Optimization of Capacity in various 802.11 Gaming Scenarios

Hanghang Qi* and David Malone

Hamilton Institute,
National University of Ireland, Maynooth, Ireland
E-mail: Hanghang.Qi@nuim.ie
E-mail: David.Malone@nuim.ie
*Corresponding author

Dmitri Botvich

TSSG,
Waterford Institute of Technology, Ireland
E-mail: dbotvich@tssg.org

Abstract: As IEEE 802.11 wireless LANs are widely deployed and the popularity of multiplayer real-time gaming continues to increase, games are increasingly played over 802.11 wireless networks. In this paper, we consider the issue of how games traffic is likely to interact with wireless networks and give a framework to study this kind of application. Using Quake 4 as an example, we produce a multiplayer game traffic model. By introducing a model of MAC layer of IEEE 802.11 networks, we show how the performance changes as the number of players increases. The performance indicators that we consider include throughput, delay, jitter and mean opinion score (MOS). Several different network scenarios are considered and discussed, including when the game server and clients are connected wirelessly, and when the server is connected to the AP through a wired link and the use of multicast. We identify issues such as the Access Point (AP) or the game server becoming the bottleneck of the network. We also present solutions to these problems based on 802.11e. Using our 802.11e testbed, we demonstrate that our solution provides a subjective improvement in game performance. We believe the technique applied in this paper could be applied to improve the performance of other real-time applications, including other games.

Keywords: 802.11;multiplayer games;bottleneck;QoS;TXOP;multicast

Reference to this paper should be made as follows: Qi, H., Malone, D. and Botvich D. (2010) '802.11 Wireless LAN Multiplayer Game Capacity and Optimization', *Int. J. Advanced Media and Communication*, Vol. x, No. x, pp.xxx–xxx.

Biographical notes: Hanghang Qi is a PhD student and Research Assistant in the Hamilton Institute at NUI Maynooth. He received his MSc in Digital Communication and Signal Processing from University of Edinburgh, UK in 2006 and B.Eng. in Electrical Engineering from Shanghai Jiaotong University, China in 2004. His research interests include wireless communication networks, networked games, resource management and channel coding.

David Malone received B.A.(mod), M.Sc. and Ph.D. degrees in mathematics from Trinity College Dublin. During his time as a postgraduate, he became a member of the FreeBSD development team. He is a Stokes Lecturer at Hamilton Institute, NUI Maynooth. His interests include mathematics of networks, network measurement, IPv6 and systems administration. He is a co-author of O'Reilly's "IPv6 Network Administration".

Dmitri Botvich received his Bachelor's (mathematics) degree and Ph.D. (mathematics) from Moscow State University, Faculty of Mechanics and Mathematics, Russia, in 1980 and 1984, respectively. He is currently a principal investigator at the Telecommunication Software and Systems Group, Waterford Institute of Technology. His research interests include bio-inspired autonomic network management, wireless networking, trust management, queueing theory, optimization methods, and mathematical physics.

1 Introduction

Multiplayer real-time games have recently become a popular network application. Online games have drawn more people and game traffic takes up a reasonable portion of the Internet traffic [22, 2, 18, 14]. Meanwhile, wireless networks, especially IEEE 802.11 WLAN, have emerged as a common last-hop in the Internet. Playing multiplayer real-time games in a wireless network is a current reality and an obvious future direction. In this paper, our interest is in the wireless LAN's performance and support for multiplayer real-time FPS games. In particular, we will consider a number of situations including (1) where a group of players, and the game server are all wirelessly connected by an 802.11 access point as might happen at university dorms or any local community and (2) when the access point and games server are connected by a wired link, as might happen at a LAN party.

We expect 802.11 networks to face certain challenges before supporting real-time multiplayer games. 802.11 uses a CSMA/CA based MAC layer. Bianchi's model of the 802.11 MAC [3] has proven to be accurate and useful, and it has been extended to different network conditions and traffic loads (e.g. [12, 17]). Our goal is to build a model for real-time multiplayer games over 802.11 using these models.

Throughput, delay (latency), jitter and packet loss are important factors for network games [11, 8]. Wireless networks may present challenges in latency-sensitive applications including VoIP and real-time games as wireless could introduce extra delay and jitter because of the CSMA/CA [1]. In this model, we focus on the WLAN's throughput, delay and jitter. As various compensation techniques are employed, packet loss is considered less important than delay and jitter [8].

Once we have built our model of the 802.11 network to predict loss, delay and jitter, we show how using the 802.11e MAC layer may improve the network performance. In [9], capacity of voice in 802.11 WLAN has been considered and it is shown that the access point can become the bottleneck. We show that with Quake 4 in an 802.11 infrastructure mode network, the AP and games server are potential bottlenecks when the number of stations increases. If it is an 802.11e network and priority is given to the AP and game server by properly selecting MAC parameters, the capacity can be increased from approximately 10 stations to 15 stations.

Besides considering the situation where the game server and clients are connected using a basic infrastructure mode network, we also explore other situations including the case where the game server is connected by a wired link to the AP (or the game server acts as AP), and the case where the game server can use broadcast/multicast to send the global game information to every client. In both these two scenarios, traffic is reduced compared to the purely wireless scheme, and thus we improve the performance and the network capacity. We show similar results demonstrating how the performance changes with the number of players.

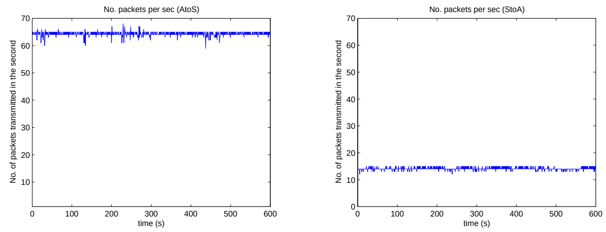
Quake 4 is used in our experiments for two main reasons: it is a typical and popular FPS game and it is well studied by the research community. Actually, the traffic model we summarize is general enough for many multiplayer real-time games and the results can be extended to games which use similar topologies. The rest of the paper is organized as follows: Section 2 presents the traffic characteristics of Quake 4. In section 3 we present our 802.11 model for network games, and throughput, delay and jitter are calculated. Section 4 presents our optimization scheme using the 802.11e MAC parameter TXOP. In Section 5, results from a normal DCF network and the optimized scheme are shown and compared. Section 5 also discuss the case that the game server is the AP or wired connected to the AP and where multicast is used. Section 6 talks about a demonstration in our 802.11 wireless game testbed. Finally, conclusions and future works are presented in Section 7. We note that this paper is an extended version of [20], which considers how to estimate the capacity when the AP and Server are connected both wirelessly and wiredly.

2 Quake 4 traffic in wireless network

In this section, we show the results of characterizing game traffic as measured in our wireless testbed in a typical indoor environment. These results are used to derive the parameters for our model and are compared to the results in wired networks. The testbed consists of 4 identical desktop PCs (Dell Inspiron 530) and a number of single board Soekris computers. All the stations are about 5–10 meters away from the AP. One PC acts as the AP, one PC acts as the game server and the other two PCs are game clients. The Soekris boxes are used later in Section 6. The Quake 4 Server is in spectate mode. Two players played the game on client PCs. Measurement is performed at the AP PC using tcpdump [16]. Key features are shown in Fig. 1 to Fig. 4.

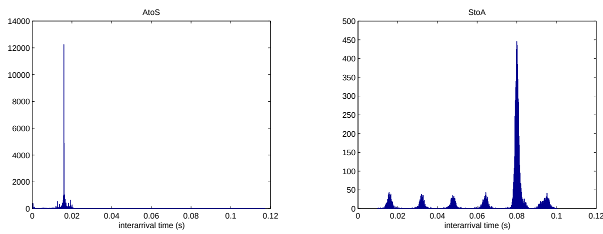
Fig. 1 illustrates the packet transmission rate. The number of packets transmitted in every second is calculated and shown. It is clear from the figure that the transmission rate is almost constant: around 65 packets per second from the clients and 14 packets per second from server. Fig. 2 shows the distribution of interarrival times. There are high peaks observed in both client-to-server and server-to-client traffic. Looking at Figs 1 and 2 together, we see that Quake 4 traffic can reasonably be considered to have constant packet rate over a good wireless network and that the rate from client to server is higher than from server to client.

Now, consider Fig. 3, which shows the packet size distribution. The packet size is spread in a range and it is larger from server to client than from client to server.



(a) Client to Server (b) Server to Client

Figure 1 Quake 4 game with two players. Packet transmission rate: the number of packets in every second during a game (10 minutes). The packet transmission rates are almost constant.



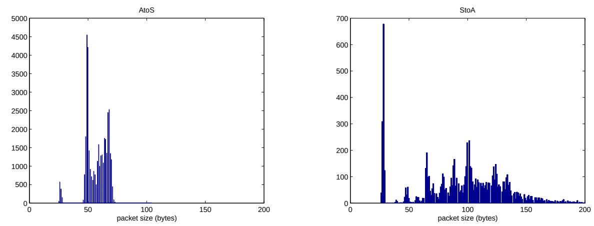
(a) Client to Server (b) Server to Client

Figure 2 Quake 4 game with two players. Packet transmission interarrival time histogram in wireless network. All the histogram figures are run over a duration of 10 minutes typical game play (which is commonly set for a real game)

It is known that the packet size exhibits correlation over time and can be modeled as an ARMA(1,1) process [10]. Fig. 4 illustrates bytes per second carried. Network throughput fluctuates in a considerable range. As the packet rate is almost constant, these fluctuations arise from variability in packet size.

Comparing our wireless results with previous studies from wired networks [13, 15, 4], we see that the game traffic behavior is similar (as expected). In our two-player testbed, game performance is good. Our assumption is that if the network can support the traffic indicated by scaling-up appropriately, then game performance will be good. When network becomes congested, as the number of clients (players) increases, delay and jitter will become too large, and the network will fail to support the games.

It was shown that the packet size from server to clients increases with the number of players [1]. In our wireless game testbed, we also find this feature. The results from wired networks show that server-to-client packet size distribution can be modeled as increasing linearly as the number of players increases [1, 10, 5]. Important quantities for the model include average packet size EP and average collision packet size EP* which means that the expected packet size for multiple packets. Following Bianchi's definition [3] , we approximate EP* by $EP* = E[max(EP1, EP2)]$. This is accurate enough as collisions are dominated by two-packet collisions. $EP1$ is the size of packet from server to client and $EP2$ is the size of packet from client to server. Using data from the SONG database [23], packet size distribution can be acquired, and EP and EP* are calculated.



(a) Client to Server (b) Server to Client

Figure 3 Quake 4 game with two players: packet size histogram in wireless network for duration of a game.

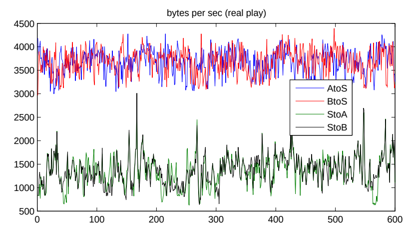


Figure 4 Quake 4 game with two players: Time series of bytes per seconds fluctuate in a range during the games. The first two lines are AtoS and BtoS. Client to server has higher traffic than from server to client. Experiments show that AtoS and BtoS (StoA and StoB) has very similar traffic.

	Client to Server	Server to Client
Packet transmission rate	λ_c=65(packets/s)	λ_s=14(packets/s)
Average packet length	EP_c=57.24(bytes)	EP_s=24.8n+45.4(bytes)
Average collision packet length	$EP*_c$=61.32(bytes)	$EP*_s$=30n+60(bytes)

Table 1 Summary of Quake 4 game traffic characteristics. n is the number of players.

Fig. 5 shows the result of the average packet size and average collision packet size from 2 players to 7 players. We use a linear fit to predict the values for larger numbers of players.

To summarize, game traffic characteristics have a constant packet rate with varying packet length (ARMA) transmission between the server and clients. Transmission rate and packet size distributions are different at server and client. The client to server packet size distribution does not change much with the number of players, while server to client packets size increases with the number of players. The Quake 4 traffic parameters are summarized in Table 1, and will be the input to our network traffic model.

Another interesting feature observed is that the server to client traffic happens in a burst, while client to server traffic is spread over time. This is shown in Fig. 6. Server to client traffic happens in a burst because the packets to each client are all

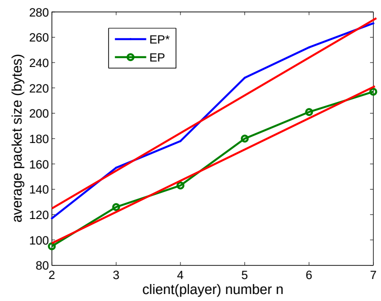


Figure 5 Server to Client average packet size is increasing as the number of clients increases. It can be linearly fitted. Over 2 players game data are used from SONG database [23].

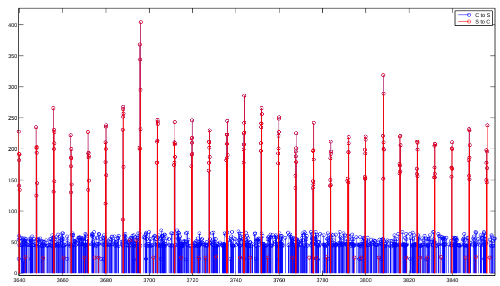


Figure 6 Packet size (bytes) time series for duration of Quake 4 game with 7 players. The high peaks are the burst of packets from server to all the clients (S to C in figure legend). The small packets, spread over time, are from all the clients to server (C to S in figure legend).

from the game server, and they are generated at almost the same time to help maintain fairness of the game [10]. Client traffic is generated on each client. Due to the lack of strict synchronization, packets do not happen in bursts but are spread over time. We do observe small fixed size packets observed from both sides which we believe are for synchronization purpose. Note, all these features are observed in both wired network and wireless network.

3 IEEE 802.11 MAC scheme and network model

3.1 Lossless Capacity of 802.11-like WLAN for game traffic

In this subsection, we examine an upper limit on the number of Quake 4 players that can be supported in an 802.11-like network. Specifically, how many players can play in the LAN, if the packets are perfectly packed onto the network with no packet losses? Let us consider the situation where a game server and many clients are playing in an infrastructure mode Wireless LAN, where the AP behaves essentially as a hub. Each

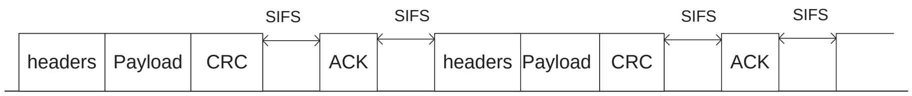


Figure 7 Packets transmission timing: Ideally, packets would be transmitted one after another with SIFS between payload packets and ACKs, with no collisions. Packets have preambles, MAC, IP and UDP headers and CRC at the end.

	802.11b	802.11g	11b short preamble	11g OFDM preamble
Preamble time	192us	192us	96us	20us
Data, MAC hdr and CRC rate	11Mbps	54Mbps	11Mbps	54Mbps
Headers	24 bytes	24 bytes	24 bytes	24 bytes
CRC	4 bytes	4 bytes	4 bytes	4 bytes
SIFS	10us	10us	10us	10us
ACK	14 bytes	14 bytes	14 bytes	14 bytes

Table 2 802.11 network MAC parameters

packet will be a UDP/IP packet, and the capacity is reached when the time required to transmit these packets fills the medium. In the 802.11 WLAN, where acknowlegements are used, optimal packet transmission pattern would be as in Fig. 7. Note this neglects the time lost to the normal CSMA/CA protocol used by 802.11, and so results in an upper bound.

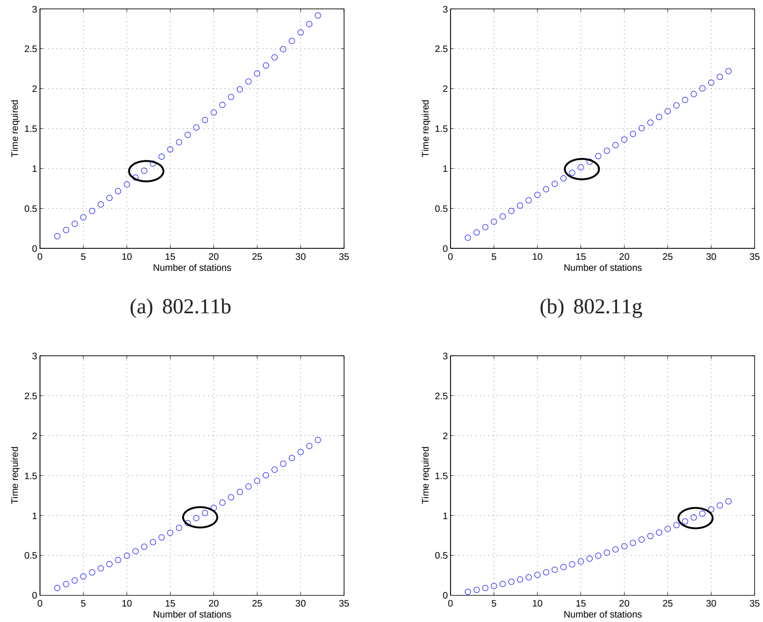
Each packet has significant overheads; they are proceeded by a PHY layer preamble and headers, and are followed by a CRC and MAC ACK frame. For a game with n players, to maintain 65 packets per second from client to server and 14 packets per second from server to client, the total time required in one second is (for 802.11b 11Mbps network)

$$\begin{aligned} t =& 2 \times (65 \times (preamble + (headers + payload(CtoS) + CRC) \times 8/11 \\ &+ SIFS + preamble + ACK/11 + SIFS) \times n \\ &+ 14 \times (preamble + (headers + payload(StoC) + CRC) \times 8/11 \\ &+ SIFS + preamble + ACK/11 + SIFS) \times n) \end{aligned}$$

The capacity result is shown in Fig. 8(a) for an 802.11b network (parameters are listed in Table 1). For 802.11g, with data speed of 54Mbps, surprisingly, the capacity only improves to 15 if the preamble can not be transmitted faster. This is shown in Fig. 8(b). This is because the payload size is small compared with the packet overheads. This kind of traffic sometimes can be called thin-stream traffic [19]

If a shorter preamble can be used, then substantial improvements in capacity are possible. 802.11g's OFDM-based rates allow the use of a 20us preamble. For 802.11b's-short preamble and 802.11g OFDM preamble use 96us and 20us, the resulting capacity is shown in Fig. 8(c)(d).

Notice that this capacity is with no collisions and no other traffic in the network with the constraint of having no packet losses.



(a) 802.11b (b) 802.11g

(c) 802.11b(fast) (d) 802.11g(fast)

Figure 8 Time used on the medium: actual time (second) needed for total game traffic transmission in one second versus the number of players for different 802.11 networks. A circle is drawn in each sub-figure to highlight the limit at 1

3.2 IEEE 802.11 CSMA/CA

An IEEE 802.11 infrastructure network with the DCF (distributed coordination function) MAC uses a CSMA/CA (carrier sense multiple access with collision avoidance) scheme with binary slotted exponential backoff. Briefly, when stations with packets to send sense the wireless medium is idle for a period of $DIFS$, each station goes to a count down state, and counts down a uniform random number chosen from the interval [0,CW-1]. While the medium remains idle, each station decreases the number by 1 after a slot time σ, until some station reaches 0 when the station transmits the packet (802.11e allows more than one packet to be transmitted). If the packet is successfully transmitted to its destination, the destination sends an ACK frame after a period of $SIFS$. Once the other stations receive the ACK frame, they know the medium is idle again, and they resume their count down. If two or more stations happen to reach 0 at the same time they transmit their packets simultaneously and a collision results. Destinations are not successful in receiving and thus no ACK frame is sent back. After a period those stations do not receive an ACK frame, then they know the transmission failed. They will try to resend their packet. They begin a new count down, where CW is doubled. After a successful transmission, CW is reset to the value CW_{min}. CW can be doubled to CW_{max} and then it will not change if there are further failures. If the number of failures of a particular packet reaches a limit, the packet is dropped and a new packet will be processed. 802.11e MAC enables the values of the MAC parameters $DIFS$

(called $AIFS$), CW_{min}, CW_{max} and $TXOP$ to be set on a per class bases for each station, with a maximum of four classes. We will be interested in the parameter $TXOP$. $TXOP$ in 802.11e stands for transmission opportunity. It allows a group of packets to be transmitted when the backoff counter reaches zero. Default value of $TXOP$ is one packet. When $TXOP$ is set to be n, it allows the station to transmit n consecutive packets (as long as there are packets in the buffer) without contention. Thus higher $TXOP$ gives this station higher priority to transmit more packets by sending more when the station seizes the channel.

Note that the 802.11 MAC in infrastructure mode also requires that stations do not communicate directly, but forward all packets through the access point. Thus, when calculating the load on our network, we must factor extra offerer load at the AP. The 802.11e MAC does include an extension, DLS (direct link setup), to allow direct station-to-station communication and we consider this as part of Section 5.3.

3.3 Two-dimensional Markov Chain model of 802.11 MAC

The MAC of 802.11 network can be modeled as a two-dimensional Markov Chain model, which can model both saturated and nonsaturated heterogeneous networks [3, 12, 17].

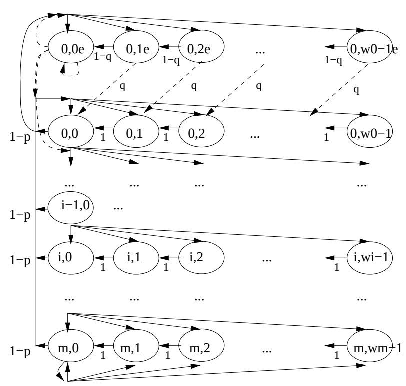


Figure 9 Markov chain model of 802.11 MAC

We will briefly summarise the model here, full details can be found in [17]. Each state in the 2-dimensional Markov Chain represents the count-down state of one station, using one dimension for the backoff counter and one for the backoff stage. Every time a packet fails to transmit, the MAC moves to next backoff stage. States are also included to model when the station does not have packets available to transmit. Different packet arrival rates lead to different parameters for each Markov chain, and so the collision probabilities p can be determined. For our wireless-networked games problem, we consider 3 classes of stations: AP, game server and clients. Each class has different input rates and so different performance. Generally, the AP and game server act like hubs of the network, where they have higher packet arrival rates than the clients.

Through the Markov chain, the collision probability p and transmission probability τ are entangled together. After solving a group of nonlinear equations, collision probability p and transmission probability τ of each class can be calculated. Using these, we can get $P_{tr} = 1 - (1-\tau)^n$, the probability that there is at least one transmission in a state, and

$$P_s = \frac{n\tau(1-\tau)^{n-1}}{P_{tr}} = \frac{n\tau(1-\tau)^{n-1}}{1-(1-\tau)^n}$$

the probability of a successful transmission occurring on the channel, conditioned on at least one station is transmitting. The normalized system throughput can then be expressed as the ratio

$$S = \frac{E[\texttt{payload information transmitted in a state time}]}{E[\texttt{length of a state time}]}$$

Using $E[P]$, the average packet payload size, σ, the duration of an empty state time, T_s, the average time the channel is considered busy during a successful transmission and T_c, the average time the channel is considered busy during a collision, we can write:

$$S = \frac{P_s P_{tr} E[P]}{(1-P_{tr})\sigma + P_{tr}P_sT_s + P_{tr}(1-P_s)T_c}. \quad (1)$$

We also calculate the channel access delay from the model. We work in terms of the length of a state L and the number of states it must wait N. Then $delay = E[D] = E[\sum_{i=1}^{N} L_i] = E[N]E[L]$ as N and L are independent and L_i are i.i.d. random variables. $E[L]$ is the average length of a state time. $E[N]$ is average average number of slots which is expressed as

$$E[N] = \frac{W_0}{2}\frac{1-(2p)^{m+1}}{1-2p} + \frac{W_0p(2p)^m}{2}\frac{1}{1-p} + \frac{1}{2}\frac{1}{1-p}. \quad (2)$$

This result is the same as Chatzimisios's [7].

We can also use this technique to calculate the jitter of the network channel access time. Jitter is defined as the variance of delay:

$$\begin{aligned} jitter = V[D] = E[D^2] - E[D]^2 = E[(\sum_{i=1}^{N} L_i)^2] - E[D]^2 \\ = E[N]E[L^2] + E[N^2]E[L]^2 - E[N]E[L]^2 - E[N]^2E[L]^2 \end{aligned}$$

where $E[L^2] = (1-P_{tr})\sigma^2 + P_{tr}P_sT_s^2 + P_{tr}(1-P_s)T_c^2$ and $E[L]$ and $E[D]$ are known, $E[N^2]$ is to be calculated as $E[N^2] = \sum_i N_i^2 p(N_i^2)$

After some algebra, including applying $\sum_{i=1}^{n} i^2 = n(n+1)(2n+1)/6$ and combining terms in the same column in two-dimensional Markov chain, we can get a formal expression of $E[N]$ (for details see [20]). Now with $E[N^2]$, $E[L^2]$ and $E[D]$, we can get jitter $V[D]$.

The MAC parameters of 802.11b network (11Mbps) used in the model are listed in Table 3. We also note that assumptions of this model include Poisson arrivals and a small amount of buffering at each station. The model has been shown to approximate constant packet rate traffic [17]. As we will see in the Section 3.4, games have been shown to be more sensitive to delay than loss, so a small buffer should be well adapted to games traffic.

802.11b parameters	Durations(μs)
Slot time, σ	20
Propagation delay, δ	1
$CW_{min} = 32\sigma$	640
DIFS(AIFS=0)	50
SIFS	10
PLCP Header@1Mbps	192
MAC Header 24 Bytes@11Mbps	17.5
CRC Header 4 Bytes@11Mbps	2.9
IP Header 20 Bytes@11Mbps	14.5
MAC ACK 14 Bytes@11Mbps	11.2

Table 3 802.11 network MAC parameters

3.4 Game MOS with Delay and Jitter

The effect of objective factors such as delay, jitter and packet loss to game performance have been widely studied [11, 8]. Other than these objective factors, people's subjective experience of games has been measured with Mean Opinion Score (MOS). MOS is based on people's judgment [21, 6, 24]. Recently, there has been work to use objective factors in order to understand MOS [6, 24]. We will use the work in [24], where the MOS score is predicted based on two quantities, the *ping average* and the *jitter average*.

The ping average is the average of 100 pings from client to server and 100 pings from server to client. These both represent full round trip time for the systems, we thus estimate them from our model using $ping\_average = D_C + D_{AP} + D_S + D_{AP}$, where D_C is the delay predicted for a client, D_{AP} is the delay predicted for an AP and D_S is the delay predicted for the server.

In [24] the jitter average is calculated as follows. 300 packets are sent from client to server, 50ms apart. The arrival times are noted and the shortest time shifted to zero. Then the mean of these is taken giving the mean client to server delay less the min client to server delay). This process is repeated with 300 packets from server to client, and the mean of the two results is taken. In our case, since the min delay is relatively small, we estimate this by $jitter\_average = (D_C + D_S)/2 + D_{AP}$.

We use these formulas directly for DCF, but when TXOP is in use, we note that the access delay is averaged over all the packets in a burst. Thus, when calculating $ping\_average$ and $jitter\_average$ for TXOP, we rescale D_S, D_{AP} by $(D_S + (K - 1)SIFS)/K$, where K is the number of packets in the TXOP burst. We believe this is a more representative scaling than the one used in [20], which rescaled delays over all packets, regardless of their transmitter.

Following [24], the *network impairment* is given by $X = 0.104 \times ping\_average + jitter\_average$, where the units are milliseconds. The mapping for the MOS is then given as

$$MOS = -0.00000587X^3 + 0.00139X^2 - 0.114X + 4.37.$$

Note that this MOS prediction takes delay into account, but does not directly account for loss, as for the range of losses considered by [24] this was found to produce satisfactory predictions.

4 Avoiding the bottleneck at AP and Server

In a normal 802.11 infrastructure network with the DCF MAC, all the stations, including the AP, have an equal opportunity to access the channel. As the MAC is based on chances to transmit single packets, when the network becomes busy each station can transmit roughly equal numbers of packets, regardless of their size. With the different packet loads at the server, client and AP, this equal sharing of transmission chances may become a problem. The AP and server will become the bottleneck of the network as they have more packets to transmit than the clients and the situation will become worse as the number of clients increases. Our idea is to effectively give more packets transmission opportunities to the AP and server using 802.11e's TXOP mechanism.

When there are n players in a game, the AP and server's TXOP value is set to be n times of the clients' TXOP value. TXOP effectively allows the AP and server to have longer packets which consists many small packets (though note that if a collision occurs, then the TXOP terminates after the first packet because of the missing ACK). Thus, in the model, the AP has the same transmission opportunity rate as clients and Server has fixed transmission opportunity rate. The effect is that the AP, server and client still have the same opportunity to access the medium, but AP and server transmit more packets in each opportunity. It has been shown that network performance can be improved for voice traffic [9] when "a suitable TXOP value" is set at the AP, which removes the network's bottle neck effect. The overall throughput and the number of conversations are improved. In following sections we show that it can also improve the network performance substantially for multiplayer games.

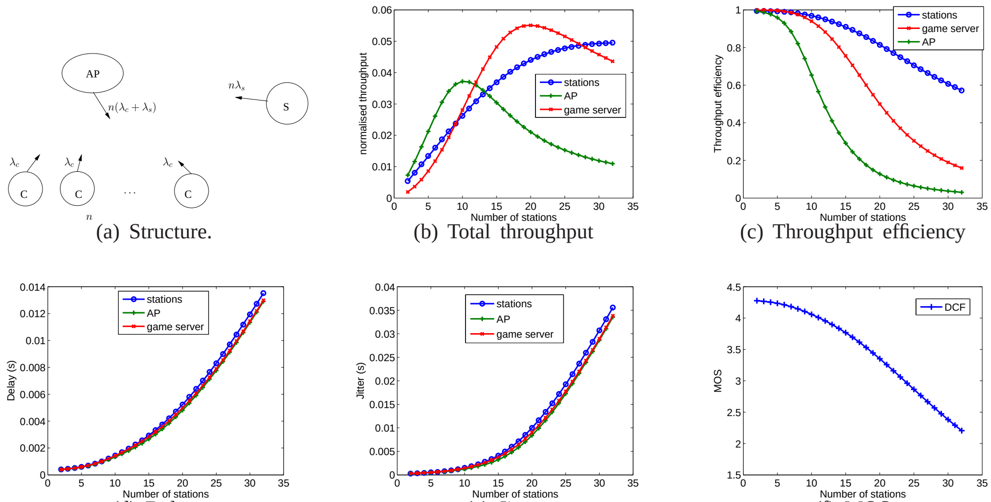
Since the games server knows the number of players, it can easily set its TXOP correctly. A question remains about how the access point would determine the number of players. One possibility is that a mechanism such as UPnP could be used by the game server to configure the AP's 802.11e settings. Another option is that the access point could identify game traffic using port numbers, or some similar mechanism, and apply an appropriate 802.11e configuration.

5 Results and Performance

5.1 Basic network with DCF

As we noted, a default infrastructure network with DCF, gives the AP, game server and clients all the same opportunity to access the medium. It is this basic DCF network we consider here. Fig. 10(a) shows the network structure and packet arrival rate at each station. Packet sizes in different classes are as indicated in Table 1. With our model, which assumes Poisson arrivals of the same rate, throughput, throughput efficiency, delay, jitter and MOS are calculated and shown from 10(b) to 10(f).

Fig. 10(b) and (c) show the throughput of each class of the three in two forms: total throughput and throughput efficiency. In Fig. 10(b), the AP reaches its peak at about 9 players and then drops; server reaches the peak at about 18 players and then drops. It indicates that they become congested and then with continued increase in number of clients, their throughput decreases. Stations' total throughput reaches a peak at about 24 players and then stays almost constant. In general, the network becomes congested before 9 players as the AP's throughput begin to decrease. Fig. 10(c) shows



(a) Structure. (b) Total throughput (c) Throughput efficiency

(d) Delay (e) Jitter (f) MOS

Figure 10 Performance of a DCF network with server, clients and AP connected wirelessly vs. the number of players.

the throughput efficiency which is the ratio of output throughput and input throughput. It is shown that the throughput efficiency of AP drops below 0.6 at about 10 players. While previous research has shown that games are resilient to loss, 40% loss indicates severe congestion and is the highest level of loss considered in [24].

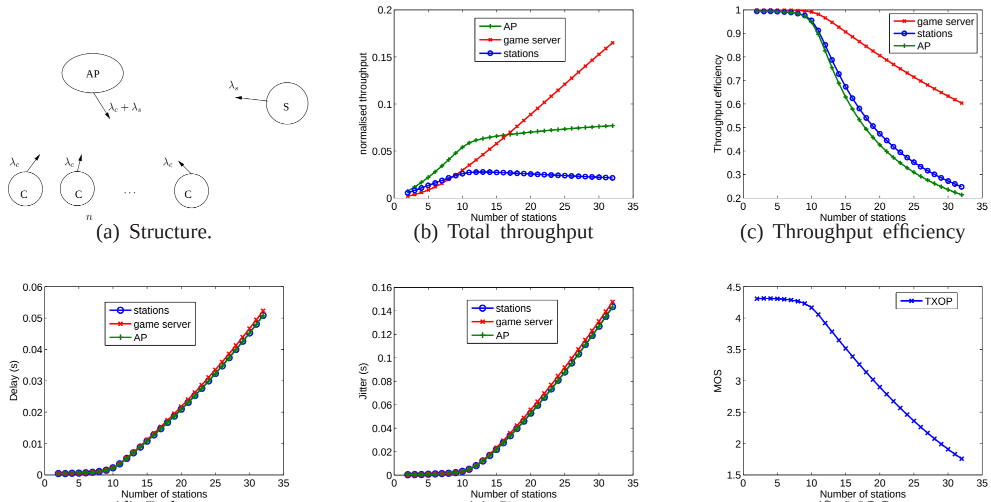
Fig. 10(d) and (e) show the delay and jitter. They initially increase gradually, but become steeper as the number of players increases. Fig. 10(f) shows the MOS. It decreases over the range, falling below a score of 4 at about 10 players. Overall, based on these results, the capacity of a default DCF 802.11b network is about 10 players.

5.2 AP and Server priority with TXOP

In this scheme, priority is given to the AP and server, as described in Section 4. The larger $TXOP$, which depends on player number n, is given to AP and server so that they can transmit multiple packets consecutively. The network structure and effective packet transmission rate are shown in Fig. 11(a). Again, the same performance indicators as in basic DCF networks are derived and shown in Fig. 11.

Fig. 11(b) and (b) show the total throughput and throughput efficiency respectively. Compared to basic DCF, the AP and server's performance are greatly improved. We see that the server and AP throughput are no longer as limited by the MAC, and it is the clients' throughput that first begins to dip. All the 3 class start to drop packets over 10 players and throughput efficiency drops below 0.6 over 15 players. Compared to the basic scheme, it appears that it can support more players before network becomes congested.

Fig. 11(d) and (e) show the access delay and jitter. Compared with Fig. 11(d) and (e), the transition to the network being congested is more obvious and consistent across AP, server and clients. For MOS, shown in Fig. 11(f), it now decreases slowly initially and then drops quickly after about 15 players. The MOS drops below 4 at about 12



(a) Structure. (b) Total throughput (c) Throughput efficiency
(d) Delay (e) Jitter (f) MOS

Figure 11 Performance of a network with server, AP and clients connected wirelessly when AP and Server priority with TXOP.

players and is about 3.5 at 15 players. However, throughput efficiency also remains above 0.6 at 15 stations. All in all, the capacity of the optimized network seems to be around 15 players. We are now operating beyond the edge of the lossless capacity of the network derived in Section 3.1, suggesting we must be close to largest usable capacity.

Comparing the MOS of both schemes, the optimized network MOS is above the default DCF network within 12 players at MOS of approximately 3.9. Though the predicted MOS is lower for the optimized scheme above 12 players, it is possible that the substantially lower levels of loss achieved by the optimized scheme may result in better game play in practice.

5.3 Server transmits via AP

Another possible topology is that the AP and the game server are co-located, from the wireless network's point of view. This may happen if the AP and server are connected by a wired link while the game clients remain connected wirelessly. This means that the AP does not need to relay packets between the AP and the stations. It is not difficult to imagine that this topology will save wireless network bandwidth. Note, this same network topology also models a network where the server is connected wirelessly, but 802.11e's DLS feature is in use. DLS is another feature in 802.11e, which allows any station to communication directly with any station without the relay of the AP.

Our network game model captures the improvement we expect to see. In Fig. 12, we see that even without using TXOP at AP, the throughput only begins to drop after 15 players, and a MOS score of above 4 is predicted for up to 19 players. Again, we expect that using 802.11e to give priority to the AP can also improve overall network performance. In this topology, Fig. 13 shows that throughput efficiency remains close to 1 to almost 18 stations if TXOP is used at the AP, and MOS scores remain above 4 up to 24 players.

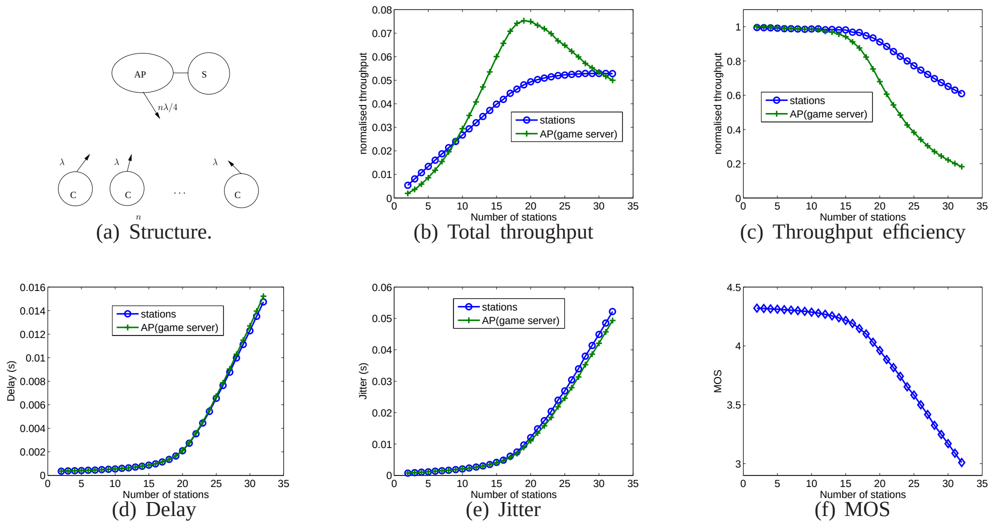


Figure 12 Results when AP transmits for the server. No TXOP in use.

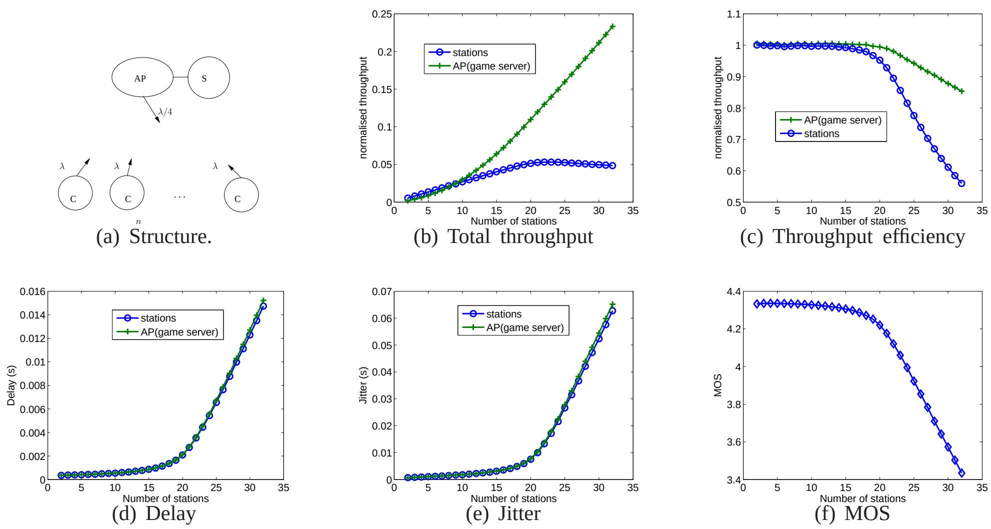
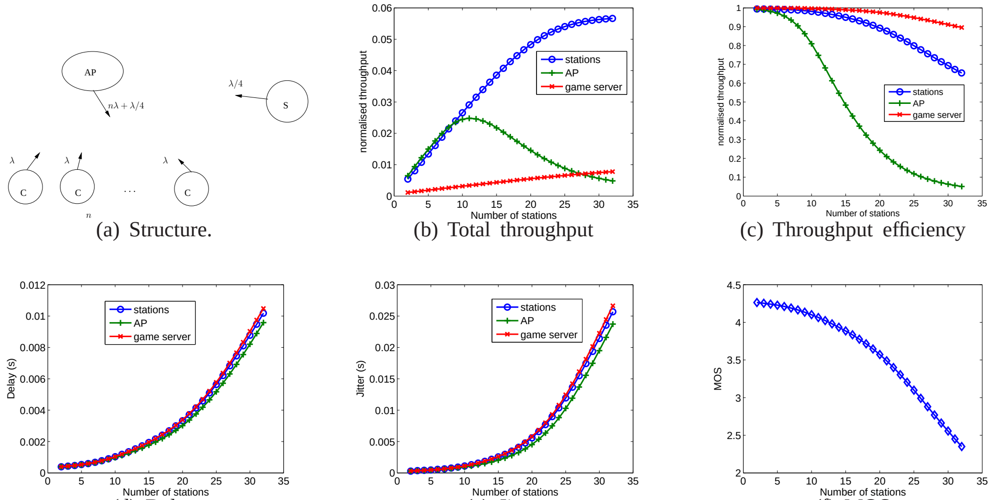


Figure 13 Results when AP transmits for the server. No TXOP in use and TXOP in use.



(a) Structure. (b) Total throughput (c) Throughput efficiency

(d) Delay (e) Jitter (f) MOS

Figure 14 Results when multicast is used, no TXOP.

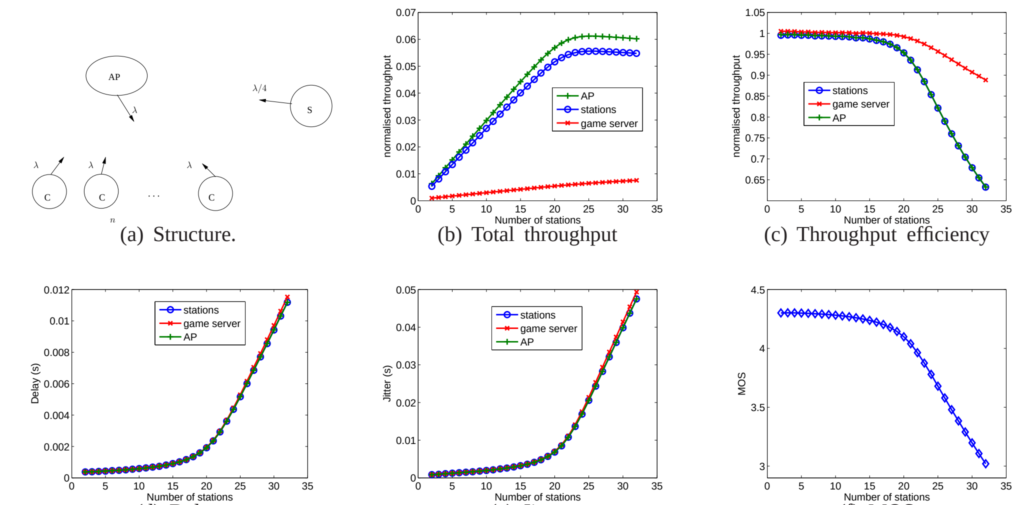
5.4 Server uses multicast

The game server holds the complete game state and needs to communicate this to every client regularly. If the server could use multicast to distribute this state to the clients, then significant wireless bandwidth could be saved. We assume that the single multicast packet size would be a little larger than the maximum size of the server-to-client packets. In 802.11, a multicast packet from the server would actually be unicast to the AP and then multicast to the whole network. This second transmission is not subject to acknowledgement or retransmission. For simplicity of modeling, we model this second transmission as a regular unicast transmission which means that we treat the packet as if it as acknowledged. However, we must check that the collision rate in the network does not become too high, otherwise multicast transmission will become unreliable.

The results of modeling with and without TXOP are shown in Fig. 14 and Fig. 15. As expected, multicast reduces the server to client traffic significantly. However, the throughput efficiency of the AP declines quickly without TXOP. Using TXOP offers a further improvement, by allowing the AP to send groups of packets to the server in a burst. We check the potential level of loss of multicast packets due to collisions, and find that it is less than 10% up to 14 stations for DCF and up to 20 stations when TXOP is used. Thus, we do not expect that multicast losses will substantially change the capacity.

6 Testbed Demonstration

Our predictions have been based on modeling of the game traffic and the 802.11 network, however we have tested our prioritised scheme in our testbed. As we did not have a large number of players available, we established a network with a server, AP, 2 real players and 8 emulated players. We chose to perform our test with a total of 10



(a) Structure. (b) Total throughput (c) Throughput efficiency

(d) Delay (e) Jitter (f) MOS

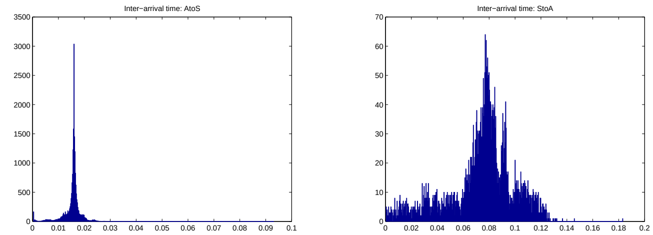
Figure 15 Results when multicast and TXOP are used.

players, because this is in the range of values where we should see poor performance with DCF alone, but see improved performance when TXOP is also used.

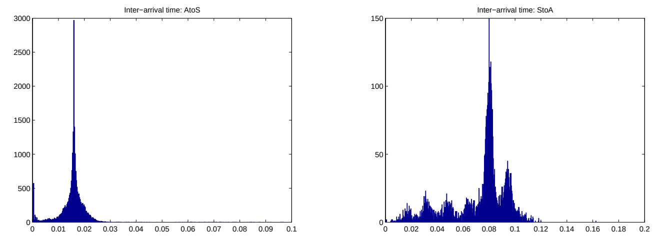
The games server, AP and real players are as described in Section 2. The emulated players are added by using Soekris based Linux boxes running mgen. We modified Mgen[1] to generate ARMA traffic, as described in [10] ($p = 0.8$, $q = 0.4$ and $var(w) = 20^2$). We also modified the driver to allow the game traffic to be sent to an 802.11e queue with a specific TXOP value. We use the default buffering for in MadWiFi driver (about 470 packets, for the version in use). Note that buffering in the real system is considerably more complex than in the model, with buffering at multiple layers, so we cannot perform a direct comparison of the results. However, we expect that both quantitatively and qualitatively to see improved performance when we enable TXOP. To be noticed, in the real 802.11 testbed measurement, the buffer size in each stations are big (larger than 15) which is different with the small buffer size (0 or 1) assumed in the model. Big buffer size maintains the networks' throughput and keeps its loss low, while it adds larger delay. This introduces some gap between the testbed results with the small buffer size model's results.

First consider the DCF network without TXOP configuration. Fig. 16(a) and Fig. 16(b) show the interarrival times of packets at the server and clients respectively. We see that the interarrival times have spread out considerably compared to those shown in Fig. 2, indicating increased jitter. When there were 10 (simulated) clients, players experienced lag and inconsistency in gameplay. The position of opponents can change abruptly. When players shoot or move fast, motion occasionally feels sluggish, as if the movement does not happen immediately. Overall performance is OK when the player is alone in the game world, but irritating when players meet.

We then set TXOP at the access point and the server. We use a value of 7000us at the access point and 4000us at the server, which corresponds to 10 (the number of total clients) packets at the server and about 10 packets from the server plus 10 packets from the clients. Each server packet takes up about 440 us and client packet is about 270



(a) Client to sever interarrival time histogram, basic DCF (b) Server to client interarrival time histogram, basic DCF



(c) Client to Server interarrival time histogram with TXOP (d) Server to client interarrival time histogram with TXOP

Figure 16 Interarrival times measured in testbed, 10 players

us. So it is about $440 \times 10 = 4400$ us and $440 \times 10 + 270 \times 10 = 7100$ us, and these values are rounded appropriately to values that can be set in the driver. The resulting interarrival times are shown in Fig. 16(c) and Fig. 16(d). The interarrival times at the clients have improved and the individual peaks from Fig. 2 are now visible again. The arrivals at the server are now slightly more jittered, however we can see some packets with very small interarrival times, corresponding to the gap between packets in a single TXOP burst.

With TXOP enabled, subjective gameplay is improved. Now the game performance is much smoother than without TXOP. The inconsistency is reduced. When players meet each other or take actions, the response is fast. Compared with just 2 or 3 players, some lag can still be felt, however the overall performance is playable.

7 Conclusions and Future Work

In this paper, we presented a theoretical model to predict the performance of 802.11 infrastructure WLAN with multiplayer real-time games (Quake 4 in this instance). We used traffic characteristics measured from a wireless testbed to establish traffic parameters and combined this with MAC and MOS modeling. We demonstrated how the distribution of transmission opportunities given by the MAC does not match the traffic load for a multiplayer game in a WLAN. Based on our model, we see that the 802.11e parameter TXOP can be used to prioritize AP and game server to improve game performance. The network capacity can improve from 10 players with normal DCF to 15 players with TXOP prioritization at the AP and the game server. Similar improvements are possible even if the server is on the wired network or if multicast packets were used to distribute game state. We implemented this scheme in a wireless testbed, where we see subjective and objective improvements in the games.

As future work, we aim to consider the interaction between available buffering and game performance. Increasing buffering offers an interesting tradeoff for games, where losses can be substantially reduced at the cost of increased delay. We are also investigated mechanisms to protect game traffic from competing traffic, such as TCP, which contend for both bandwidth and buffer space in the network.

Acknowledgements

The authors wish to acknowledge funding support from the Irish HEA PRTLI Cycle 4 FutureComm (http://futurecomm.tssg.org) programme.

References

[1] G. Armitage, M. Claypool, and P. Branch. *Networking and online games: understanding and engineering multiplayer Internet games*. John Wiley & Sons, Ltd., Chichester, England, 2006.

[2] Enterrainment Software Association. 2005 sales, damographics and usage data — essential facts about the computer and vedeo game industry. 2005.

[3] G. Bianchi. Performance analysis of the IEEE 802.11 distributed coordination function. *Selected Areas in Communications, IEEE Journal on*, 18(3):535–547, Mar 2000.

[4] Michael S. Borella. Source models of network game traffic. *Computer Communications*, 23:403–410, 2000.

[5] P. Branch and G. Armitage. Extrapolating server to client IP traffic from empirical measurements of first person shooter games. In *NetGames '06*, NY, USA, 2006. ACM.

[6] B. Carrig, D. Denieffe, and J. Murphy. Supporting first person shooter games in wireless local area networks. In *Personal, Indoor and Mobile Radio Communications, 2007. PIMRC 2007. IEEE 18th International Symposium on*, pages 1–6, Sept. 2007.

[7] P. Chatzimisios, A.C. Boucouvalas, and V. Vitsas. Packet delay analysis of IEEE 802.11 MAC protocol. *Electronics Letters*, 39(18):1358–1359, Sept. 2003.

[8] M. Claypool and K. Claypool. Latency and player actions in online games. *Commun. ACM*, 49, 2006.

[9] P. Clifford, K. Duffy, D. Leith, and D. Malone. On improving voice capacity in 802.11 infrastructure networks. In *Wireless Networks, Communications and Mobile Computing, 2005 International Conference on*, 2005.

[10] A. Cricenti and P. Branch. ARMA(1,1) modeling of Quake 4 server to client game traffic. In *NetGames '07*, NY, USA, 2007. ACM.

[11] M. Dick, O. Wellnitz, and L. Wolf. Analysis of factors affecting players' performance and perception in multiplayer games. In *NetGames '05*, NY, USA, 2005. ACM.

[12] K. Duffy, D. Malone, and D.J. Leith. Modeling the 802.11 distributed coordination function in non-saturated conditions. *Communications Letters, IEEE*, 9(8):715–717, Aug 2005.

[13] W. Feng, F. Chang, W. Feng, and J. Walpole. A traffic characterization of popular on-line games. *IEEE/ACM Transactions on Networking*, 13:488–500, 2005.

[14] S. Gargolinski, C. St. Pierre, and M. Claypool. Game server selection for multiple players. In *NetGames '05*, New York, NY, USA, 2005. ACM.

[15] C. Ibrahim, E. Melike, and O. F. Sema. Analysis of multi-player online game traffic based on self-similarity. In *NetGames '06*, New York, NY, USA, 2006. ACM.

[16] Van Jacobson, Craig Leres, and Steven McCanne. Tcpdump. Technical report, http://www.tcpdump.org/, 2006.

[17] D. Malone, K. Duffy, and D. Leith. Modeling the 802.11 distributed coordination function in nonsaturated heterogeneous conditions. *Networking, IEEE/ACM Transactions on*, 15(1):159–172, Feb. 2007.

[18] S. McCreary and K. Claffy. Trends in wide area ip traffic patterns: A view from ames internet exchange. In *Proceedings of ITC Specialist Seminar on Measurement and Modeling of IP Traffic*, Sept 2000.

[19] Andreas Petlund, Kristian Evensen, Pål Halvorsen, and Carsten Griwodz. Improving application layer latency for reliable thin-stream game traffic. In *NetGames '08: Proceedings of the 7th ACM SIGCOMM Workshop on Network and System Support for Games*, pages 91–96, New York, NY, USA, 2008. ACM.

[20] Hanghang Qi, David Malone, and Dimitri Botvich. 802.11 wireless LAN multiplayer game capacity and optimization. In *NetGames*, 2009.

[21] C. Schaefer, T. Enderes, H. Ritter, and M. Zitterbart. Subjective quality assessment for multiplayer real-time games. In *NetGames '02*, NY, USA, 2002. ACM.

[22] N. Shah and C. Haigh. The video game industry, an industry analysis, from a VC perspective. 2005.

[23] Smart Networks. Simulating online networked games (SONG) database. *http://caia.swin.edu.au/sitcrc/song/download_traces.html*.

[24] A. F. Wattimena, R. E. Kooij, J. M. van Vugt, and O. K. Ahmed. Predicting the perceived quality of a first person shooter: the Quake IV G-model. In *NetGames '06*, NY, USA, 2006. ACM.