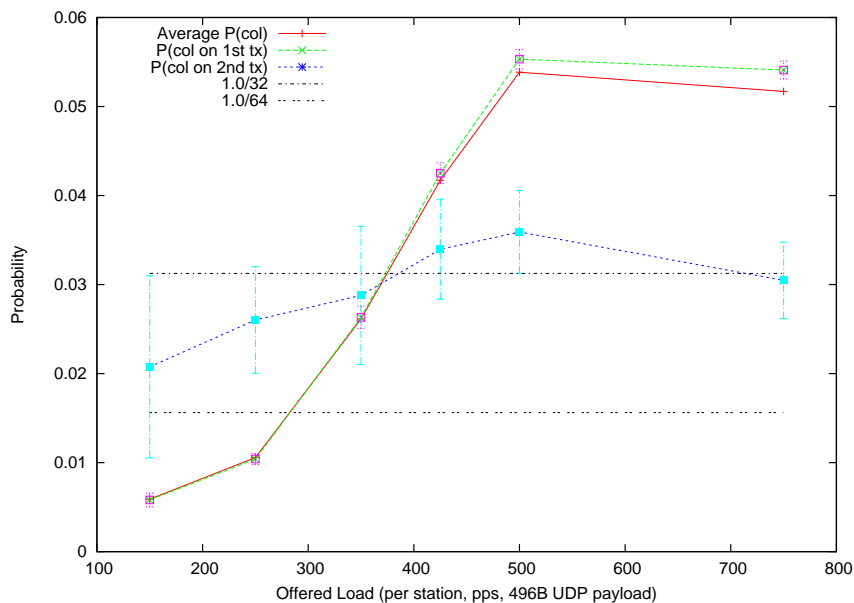


Figure 3.11: 2 STA P(collision) versus offered load

also calculate the probability of a collision for each back-off stage. If R_i is the number of transmissions a packet has experienced then this is done according to

$$p(\text{collision } n^{\text{th}} \text{ transmission}) = \frac{\#(\text{packets with } R_i \geq n)}{\#(\text{packets with } R_i = n - 1) + \#(\text{packets with } R_i \geq n)}$$

We exclude packets with less than n retransmissions because we are effectively conditioning the probability on having had n collisions already.

In figure 3.11 we can see the average collision probability for the case where there are two stations on the network as we vary the load. We can also see the first two conditional collision probabilities, the probability of a collision occurring on the 1st and 2nd back-off stages. The error is estimated using $\frac{1}{\sqrt{N}}$ with N being the number of packets having at least n retries.

We can see clearly that the probability changes as we move from the first to the second back-off stage. The collision probability for the first back-off stage is very close to the average collision probability, however the collision probability for the second back-off stage is much flatter (given the much larger error bars). This is expected since the collisions are dominated by collisions at the first back-off stage.

The difference can be understood in this case due to the fact that there are only two stations on the network. If one of the stations experiences a collision then the other station has also experienced

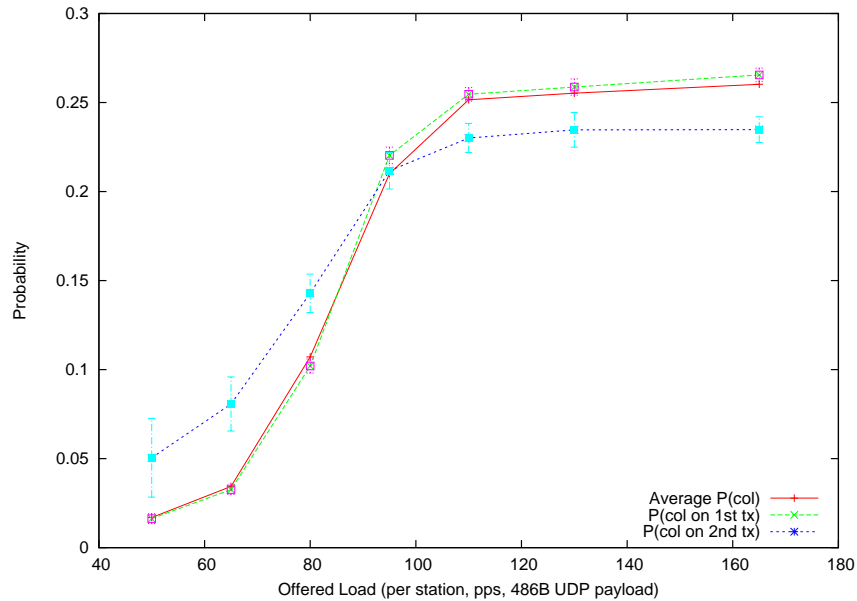


Figure 3.12: 10 STA P(collision) versus offered load

one, and so the stations behaviour in the second back-off stage is very closely coupled. We would expect that the collision probability at this back-off stage be less sensitive to the offered load since we know that both stations are attempting transmission. In this situation we might expect the collision probability to be $\frac{1}{64}$ since both stations are in a second stage back-off. However the measured collision probability is higher than this, even given the error bars, for almost all the offered load. A possible explanation for this is that if one of the stations has a successful transmission and has another packet waiting, there is a chance that this second packet could collide. Thus we would expect the collision probability to rise as the load increases, even in such a tightly coupled situation, and this is what we see in figure 3.11.

Next we look at the same collision probabilities but with ten stations on the network. Now all three of the collision probabilities are closely related, again with the collision probability at the first back-off stage very closely matching the average collision probability. Although the second back-off stage collision probability is much closer to the first and average collision probabilities it is still different as can be seen from the error bars. Since there are now ten stations sharing the medium explicit reasoning for this behaviour is more difficult. We do see the same trend in that there is a sharp increase in the collision probabilities as the offered load is increased. The collision probabilities also level out as the network becomes saturated.

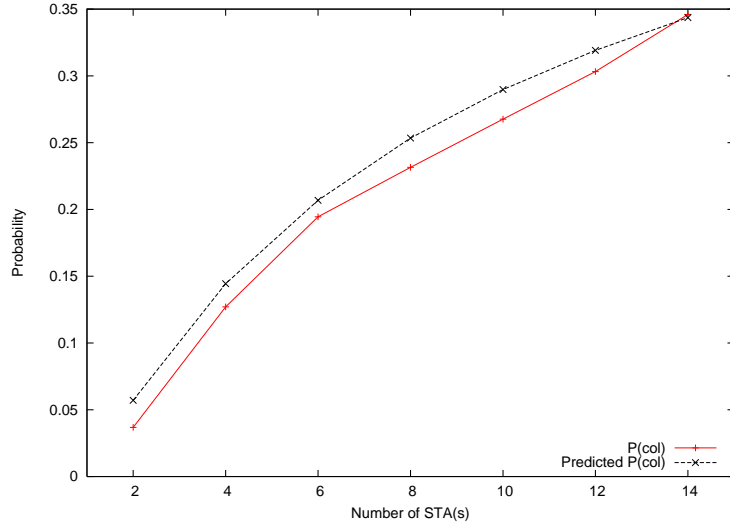


Figure 3.13: Measured versus expected collision probability

We can take from this that, while it may not be strictly true that the collision probabilities are independent of the back-off stage, the average collision probability is heavily dominated by the first stage collision probability and so it is a reasonable approximation. As reported in [7], there exists differences in implementation between different wireless cards, some non-standard. It is possible that some non-standard implementation to do with the precise value of `ACK_TIMEOUT` could affect the behaviour depending on whether the station was a sender in a collision, or simply observed a collision on the medium. This could lead to a slight change in the collision probability for retransmissions, however a full investigation of this is left as future work.

The next thing we can do is examine the relationship between p and τ the transmission probability. The model, especially the extensions described in chapter 2, makes use of the assumption that the probability of a collision occurring is the chance that a station would experience a collision if it attempted to transmit a packet. In other words that

$$1 - p_i = \prod_{k \neq i} (1 - \tau_k)$$

The model calculates τ from the parameters given to the model, the window size and number of nodes among others, and then uses τ to estimate the value of p by solving the two expressions. We can write the relationship for the throughput in terms of τ , using the above relationship

$$S = \frac{E(\text{payload})n\tau(1 - \tau)^{n-1}}{\sigma(1 - \tau)^n + T_s n\tau(1 - \tau)^{n-1} + (1 - (1 - \tau)^n - n\tau(1 - \tau)^{n-1})T_c}$$

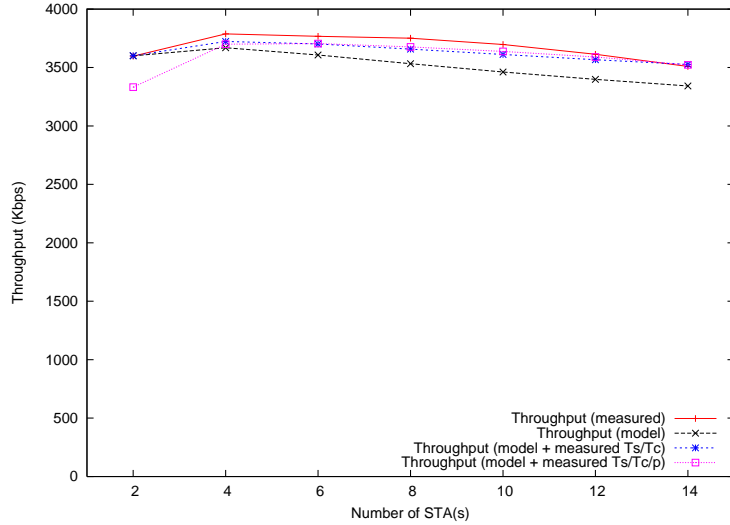


Figure 3.14: Measured versus expected throughput

Here $E(\text{payload})$ is the time taken to transmit the payload in a packet, σ is a MAC level slot $20\mu s$ in our case, T_s is the time for a successful transmission and T_c the time for a collision.

Figure 3.13 shows the measured collision probabilities compared with the estimated collision probabilities using the model. The model seems to overestimate the collision probability yet it is a close match.

Figure 3.14 shows the measured throughput of the network against the models throughput predictions. In order to predict the throughput we need estimates of T_s and T_c . We can measure these values from the test-bed itself. T_s can be estimated as the shortest time measured for a packet to be sent, ensuring the back-off is not included. T_c can be estimated by subtracting T_s from the time measured for a packet with 1 retry. Initially using the standard values for 802.11b we should get $T_s = 908\mu s$ and $T_c = 964\mu s$. This yields a fairly close match. However measured values for T_s and T_c from the testbed were different to the predicted ones, particularly in the case of T_c . They were $T_s = 916\mu s$ and $T_c = 677\mu s$. Using these values the match to the measure throughput results in a better match. This indicates that the hardware is resuming the back-off procedure more quickly than the standard requires, it seems that it does not wait the full time to transmit an ACK at $1Mbps$.

Another test is to use the measured value of the collision probability instead of calculating it using the model. Using this and the measured values of T_s and T_c results in a close match for all but the

situation with 2 stations. It is not noticeably closer than simply using the model though, indicating that the model is not sensitive to changes in these values.

Given these results we can say that the collision probability does have some dependency on the back-off stage however they are dominated by collisions at the first stage and so this seems a good approximation to make. If a small number of stations are being used and there was some reason to expect a large proportion of second and higher stage retransmissions then this may become an issue, however this does not seem to correspond to any set of real world network conditions. The throughput predictions match the model predictions very closely in saturated conditions. They seem to be fairly insensitive to small changes in p , measured versus predicted, although there is a definite difference when only 2 stations are present.

3.4 Issues and Improvements

Some issues with the Atheros cards were encountered during the operation of the test-bed. Firstly, there is an as yet unresolved issue with one of the access category queues when the card is operating as an AP. $CW_{min/max}$ *AIFS* and *TXOP* cannot be set on the best effort, *BE*, queue. This can be solved by forcing packets through the three other queues that the driver provides, but is a problem when several types of traffic with varying priorities are going through the AP.

There has also be a reported issue with Atheros cards which result in strange behaviour if the cards are used over distance. See Bianchi et al [7]. Again this issue was not a major concern since all the stations were within several meters of each other.

The methods employed to collect data from the driver could be improved. In this case our method of using `sysloyd` and `dmesg` was sufficient due to the relatively low bit-rate employed. For higher performance RelayFS [3] or similar kernel extension should be used. This may also enable real time measurements which would also be a significant improvement.

3.5 Summary

In this chapter we have demonstrated an efficient way of measuring delay statistics accurately and using commodity hardware. The approach to delay measurement is efficient and with modification could be used on different platforms and in a real-time manner.

We have also tested the Atheros PCI cards adherence to the 802.11e standard and confirmed that the cards when used with the MadWiFi Linux driver, allow control over the various 802.11e parameters.

We were then able to use the test bed to investigate the assumption, implicit in the analytical models, that the collision probability is independent of the back-off stage. This was found to be generally not true, however because transmissions are dominated by the first back-off stage, any corrections at higher orders would not be likely to have a significant impact. This confirms that it is an assumption but it is justified for the scenarios we consider.

In the next chapter we will investigate VoIP traffic over the testbed, on its own and in conjunction with saturated traffic, and investigate how to prioritise and maintain acceptable performance using the 802.11e extensions.

Chapter 4

Experimental VoIP measurements

4.1 Introduction

In this chapter we are going to take a look at the experimental results which show how well VoIP operates over a real wireless network. For these experiments we used the older 802.11b standard and associated parameters. The main reason for this is the maximum bitrate of 11Mbit/second which, when combined with a testbed of about 20 nodes, allows us to comfortably saturate the medium. There are a few settings which deserve mention. Firstly a slot length of $20\mu s$ was used. Secondly, as previously discussed, a long preamble was also used. The more recent 802.11g standard allows a slot length of $9\mu s$ and 802.11b/g allow shorter preambles. Either of these features would be expected to improve performance.

Here we will first look at the performance of VoIP traffic on its own and look at a variety of prioritisation schemes. We will also look at the impact of buffer size on VoIP traffic to investigate what gains may be made in this regard. This also allows us to see how many VoIP conversations we can have over our testbed, given the parameters we have chosen. Next we move on to VoIP in competition with other saturated traffic, and how to prioritise the VoIP traffic so that it is able to function despite sharing the medium with other, potentially badly behaved, traffic.

4.2 Performance of VoIP over 802.11

As a starting point it is very important to understand how VoIP would perform in isolation on a network, so that we can compare the performance of various schemes to prioritise the traffic. This section is based in part on results presented in [10]. In our setup we have two distinct classes of station, ordinary stations and the access point. The access point handles half of all the traffic because all the VoIP traffic passes through it. This results in a well documented unfairness between the AP and the stations, as the number of stations increases [24]. In our testbed each VoIP conversation involves one station and the AP. This is natural since wireless technology commonly provides the last hop connection for networked services.

Consider a network with n stations carrying VoIP calls and an AP. We vary the number of stations and check the performance at the stations and the AP. We also want to investigate buffer sizing to see if gains could be made by adjusting it. Initial experiments prioritised the AP using *TXOP*, *CWmin* and adjusted the buffer size at both the AP and the stations. Two different prioritisation schemes were tried. First, *TXOP* at the AP was set to be equal to the number of stations on the network. This should give half the network throughput to the AP. However using *TXOP* ties up the network for longer periods of time, and since VoIP is very time sensitive an alternative where *TXOP* was half the number of stations and *CWmin* was halved was also examined. The results can be seen in figure 4.1 and figure 4.2, where MAC delay is shown for various combinations of buffering prioritisation schemes.

Figure 4.1 is perhaps the most striking. Despite having three prioritisation schemes and different buffer sizes, the results clearly separate into only two groups. The worse performing group, allowing at most 10 conversations, consists of all the situations where *TXOP* and *CWmin* were left at their default values and were the same values used for the stations. The second group, allowing at most 12 voice conversations, consists of all the experiments where some prioritisation was applied to the AP. There seems to be no difference in terms of MAC delay as regards the use of *TXOP* in these schemes. The buffer size at the AP seems to have no impact in terms of the MAC delay performance. Figure 4.2 shows all the schemes performing roughly on a par until large numbers of stations are present, when we know that the performance at the AP is already unacceptable.

We need to look at the throughput performance at the AP to see the impact of the buffer size. Figure 4.3 compares the different buffer size experiments to the model with a single buffer, discussed

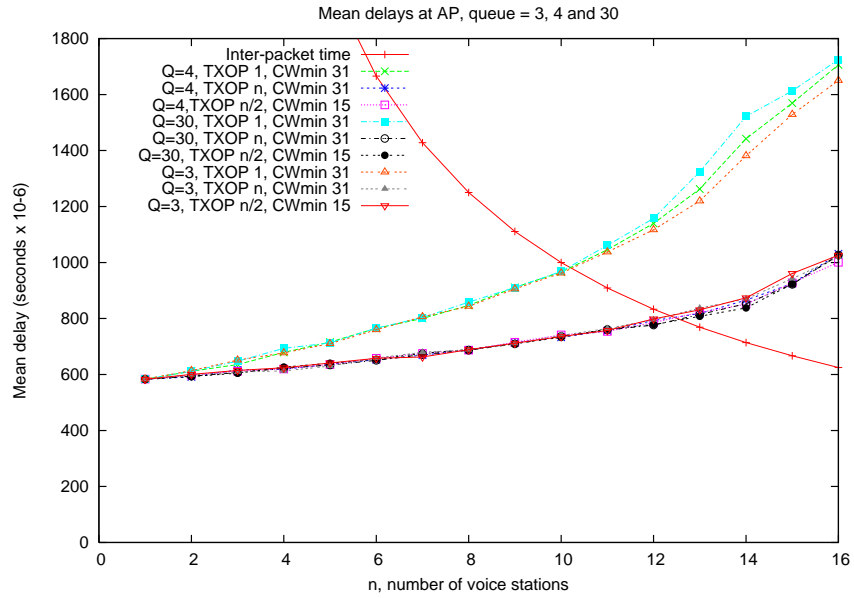


Figure 4.1: Mean MAC delays at AP for variable buffer sizes

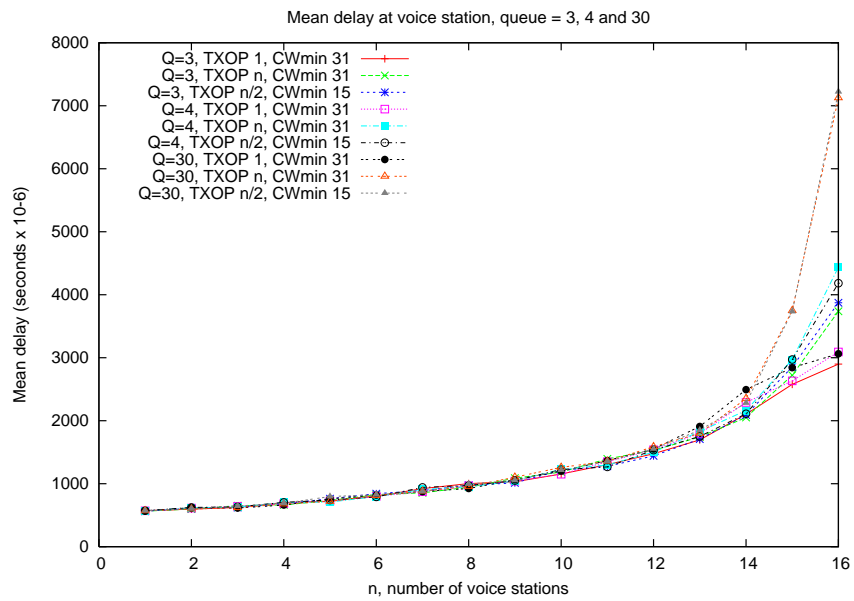


Figure 4.2: Mean MAC delays at a STA for variable buffer sizes

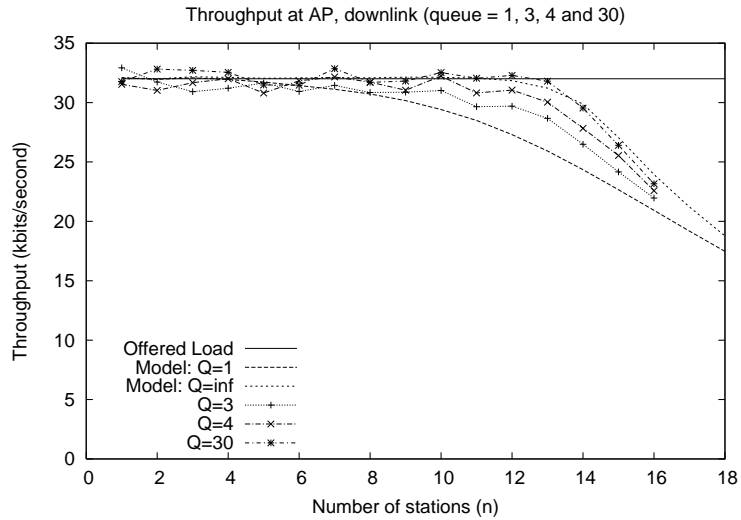


Figure 4.3: Comparison between model predictions and throughput at an unprioritised AP

in Chapter 2, and an infinite buffer[11]. We can see that the performance of the 30 packet buffer is very close to the predicted by the infinite buffer case, and indeed the experiments results are bounded by the model predictions. This indicates that once that a queue is big enough little is gained by increasing the buffers size. However, if the buffer is very small, 3 or 4 in our experiments, then there will be an impact on the performance.

Perhaps the most significant lesson to be taken from these experiments is a little less obvious. Figure 4.3 shows the throughput for an unprioritised AP, the better to see the impact of the buffer size. It shows the throughput remaining acceptable up to 13 stations when the buffer is 30, and 10 when the buffer is 3 or 4. However in figure 4.1 the unprioritised AP cases have a mean MAC delay which is larger than the inter-packet time for all cases where more than 10 stations are present.

The explanation for this is that VoIP traffic is quite bursty, and so having a sufficient buffer allows the throughput to be maintained even if the mean MAC delay is becoming longer than the desired inter-packet time. In the range 11 to 13 stations, where we have a sufficient buffer, we should not expect the quality to be acceptable since on average the packets are taking too long to be transmitted.

In figures 4.4 and 4.5 we can see the performance for a prioritised AP. In both cases *TXOP* is used to prioritise the AP, and its impact on the performance seems less than having an adequate buffer. We can take from this that the single most important step to take in an all VoIP network is to prioritise

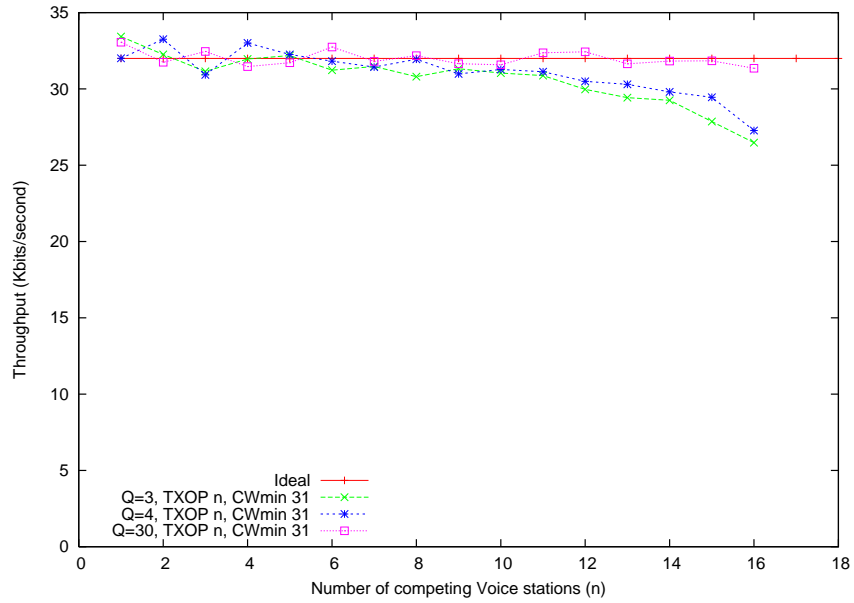


Figure 4.4: Throughput at prioritised AP, TXOP=N, CW=31

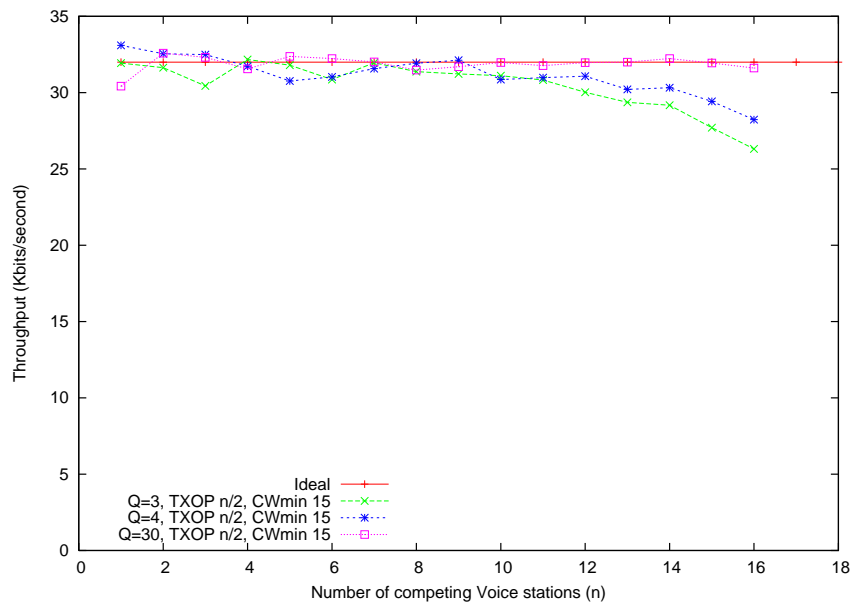


Figure 4.5: Throughput at prioritised AP, TXOP=N/2, CW=15

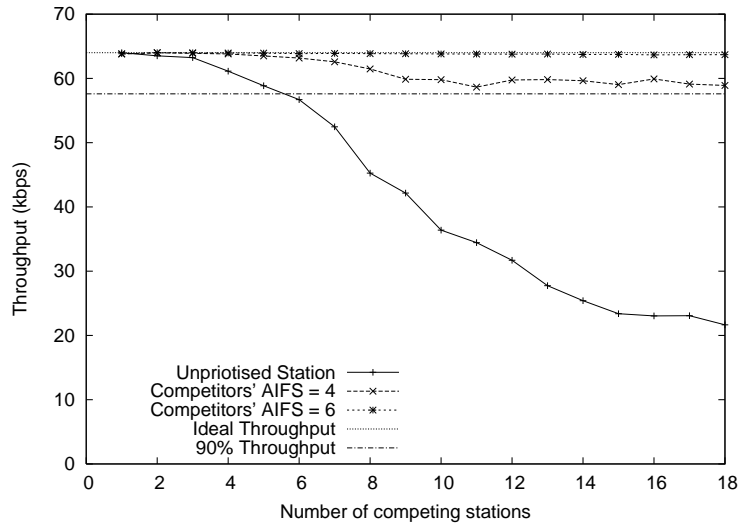


Figure 4.6: Throughput of a single VoIP station versus N saturated stations

the AP, and using *TXOP* does not appear to harm VoIP performance. In figures 4.4 and 4.5 the throughput is maintained all the way to 16 VoIP stations while in figure 4.1 we can see that the delay performance is only acceptable to at most 12 VoIP stations.

4.3 Single VoIP vs Data behaviour

We will briefly look at the behaviour of a single VoIP station when in competition with a variable number of saturated stations. This work is presented in [9].

Since we are dealing with a single VoIP station, a straight forward prioritisation scheme was used: the saturated stations had their *AIFS* increased. Since increasing *AIFS* increases the delay that station experiences every time a packet is transmitted the effect should be load dependent and sufficient to protect the VoIP traffic. We looked at the basic behaviour of *AIFS* in Chapter 3.

Figure 4.6 shows the throughput the VoIP station obtains when unprioritised and when prioritised via *AIFS* increases on the saturated stations. The result is quite dramatic. The throughput performance becomes unacceptable at around 6 competing stations. Using an *AIFS* of 6 results in the throughput remaining at over 90% while increasing this to 8 effectively maintains the throughput across the entire

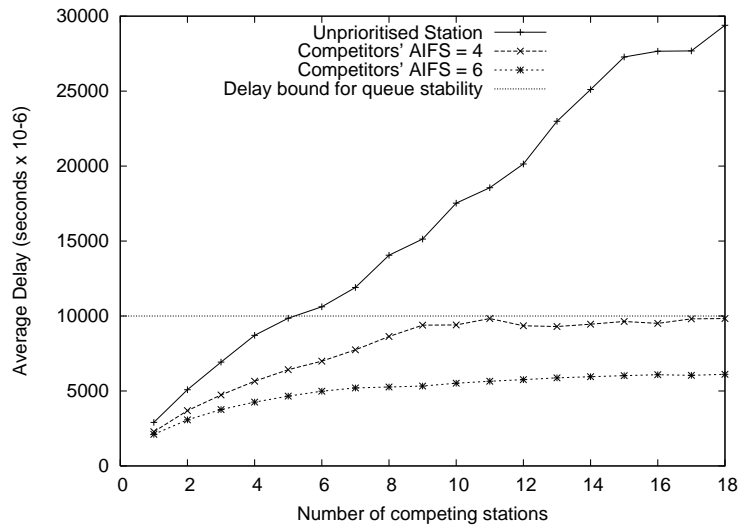


Figure 4.7: Delay experienced by a single VoIP station versus N saturated stations

range.

Figure 4.7 shows the mean MAC delay experienced by the VoIP packets. We can see that in both of the prioritised cases the mean delay is kept below the desired inter-packet time of $10ms$, only just in the case where an *AIFS* of 6 is used. Looking at the cumulative distributions of the delays also shows a significant difference. In figure 4.8, which shows the cumulative distribution of MAC delays experienced by an unprioritised station, we can see that a large proportion of the packets, up to about 60%, have delays greater than the desired inter-packet time. In contrast the distribution for a station with *AIFS* 6, shown in figure 4.9 the percentage is cut to roughly 20%. Increasing *AIFS* further, as shown in figure 4.10, this is cut again to 10%.

These percentages may seem quite high however what this means is that only 10%, or 20%, of the packets arriving have to wait for the previous packet to be sent. In the unprioritised case this percentage increases to over 50% as the number of stations increases. Note that here we are talking about the MAC level delay not the total delay experienced by the packets being transmitted. In the next section we will look at the MAC and queueing delays in more detail.

One other concern is that the delays at the MAC layer might be correlated in some way. The design of the 802.11 MAC indicates strongly that this should not be the case, it is modeled successfully by a memory-less Markov Chain. Figure 4.11, showing the autocorrelation of the sequence of packet times,

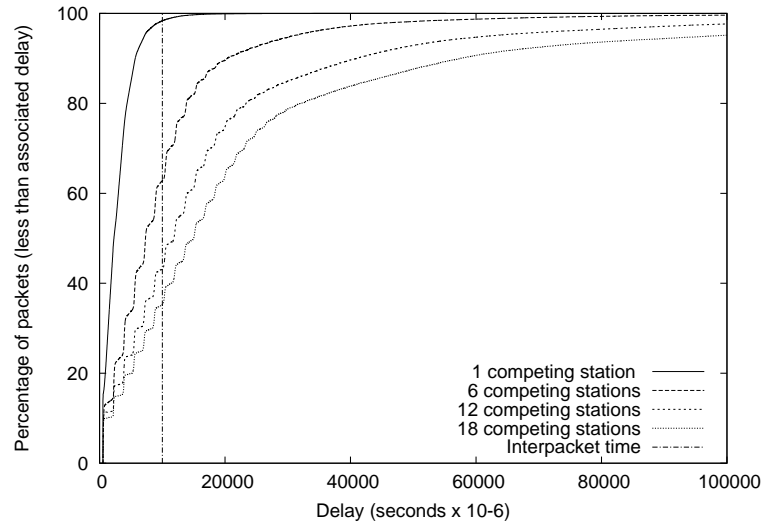


Figure 4.8: CDF for an unprioritised VoIP station

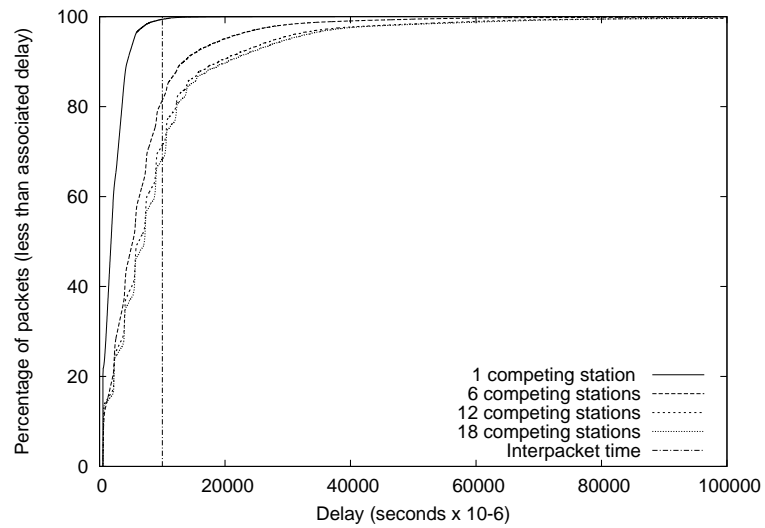


Figure 4.9: CDF for a prioritised VoIP station, AIFS=6

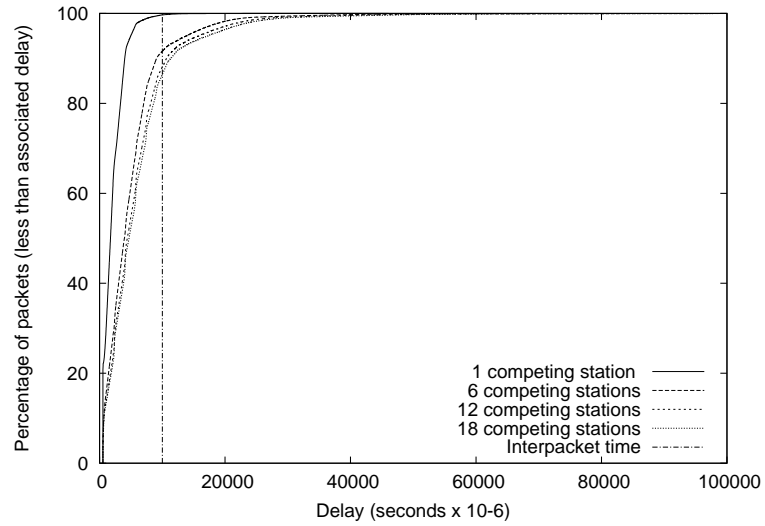


Figure 4.10: CDF for a prioritised VoIP station, AIFS=8

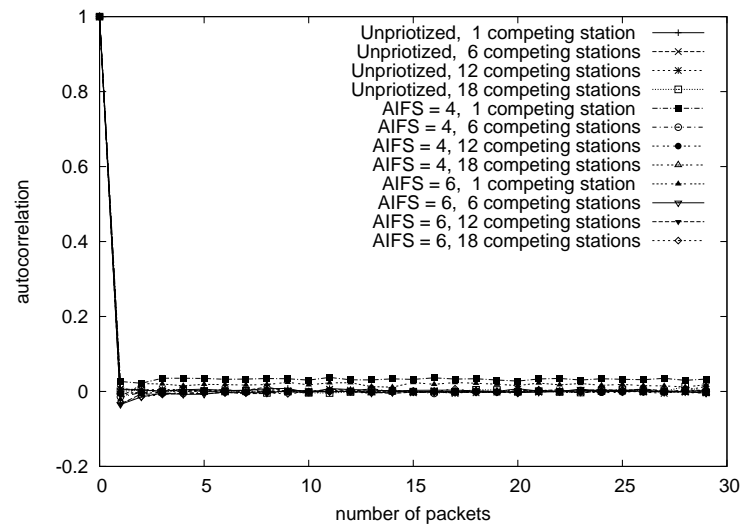


Figure 4.11: Autocorrelation of packets over a number of schemes

shows that there is practically no correlation between packets, as was expected given the design of the 802.11 MAC.

4.4 VoIP versus saturated traffic

From the VoIP experiments we know that we can fit about 12 conversations over the wireless network when some simple prioritisation is given to the AP. Specifically to give the AP about half the throughput in the cases that we consider. In this section we will examine the effect of adding saturated nodes to the network and see how VoIP traffic behaves. Here aggressive traffic in this case is UDP and with a large packet size, 1470 bytes. The purpose of this is to show how well VoIP traffic can be protected in the most adverse situations.

We will apply a number of prioritisation schemes informed by simulation and then compare the experimental results to see which perform the best in terms of VoIP traffic. We will compare the schemes first in terms of throughput then both the MAC layer and the total queuing delay and finally the queue occupancy in order to identify the critical quality of service issues that face VoIP traffic. The parameters chosen come from the analytical models, introduced in chapter 2, as likely combinations which will protect the VoIP traffic from the saturated traffic. Note that here the saturated traffic should be completely starved if the schemes are working correctly since the only concern is to protect the VoIP traffic.

A number of prioritisation schemes were used and it is worth mentioning them here before we discuss the experimental results. These schemes all use the 802.11e parameters mentioned in previous chapters and the combinations are listed below.

AP	VoIP STA	DATA STA
CWmin = 2^3	CWmin = 2^3	-
TXOP = N(VoIP)	-	-
-	-	AIFS = 8
CWmin = 2^3	CWmin = 2^3	AIFS = 8
CWmin = 2^3 and TXOP = N(VoIP)	CWmin = 2^3	AIFS = 8

The above table shows the 802.11e parameters that correspond to the schemes depicted in the graphs.

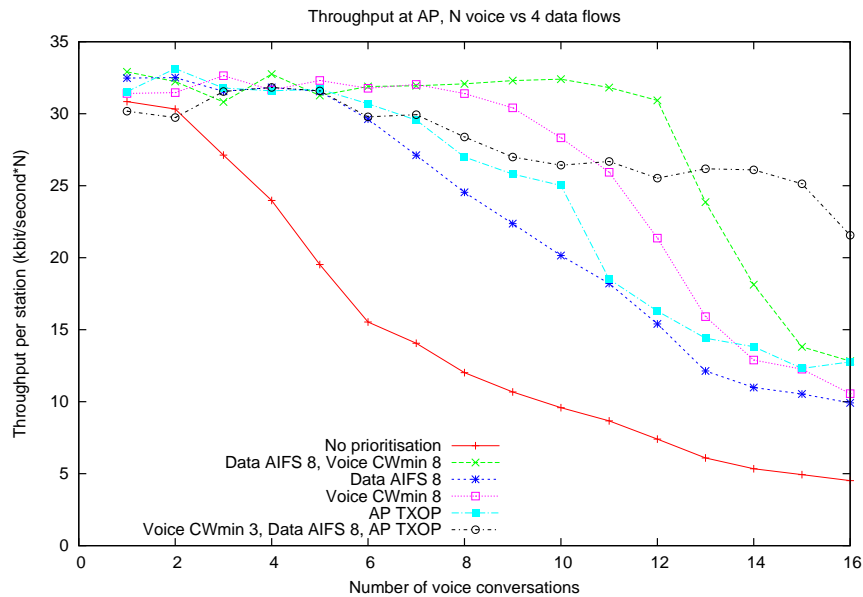


Figure 4.12: AP throughput per voice sta versus N voice conversations and 4 saturated stations

The changes to the default 802.11b parameters are shown. For each point in the graphs the experiment was run for at least 20 minutes. In some cases points were run for 30 minutes where the STA or AP was performing poorly, in order to give a better chance of seeing the long-run behaviour.

4.4.1 VoIP throughput

Throughput is the most obvious measure of performance in the network. VoIP has a clear throughput requirement of $32Kbps$ and if this is not being met then the performance will definitely suffer. We should mention that due to the bursty nature of the traffic the throughput requirement is actually $64Kbps$ while the station is transmitting and $0Kbps$ otherwise. All the experiments were run for at least 600 seconds in order to obtain average results.

Figures 4.12 and 4.13 show the throughput at the AP and for a single station for various priority schemes and for 4 and 10 saturated stations. Figure 4.12 shows the throughput at the AP and we can see that in the unprioritised case only 2 voice conversations are required before the throughput drops significantly. There is some variation in the performance of the various schemes however. All the schemes performed better than the unprioritised case, with the scheme penalising the data traffic using AIFS maintains throughput out to 12 stations, more than any other scheme. The scheme which

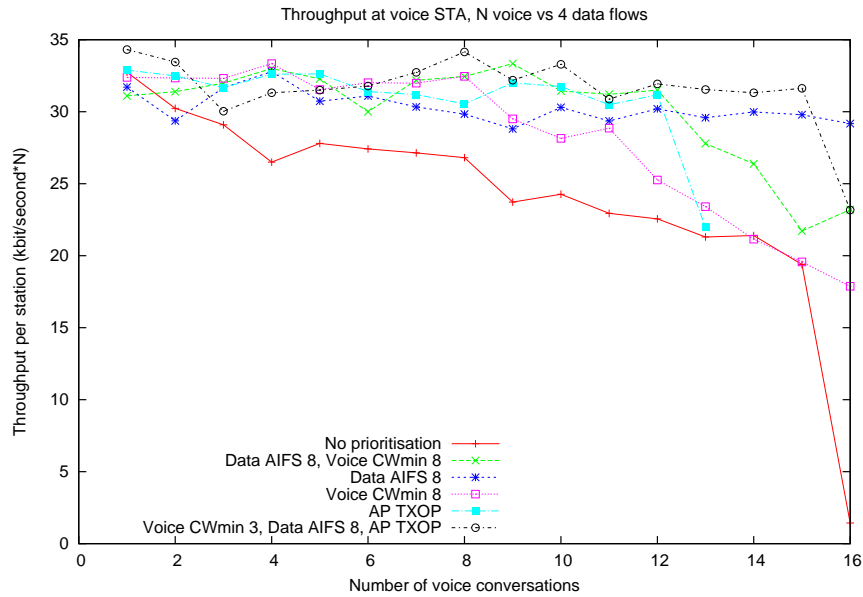


Figure 4.13: Voice throughput versus N voice conversations and 4 saturated stations

combines $AIFS$, $TXOP$ and CW_{min} is also interesting since although the throughput drops below what is required, it maintains the highest throughput after about 13 stations.

In figure 4.13 we can see that in the unprioritised case the throughput drops significantly once about 4 other voice stations are started. In the cases where we have applied some prioritisation the throughput remains close to $32Kbps$ out to about 11 or 12 which is what we would expect for a network with only VoIP. An exception to this is the case where we prioritise the AP and the voice stations by giving them a CW_{min} of 2^3 . This may be because as the number of voice stations rises above 8 the number of collisions experienced increases rapidly. We will examine the collision probability statistics later in the chapter.

Figures 4.14 and 4.15 show similar experiments except that now 10 data stations are used. The most striking aspect of these graphs is that the throughput is clearly insufficient in the unprioritised case for either the AP or a VoIP station right from the start. Some form of prioritisation is required for VoIP to operate at all.

In figure 4.14 we can see that the prioritisation schemes are quite different in terms of their effect on the throughput. Firstly, just using $TXOP = N$ is not sufficient at all. This is to be expected since there are now a significant number of saturated stations that the AP must compete with. The remaining

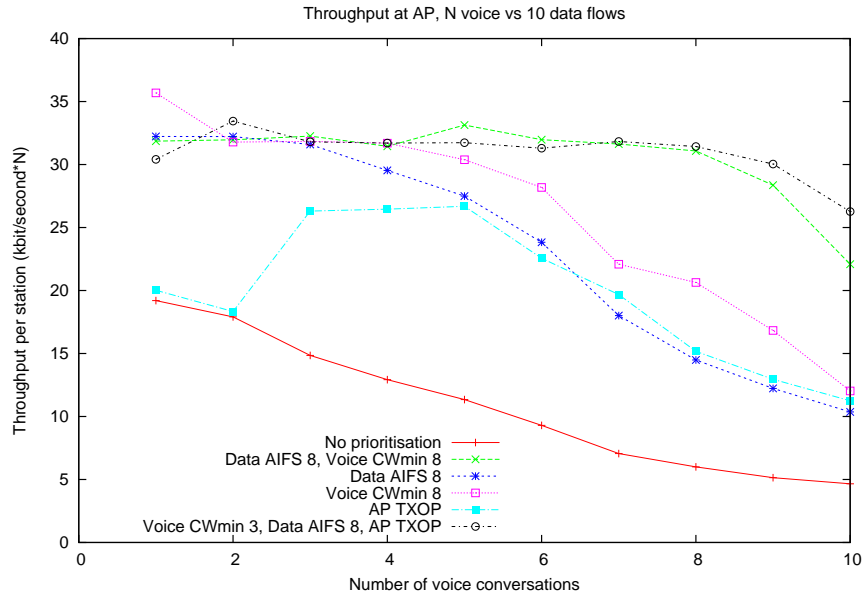


Figure 4.14: AP throughput per voice sta versus N voice conversations and 10 saturated stations

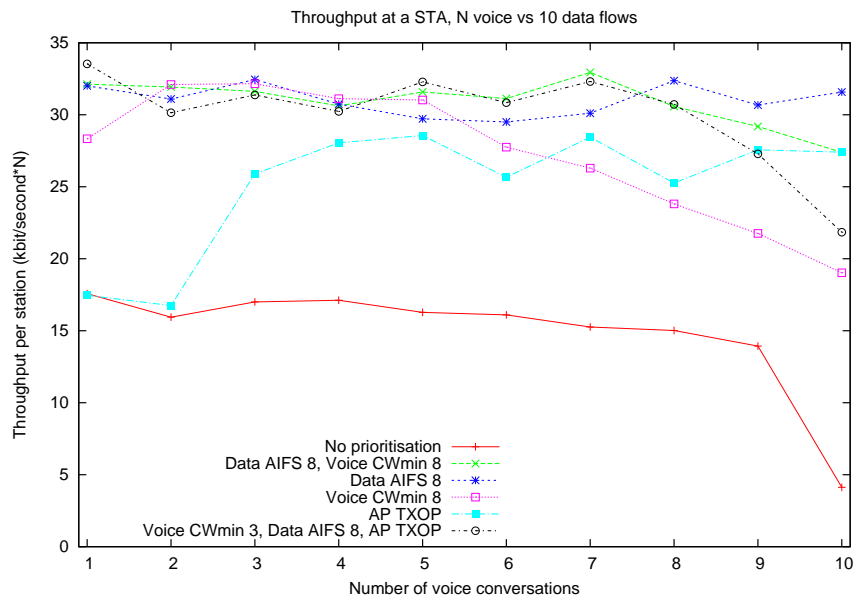


Figure 4.15: Voice throughput versus N voice conversations and 10 saturated stations

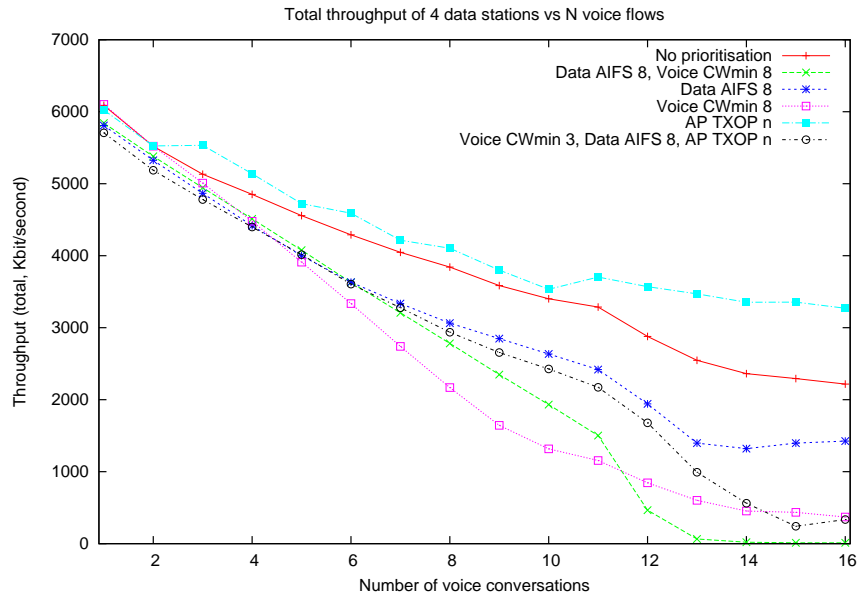


Figure 4.16: Throughput for 4 saturated stations

schemes separate out into two groups. Using just *AIFS* or CW_{min} gives the required throughput out to about 3-4 VoIP stations, whereas using a combination of these maintains throughput out as far as 8 or 9 stations. This is more than twice the number of VoIP stations so is very significant improvement.

The situation in figure 4.15 is not quite the same. Here the best performing scheme is the scheme where only *AIFS* is used, however again the schemes using *AIFS* and CW_{min} in combination perform well out to about 8 VoIP stations. Using CW_{min} only is not sufficient past about 5 stations. Again we would expect a significant increase in collisions in this case since due to the low value of CW_{min} relative to the number of stations.

We can also look at the total throughput of the badly behaved data stations, to gauge the impact of the prioritisation schemes. The throughputs are shown in figures 4.16 and 4.17, for 4 and 10 of these stations respectively. Interestingly in both cases the schemes which employ a combination of *AIFS* and CW_{min} result in the lowest data throughput when the number of VoIP stations is large. Indeed these schemes succeed in taking almost all the bandwidth away from the saturated stations. In order to allow 12 VoIP calls, this approach effectively denies the saturated stations any throughput at all.

Using CW_{min} alone reduces the throughput most quickly initially but then levels out. However we can see from before that the VoIP throughput performance using this scheme rapidly becomes

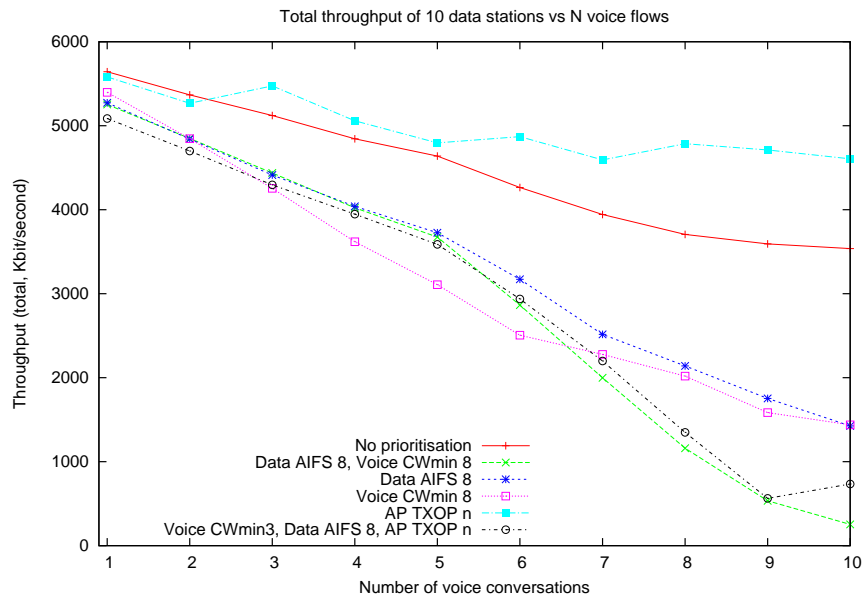


Figure 4.17: Throughput for 10 saturated stations

unacceptable. This scheme gives the AP and the VoIP stations a strong prioritisation but quickly results in increased collisions wasting a lot of bandwidth. Using *TXOP* on the AP only actually increases the total throughput for the data stations however this does not perform well enough for the VoIP stations, never achieving enough throughput in the 10 data station case. This is the reverse of the issue with CW_{min} , the AP with half the traffic is contending for access less often and so making more efficient use of the available bandwidth, yet still does not perform as well as the combination schemes. Using *AIFS* on its own in both cases maintains the highest throughput for the saturated stations, yet again we can see that the performance for the VoIP stations is inferior to the other schemes tested.

In summary, the throughput shows us the relative performance of the schemes but only so far as to show when a particular scheme is definitely not performing sufficiently well. Due to the bursty nature of VoIP traffic and buffering at the stations the performance may yet be unacceptable due to long delays in transmitting the packet. This is a clear worry where *TXOP* is being used and so we must look at other statistics to determine whether or not a scheme is actually performing well.

4.4.2 Delays for VoIP

In the previous section we looked at the throughput characteristics, here we will look at the delays for the same scenarios. As discussed in chapter 3 there are two separate delays that we can examine. First is the MAC level delay which is the delay resulting from contention and retransmissions at the 802.11 MAC layer. The second is the total delay which includes both the MAC level delay and the delay resulting from queuing. We will first look at the MAC level delay. Although the total delay may seem to be the important statistic, looking at the MAC delay has the advantage of showing how the 802.11e parameters effect the traffic.

Figures 4.18 and 4.19 show the MAC delays at the AP and VoIP stations respectively when 4 competing data stations are present, as in the previous section. Again we can see a distinct difference between the prioritised and unprioritised cases. Interestingly for both the AP and a VoIP station we can see that the best performing scheme is now the combination scheme where all three parameters are being used. This is keeping the average delay at the MAC layer low for the whole range of the experiment. Here it should be noted again that because the AP has half the traffic, its inter-packet time is linearly dependent on the number of VoIP stations on the network. For example, with 10 VoIP stations, it would need an average inter-packet time of at most $1000\mu s$, a target which it just reaches with two of the schemes. In figure 4.19, at a VoIP station, the same scheme yields unacceptable performance only with 16 stations and using *AIFS* only acceptable performance across the entire range. However performance is not acceptable at the AP in these cases.

In figures 4.20 and 4.21 we can see the same results for 10 data stations. Again there is a clear difference between the prioritised and unprioritised cases, with the two combination schemes performing the best. In figures 4.19 and 4.20 we can see irregularities developing in the unprioritised cases, as seen from the spikes in the data. This is because here the performance is so poor that the 30 minutes for an experiment is actually becoming too short. Longer experiments were not carried out due to the fact that the performance is quite clearly unacceptable and potentially quite a large amount of time would be needed to obtain significantly better results.

We can see from these results that the AP is indeed the limiting factor. Most of the schemes provide acceptable inter-packet delays from the point of view of the VoIP stations, it's on the AP where the inter-packet times become an issue.

The next thing to look at are the total delays, including the queuing delays. Figures 4.22 and 4.23

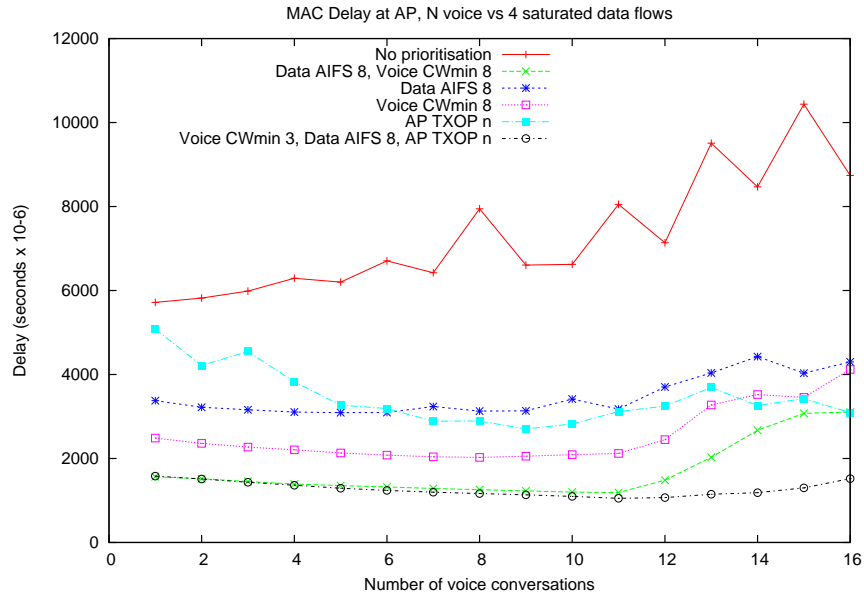


Figure 4.18: AP MAC delay versus N voice conversations and 4 saturated stations

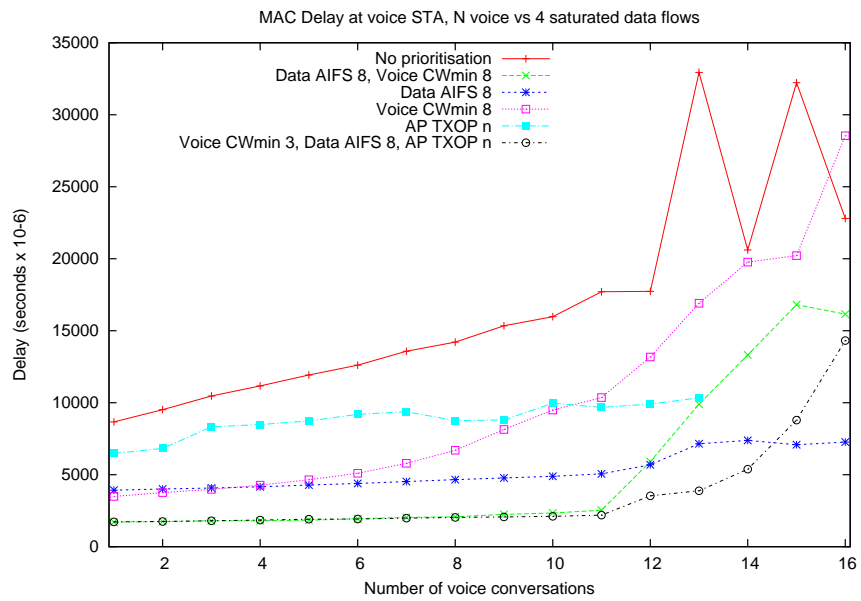


Figure 4.19: STA MAC delay versus N voice conversations and 4 saturated stations

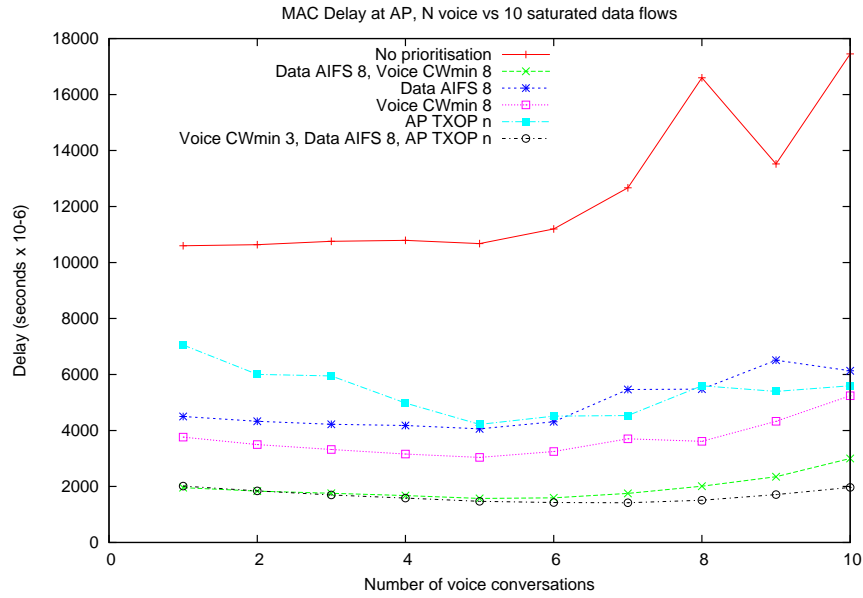


Figure 4.20: AP MAC delay versus N voice conversations and 10 saturated stations

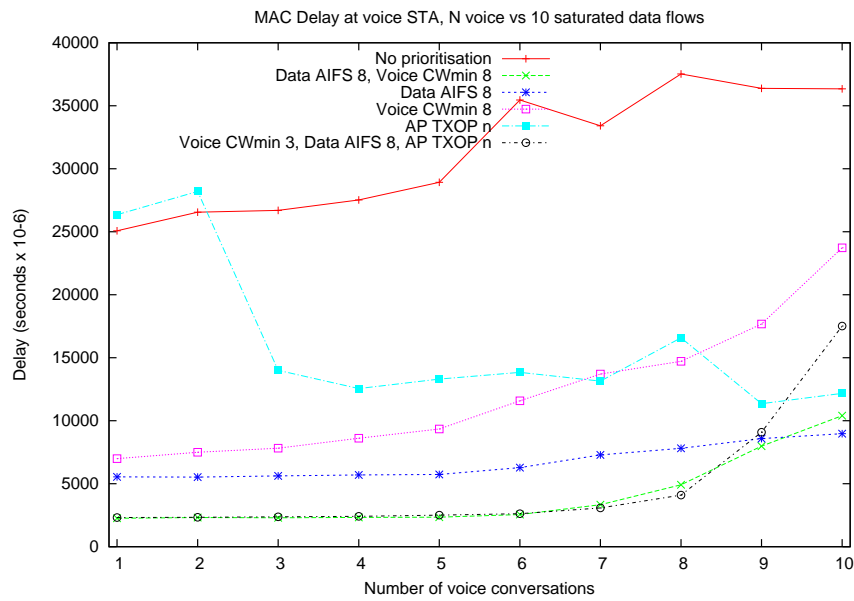


Figure 4.21: STA MAC delay versus N voice conversations and 10 saturated stations

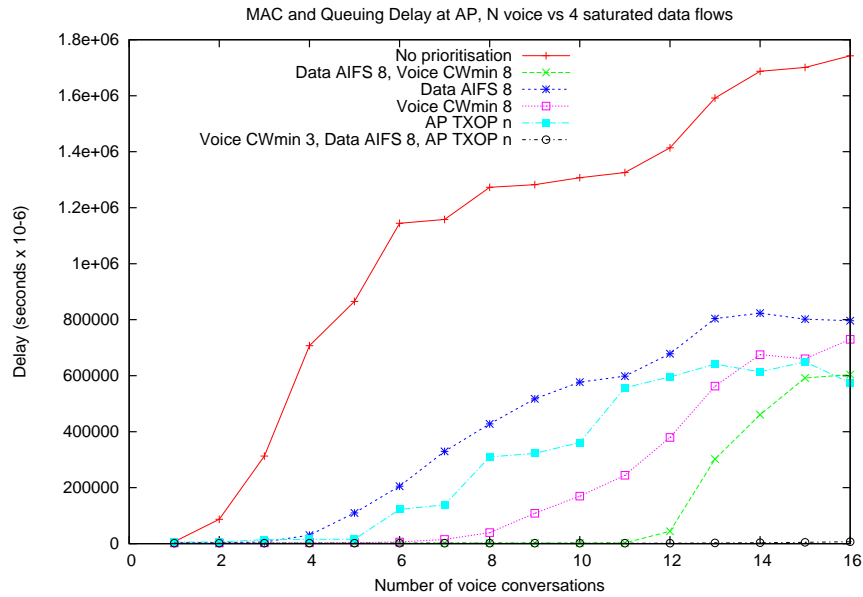


Figure 4.22: AP queuing delay versus N voice conversations and 4 saturated stations

show the total delays experienced by the AP and a VoIP station when 4 data stations are present. Whereas the MAC level delays were relatively close, the total delay values are much more dramatic. In both figures we can clearly see that the total delays increase rapidly, roughly at the points where the throughput and MAC delays would suggest the performance becomes unacceptable

Again in figures 4.24 and 4.25, now showing the situation with 10 data stations, a dramatic increase in the delay can be seen. Interestingly in figure 4.25 the delay actually decreases very significantly for the case where only *TXOP* is being used. This is perhaps not surprising since the APs *TXOP* value is related to the number of VoIP stations and so is able to make more efficient use of the medium as this number increases. However for *TXOP* alone the delay is *never* acceptable across the whole range of the experiments.

In the previous figures the scale was such that it is difficult to see at what point the delay might become unacceptable. By limiting the range to $50000\mu s$ we can see more easily the dramatic increase that occurs. The ITU acceptable delay is listed as $150ms$, however $50ms$ is often all that is allocated for network delay: the remainder being allocated to encoding, packetisation decoding and so on. This is therefore a reasonable timescale to look at. However it makes little difference in most cases since the delays increase well beyond even these values.

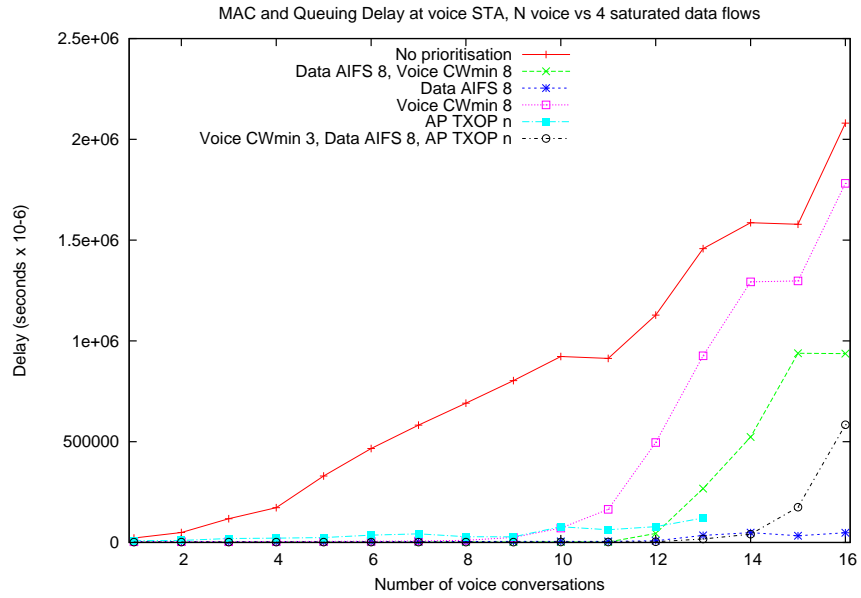


Figure 4.23: STA queuing delay versus N voice conversations and 4 saturated stations

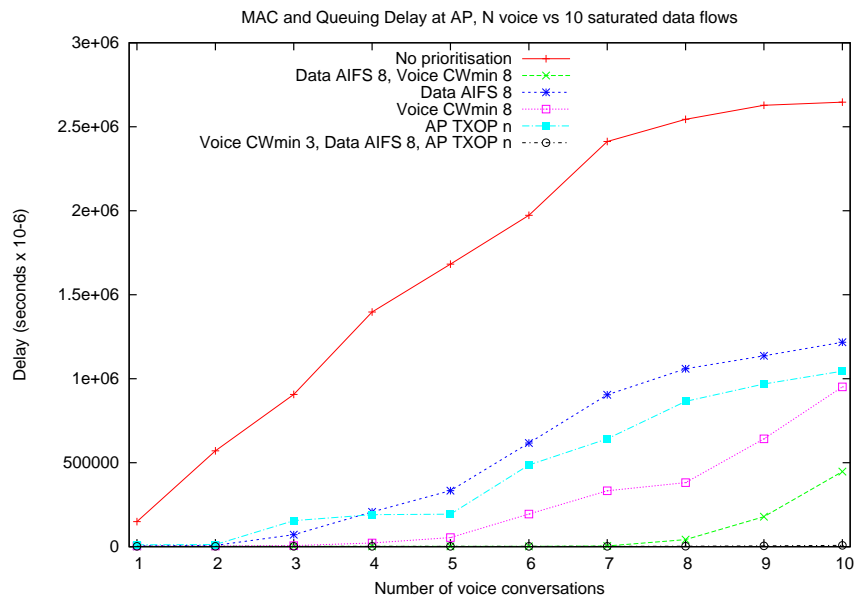


Figure 4.24: AP queuing delay versus N voice conversations and 10 saturated stations

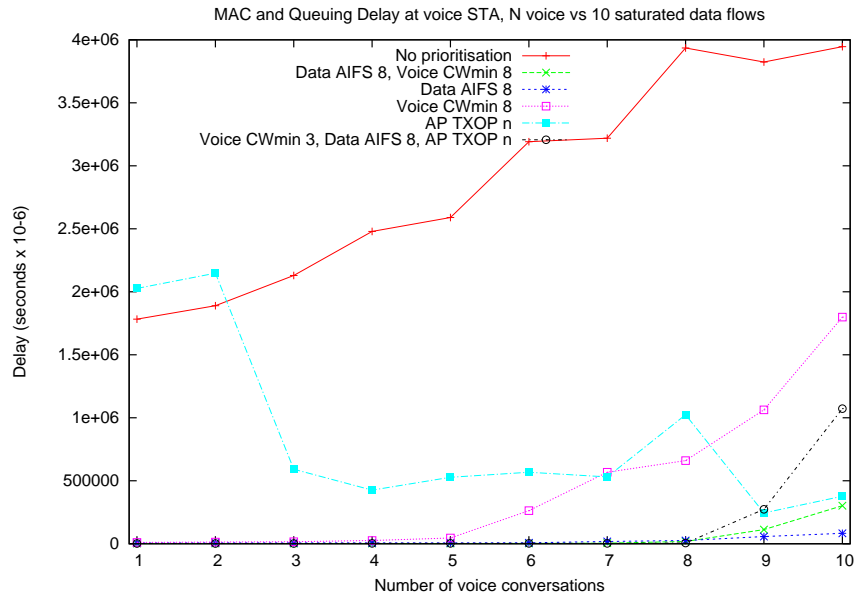


Figure 4.25: STA queuing delay versus N voice conversations and 10 saturated stations

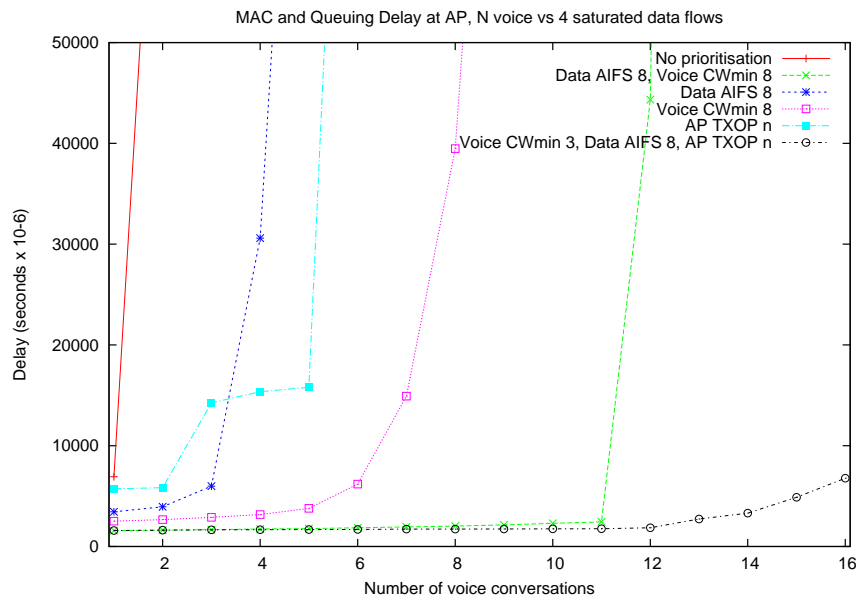


Figure 4.26: Close up of AP queuing delay (N voice conversations and 4 saturated stations)

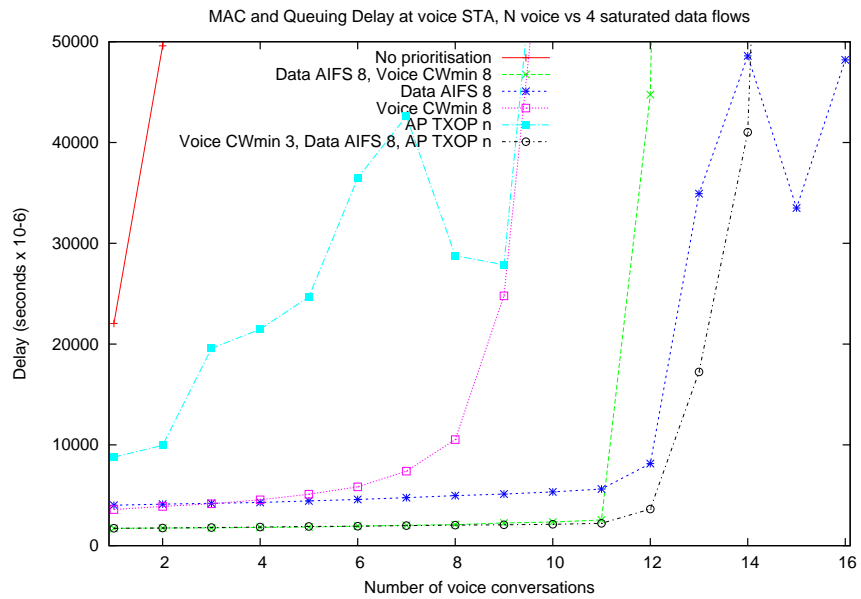


Figure 4.27: Close up of STA queuing delay (N voice conversations and 4 saturated stations)

Figures 4.26 and 4.27 again show the 4 data station case. In these graphs we can clearly pick the points where the performance has become unacceptable. In figure 4.26 we can see quite a dramatic difference between the two schemes which combine various parameters, and interestingly it is the scheme which includes *TXOP*, as well as *AIFS* and CW_{min} which performs best, not showing a rapid increase in delay across the whole range.

Figures 4.28 and 4.29 show the situation when 10 saturated stations are present, and again the best performance is reached by using a combination of the parameters. While the AP's delay does not dramatically increase when all three parameters are being used, for the VoIP station none of the schemes are able to keep the delay down past 8 stations. Also, *AIFS* and CW_{min} have quite a strong effect with the addition of *TXOP* only having an impact at the AP when the performance at the station is already poor.

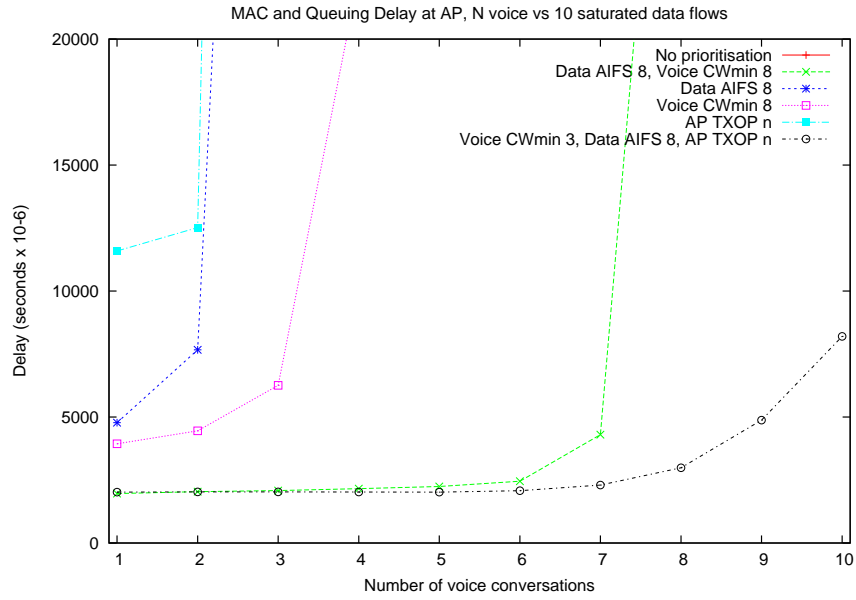


Figure 4.28: Close up of AP queuing delay (N voice conversations and 10 saturated stations)

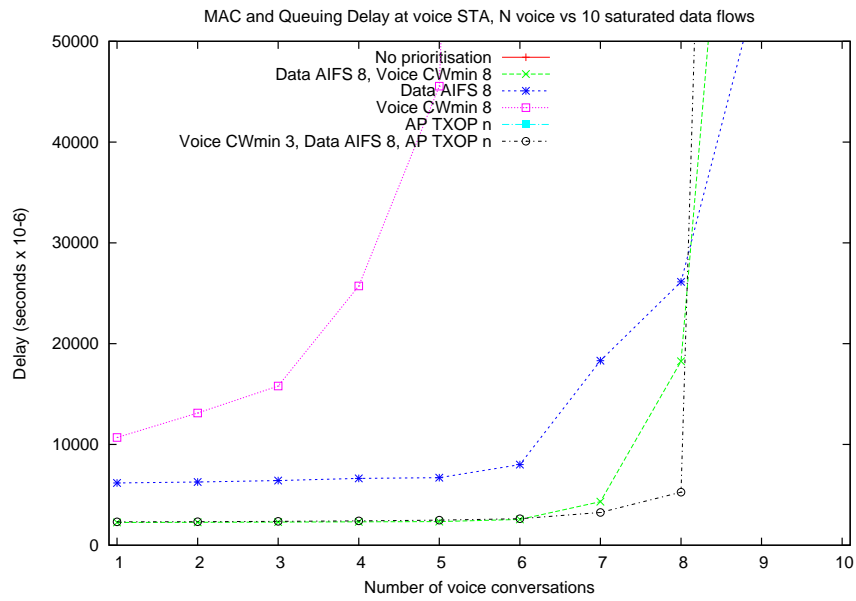


Figure 4.29: Close up of STA queuing delay (N voice conversations and 10 saturated stations)

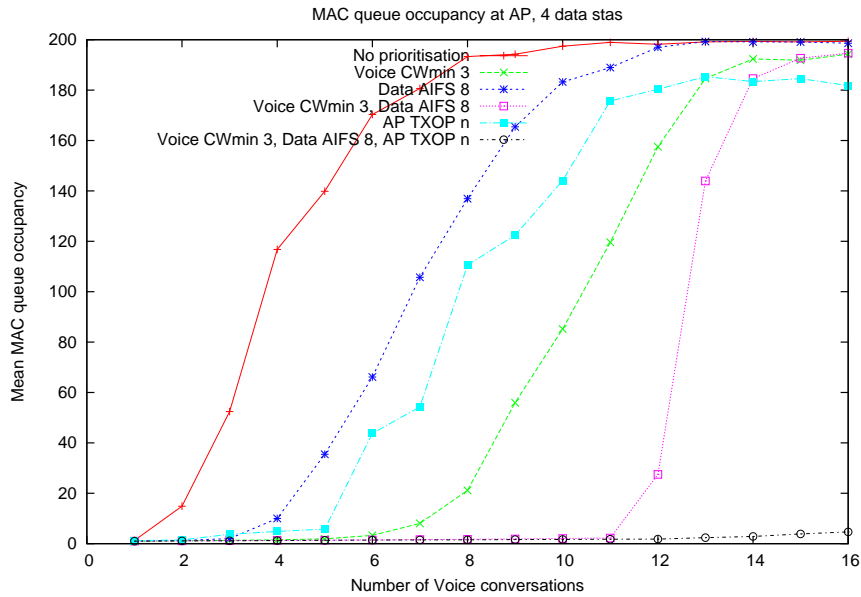


Figure 4.30: Average MAC queue occupancy at AP versus 4 saturated stations

4.4.3 Queue occupancy

The MAC delay and the queuing delay represent the effective total delay associated with transmitting the packet. This is regarded as the most important statistic for VoIP traffic, however the total delay is determined by queue occupancy and the service rate. In our case the service rate is directly related to the MAC delay. The next important statistic to consider is therefore the queue occupancy. For these experiments, and the previous ones, the queue length was left at the driver default value of 200. This is reasonably large so it enables us to see how the queue fills.

Figures 4.30 and 4.31 show the average queue size for the AP and a VoIP station respectively, when 4 saturated stations are present. In the case of 4.30 the queue fills for all of the prioritisation schemes with the exception of when all three parameters are used. The AP is able to keep its queue close to empty across the entire range examined. When *TXOP* alone is being used the queue occupancy seems to level out below 200. This is a product of when the queue occupancy is being recorded, after each successful transmission. It is not the case that the queue is being kept just under full by this scheme. In figure 4.31 we can see that the queue occupancy starts to increase for the combination scheme at about 14 stations. Interestingly, for the scheme where *TXOP* alone is used, the queue at the station remains small. Only the AP has this *TXOP* advantage and its performance using this

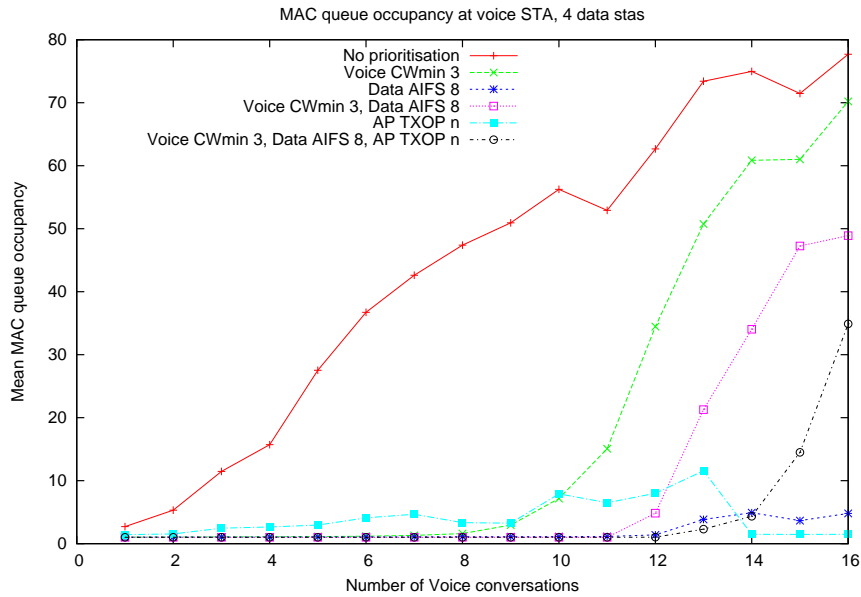


Figure 4.31: Average MAC queue occupancy at STA versus 4 saturated stations

scheme is not nearly as good. A possible explanation for this is that the AP is contending for access less often than it would using other schemes and this helps reduce collisions, benefiting the rest of the stations on the network.

We can see a similar situation when we look at figures 4.32 and 4.33 where 10 saturated stations are competing with the VoIP stations. Again the best performing scheme is the combination of all three parameters. In 4.33 we see an initial reduction in the queue occupancy for a station, in the case where the AP is prioritised using *TXOP*. Again it seems that the fact that the AP is contending less often for access is having a positive effect on the other stations. We know however that the performance is never acceptable in this situation.

The single most important statistic that applies to VoIP traffic is the delay. However, we were able to measure the delay in two separate ways the MAC delay and the total queuing delay. The total delay may be seen as the critical statistic for VoIP, since it is the time it takes the driver to actually send a packet and thus directly relates to delay experienced by the voice codec sending and receiving voice packets. When this delay is kept low we can be confident that the quality of the call is acceptable and when the delay starts to increase it increases dramatically giving a clear indication that the quality will no longer be acceptable. The MAC delay does not behave in this manner, it does not exhibit

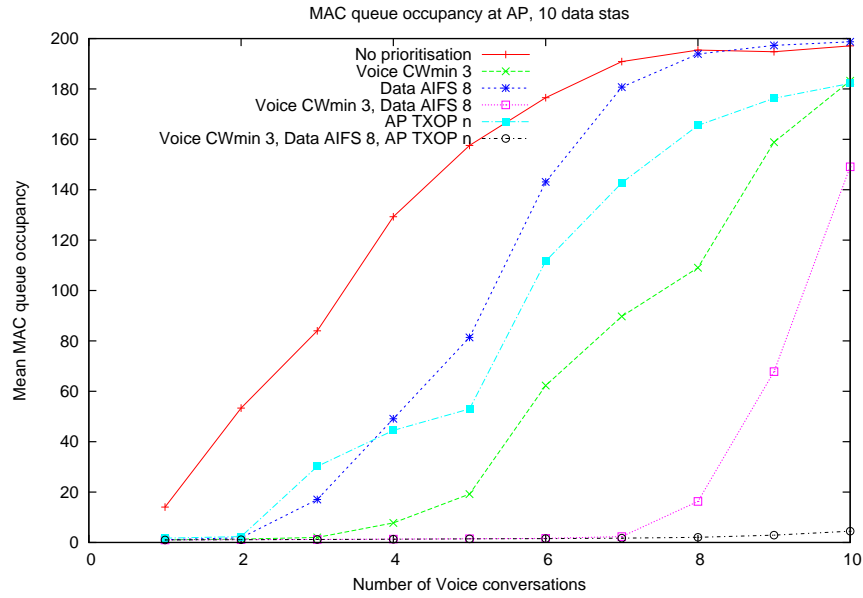


Figure 4.32: Average MAC queue occupancy at AP versus 10 saturated stations

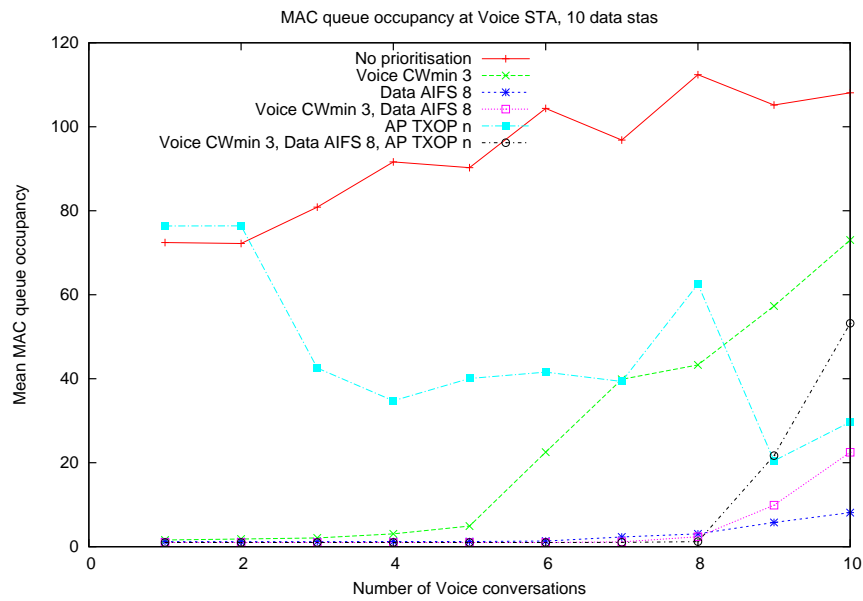


Figure 4.33: Average MAC queue occupancy at STA versus 10 saturated stations

the dramatic increases that the total delay does. This is because the MAC delay corresponds to the service rate at which the MAC level queue is being served. In the case of the AP, as the load increases so does the total delay, however the MAC delay experienced by the AP does not increase greatly. The reason for this is the MAC delay experienced by any particular station depends on the chance that *other* stations are attempting transmission. In contrast, the queueing delay will depend strongly on the rate at which the station itself is attempting to transmit packets.

This can be seen by comparing 4.20 and 4.24, which show the MAC delay and the total delay at the AP. In the first half of the graphs the MAC delay is effectively constant, but the queueing delay increases linearly. This is due to the fact that the load at the AP is increasing, yet adding more VoIP stations to the network which contains 10 saturated stations, has a fairly small impact on the the chance that other stations are attempting transmission. In the second half of the queueing graph we can see that half of the delays level out, two others look set to increase more rapidly and the remaining set doesn't increase greatly at all. To understand this we need to look at the queue occupancy statistics. Comparing 4.24 and 4.32, showing the queueing delay and the average queue occupancy, we can see that in three cases the queue becomes close to full at the same point the delay levels off. Only in the case where all three prioritisations are used does the queue remain close to empty.

Since we have a queue with a service time we can use Little's Law, [17], to relate these statistics. According to this, the total delay experienced by a packet in the queue will simply be the product of the average number of packets in the queue and the service time of that queue. We have these in the form of the MAC delay and the queue occupancy. This has a very important consequence, although the total delay is what we need to control, we can do it by controlling the MAC delay and the size of the queue. In practice, considering the relatively poor improvement that adjusting the buffer sizes had, optimising the MAC delay seems to be the correct approach. This also indicates that buffering will not work as an approach for QoS, unless it is combined with techniques to improve the MAC delay.

4.4.4 Collision probabilities

In the previous sections we saw several quite surprising results in terms of total delay and queue occupancy, in particular where *TXOP* alone was being used. In order to understand why some of

these schemes work well, and why some produce better results than others, it is important to look at the collision probability.

In figures 4.34 and 4.35 show the collision probability experience at the AP and a VoIP station respectively, when competing with 4 saturated stations. At first glance the two figures look very similar, however the scales are very different. The collision probability that each experiences is very close when there is only one VoIP station, however the collision probability increases more rapidly for the station than it does for the AP.

Using *TXOP* alone has the effect of reducing the collision probability for everybody, not just the AP. It should be noted that the value of *TXOP* is increased with the number of VoIP stations on the network and the AP also has more traffic to send.

Combining CW_{min} and *AIFS* to prioritise the traffic results in low collision probability until about 11 stations. After this the collision probability goes up rapidly. *AIFS* allows the voice stations to largely contend with one another, rather than with the data stations. However, as the number of VoIP stations increases a low CW_{min} means VoIP stations are increasingly likely to collide with each other and the value of CW_{min} is no longer sufficiently large to keep the inter VoIP collision rate small. This also indicates why the data stations throughput does not go to zero. Since the VoIP stations are colliding more regularly, they have increased CW values and the data stations can contend with them.

Combining *TXOP* CW_{min} and *AIFS* results in a consistently low collision probability across the range. It does start to increase rapidly at about 12 stations in 4.35, again indicating that the value of CW_{min} is now too low.

We have a broadly similar situation in figures 4.36 and 4.37 which show the collision probabilities at the AP and a VoIP station competing with 10 saturated stations. The collision probabilities do not rise as rapidly, only to the level of the unprioritised case. This indicates that the high priority VoIP stations are driving up the collision probability in the previous case. Again we see that *TXOP* maintains a low collision probability, due to reducing the number of times the AP must contend for access.

The best performing scheme, combining CW_{min} *AIFS* and *TXOP* at the AP, again significantly reduces the collision probability over the unprioritised case.

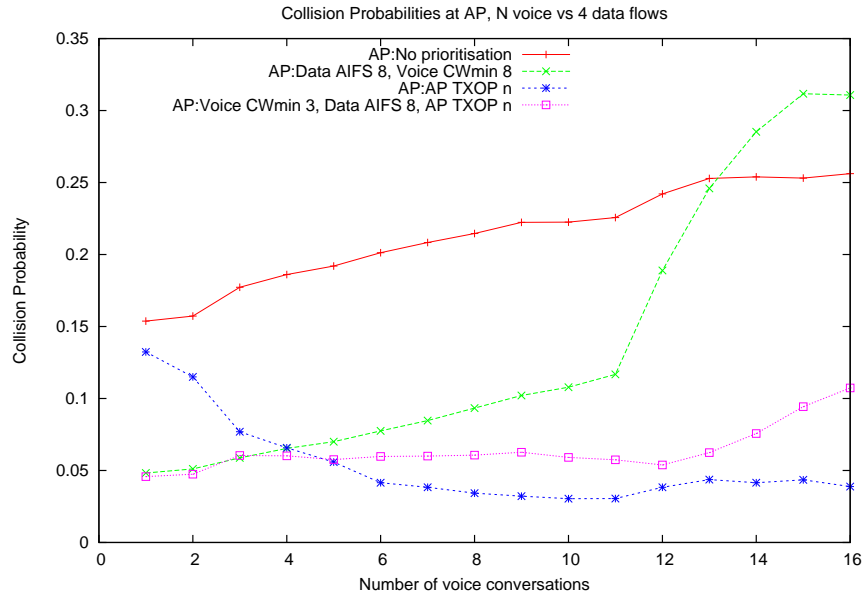


Figure 4.34: Collision Probability at AP versus 4 saturated stations

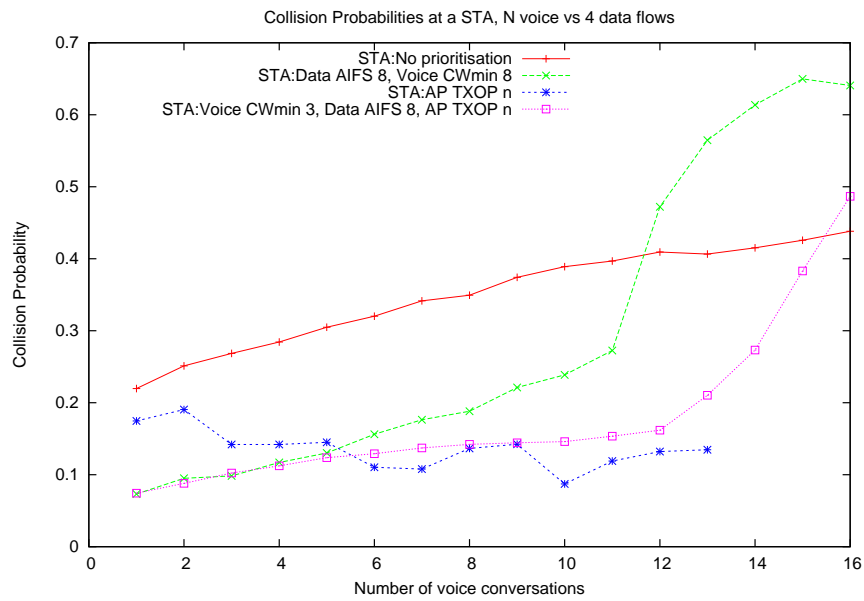


Figure 4.35: Collision Probability at VoIP STA versus 4 saturated stations

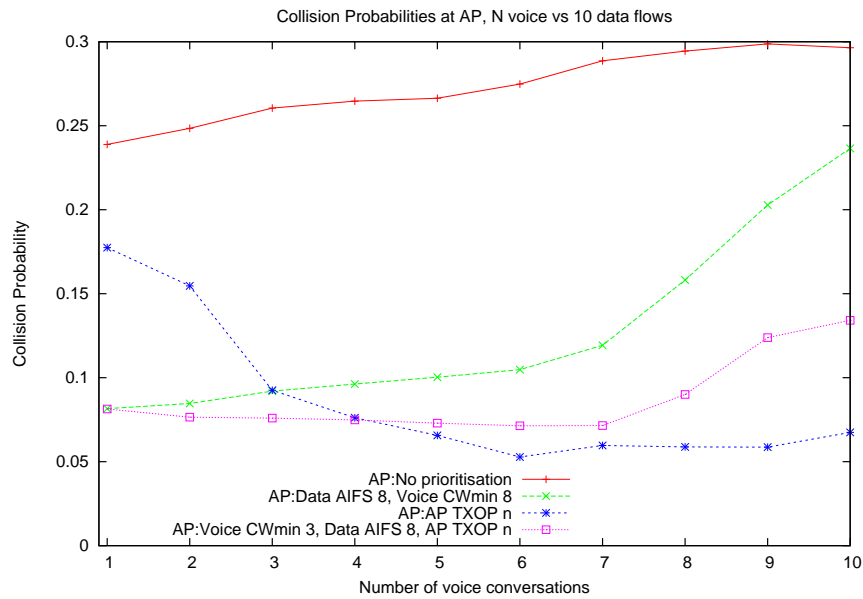


Figure 4.36: Collision Probability at AP versus 10 saturated stations

It is interesting to note that the collision probabilities measured at the AP and at a VoIP station are not the same. The graphs have the same trends but the scales are markedly different. This is due to the fact that the collision probability depends on the τ value, the probability that a transmission is attempted, of all the *other* stations on the network. Since the AP has more traffic to send than a station its τ value will be different so causing a difference in the collision probability experienced by the AP and that of a VoIP station.

The most striking feature of the collision probability graphs is the large reduction in collision probability for most of the schemes when compared to the standard DCF parameters. This shows that for this sort of traffic the medium is not being used efficiently. A notable exception is the case where CW_{min} is set to 7 for the VoIP stations. In this case the collision probability dramatically increases after 11 VoIP stations are introduced. This is simply because the value of CW_{min} is now too low, and collisions becoming extremely common. Raising CW_{min} to 15 could be expected to reduce this significantly.

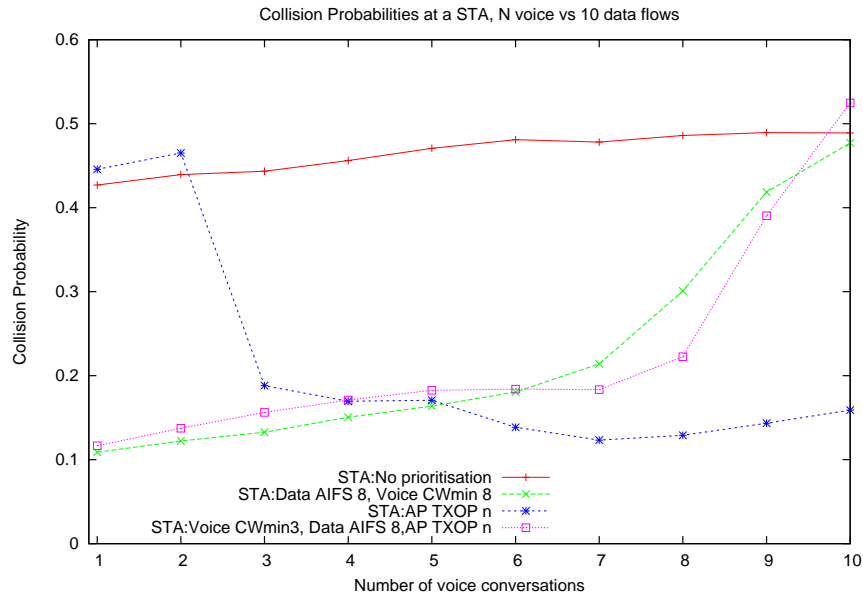


Figure 4.37: Collision Probability at VoIP STA versus 10 saturated stations

4.5 Conclusions

In this chapter we have seen how VoIP traffic behaves in a number of different situations. With only VoIP traffic on the network we investigated the impact of buffering as it applies to QoS and discovered that although poor performance can be caused by having insufficient buffer space, increasing the buffer beyond a certain point results in effectively no gains. By prioritising the AP, using $TXOP$ and CW_{min} the number of VoIP conversations can be increased from 10 to 12.

The situation where one VoIP station was in competition with a number of saturated stations was then considered. The VoIP station can be quite effectively protected by using an increased $AIFS$ for the saturated stations. Without this prioritisation only about 5 competing stations were possible, but giving the VoIP station an $AIFS$ advantage of 4, the throughput and delays were maintained across the whole range, up to 16 competing stations.

Next we considered cases where that VoIP stations were in competition with saturated traffic. Here we used combinations of three 802.11e parameters, $TXOP$, CW_{min} and $AIFS$, in order to see if the VoIP stations could be effectively protected from the saturated stations through using these parameters. This showed that the VoIP stations could indeed be protected using a combination of all three of the

parameters. More importantly, in this section we were able to break down the results by the different measures we were able to obtain from the experiments. This gives an indication of what measures are important for QoS as regards interactive traffic like VoIP, as used here. The MAC delay and the queue occupancy provide important information, as they capture both the throughput and delay constraints experienced by the station on the network.

The main advantage of 802.11 wireless networks is their distributed nature. This allows easy setup and interoperability, however makes QoS difficult. Both the MAC delay and the queue occupancy are statistics which can be gathered in a distributed manner, in other words, each node on the network can get these independently. From these statistics a node can independently estimate if it is meeting delay criteria for high priority traffic. We have seen that the 802.11e extensions allow high priority traffic to be protected against saturated traffic, and even make better use of the medium when only high priority traffic is being sent. The combination of these facts provides a good basis to provide QoS for traffic over 802.11 wireless networks, without having to resort to a centralised scheme, and therefore maintaining the key advantage that 802.11 networks enjoy.

Chapter 5

Future work

Models of the 802.11 MAC, including the 802.11e enhancements, have been reviewed here, and some of the assumptions made by the models were tested. Notably the assumption that the collision probability is independent of the back-off stage was found not to be the case. It may be worth investigating models with a different collision probability, dependent on back-off stage. This is unlikely to be productive for two reasons however. Firstly, collisions are typically dominated by collisions at the first back-off stage, and so the assumption works well in practice. Secondly, latter back-off stages imply multiple collisions which waste time on the medium. The collision probability results in chapter 4 show that the prioritisation schemes that worked well, significantly reduced the collision probability.

The test-bed provided a platform to measure many of the statistics that determine the performance of traffic on the network. However there are several other measurements which could prove useful. First, the error rate, which was measured relatively easily on a quiet network, may not remain the same on a busy network due to the potentially noisier environment. It is important that this is accurately measured since it would benefit rate control algorithms in particular, to be able to distinguish between losses due to collisions and losses due to interference. This would also have applications for channel selection algorithms, since certain channels may be noisier than others.

There remain some issues which will effect the performance of 802.11 networks which are not dealt

with at all here. First, the impact of broadcast or multicast packets should be considered for other QoS sensitive applications. These pose a particular problem since because they are not explicitly acknowledged at the MAC layer, the MAC delay scheme described here would not work for these types of transmissions. Also, all the experiments were carried out with all the stations using the same transmission rate, 11Mbps . It is easy to imagine that real infrastructure networks will have stations operating at a number of different transmission rates. This will impact the fairness considerations since the 802.11 MAC is per-packet fair by default, yet transmissions at lower rates take significantly longer. For QoS purposes, the time spent on the medium is the vital consideration, so in this situation it may be useful to measure the distribution of MAC delays. This may enable better parameter choice in mixed rate environments.

Finally, as noted before, monitoring the MAC delay and the queue occupancy forms a basis for providing QoS in a distributed manner, when paired with the extended 802.11e MAC. Here, particular combinations of parameters were tested, however this paves the way for the development of adaptable schemes, which might change the parameters automatically in response to changing network conditions. It is important that these type of distributed wireless networks can be adapted for more efficient use of the medium as due to collisions taking significant amounts of time and high per-packet overheads relative to wired networks, increases in the physical rate of the network has been shown to scale poorly [26]. This would represent a significant improvement, since this would help overcome this major shortcoming in 802.11 networks.

Bibliography

- [1] 802.11 standard. <http://standards.ieee.org/getieee802/802.11.html>.
- [2] 802.11e standard. <http://www.ieee802.org/11>.
- [3] Relayfs. <http://relayfs.sourceforge.net>.
- [4] Skype. <http://www.skype.com>.
- [5] R. Battiti and B. Li. Supporting service differentiation with enhancements of the IEEE 802.11 MAC protocol: models and analysis. *University of Trento, Tech. Rep. DIT-03-024*, 2003.
- [6] G. Bianchi. Performance analysis of IEEE 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications*, 18(3), March 2000.
- [7] G. Bianchi, A. Di Stefano, C. Giaconia, L. Scalia, G. Terrazzino, and I. Tinnirello. Experimental assessment of the backoff behavior of commercial IEEE 802.11b network cards. *IEEE INFOCOM 2007*, 2007.
- [8] Peter Clifford, Ken Duffy, John Foy, Douglas J. Leith, and David Malone. Modeling 802.11e for data traffic parameter design. *Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt 2006)*, 2006.
- [9] Ian Dangerfield, David Malone, and Doug Leith. Testbed evaluation of 802.11e EDCA for enhanced voice over WLAN performance. *International Workshop On Wireless Network Measurement (WiNMee 2006)*, 2006.
- [10] Ian Dangerfield, David Malone, and Doug Leith. Understanding 802.11e voice behaviour via testbed measurements and modeling. *Workshop on Wireless Network Measurement (WiNMee 2007)*, 2007.

- [11] K. Duffy and A. J. Ganesh. Modeling the impact of buffering on 802.11. *IEEE Communications Letters*, 11 (2), 219-221, 2007.
- [12] Rick Durrett. Essentials of stochastic processes. 1999.
- [13] Matthew S. Gast. 802.11 wireless networks. 2005.
- [14] Z. Hadzi-Velkov and B. Spasenovski. Capture effect with diversity in IEEE 802.11b DCF. *Eighth IEEE International Symposium on Computers and Communication, Proceedings.*, 2003.
- [15] ITU. G.711 recommendation. <http://www.itu.int/rec/T-REC-G.711-198811-I/en>.
- [16] Van Jacobson, Craig Leres, and Steven McCanne. Tcpdump. <http://www.tcpdump.org>.
- [17] JDC Little. A proof for the queuing formula: $L = \lambda w$. *Operations Research*, 09(3):383-387, 1961.
- [18] David Malone, Peter Clifford, David Reid, and Douglas J. Leith. Experimental implementation of optimal WLAN channel selection without communication. *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN '07)*, 2007.
- [19] David Malone, Ian Dangerfield, and Doug Leith. Verification of common 802.11 MAC model assumptions. *Passive and Active Measurement Conference (PAM 2007)*, 2007.
- [20] David Malone, Ken Duffy, and Doug Leith. Modeling the 802.11 distributed coordination function in non-saturated heterogeneous conditions. *IEEE/ACM Transactions on Networking*, 15(1), 2007.
- [21] A. Markopoulou, F. Tobagi, and M. Karam. Assessing the quality of voice communications over internet backbones. *IEEE Transactions on Networking*, 11(5):747-760, 2003.
- [22] Hugh Melvin and Liam Murphy. Time synchronization for voip quality of service. *IEEE Internet Computing*, 06(3):57-63, 2002.
- [23] Robert M. Metcalfe and David R. Boggs. Ethernet: distributed packet switching for local computer networks. *Commun. ACM*, 19(7), 1976.
- [24] A. Ng, D. Malone, and D. Leith. Experimental evaluation of TCP performance and fairness in an 802.11e testbed. *ACM SIGCOMM Workshops*, 2005.
- [25] Y Sun, I Sheriff, EM Belding-Royer, and KC Almeroth. An experimental study of multimedia traffic performance in mesh networks. *International Workshop on Wireless Traffic Measurements and Modeling (WitMeMo)*, 2005.

- [26] Li T, Ni Q., Malone D., D.J. Leith, Xiao Y, and Turletti T. A new MAC scheme for very high-speed WLANs. *Proceedings WoWMoM (International Symposium on a World of Wireless, Mobile and Multimedia Networks)*, 2006.
- [27] Ajay Tirumala, Feng Qin, Jon Dugan, Jim Ferguson, and Kevin Gibbs. Iperf. <http://dast.nlanr.net/Projects/Iperf/>.
- [28] Yang Xiao. Performance analysis of IEEE 802.11e EDCF under saturation condition. *2004 IEEE International Conference on Communications*, 2004.
- [29] K. Xu, M. Gerla, and S. Bae. How effective is the IEEE 802.11 RTS/CTS handshake in ad hoc networks? *IEEE GIOBECOM*, 1, 2002.
- [30] Hubert Zimmermann. OSI reference model the ISO model of architecture for open systems interconnection. *IEEE Transactions on Communications*, 28(4), 1980.