A Study of EDCA and DCF in Multihop Ad Hoc Networks

Tommi Larsson

Department of
Electrical and Information Technology
Lund Institute of Technology

Yusheng Liu

Department of
Signal and System
Chalmers University of Technology

Supervisor: Ali Hamidian

Examiner: Christian Nyberg, Sven Tafvelin

March 13, 2008

Abstract

The IEEE 802.11 based Wireless Local Area Network (WLAN) and ad hoc networks have been widely used and still continuously attract more and more interest. In recent years, along with the development of information technology, the requirements on the size of network, the speed of connection and the complexity of application are growing very quickly. Thus the Quality of Service (QoS) issue is paid much attention by researchers to achieve better performance in WLAN and ad hoc networks. The IEEE 802.11e standard which provides QoS enhancement in the original IEEE 802.11 legacy was released in 2005. In this thesis, the aim is to analyze the QoS performance gain provided by Enhanced Distributed Channel Access (EDCA) of IEEE 802.11e in ad hoc network compared to the original IEEE 802.11 standard, and the impact of changing low priority traffics on high priority traffics. Protocols are implemented in Network Simulator 2 (ns-2) to simulate the network and communication procedures.

Acknowledgments

We have had a great time working with this thesis and we have learnt very much about the IEEE 802.11 standard, protocols, and ns-2 simulation environment. We would like to thank our supervisor Ali Hamidian at the department of Electrical and Information Technology for his professional advice on the thesis, his great patience with us and his great support of always being ready to answer our questions. We also want to thank him for all the feedback and encouragement he gave us during our work with this thesis.

Table of Contents

Chapter 1

Introduction

There had been an huge increase of the usage of the Internet since the 90's of last century. Nowadays many people take the Internet for granted and expect it to be working all the time. E-mail is a very common way to communicate today and it is also very common to share files with other people. More and more people are also starting to use IP-telephony. In the beginning you needed a stationary computer to be able to connect to the Internet, which limited the possibility to be connected to the Internet. But in 1997 a new standard was developed, the IEEE 802.11, which defined how to communicate with other devices in a wireless network. Now you did not need a stationary computer to connect to the Internet anymore, and you could connect to the Internet with portable devices such as mobile phones, laptops or personal digital assistants (PDAs). In the 21st century the usage of Wireless Local Area Networks (WLANs) has increased very much. Larger and larger quantity of data is being sent and the number of users is increasing. This can be a problem sometimes in realtime applications as telephony or video transmissions, since the data from these kind of applications is very delay-sensitive. If the delay in a phone conversation is too large it is impossible to communicate, which means it is extremely important that the data is delivered in time. Therefore in 2005 a new standard was developed, the IEEE 802.11e, which is an amendment to the original IEEE 802.11. The expression Quality of Service (QoS) is introduced in IEEE 802.11e and its purpose is to increase the performance for delay-sensitive applications.

1.1 Problem Description

The major task in this project is to investigate the benefits from using the new MAC sublayer protocol EDCA from the IEEE 802.11e standard instead of the old MAC sublayer protocol DCF from the original IEEE 802.11 standard. These two protocols will be explained in more details later in this report. To fulfill this task we needed to setup a wireless network and simulate it with the two different MAC sublayer protocols in different situations. The simulations were going to be made in the ns-2 network simulator. But the ns-2 version 2.26 is based on the IEEE 802.11 so we had to add support for the new IEEE 802.11e to be able to do the simulations. The tasks that needed to be completed were to add support for the IEEE 802.11e standard in ns-2, set up a wireless network with different simulation scenarions, run all the simulations and finally analyze the simulation result data and present it with graphs or diagrams.

1.2 Disposition of the Report

Recent research works in this field are reviewed and summarized in Chapter 2. Chapter 3 and Chapter 4 studies the IEEE 802.11 standard and the IEEE 802.11e enhancement to the original standard. The principles of DCF and PCF are elaborated in detail and HCF is introduced to combine the advantages of both DCF and PCF. EDCA and HCCA are both discussed in Chapter 4, but only EDCA is analyzed in the simulation. In Chapter 5, network layer and MAC layer protocols are studied with a focus on ad hoc networks. In Chapter 6, the network simulator ns-2 is studied, and the simulation results are provided. And finally the whole thesis is concluded in Chapter 7.

Chapter 2

Related Works

Many simulation and analysis work have been done regarding the performance of IEEE 802.11e EDCA. In [10], the MAC level performance gain of IEEE 802.11e is analyzed. And in [11], the network level QoS performance supported by 802.11e EDCA is enhanced by using a parameter configuration algorithm to provide guaranteed throughput, and the optimal parameter selection algorithm is analyzed to efficiently utilize the network resources. In [12], the analysis is based on the application level comparison between EDCA and DCF to represent the perceptual quality of the end users. Also, many research work have been done to analyze the performance of IEEE 802.11e in the condition of saturated network load. In [13], the author proposes an analytical model to evaluate the saturation throughput which is based on the use of mean value analysis, and this scheme accurately demonstrated the effects of the change of contention window size and Arbitration Interframe Space. And in [14], a novel Markov chain based model with a simple architecture for EDCA performance analysis under the saturated traffic load is proposed. It incorporates more features of EDCA into the analysis and has better performance accuracy.

Not only the IEEE 802.11e supported nodes is analyzed in this field, in [15], the coexistence scenario of IEEE 802.11b and IEEE 802.11e nodes is analyzed to represent the compatibility and QoS support provided by 802.11e. This coexistence may deteriorate the QoS provision and the influence of 802.11b DCF stations is analyzed and new mechanism for performance improvement of IEEE 802.11 legacy stations is also proposed.

In [16], the performance is analyzed by varying the EDCA parameters in ad hoc network. Also, [17] presents a preliminary simulation study of a star topology based on the IEEE 802.11e standard regarding the degrading impact of the presence of the hidden stations on the observed delay and throughput, and [18] presents an analytical model for the performance evaluation of IEEE 802.11e EDCA scheme under finite load conditions on the basis of various instances of delay metric.

In this thesis, the focus is on analyzing the performance in the environment of random node movement in an area larger than the node radio transmission range, with random generated data connection.

Chapter 3

IEEE 802.11 Standard

IEEE 802.11 denotes a set of Wireless LAN (WLAN) standards developed by working group 11 of the IEEE LAN/MAN Standards Committee (IEEE 802). The IEEE 802.11 standard defines the media access control (MAC) and physical (PHY) layer for a local area network (LAN) with wireless connectivity [1]. It addresses local area networking where the connected devices communicate over the air interface to other devices that are within close proximity to each other using carrier sense multiple access protocol with collision avoidance (CSMA/CA) medium share mechanism.

3.1 Overview of IEEE 802.11 Standard Family

The 802.11 standard family includes six over-the-air modulation techniques that use the same basic MAC protocol. The first version of the 802.11 standard was released in 1997. It specified three PHY layer options: infrared (IR), frequency hopping spread spectrum (FHSS) and direct sequence spread spectrum (DSSS). FHSS and DSSS operate at the industrial, scientific and medical (ISM) band at 2.4 GHz, and IR uses near-visible light in the 850 nm to 950 nm range for signalling. The three basic options did not become widely used mostly because of the low data rate they can provide, maximum 2 Mbps. Then it is rapidly supplemented by 802.11b in 1999.

While in the same year, another amendment to the 802.11 legacy, 802.11a is also released. It operates in 5 GHz band to avoid the crowded 2.4 GHz band. Also, 802.11a uses the OFDM modulation scheme, the thus gives a much higher data rate of 54 Mbps. However, despite of the promising specifications and significant advantages against 802.11b, 802.11a did not become widely used. The main reason is that its carrier frequency is so high which yields a shorter transmission range and can be easily blocked or attenuated by walls and other solid objects, and moreover, it is not compatible with the original 802.11 legacy and 802.11b, which operates at a different frequency band but is widely popularized.

The first mass used amendment to the 802.11 standard is 802.11b. It operates at 2.4 GHz and supports the maximum data rate of 11 Mbps. 802.11b directly uses the DSSS modulation technique of the original 802.11 legacy standard and largely improves its data rate. At that time it can satisfy most of the network applications, thus not long after its release, the 802.11b products began to appear on the market and then became very widely used. This results in that the upcoming 802.11 amendments tend to choose to be backward compatible to 802.11b.

To further improve the data rate, in June 2003, 802.11g was released. It uses OFDM as the modulation scheme, and supports the maximum data rate of 54 Mbps, which is the same as 802.11a. It operates at 2.4 GHz and can be viewed as an enhancement of 802.11b. Actually, in a very short time it replaces 802.11b in the market, even before this standard is ratified by the standard group. And currently it is the most popular and de facto WLAN standard.

The next generation to improve the data rate is 802.11n. This new amendment to the 802.11 standard is estimated to reach a theoretical data rate of 540 Mbit/s (which may require an even higher raw data rate at the physical layer), which is up to 50 times faster than 802.11b, and up to 10 times faster than 802.11a or 802.11g. 802.11n builds upon previous 802.11 standards by adding MIMO (multiple input multiple output). MIMO is the use of multiple antennas at both the transmitter and receiver to improve communication performance. It is not only simply employing transmitter diversity and receiver diversity, but introducing space time coding to provide coding gain. This MIMO technology enhances MAC sublayer by coding multiple frames and send them in one physical layer packet, which largely improves the throughput.

There are some other important amendments that provide new functionalities to the original 802.11 legacy. 802.11i is an amendment to provide security network access. It includes the solution of WiFi Protected Access 2 (WPA2) and makes use of the Advanced Encryption Standard (AES) block cipher.

802.11e, the core of this thesis, provides Quality of Service (QoS) enhancements and will be explained in detail in the next chapter.

There are lots of other amendments of original 802.11 standard (actually it already consumes nearly all letters). In 2007, Task group TGma merged 8 main amendments (802.11a, b, d, e, g, h ,i, j) to one single document and renames it to IEEE 802.11-2007. This latest document contains cumulative changes from the amendments mentioned above.

Also, to provide interoperability with external networks, 802.11u is created to allow devices such as GSM and WCDMA cellular phones to join a WLAN. It assumes that a user is not pre-authorized to use the network, and allows access based on the user's relationship with the network, for example hotspot roaming agreements. It largely improves the connectivity of a travelling mobile station.

3.2 MAC Layer of IEEE 802.11 Legacy

The original 802.11 standard [1] is also referred as "802.11 legacy". The 802.11 standard specifies a common medium access control (MAC) layer, which provides a variety of functions that support the operation of 802.11-based WLANs. In general, the MAC layer coordinates access to a shared radio channel and utilizes protocols that enhance communications over a wireless medium to manage and maintain communications between 802.11 stations (end users and access points). Although there has been several enhancements to the PHY layer in 802.11a/b/g focusing on higher data rate, the medium access mechanism in the MAC sublayer does not have much changes.

The original 802.11 standard has defined two medium access mechanisms: the mandatory distributed coordination function (DCF), and the optional point coordination function (PCF). The DCF is the basis for the PCF and is used for best effort contention services. Since the DCF is a distributed access method, it can be used not only in infrastructure network configurations, but also in ad hoc network configurations. The PCF on the other hand, depends on the DCF and is required for contention-free services. Furthermore, it is only usable in infrastructure network configurations.

3.2.1 Distributed Coordination Function (DCF)

DCF is a method used by the stations contending to access the network resources and attempt to send frames when there is no other station transmitting. Carrier sense multiple access with collision avoidance (CSMA/CA) is the basic multiple access method used by DCF. Since it is contention-based, it can be used in both infrastructure and ad hoc networks.

CSMA/CA

In CSMA/CA, a station wishing to transmit has to listen to the channel for a predetermined amount of time so as to check for any activity on the channel, which is a radio medium that stations share. If the channel is sensed "idle" then the station is permitted to transmit. If the channel is sensed as "busy" the station has to defer its transmission. This is the essence of the "collision avoidance" part of the protocol.

RTS/CTS

One of the problems is that it is not possible to listen while sending, therefore collision detection is not possible. Moreover, the DCF suffers from hidden station problem that exists in contention based protocols. Two stations are hidden from each other if they are out of signal range and thus cannot hear each other. In this situation the carrier sense mechanism will not work properly and the hidden stations may both sense the medium free and start transmitting at the same time to a common receiver causing a collision. Therefore a Request to Send and Clear to Send (RTS/CTS) exchange mechanism can be optionally adopted by CSMA/CA to counter the hidden station problem, and to avoid collision by announcing the impending transmitting status of the neighbour stations. Before sending a data or a management frame, a station can transmit an RTS frame and await a CTS frame. The RTS and CTS frames contain a Duration/ID field that defines the period of time that the medium is to be reserved to transmit the actual data frame and the returning ACK frame. All stations within the reception range shall update their network allocation vectors (NAVs) so that they consider the medium busy until the end of the transmission. Thus, collisions occur only on RTS frames and are detected by the absence of a CTS frame.

However, RTS/CTS mechanism introduces large overhead especially for short data frames. This mechanism is only recommended when the data or management frame is greater than the threshold dot11RTSThreshold and only for directed frames.

Interframe Space

To control the waiting time before medium access, the DCF use the following parameters called interframe spaces (IFS): short interframe space (SIFS), DCF interframe space (DIFS), and extended interframe space (EIFS). The different length of the IFSs are defined to provide priority levels for access to the wireless media.

SIFS has the shortest waiting time, and is thus used to give the highest priority for medium access. It is used by short control messages, such as ACK and CTS frames. DIFS is a waiting time longer than SIFS and thus has lower priority for medium access. It is used to transmit data or management frames by the stations operating DCF. EIFS has the longest waiting time and is only used when a transmission failure occurs. A station that receives an incorrect frame must wait for EIFS before starting its transmission in order to give other stations enough time to acknowledge the frame that the station received incorrectly.

DCF Access Procedure

Before transmitting, each station in the network has to sense the medium. There are two carrier sense mechanisms in PHY layer and in NAV which is virtually provided by MAC layer. If both of the two functions indicate an idle medium, the medium is considered idle; otherwise the medium is considered busy. When the station determines that the medium is sensed to be idle for greater than or equal to a DIFS period, it can initiate a transmission immediately. Otherwise, if the medium is sensed to be busy or if it is sensed to be idle but becomes busy before it reaches the DIFS period, the station must defer the pending transmission by itself, waits until the end of the current ongoing transmission which occupies the medium, and starts to sense the medium to be idle for a DIFS period without interruption. Then a random backoff process is invoked to reduce the possibility of collision, since in this situation (after the medium becomes idle following a busy medium) several stations may be waiting for the medium to become idle and there is a high probability of collision. The random backoff time is calculated as below:

$$backoffTime = random() \times aSlotTime$$

where random() is a uniformly distributed pseudo-random integer between zero and contention window (CW), and aSlotTime is a PHY-dependent value. It means when a station senses the medium to be idle again for a DIFS period, then it begins to wait another random times of the slot time. During the backoff time, if the medium is sensed to be idle for a whole slot, the generated random number is decremented with one, and if the medium is sensed to be busy, the backoff timer becomes suspended until the medium is sensed to be idle for the duration of DIFS and then the backoff procedure resumes. The station can start to transmit the pending frame if the backoff timer reaches zero. This procedure ensures that the station selecting the smallest backoff time using the random function will win the contention.

The value of CW varies between CWmin and CWmax, which are PHY-dependent. It is possible that two stations which choose the same random number finish the backoff procedure and start to transmit simultaneously, which causes collision again. To reduce the possibility of collision in this situation, each station maintains a CW window with different length. Initially CW is set to CWmin and its value doubles if a collision occurs until CWmax is reached. If the transmission is successful, the CW value reverts to CWmin before a new random backoff interval is chosen.

Upon receiving a frame, the destination station waits for the duration of SIFS and responds with an ACK frame to notify the sender of a successful reception.

3.2.2 Point Coordination Function (PCF)

PCF uses polling to regulate access to the shared medium. It relies on a central node, which is called a point coordinator (PC), to poll a registered station to communicate. So PCF is restricted to the 802.11 infrastructure mode, and the PC is located at the access point (AP). When PCF is used, the time is divided into a contention period (CP), during which the DCF manages multiple access, and a contention free period (CFP), during which the PCF manages multiple access. During the CFP, every station maintains a parameter that indicates the duration of the CFP. It prevents stations to contend for access to the medium and thus they will not initiate a transmission unless they are polled by the PC. However, PC determines the maximum duration of a CFP, so it can terminate a CFP at or before it reaches the limit, based on available traffic and size of the polling list.

In stations operating PCF, a new interframe space called PCF interframe space (PIFS) is introduced. PIFS is a waiting time longer than SIFS but shorter than DIFS, which results in a medium priority.

The PC maintains a polling list for CF pollable stations and these stations registers with an association message and gets an Association ID (AID) from the PC, and it controls the length of the CFP. At the beginning of each CFP, the PC shall sense the medium. When the medium is determined to be idle for a PIFS period, the PC shall transmit a beacon frame containing the CF parameter set element and a DTIM element. Thus the PC has a higher priority to access the medium than the other stations which wait for a DIFS period. Then the PC shall wait for at least one SIFS period, and then transmit one of the following frames: a data frame, a CF-poll frame, a data plus CF-poll frame, or a CF-end frame. If the CFP is null (i.e. there is no traffic buffered and no polls to send at the PC) a CF-end frame shall be transmitted immediately after the initial beacon.

3.2.3 Limitations of IEEE 802.11 Legacy

The two multiple access methods of DCF and PCF both have their limitations. DCF provides only best effort service and all stations contend for access to the medium with the same priority. Thus the DCF does not guarantee bandwidth, packet delay and jitter, and will have a throughput degradation in the heavy load environment. Although PCF allows for a better management of the QoS, the polling scheme in PCF is not efficient enough. So it has only been implemented in very few hardware devices as it is not part of the WiFi Alliances's interoperability standard.

Chapter 4

IEEE 802.11e Amendment: QoS Support

4.1 Overview of IEEE 802.11e: MAC QoS Enhancements

The IEEE 802.11e has been approved as a standard that defines a set of Quality of Service (QoS) mechanisms for WLANs [2]. It differentiates traffic types and sources and is considered to be of critical importance for delay-sensitive and bandwidth-sensitive applications, such as streaming multimedia and wireless Voice over IP (VoIP). It addresses the QoS issues in the MAC sublayer.

4.2 Hybrid Coordination Function (HCF)

To improve the QoS limitations of DCF and PCF, IEEE 802.11e specifies a new coordination function, the hybrid coordination function (HCF). It provides a hybrid access method approach to achieve a better QoS performance.

The purpose of HCF is to combine contention-based and contention-free medium access method, and to replace the legacy DCF and PCF in a QoS Station. Within the HCF, there are two methods of channel access, Enhanced Distributed Channel Access (EDCA) and HCF Controlled Channel Access (HCCA). EDCA, which provides distributed access method, can be viewed as an enhancement of DCF and can be used in both infrastructure networks and ad hoc networks. HCCA provides centralized access method and can be only used in infrastructure networks.

802.11e defines other new features to give better QoS performance. A transmission opportunity (TXOP) is a bounded time interval reserved for a specific station. If the frame length is shorter than the TXOP, the station is allowed to send as many frames as it can during its TXOPs. If the frame length is larger than the TXOP, the station must fragment the large frame into smaller blocks each of which can be sent in the length of TXOP. The introduction of TXOP reduces the problem of low rate stations gaining an inordinate amount of channel time in the 802.11 legacy DCF.

Admission control is negotiated by the usage of a Traffic Specification (TSPEC). TSPEC describes the QoS requirements of a traffic stream by specifying a set of parameters such as nominal/maximum frame size, minimum/maximum service interval, service start time, minimum/mean/peak data rate, burst size, delay bound, minimum PHY rate and medium time. Most of the above mentioned parameters are typically set according to the requirements from the application while some are generated locally within the MAC. The parameter minimum/maximum service interval specifies the minimum/maximum time interval between the start of two consecutive TXOPs and service start time specifies the time when the service period starts. i.e. when the station expects to be ready to send frames. Burst size specifies the maximum size of the data burst that can be transmitted at the peak data rate. Medium time is the amount of time admitted to access the medium.

In addition to the mandatory features stated above, there are also four optional features for 802.11e MAC layer QoS. Automatic Power Save Delivery (APSD) is a more efficient power management method. It is very useful for the VoIP stations where data rates are roughly the same in both directions. It achieves power saving by making the VoIP station enter a doze state after it sends a block of data. The AP is triggered to send the buffered data in the other direction, which also trigger the VoIP station to send the next voice data.

Block Acknowledgement (BA) allows an entire TXOP (up to 64 frames) to be acknowledged in a single ACK frame, which provide less protocol overhead when longer TXOPs are specified. The BA mechanism is initialized by an exchange of add block acknowledgement (ADDBA) request/response frames, multiple data frames can be transmitted. When the sender needs an ACK, it sends a block acknowledgement request (BlockAckReq) control frame to the receiver, and the receiver replies with a block acknowledgement (BlockAck) control frame acknowledging the successfully received data frames. The BA mechanism has two implementations, namely immediate BA and delayed BA. In immediate BA, the BlockAck frame is sent to the sender immediately after it receives the BlockAck-Req. However in delayed BA, which is designed for the stations with low processing power, an ACK is used to response to the sender to save calculation power before the content of a BlockAck is ready to be sent after the station finishes the calculation of the BlockAck. A BA setup can be torn down, e.g. when there are no more data frames to be sent, by sending a delete block ACK (DELBA) frame.

In 802.11e, MAC-level Acknowledgement become optional, service class for frames to send can have two values: QosAck and QosNoAck. Frames with QosNoAck are not acknowledged when it has correctly received a frame. This also means that reliability of QosNoAck is reduced. But it improves the overall MAC efficiency and avoids retransmissions of highly time-sensitive data, such as VoIP, where the data has certain very strict lifetime.

Direct Link Setup (DLS) allows direct station-to-station frame transfer within an infrastructure network without relying on the AP to forward the frames. This is designed for consumer use, where station-to-station transfer is more commonly used. When a station needs to setup a direct link to another station, it sends a DLS request action frame to the QoS access point (QAP). The QAP relays the request to the latter station, which replies a DLS response back to the QAP. Then the QAP forward the DLS response to the initializing station to finish this handshake procedure and finally the two stations can communicate with each other directly.

4.2.1 Enhanced Distributed Channel Access (EDCA)

The main drawback of the DCF, regarding QoS provisioning, is that it can only support random access and cannot provide any service differentiation since all stations have the same priority, for example, the same CWmin, CWmax and waiting time before backoff or transmission (DIFS) [3].

EDCA is also a differentiated, distributed and contention-based medium access mechanism, but it enhances the original DCF to provide prioritized QoS. The EDCA mechanism defines four access categories (ACs) that provide support for the delivery of traffic with user priorities (UPs) at the stations. An AC is assigned to each frame before it enters the MAC layer based on its user priority (UP) or its frame type according to the table below:

(Priority)	(User Priority (UP))	(Access Category (AC))	(Designation)
lowest	1	AC_BK	Background
	2	AC_BK	Background
	0	AC_BE	Best effort
	3	AC_BE	Best effort
	4	AC_VI	Video
	5	AC_VI	Video
	6	AC_VO	Voice
highest	7	AC_VO	Voice

Table 4.1: UP to AC mappings

There are four traffic types corresponding to the four ACs, AC_BK stands for background traffic, AC_BE stands for best effort traffic, AC_VI stands for video traffic and AC_VO stands for voice traffic. For example, management frames should be sent with the highest priority, so AC_VO is selected for management frames.

An enhanced variant of the DCF, Enhanced Distributed Channel Access Function (EDCAF), is assigned to each AC. This enhanced DCF variant is called an EDCA parameter set. It is used by each AC to contend for medium access.

The AC parameter set contains the following parameters:

Arbitrary interframe space number (AIFSN): the number of time slots after a SIFS duration that a station has to defer before either invoking a backoff or starting a transmission. AIFSN affects the arbitration interframe space (AIFS), which specifies the duration (in time instead of number of time slots) a station must defer before backoff or transmission:

$$AIFS = SIFS + AIFSN \times SlotTime$$

So an AC with lower value of AIFSN has less AIFS and is thus given a high priority.

Contention Window (CW): a random number is drawn from this interval for calculating the total backoff time:

$$Backoff = AIFS + random[CWmin, CWmax]$$

An AC with lower values of CWmin and CWmax has higher probability to draw a smaller random number, thus it is given higher priority.

TXOP limit: the maximum duration for which a station can transmit after obtaining a TXOP. A value of zero means when this AC gets access to the medium, it is allowed to send only one frame from the AC queue. This is to limit the low priority traffic. And a value higher than zero means that an AC may transmit multiple frames only from its AC queue since a TXOP is given to an EDCAF in a specific AC, not to a station. As long as the duration of the transmissions does not exceed the TXOP limit, the station is allow to transmit frames from the specific AC queue. Thus an AC with a higher value of TXOP limit has a higher priority.

The EDCA parameters for each AC is shown in the table below:

(AC)	(CWmin)	(CWmax)	(AIFSN)	(TXOP limit (ms))	
				(802.11b)	(802.11a/g)
AC_BK	CWmin	CWmax	7	0	0
AC_BE	CWmin	CWmax	3	0	0
AC_VI	(CWmin+1)/2-1	CWmin	2	6.016	3.008
AC_VO	(CWmin+1)/4-1	(CWmin+1)/2-1	2	3.264	1.504

Table 4.2: Default EDCA parameter set

The values of these parameters can be updated by exchanging information through beacon frames. The detail of beacon frame handling is studied in Section 5.2. The EDCA parameter set information is only present in infrastructure mode. With proper tuning of AC parameters, traffic performance from different ACs can be optimized and prioritization of traffic can be achieved. This requires the QAP to maintain a common set of AC parameters to guarantee fairness of access for all stations. Also in order to address the asymmetry between uplink and the much heavier downlink traffic, a separate set of EDCA parameters is defined for the QAP only, which takes this asymmetry into account.

The EDCA implementation model is shown in Figure 4.1:

The procedure of EDCA can be summarized as follows. When data arrives at the MAC sublayer, it is firstly classified into an appropriate AC, and then pushed into the AC transmission queue. Frames from different ACs contend for channel access internally within the station. The internal contention algorithm calculates the backoff, independently for each AC, based on AIFSN, contention window, and a random number. The backoff procedure is similar to that in DCF, and the AC with the smallest backoff wins the internal contention. If the backoff time of more than one AC counts down to zero at the same time, an internal collision occurs within a station. These collisions are resolved such that the frames in the high-priority AC receive the TXOP whereas the frames in the low-priority AC act as if there was an external collision on the wireless medium. The winning AC would then contend externally for the wireless medium. The external contention algorithm has not changed significantly compared to DCF, except that in DCF the backoff and deferral are constant for a particular PHY, while 802.11e has changed the backoff and deferral to be variable, and the values are set according to the appropriate AC.

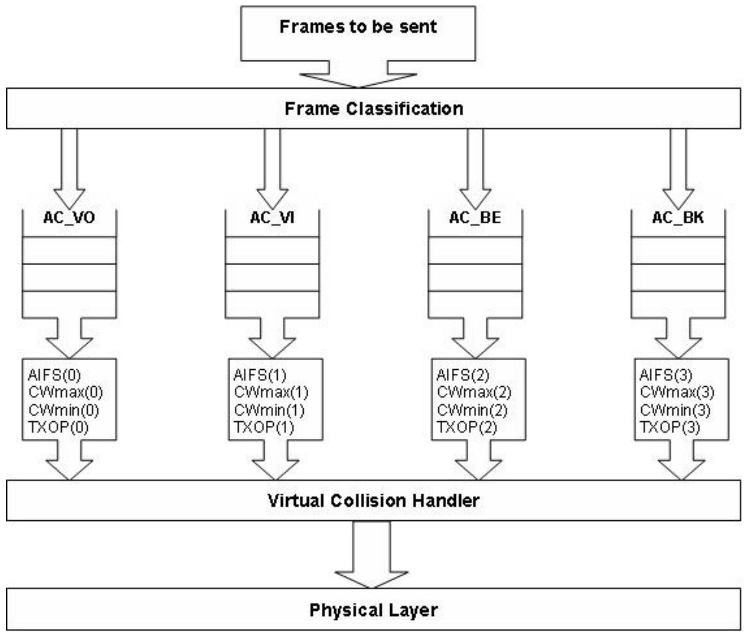

Figure 4.1: EDCA implementation model

4.2.2 HCF Controlled Channel Access (HCCA)

The HCCA is the centralized, contention-free medium access mechanism of the HCF and uses a hybrid coordinator (HC) which is collocated with the QAP, to manage access to the medium. It can be view as an enhancement of the original PCF to provide parameterized QoS. Similar to PCF, HCCA provides polled access to the wireless medium. But unlike PCF, QoS polling which takes place during CP and scheduling of packets is based on admitted TSPECS.

The central concept of HCCA is controlled access phase (CAP), which is a bounded time interval and formed by concatenating a series of HCCA (polled) TXOPs. Scheduling of HCCA TXOP and formation of CAP are performed by the HC. The HC gains control of the wireless medium as needed to send QoS traffic to non-AP stations and to issue QoS CP-Poll frames to non-AP stations by waiting PIFS, which is shorter than DIFS and AIFS that other stations using EDCA procedures must wait. The duration values used in QoS frame exchange sequence reserve the medium to permit completion of the current sequence. The HC may include a CF Parameter Set element in the beacon frame it generates. This causes the infrastructure network to appear to be a point-coordinated network to stations. And it also causes the stations to set their NAVs to the CFPDurRemaining value in the CF Parameter Set element value at TBTT, as specified in Section 5.2.2. this prevents most contention in the CFP by preventing nonplooed transmissions by non-AP stations whether or not they are CF-pollable.

Chapter 5

Mobile Ad Hoc Networks

Mobile Ad hoc Network (MANET), is a temporary network formed by a collection of mobile nodes. Unlike its counterpart infrastructure mode as defined in the IEEE 802.11 standard, ad hoc network is a network with a dynamic topology and without centralized administration or standard support services. Each node can be viewed as an independent router to support connectivity to other mobile nodes that are out of range [3]. Originally conceived for mostly military purposes, ad hoc networks allow for ease of deployment and appeal to various commercial applications such as convention meetings, electronic classrooms, search-and-rescue efforts, disaster relief, and law enforcement.

5.1 Ad Hoc Routing Protocols

As radio coverage is usually limited, multi-hop routing is often needed. This is achieved by an ad hoc routing protocol, which automatically discovers the neighbours and sets up routes, and thus mobility is supported, as the information about routes and topology are exchanged among the wireless nodes. The routing protocols of ad hoc networks can be classified into two main classes: proactive and reactive routing protocols.

In proactive routing, routing tables are updated periodically and are almost always up to date. Thus, the delay before sending a packet is minimal but at the cost of the increased routing overhead, and to increase mobility, more updates are needed which consumes more power and bandwidth of mobile nodes.

In reactive routing, routes are founded on-demand, for example, the sending station searches for a route to the destination station only when it needs to communicate. Hence no control massage is needed for non-active routes, and the routing overhead is minimized which saves power and bandwidth. The drawback is that a considerable delay is introduced when performing the route discovery procedure. So it may not be appropriate for real time communication.

Many routing protocols have been proposed. For example, Optimized Link State Routing Protocol (OLSR) and Topology dissemination Based on Reverse-Path Forwarding (TBRPF) are proactive routing protocols; and Dynamic Source Routing (DSR), Ad hoc On-demand Distance Vector (AODV) and DYnamic MANET On-demand (DYMO), are reactive routing protocols. Moreover, there is a kind of hybrid routing protocols which combines the proactive and reactive routing mechanisms together, for example Hazy Sighted Link State routing protocol (HSLS) and Zone Routing

Protocol (ZRP).

5.1.1 OLSR

OLSR is a proactive routing protocol developed for MANETs. It is an optimization of a pure link state routing protocol which is based on the concept of multipoint relays (MPRs).This is done by allowing only some selected stations to forward the broadcast messages during the flooding process. Each station in the network selects a subset of its neighbors as MPRs. To avoid problems associated with uni-directional links, the candidates for MPRs must have bi-directional links.

Using multi-point relays reduces the size of the control messages. It also minimizes flooding of control traffic. This technique significantly reduces the number of retransmissions of broadcast control messages.

5.2 MAC Layer Synchronization in Ad Hoc Networks

In an IEEE 802.11 WLAN, each station (STA) maintains a timer with modulus 264 counting in increments of microseconds. Timer synchronization is necessary for power management, synchronization of frequency hopping, and PCF polling, to predict the start of a frame.

5.2.1 The TSF of IEEE 802.11

In order to synchronize the timers, IEEE 802.11 adopts the Timing Synchronization Function (TSF).

In infrastructure network, TSF is easy to implement. The access point (AP) periodically broadcasts a beacon frame which contains the timing information, and all the other STAs that receive the beacon frame adopt the access point's timer value.

In Ad-hoc network, or an independent basic service set (IBSS, ad hoc network is called an IBSS in the IEEE 802.11 standard, therefore we use ad hoc and IBSS interchangeably in this thesis), there is no access point to perform such a centralized management, so a distributed algorithm is employed for timing synchronization. Basically, timing information is exchanged through periodically transmitted beacon frames in every STA. Upon receiving a beacon frame, an STA shall adopt the timing information if the received time is later than its own TSF timer.

5.2.2 Beacon Generation Procedure in Ad Hoc Networks

STAs expect to receive beacons at a nominal rate. The interval between beacons is defined by the aBeaconPeriod parameter of the STA. This value, established by the station that initiates the IBSS, defines a series of Target Beacon Transmission Times (TBTTs) exactly aBeaconPeriod time units apart. Time zero is defined to be a TBTT. Figure 5.1 shows beacon frame timing in an IBSS. An STA sending a beacon shall set the value of the beacon's timestamp so that it equals to the value of the STA's TSF timer at the time when transmitting the first bit of the timestamp to the air. This TSF timer equals to the time when the first bit of the timestamp is transmitted to the PHY plus the transmitting STA's delay through its local PHY from the MAC-PHY interface to its interface within the wireless medium.

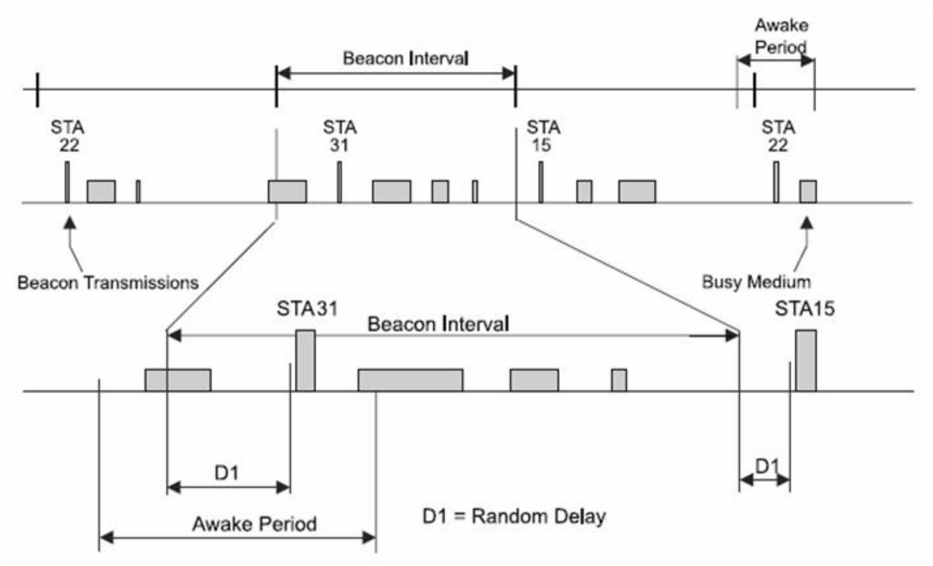

Figure 5.1: Beacon frame timing in an IBSS

Beacon generation in an IBSS is distributed. The beacon period is included in Beacon and Probe Response frames, and STAs shall adopt that beacon period when joining the IBSS. All members of the IBSS participate in the process as follows.

Beacon Generation and Clock Synchronization:

1. At each TBTT each STA calculates a random delay uniformly distributed in the range between zero and $2 * aCWmin * aSlotTime$.
2. The STA waits for the period of the random delay.
3. If a beacon arrives before the random delay timer has expired, the STA cancels the pending beacon transmission and the remaining random delay.
4. When the random delay timer expires, the STA transmits a beacon with a timestamp equal to the value of the station's TSF timer.
5. Upon receiving a beacon, a STA sets its TSF timer to the timestamp of the beacon if the value of the timestamp is later than the STA's TSF timer.

5.2.3 Beacon reception procedure in ad hoc networks

On receiving a beacon frame with a matching SSID and with the IBSS subfield of the Capability field set to 1, STAs should compare the Timestamp field with its own TSF time. If the Timestamp field of the received beacon frame is later than its own TSF time, the STA shall discard the beacon frame it schedules currently, and adopt all the parameters contained in the received beacon frame.

Chapter 6

Simulations

6.1 Introduction to ns-2 Software

ns-2 (Network Simulator-2) is a discrete event and object oriented network simulator. ns-2 uses a full set of C++ class libraries to implement networking protocols and data link layer models, and with the instances of the classes we can set up the model of the whole wired and wireless network, including every detailed aspect. ns-2 is a good tool to have deeper understanding of existing networking protocols, and furthermore to help researchers and engineers to speed up the process of developing new networking protocols and simulating the behavior of the networks. As opposed to the corresponding commercial simulation tools such as OPNET and QualNet, ns-2 is a free and open-source software. It provides all the source codes, and is widely used within the academic community as a reliable simulation tool.

6.1.1 Assumptions of ns-2 Wireless Networks

ns-2 is a packet-level simulator and essentially a centric discrete event scheduler to schedule the events such as packet reception and timer expiration. Events are handled one by one instead of accurately emulating at the same time as in the real world. However this is not a serious problem in most network simulations, because the events here are often of very short time. Also notably, to simulate wireless scenarios, there are two assumptions to simplify the physical world:

1. Nodes do not move significantly over the duration of time they transmit or receive a packet. This assumption holds only for mobile nodes of high-rate and low-speed. Consider a node with the sending rate of 10 kbps and moving speed of 10 m/s, during its receiving a packet of 1500 B, the node moves 12 m. Thus the surrounding can change significantly and cause reception failure.

2. Node velocity is insignificant compared to the speed of light. In particular, none of the provided propagation models include Doppler effects, although they could.

3. Wireless stations use omni-directional antenna having unity gain, and can only communicate to each other within the range of 250 m, and the received signal strength is the same within this range.

4. There is no noise or fading effects during the communication between stations.

6.2 ns-2 and Split Object Model

To ensure both the configuration flexibility and the efficiency, ns-2 adopts the OTcl/C++ split object model to give C++ and OTcl the ability to interpret and operate the data defined by each other, and to link the C++ classes and OTcl classes. C++ is used to perform per packet action, OTcl is used for periodic or triggered action, and TclCL is used as C++ and OTcl linkage.

The reason why using two programming language to implement ns-2 is that the simulator has two different kinds of work it needs to do. On one hand, detailed simulations of protocols requires a system programming language which can efficiently manipulate bytes, packet headers, and implement algorithms that run over large data sets. For these tasks run-time speed is important and turn-around time (run simulation, find bug, fix bug, recompile, re-run) is less important.

On the other hand, a large part of network research involves slightly changing parameters or configurations, and quickly exploring a number of scenarios. In these cases, iteration time (change the model and re-run) is more important. Since configuration runs once (at the beginning of the simulation), run-time of this part of the task is less important.

NS meets both of these needs with two languages, C++ and OTcl. C++ is fast to run but slower to change, making it suitable for detailed protocol implementation. OTcl runs much slower but can be changed very quickly (and interactively), making it ideal for simulation configuration. NS (via TclCL) provides glue to make objects and variables appear on both languages.

Network components in ns-2 are node, link, queue, etc. Network components can be created from the corresponding C++ classes, or composed of multiple simple C++ classes. For example, link components are composed of delay and queue components. In general, all network components are created, plugged and configured from Tcl.

6.3 802.11 Implementations and Modifications in ns-2

In this section, the original CMU wireless extension of the ns-2 is studied, and then Mike Moreton's change of the 802.11 code of ns-2 which gives the 802.11e function of prioritized medium access is studied and used in the simulations.

The wireless extension of ns-2 derived from CMU Monarch Project is used to simulate pure 802.11 ad hoc network. It is developed at the University of California at Berkeley and extended at Carnegie Mellon University to simulate wireless networks.

The physical layer of the 802.11 implementation is provided by channel.cc and wireless-phy.cc. The class 'Channel' is to deliver packets from a wireless node to its neighbours within the sensing range, which is defined as 250 m in ns-2. The class 'WirelessPhy' is to send packets to 'Channel' and receive packet from 'Channel'. The MAC sublayer class 'mac-802_11' has two functions to perform CSMA/CA based medium access and SIRthreshold based reception. MAC timers are used to trigger channel access. The BackoffTimer and DeferTimer give basic functions of the medium access. IFTimer sets interface state active when transmitting and NavTimer is set by RTS or CTS to indicate the residual time of data transmission. RxTimer indicates the completion of incoming packets, while TxTimer indicates sending timeout. The recv() function is generally the entry of most network protocols, and for outgoing packets, they exit from the send() function.

The 802.11e EDCA implementation in ns-2 is developed by Mike Moreton from Synad Technologies Ltd in 2003. It replaces the original mac-802_11.cc in ns-2 with new 802.11e MAC layer access functionalities, and moves the corresponding physical layer functionalities, such as carrier sense, transmission and reception of frames, out to two separate files. This abstraction provides an interface between PHY and MAC layers, and helps to analyze impacts on the MAC layer protocol parameter changes. The aim of this thesis is to simulate 802.11e EDCA in random environment which corresponds to the environment in the real world. And to see the effect of 802.11e EDCA, the same simulation is run with 802.11 legacy as well so that the results can be easily compared.

The simulation in this thesis is based on ns-2.26 and Mike's modification to its MAC layer wireless module. The network layer protocol OLSR is selected for the simulations, and the OLSR patch UM-OLSR v.0.8.7 developed by Franscisco J. Ros [7] for ns-2 is used and modified to be compatible to ns-2.26. Also, to simulate the impact of webpage surfing, the random HTTP traffic generating patch NsWeb developed by Joerg Wallerich [8] is used and modified with the same purpose.

In the simulation scenarios, there are 30 nodes which constitute a medium size MANET to have a reasonable network load. According to Section 4.1 of [4], in an ad hoc network with $n >> 1$ nodes and homogeneous node density, the transmission range of each node r_0 can be calculated from the equation below:

$$r_0 \geq \sqrt{\frac{-\ln(1-p^{1/n})}{\rho\pi}}$$

where p is the isolation probability which means no node in this ad hoc network is isolated with a probability of at least p, n is the number of nodes, and ρ is the node density. Let n=30, p=99%, A=node movement area, and knowing that r_0=250m in ns-2 implementation, we have

$$\rho = \frac{n}{A} \geq \frac{-\ln(1-p^{1/n})}{r_0^2\pi}$$

So we got $A \leq \frac{nr_0^2\pi}{-\ln(1-p^{1/n})} = 736310\ m^2$

Assuming rectangular simulation area, the length and width of the simulation area is set to $900\ m\ *\ 800\ m\ =\ 720000\ m^2$ to make it easier to read and simulate.

To simulate random environment, two models are used to generate random node movement and originate random CBR connection which simulate random voice call duration in the real case. The random node movement is generated using the setdest function version 2, with normal distributed node speed within [1,30] m/s and uniform distributed pause time within [0,10] second. The minimum speed is chosen to be positive in order to avoid the unsteady state reached by the Random Waypoint model which is examined in [9]. The random CBR connection is generated by the model in [5], with the packet size 180 byte and data rate 64 kbps according to G.711 audio companding standard used in telephony [6] to simulate VoIP (Voice over IP) traffic. The packet size used in G.711 codec is 160 byte. We use 180 byte instead because ns-2 does not add RTP header (12 byte) and UDP header (8 byte) in the transport layer. In this model, the calls are generated between two randomly selected nodes and the call time is exponential distributed with an average value of 30 seconds.

The nodes send and receive three different kinds of data: random VoIP traffic, randomly generated HTTP downloads and background FTP downloads. The network traffic is classified into high-priority traffic and low-priority traffic. In the real case the VoIP traffic is very time-sensitive so it is defined as high-priority, and the HTTP and FTP downloads are defined as low-priority.

The simulation output is a trace file, which has all the detailed information of each packet. An awk script is used to analyze the trace file and calculate the simulation results. The basic functionality of awk script is to search specific patterns in the trace file line by line. When there is a matched pattern in one line, the script takes specific actions and continues to search until the end of the file. Thus it is a good tool for analyzing ns-2 trace files.

Since we were going to run several simulations with different scenario parameters and traffic patterns to get stable average simulation results, we needed to create a perl script which runs multiple simulations with proper parameters. We created a start script where we set the number of simulations to 10 that we were going to run and the simulation time to 500 for each simulation. Then we let a loop run all the ns-2 simulations, with different scenario and traffic pattern for each simulation. When each simulation is completed, the awk script is called to analyze the trace file from the simulation. This awk script calculates the average delay time, the jitter, the delivery ratio and the throughput for both high priority (HP) and low priority (LP) packets. We also calculate the number of dropped high- and low priority packets. These results were calculated like this:

1. For all the packets the delay was calculated as the difference between the time a packet was received by the destination and the time it was sent by the source.

2. The jitter was calculated with the following formula

$$jitter = \frac{sum_{n=0}^{nbrPackets-1}(avgDelay - delay[n])^2}{nbrPackets - 1}$$

   where $nbrPackets$ is the total number of packets, and $delay[n]$ is the delay time for packet number n.

3. The delivery ratio was calculated as the number of received packets divided by the number of generated packets.

4. The throughput was calculated with the following formula

$$throughput = \frac{nbrReceivedBytes \times 8}{simulationTime \times 1024} kbps$$

5. The number of dropped packets was calculated as the number of packets that did not reach the receiver.

All these results were stored in a text file for further processing. When all the simulations were completed, the text file contained one row with results from each simulation. This text file was then processed by another awk script to get the final results.

In the second awk script, the average delay time, the average throughput, the average delivery ratio, the average jitter were calculated by averaging the corresponding number in the output text file of the first awk script.

6.4 Simulation Results

In the first scenario, randomly generated VoIP traffic and 10 HTTP connections are scheduled. As described above, the random VoIP is to simulate the voice call and is given AC_VO, the highest priority. The HTTP traffic is given AC_BE as the LP traffic. The simulation is run in both 802.11 legacy and 802.11e EDCA environment, and the average delay, average throughput, delivery ratio, and average jitter is calculated and analyzed from the ns-2 trace file generated by the simulation. The simulation time is set to be 500 seconds and 10 simulations are run for each case. This is chosen by doing trial simulations and increasing the simulation time, until the analytical results reach a stable level.

Figure 6.1 shows the comparison between 802.11e EDCA and 802.11 legacy with random VoIP traffic and 10 background HTTP connections. For HP packets, the average delay is largely shortened by EDCA, and the average throughput, jitter and delivery ratio are improved. Correspondingly, for the LP packets, we have worse performance for 802.11e EDCA compared to 802.11 legacy. It validates that the AC setting gives higher priority to the VoIP traffic and lower priority to the HTTP traffic.

But it is quite common that some people download files while other people have a voice chat. In the second scenario, let us take the example that two FTP connections are established in this MANET together with 10 HTTP connections, and see the performance difference between 802.11e EDCA network and 802.11 legacy network. The FTP downloads exist throughout the whole simulation period between randomly selected nodes in the 802.11e EDCA network. They are not as time-sensitive as either the voice call or the HTTP web surfing, so it is given the lowest priority, AC_BK.

Figure 6.2 shows the comparison between 802.11e EDCA and 802.11 legacy with random VoIP traffic, 10 background HTTP connections and 2 background FTP connections. Because the LP traffic load is largely increased, we see that the HP performance gain of EDCA is even larger compared to the first scenario. In 802.11 legacy, all nodes have the same priority, the higher the traffic load, the lower the performance for all nodes. When using EDCA, the LP traffic competes for network access in their own LP domain, so that the LP performance decreases as the LP traffic load increases, but the HP traffic still has good performance and wins even larger than in 802.11 legacy.

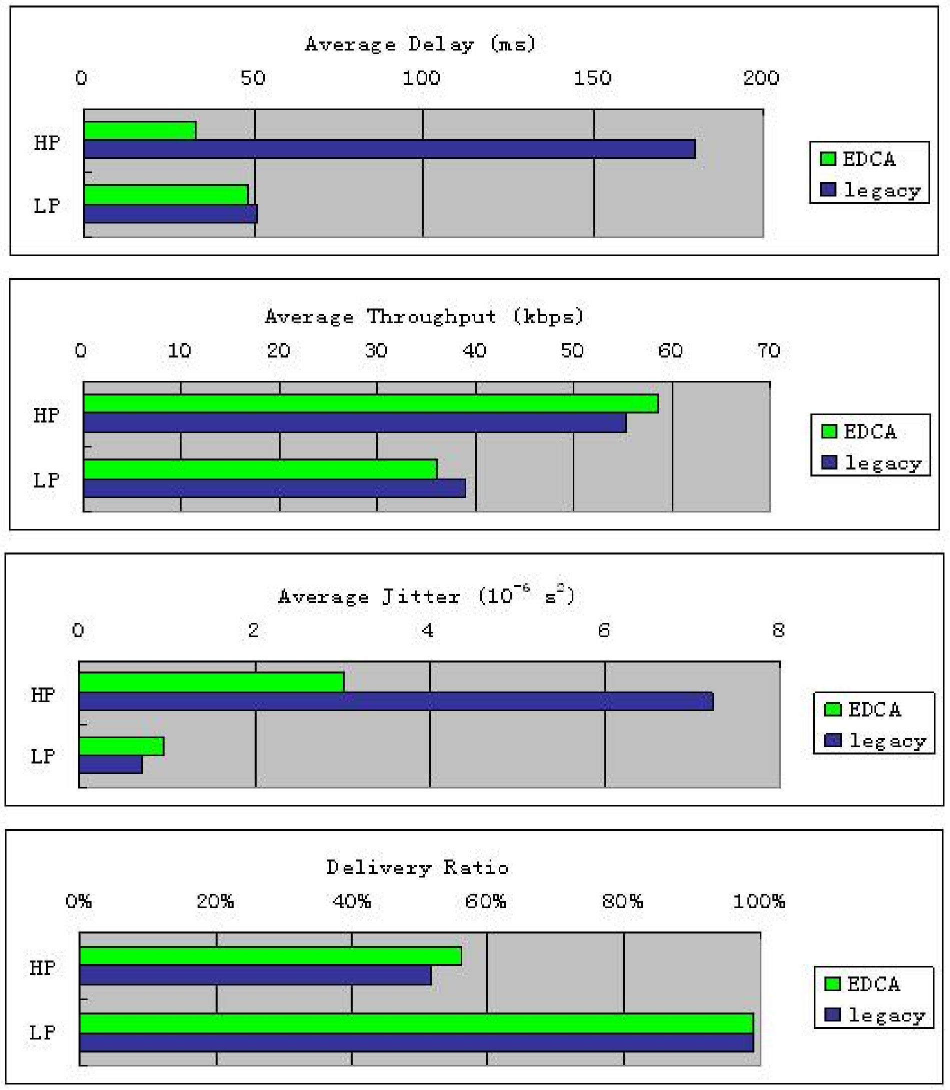

Figure 6.1: Comparison between 802.11e EDCA and 802.11 legacy with random VoIP traffic and 10 background HTTP connections

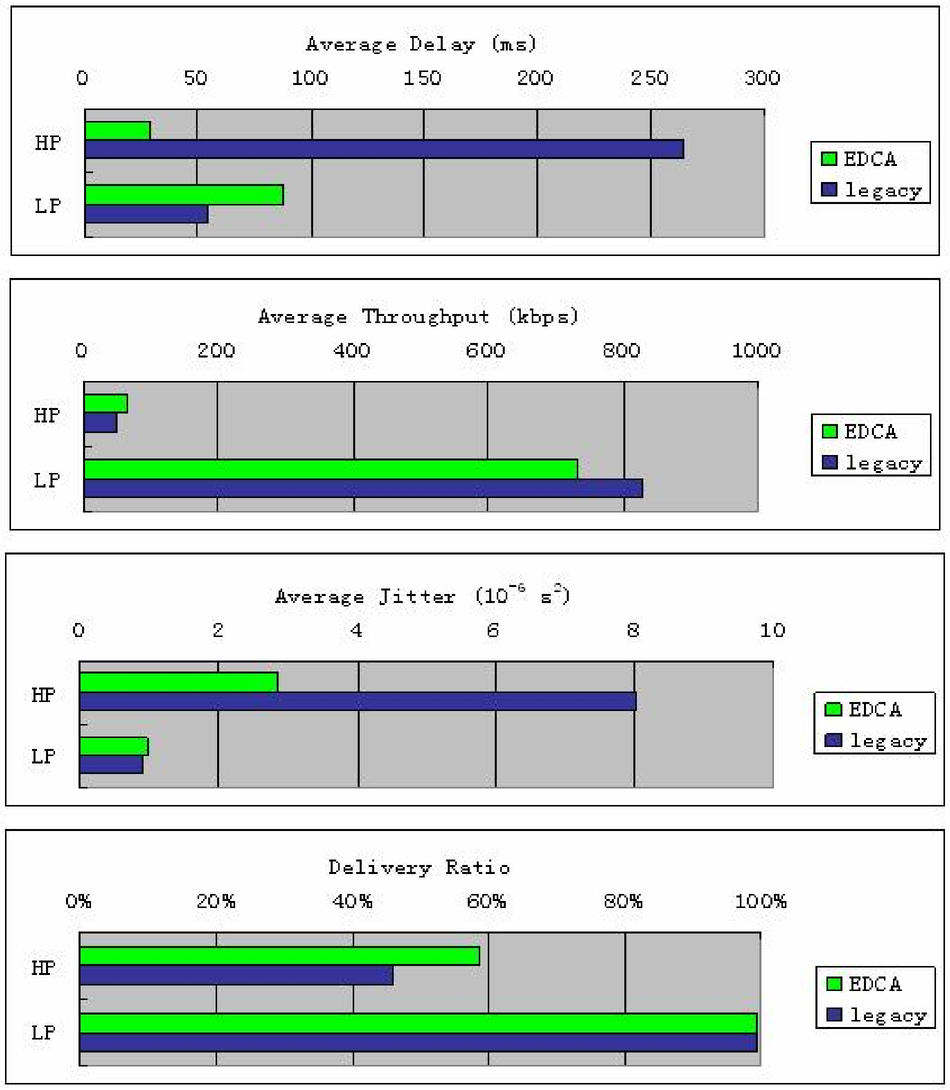

Figure 6.2: Comparison between 802.11e EDCA and 802.11 legacy with random VoIP traffic, 10 background HTTP connections and 2 background FTP connections

The delivery ratio of HP packets is about 60%, and the delivery ratio of LP packets is nearly 100%. HP packets are sporadically transmitted between two randomly selected nodes. If the two nodes could not reach each other, then all the packets will be dropped for this connection, and the number of lost HP packets is a considerable amount compared to all the HP packets that are scheduled in one simulation. LP packets are transmitted as background traffic throughput the whole period of a simulation with a much higher average throughput. So the number of lost LP packets is only a small portion of the number of scheduled LP packets in one simulation. That is why the delivery ratio of HP packets is less than the delivery ratio of LP packets.

Figure 6.3 shows the throughput over time in one simulation. When HP starts to transmit, the LP traffic fluctuates a lot to guarantee the transmission of the HP packets. The throughput of HP is almost constant during its transmission. Moreover, from the delay comparison in Figure 6.1, the average delay of HP traffic using 802.11e EDCA is much smaller than in 802.11 legacy, which means that people will have much better voice quality with the help of 802.11e EDCA.

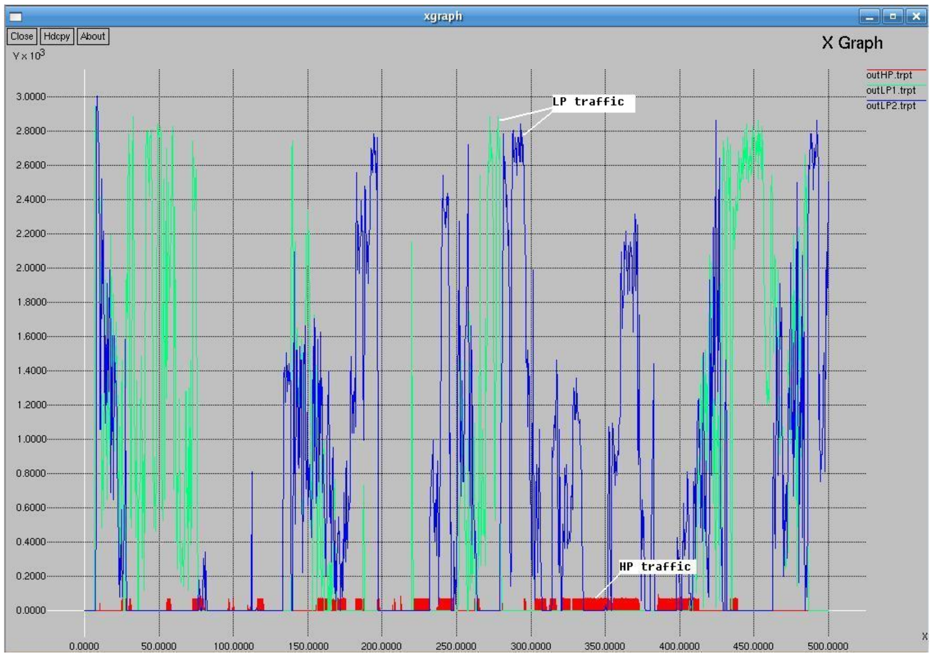

Figure 6.3: Throughput over time showing random HP traffic and 2 LP FTP connections

In order to see the impact of the increasing load of the LP traffic, we vary the number of FTP downloads and HTTP connections and compare the simulation results which are shown in Figure 6.4 and Figure 6.5.

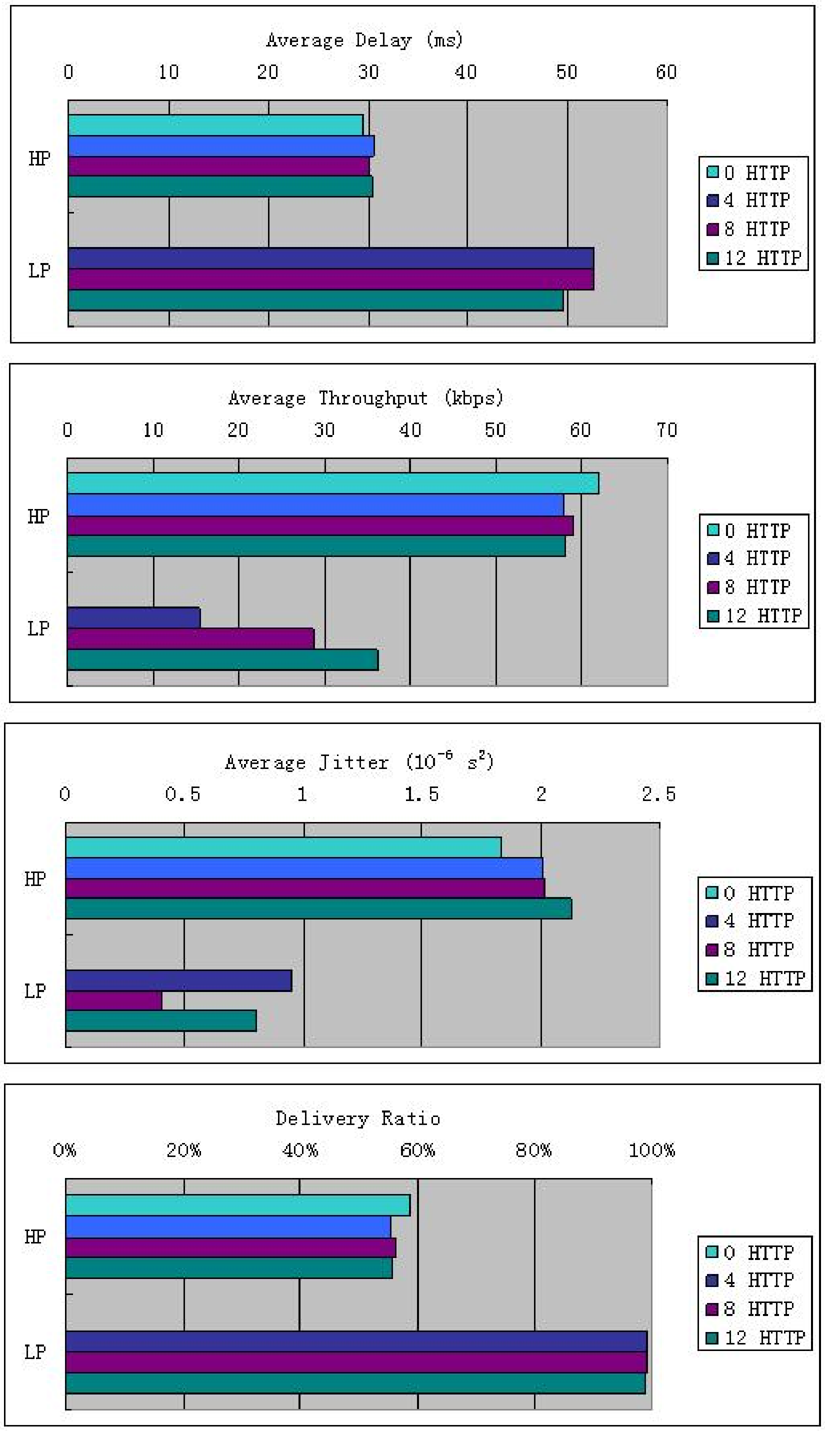

Figure 6.4: 802.11e EDCA performance analysis with different number of HTTP connections

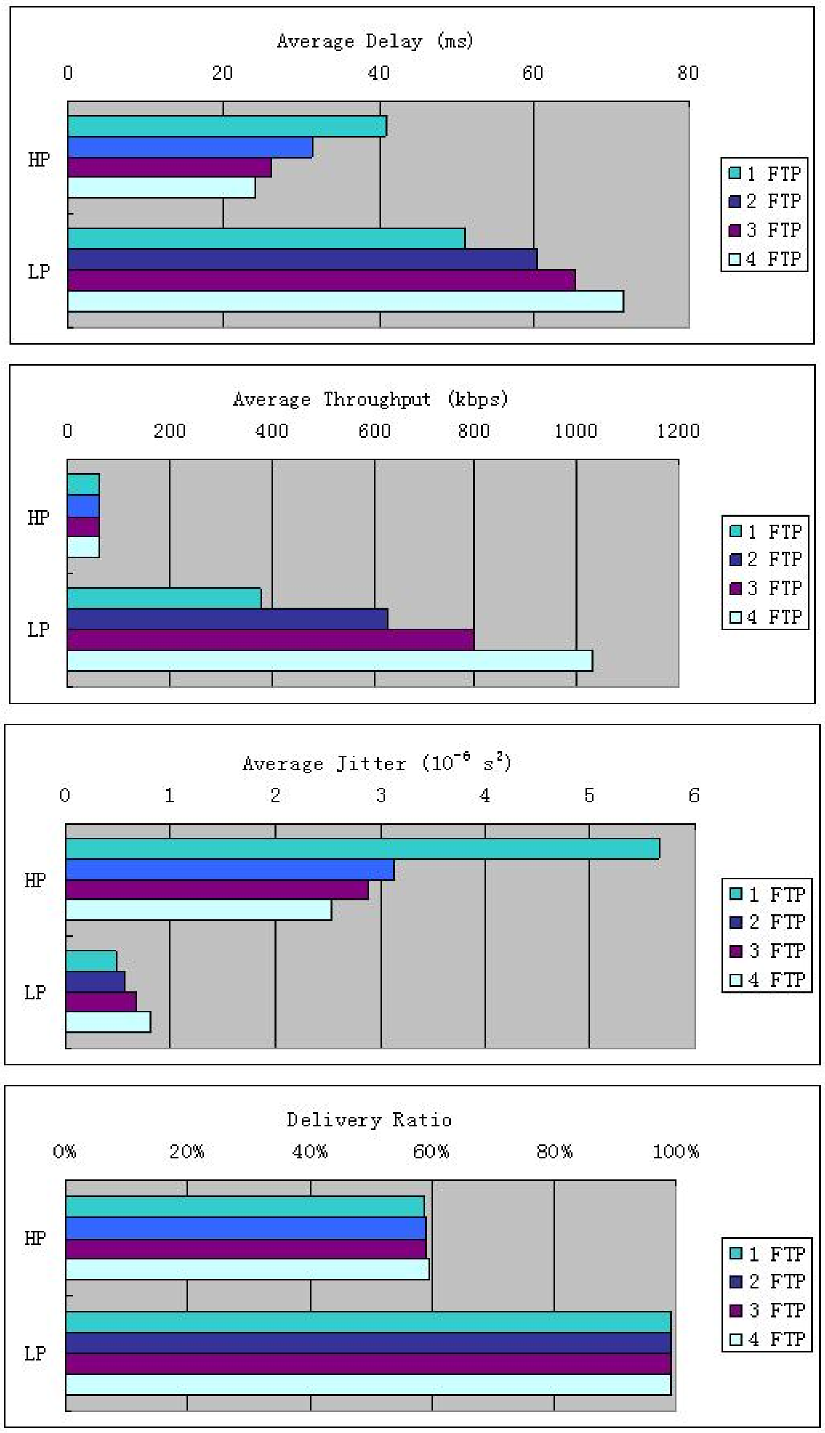

Figure 6.5: 802.11e EDCA performance analysis with different number of FTP downloads

From Figure 6.4, we can see that the HP and LP performance remains at the same level as the number of HTTP connections increases. That is because both the Voice and HTTP are randomly generated in the random node movement environment, so the HP and LP had hard time to meet each other. And also, the LP traffic load is low, thus EDCA will not give much advantage to HP traffics.

From Figure 6.5, we can see that the average HP delay and jitter improves as the number of FTP connections increases, and the average throughput slightly increases but remains at the same level. In an IEEE 802.11b network, CWmin and CWmax is set to 31 and 1023 by default. Each time a packet is lost, a new Contention Window (CW) is calculated by

$$CW_{new} = (CW_{old} + 1) * 2 - 1$$

until it reaches CWmax. From Table 4.2, we can calculate that CWmax for AC_BK is 1023, and for AC_VO is 15. As the number of FTP downloads increases, more and more packets are lost due to the increasing traffic load. But when the CW for HP traffic reaches CWmax, the increasing number of lost packets stop affecting it, while the CW of LP traffic can still increase as the traffic load increases and gives even higher backoff time for LP packets. That means the increasing LP traffic load affects LP itself much more negatively than the HP traffic. That is why the performance of HP traffic improves as the number of FTP downloads increases.

Chapter 7

Conclusions

The IEEE 802.11e EDCA improves the overall performance of the HP traffic in the MANET by introducing a MAC layer access control mechanism, and will give better end-user's sense quality if the high-priority traffic is voice data call or even video call. By simulating the random call generation in MANET at the same time of different number of FTP and HTTP connections, the average delay, throughput, jitter and delivery ratio of both HP and LP traffic are analyzed, validating the HP performance gain achieved by IEEE 802.11e EDCA.

But EDCA is still not perfect. The EDCA algorithm is also based on random access technology, and cannot guarantee a quality level for high-priority traffic, for example how much minimum throughput it must provide or how much maximum delay it has to introduce during the connection. It still needs further research and simulation work to provide real QoS to IEEE 802.11 wireless networks.

Voice and video call quality is a very interesting field and much have yet to be accomplished by increasing the bandwidth of both wired and wireless connections. But the bandwidth is very hard to be greatly improved because of the interference and fadings, and also more and more developed application level service consumes lots of the network resources. So it is really critical to have real QoS to give differentiated service and guarantee the performance in cooperation of both MAC layer and network layer functions. It will be interesting to see where the research will take in this area in the future.

References

[1] ANSI/IEEE Std 802.11, "Part11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", 1999

[2] ANSI/IEEE Std 802.11e, "Part11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements", November 2005

[3] A. Hamidian, "Internet Access and QoS in Ad Hoc Networks", Licentiate thesis, Department of Communication Systems, Lund University, April 2006

[4] C. Bettstetter, "On the Minimum Node Degree and Connectivity of a Wireless Multihop Netweork", The ACM Symp. on Mobile Ad Hoc Netw. and Comp. (MobiHoc), June 2002

[5] P. Tran and C. Wibom, "Simulation and Analysis of a Wireless Ad Hoc Network using DYMO", Master's thesis, Department of Communication Systems, Lund University, June 2007

[6] International Telecommunication Union Telecommunication Standardization Sector (ITU-T), http://www.itu.int/rec/T-REC-G.711/e, November 1988

[7] F. J. Ros, "UM-OLSR documentation", http://masimum.dif.um.es/um-olsr/html/index.html

[8] J. Wallerich, "NsWeb documentation", http://www.net.in.tum.de/j̃w/ nsweb/

[9] J. Yoon, M. Liu and B. Noble, "Random Waypoint Considered Harmful", 21st Ann. Joint Conf. IEEE Computer and Comm. Soc. (Infocom), April 2003, pp. 1312-1321

[10] P. Ferre, A. Doufexi, A. Nix and D. Bull, "Throughput analysis of IEEE 802.11 and IEEE 802.11e MAC", Wireless Communications and Networking Conference, 2004, pp. 783-788

[11] A. Banchs, X. Perez-Costa and D. Qiao, "Providing throughput guarantees in IEEE 802.11e wireless LANs", 18th International Teletraffic Congress, Semptember 2003

[12] H.M. Liang, C.H. Ke, C.K. Shieh, W.S. Hwang and N.K. Chilamkurti, "Performance Evaluation of 802.11e EDCF in Infrastructure Mode with Real Audio/Video Traffic", International conference on Networking and Services, 2006, pp. 92-92

[13] Y. Lin and V. Wong, "Saturation Throughput of IEEE 802.11e EDCA Based on Mean Value Analysis", IEEE Wireless Communications and Networking Conference (WCNC), April 2006

[14] L. Xiong and G. Mao, "Saturated throughput analysis of IEEE 802.11e EDCA", Computer Networks, August 2007, pp. 3047-3068

[15] J. Majkowski and F. C. Palacio, “Coexistence of IEEE 802.11b and IEEE 802.11e Stations in QoS Enabled Wireless Local Area Network”, Communication Systems and Applications (CSA), 2006

[16] M. P. da Silva and C. B. Westphall, “Performance Analysis and Service Differentiation in the MAC SubLayer of. IEEE 802.11e Ad Hoc Networks”, Advanced Industrial Conference on Telecommunications/Service Assurance with Partial and Intermittent Resources Conference/E-Learning on Telecommunications (AICT/SAPIR/ELETE), 2005, pp. 434-440

[17] K. Kosek, M. Natkaniec, L. Vollero and A. R. Pach, “Simulation Study of 802.11e in the Presence of Hidden Terminals”, Mediterranean Ad Hoc Networking (MedHocNet), June 2007

[18] D. Vassis and G. Kormentzas, “Delay Performance Analysis and Evaluation of IEEE 802.11e EDCA in Finite Load Conditions”, Wireless Personal Communications, July 2005, pp. 29-43

Appendix A

Program listing

A.1 simulation.tcl

```
set val(x) 900
set val(y) 800
set val(nbrOfNodes) 30
set val(endTime)  [lindex $argv 0]
set val(scen)  [lindex $argv 1]
set val(trap) [lindex $argv 2]

set val(nsweb) "HTTP_on"
set val(record) "on"

#if NSWEB is installed
CMUTrace set long_format_ 0

#
#Create a simulator object
#
set ns_ [new Simulator]

#
#Define different colors for data flows (for NAM)
#all packets are sent with color 0
#
$ns_ color 0 Blue

#
# Create trace object for NS and NAM
#
set nstrace [open sim.tr w]
$ns_ trace-all $nstrace
#set namtrace [open sim.nam w]
#$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)

#
#Set up topography object
#
set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)

#
# Create God
#
set god_ [create-god $val(nbrOfNodes)]

#
# Configure nodes
#
$ns_ node-config -adhocRouting OLSR
$ns_ node-config -propType Propagation/TwoRayGround
$ns_ node-config -llType LL
$ns_ node-config -macType Mac/802_11
$ns_ node-config -ifqType Queue/DropTail/PriQueue
$ns_ node-config -ifqLen 50
$ns_ node-config -phyType Phy/WirelessPhy
$ns_ node-config -channel [new Channel/WirelessChannel]
$ns_ node-config -antType Antenna/OmniAntenna
$ns_ node-config -topoInstance $topo
$ns_ node-config -agentTrace ON
$ns_ node-config -routerTrace ON
$ns_ node-config -macTrace ON
$ns_ node-config -movementTrace OFF
$ns_ node-config -eotTrace OFF

#Must be defined if you want to use priorities higher than one!
Node/MobileNode set numQueues_ 4

#
# Create nodes
#
for {set i 0} {$i < $val(nbrOfNodes)} {incr i} {
set node_($i) [$ns_ node]
$node_($i) random-motion 0 ;# disable random motion

set mac [$node_($i) getMac 0]
$mac BasicRateSet 1b 2b
$mac ExtendedRateSet 5.5b 11b
$mac set Short11bPreamble_ true

set mac_($i) $mac
}
#addAC <priority> <AIFSN> <CWmin> <CWmax> <TXOPlimit>
for {set k 0} {$k < $val(nbrOfNodes)} {incr k} {
    $mac_($k) addAC 0 2 7 15 3264
    $mac_($k) addAC 1 2 15 31 6016
    $mac_($k) addAC 2 3 31 1023 0
    $mac_($k) addAC 3 7 31 1023 0
}

if {$val(nsweb) == "HTTP_on"} {
$ns_ at 0.0 "$node_(0) label \"client\""
$ns_ at 0.0 "$node_(1) label \"client\""
$ns_ at 0.0 "$node_(2) label \"client\""
$ns_ at 0.0 "$node_(3) label \"client\""
$ns_ at 0.0 "$node_(4) label \"client\""
$ns_ at 0.0 "$node_(5) label \"client\""
$ns_ at 0.0 "$node_(6) label \"client\""
$ns_ at 0.0 "$node_(7) label \"client\""
$ns_ at 0.0 "$node_(8) label \"client\""
$ns_ at 0.0 "$node_(9) label \"client\""
$ns_ at 0.0 "$node_(14) label \"server\""

#
# Nsweb
#
set web [new GPool]

# initialize event logging
set log [open "sim.log" w]
$web log $log

# server, node_(14) is server node
set s1 [$web server $node_(14) $CONN_PIPELINED 5 15]

Agent/TCP/FullTcp set prio_ 2
Agent/TCP/FullTcp set fid_ 2

# client, node(0)~node_(13) is client node
set c1 [$web client $node_(0) 3 $CONN_PIPELINED]
set c2 [$web client $node_(1) 3 $CONN_PIPELINED]
set c3 [$web client $node_(2) 3 $CONN_PIPELINED]
set c4 [$web client $node_(3) 3 $CONN_PIPELINED]
set c5 [$web client $node_(4) 3 $CONN_PIPELINED]
set c6 [$web client $node_(5) 3 $CONN_PIPELINED]
set c7 [$web client $node_(6) 3 $CONN_PIPELINED]
set c8 [$web client $node_(7) 3 $CONN_PIPELINED]
set c9 [$web client $node_(8) 3 $CONN_PIPELINED]
set c10 [$web client $node_(9) 3 $CONN_PIPELINED]

# this callback procedure initiates a new request
# some time after the previous one has finished
proc session_handler { client } {
    global ns_ web u inter_request
    puts stderr "Call to session handler: [$ns_ now]"
    if { $u > 0 } {
# select next Page
set page [$web select-page]
set nextpage [$page getID]
# time for next request
set wait [$inter_request value]
set nt [expr [$ns_ now] + $wait]

puts "Scheduling next request for page \
$page/$nextpage at $nt (now + $wait)\n"
$ns_ at $nt "$client request-page $page \"session_handler $client\""

incr u -1
    } else {
# let established connections time out and become closed
$ns_ at [ expr [$ns_ now] + 30.0 ] "finish"
   }
}

# populate server
# page size generator
set pagesize_gen [new RandomVariable/UdSPareto]
$pagesize_gen set scale_ 13300
$pagesize_gen set alpha_ 1.2

# number of embedded objects per page
set embcount_gen [new RandomVariable/UdSPareto]
$embcount_gen set scale_ 2
$embcount_gen set alpha_ 1.5

# size of embedded objects
set embsize_gen [new RandomVariable/UdSPareto]
$embsize_gen set scale_ 133000
$embsize_gen set alpha_ 1.05

# create 100 pages with EmbeddedObjects on server
# using the high level method
$web populate-server $s1 100 $pagesize_gen $embcount_gen $pagesize_gen

# dump scenario to file
set dump [open "sim.dump" w]
$web dump-scenario $dump
close $dump

# set default page selection distribution
set page_gen [new AccessGenerator/Uniform 0.0 1.0]
$web set-page-generator $page_gen

# time between two requests.
# this is the equivalent to SURGE-passive-off
set inter_request [new RandomVariable/UdSPareto]
$inter_request set scale_ 10
$inter_request set alpha_ 1.5

# number of requests for this single session
set u 5000
# initiate first request
$ns_ at 6.0 "session_handler $c1"
$ns_ at 8.0 "session_handler $c2"
$ns_ at 10.0 "session_handler $c3"
$ns_ at 12.0 "session_handler $c4"
$ns_ at 14.0 "session_handler $c5"
$ns_ at 16.0 "session_handler $c6"
$ns_ at 18.0 "session_handler $c7"
$ns_ at 20.0 "session_handler $c8"
$ns_ at 22.0 "session_handler $c9"
$ns_ at 24.0 "session_handler $c10"
}

#
# Define node movement model and traffic pattern
#
puts stderr "Loading scenario file..."
source $val(scen)
puts stderr "Loading scenario file finished"
puts stderr "Loading traffic pattern..."
source $val(trap)
puts stderr "Loading traffic pattern finished"

#
# FTP 1
#
set src(12) [new Agent/TCP]
set dst(12) [new Agent/TCPSink]
$src(12) set prio_ 3
$src(12) set fid_ 3
$src(12) set packetSize_ 1000
$ns_ attach-agent $node_(28) $src(12)
$ns_ attach-agent $node_(29) $dst(12)
$ns_ connect $src(12) $dst(12)

set ftp1 [new Application/FTP]
$ftp1 attach-agent $src(12)

$ns_ at 6.5 "$ftp1 start"
```

```
$ns_ at 500 "$ftp1 stop"

if {$val(record) == "on"} {
    set trpttrace(1) [open ./outLP1.trpt w]
}

#
# FTP 2
#
set src(13) [new Agent/TCP]
set dst(13) [new Agent/TCPSink]
$src(13) set prio_ 3
$src(13) set fid_ 3
$src(13) set packetSize_ 1000
$ns_ attach-agent $node_(27) $src(13)
$ns_ attach-agent $node_(26) $dst(13)
$ns_ connect $src(13) $dst(13)

set ftp2 [new Application/FTP]
$ftp2 attach-agent $src(13)

$ns_ at 7.5 "$ftp2 start"
$ns_ at 500 "$ftp2 stop"

if {$val(record) == "on"} {
    set trpttrace(2) [open ./outLP2.trpt w]
}

#
# Define Initial Node Position in NAM
#
for {set i 0} {$i < $val(nbrOfNodes)} {incr i} {
    # 20 defines the node size in nam, must adjust it according to your scenario
    # The function must be called after mobility model is defined
$ns_ initial_node_pos $node_($i) 20
}

#
# Tell nodes when the simulation ends
#
for {set i 0} {$i < $val(nbrOfNodes)} {incr i} {
$ns_ at $val(endTime).0 "$node_($i) reset" ;
}
$ns_ at [expr .005*$val(endTime)] "puts stderr \"|||  0.5 % |||\n\""
$ns_ at [expr .01*$val(endTime)] "puts stderr \"|||  1 % |||\n\""
$ns_ at [expr .02*$val(endTime)] "puts stderr \"|||  2 % |||\n\""
$ns_ at [expr .03*$val(endTime)] "puts stderr \"|||  3 % |||\n\""
$ns_ at [expr .04*$val(endTime)] "puts stderr \"|||  4 % |||\n\""
$ns_ at [expr .05*$val(endTime)] "puts stderr \"|||  5 % |||\n\""
$ns_ at [expr .06*$val(endTime)] "puts stderr \"|||  6 % |||\n\""
$ns_ at [expr .07*$val(endTime)] "puts stderr \"|||  7 % |||\n\""
$ns_ at [expr .08*$val(endTime)] "puts stderr \"|||  8 % |||\n\""
$ns_ at [expr .09*$val(endTime)] "puts stderr \"|||  9 % |||\n\""
$ns_ at [expr .10*$val(endTime)] "puts stderr \"||| 10 % |||\n\""
$ns_ at [expr .25*$val(endTime)] "puts stderr \"||| 25 % |||\n\""
$ns_ at [expr .50*$val(endTime)] "puts stderr \"||| 50 % |||\n\""
$ns_ at [expr .75*$val(endTime)] "puts stderr \"||| 75 % |||\n\""
$ns_ at [expr .95*$val(endTime)] "puts stderr \"||| 95 % |||\n\""
$ns_ at $val(endTime).0001 "finish"
$ns_ at [expr $val(endTime)+0.0002] "puts stderr \"\nSimulation finished\" ; $ns_ halt"

#
# Define a 'record' procedure
#
proc record {} {
    global ns_ null_ dst trpttrace bw val

    #set the time after which the procedure should be called again
    set record_period 1

    #get the current time
    set now [$ns_ now]

    for {set i 0} {$i < 104} {incr i} {
     set bw(0) [$null_($i) set bytes_]
     puts $trpttrace(0) "$now [expr $bw(0)*8/$record_period/1024]"
        $null_($i) set bytes_ 0
    }

    set bw(1) [$dst(12) set bytes_]
    puts $trpttrace(1) "$now [expr $bw(1)*8/$record_period/1024]"
    $dst(12) set bytes_ 0

    set bw(2) [$dst(13) set bytes_]
    puts $trpttrace(2) "$now [expr $bw(2)*8/$record_period/1024]"
    $dst(13) set bytes_ 0

    #reschedule the procedure
    if {$now+$record_period < $val(endTime)} {
$ns_ at [expr $now+$record_period] "record"
    }
}

#
# Define a 'finish' procedure
#
proc finish {} {
global ns_ nstrace log trpttrace mac_ val
$ns_ flush-trace
close $nstrace
#Close the NAM trace file
# close $namtrace
#Execute NAM on the trace file
close $log
# exec nam sim.nam &
    if {$val(record) == "on"} {
    for {set i 0} {$i < 3 } {incr i} {
     close $trpttrace($i)
}
}

exec xgraph outHP.trpt outLP1.trpt outLP2.trpt -geometry 600x400 &

exit 0
}

if {$val(record) == "on"} {
    $ns_ at 0.0 "record"
}

puts "Starting simulation..."
$ns_ run
```