Towards a Fully Distributed QoS-Aware MAC Protocol for Multihop Wireless Networks

Fethi Filali
Institut Eurécom
E-mail: Fethi.Filali@eurecom.fr
2229 Route des Crêtes, BP-193, 06904 Sophia-Antipolis, France

Abstract— **In this paper we propose a new MAC protocol designed for multihop wireless networks called QAMP (QoS-Aware MAC Protocol). It is a distributed protocol and it supports QoS using a distributed reservation mechanism. Although QAMP could be implemented using a single channel, this paper focus on the QAMP's version using a common reservation channel and at least one data channel.**

QAMP's performance evaluation results are presented for several scenarios using analytical analysis and ns-2 simulations. We measure the saturation throughput and the delay of QAMP using the analytical analysis and based on the simulations, we show that our proposal outperforms 802.11 as it has a saturation throughput of about 97% of the physical capacity.

Keywords: multi-hop wireless networks, medium access control, quality of service, reservation-based protocols, collision-free protocols.

I. Introduction

One of the most striking changes in the use of technology in the five last years or so has been the explosive growth in the use of wireless networks for Internet and local network access. The promise of ubiquitous wireless networks dramatically enhances the usefulness of small Internet-capable devices.

One of the most important critical components of wireless networks is the lack to incorporate an efficient medium access control (MAC) protocol. Existing MAC protocols tend to optimize fairness against efficiency or vice-versa. The most commonly used Wireless LANs (WLAN) today is IEEE 802.11b with bandwidth of 2-11 Mbps [5]. In general, WLANs have a low latency of 3-100 ms and bandwidth in the range of 1-50 Mbps. WLAN uplink and downlink channels are not independent as in cellular or satellite, but compete with each other for shared bandwidth. The coverage radius of a single base station varies from tens to hundreds of meters. The link error control of 802.11b is tightly coupled with the MAC mechanism. There are at most three retransmission attempts per data frame [8]. Packet fragmentation is supported for higher efficiency of error recovery, but it is not commonly used.

The 802.11 de-facto standard suffers from high network load and it cannot support hard-QoS in infrastructureless environment because it was not design for. This paper presents an alternative **fully distributed Multiple Access Control (MAC) protocol** that supports service differentiation in multihop wireless networks. There are in a way two extreme points in designing such protocol. A centralized MAC protocol with reservation for hard-QoS support and decentralized MAC without reservation for soft-QoS support. QAMP belongs to distributed MAC protocols class but at the same time it is able to provide hard-QoS with a distributed reservation mechanism.

Reducing collision rate and maximizing network utilization are of particular importance to MAC protocols designers. Latency can be important or unimportant depending on what application is running and the node state. During a period that there is no sensing event, there is normally very little data flowing in the network. Most of the time nodes are in idle state. Sub-second latency is not important, and we can trade it off for energy savings. QAMP therefore lets nodes periodically sleep if otherwise they are in the idle listening mode. In the sleep mode, a node will turn off its radio. The design reduces the energy consumption due to idle listening. However, the latency is increased, since a sender must wait for the receiver to wake up before it can send out data. Last but not least QAMP provides a utilization rate which independent of the network load. With that being said, QAMP is designed to follow this general rule which is one of its main goals.

Although our proposal could be implemented using a single channel for reservation and data frames (the time is divided into data transmission and reservation transmission periods), we limit the discussion given in this paper to the QAMP's version with two channels: a reservation channel and a data channel[1]. Nodes that have packets to send should make reservation request in the reservation channel specifying the required number of slots and the destination

[1]It is possible to use more than one channel for data transmission.

address. To solve the well-know hidden problem, each node knows all reservation requests in its two neighborhood. After sending a packet, a node switch back to the reservation channel and receives all reservation requests sent while it was sending its data packet. Reservation control messages are sent using the 802.11 DCF basic access function in order to avoid collisions between these messages. A stop-and-wait ARQ (Automatic Repeat reQuest) mechanism is used as an error control scheme in the reservation channel.

Throughout this paper, we refer the stations which always have packets backlogged as "active stations". We also borrow the "Saturation Throughput" defined in [2] as our performance metric. The "Saturation Throughput" was defined as the "maximum throughput" the system can carry while the offered load increases.

The remainder of this paper is structured as follows. In Section II, we describe QAMP and detail its main features. We also give as illustration example showing the operations of QAMP. Performance based on analytical analysis of QAMP is provided in Section III while Section IV details some preliminaries simulation results using ns-2 through the comparison of QAMP and 802.11. Section V, concludes this paper and outlines future works.

II. QAMP

A. *Preliminaries and terminology*

Before embarking into the protocol details, it is meaningful to provide some preliminaries and the terminology that will be used to describe our protocol. In QAMP, two channels are needed:

- a reservation channel: this channel is dedicated for sending and receiving reservation requests. It is a common channel for all nodes in the wireless network.
- a data channel: at least one channel is needed to transmit data frames. QAMP's details given in this paper are based on a single data channel.

At this stage, assume that both channels have the same transmission range. Explanations on how we can configure the transmission range of the reservation channel in order to solve the hidden node problem will be given later. Note that if it is impossible, for some reasons, to use two channels, QAMP has a one-channel version which is not described in the paper.

Assume that each node[2] in the network has an unique identifier (ID) which could be the MAC address, a random value, or a value attributed by a centralized entity. The manner how node identifiers are obtained does not affect the behavior of our protocol and the optimal method to attribute them is out of the scope of this paper.

[2]Throughout this paper the terms station, node and terminal are used interchangeably.

For broadcast frames, a specific identifier known and pre-configured by all nodes. Multicast sessions identifier could be obtained from the corresponding MAC multicast address mapped from the multicast IP address according to RFC 1112 [4].

Assume an ad hoc network of n nodes and let us denote the list of reservations known by node i by $\Re_i$, $1 \leq i \leq n$. An entry in the list of reservations is denoted (S, D, T) where S is the sender ID, D is the receiver ID, and T is the number of reserved slots. Initially, this set is empty and when a node switches from active mode to sleep mode, it empties this list. QAMP's reservation mechanism described later allows only one reservation per node to appear in $\Re_i$ at the same time. The maximum size of this list is then equal to the number of stations that are in the same transmission range of given node.

Figure 1 shows QAMP components at node 2. As we can see, reservation requests information, including that of node 2, is added to the list.

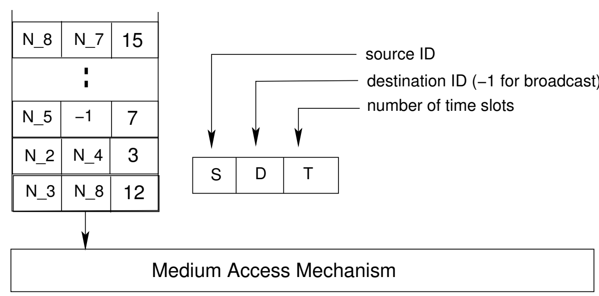

Fig. 1. QAMP components at node 2

Reservation entries are ordered according to a pre-configured scheduling policy that will be detailed in Section II-E. QAMP is independent of the scheduling mechanism to be used, however it requires that the same algorithm is used by all nodes.

Hereafter, the following terms are used:

- **authorized transmission:** refers to the data frame transmission that will happen (S, D, T).
- **authorized sender:** refers to sender of the authorized transmission.
- **authorized destination(s):** refers to destination(s) of the authorized transmission. In case of multicast and broadcast transmissions, several stations may be referred as authorized destinations.

The next sub-sections describes the reservation mechanism part of QAMP.

B. *Medium access reservation*

All control messages sent within the reservation channel use the 802.11 DCF basic mechanism in order to avoid collision between these messages.

When a node has a packet to send, it broadcasts an AREQ (Access REQuest) message via the reservation channel. In this message, the node indicates its ID, the ID of the destination station, the number of slots to be reserved. The number of slots may also include those required to send an ACK back from the destination station in case of using a reliable link-level transmission such as stop-and-wait ARQ.

Each node in the reservation channel which receives the AREQ message, adds a new reservation entry to its list of reservations after analyzing AREQ fields. On the other hand, only the node with the lowest ID[3] among nodes which are sources or destinations of the entries in the list of reservations $\Re$ sends an AREP (Access REPly) message back to the requesting node. At this stage, we assume that this list is up to date. We will explain later how a node updates its list of reservations using LREQ and LREP messages. This method prevents the well-known feedback implosion problem and ensures a reliable transfer of reservation messages. If the list is empty, the requesting node does not wait for an AREP message.

In case when the requesting node does not receive an expected AREP message during a `AREPTimeOut`, it assumes that its AREQ message was lost and resends a new reservation request message. The maximum number of retransmissions is limited to `AREQMaxRetry`. This value is bounded by taking into account the number of slots reserved by the node sending its frame in the data channel. The requesting node switches back to the data channel even if it did not receive an AREP. When it switches back to the reservation channel, it retries to send its reservation request.

Based on reservation requests that have been sent, active nodes know the duration of each frame transmission. They stay in the reservation channel during the transmission of the current data frame and switch back to the data channel at the expected end time of the authorized transmission.

If the authorized source starts sending its frame, others (except the authorized destination(s)) switch immediately to the reservation channel, otherwise they wait for a maximum time of IFAS (Inter Frame Access Space) slots and the new authorized source is allowed to send its frame. If an authorized transmission is detected or it does not happened during IFAS, its reservation entry is deleted from the $\Re$ list of all active stations. Note that some nodes may not be able to hear data frames sent by the authorized node because they have been down or they moved so that they become out of range of the authorized node. The same procedure is applied recursively until an authorized transmission takes place or the list of reservations becomes empty. In both cases, all active stations switch immediately to the reservation channel.

An example showing the exchange of control and data frames is given in Figure 2.

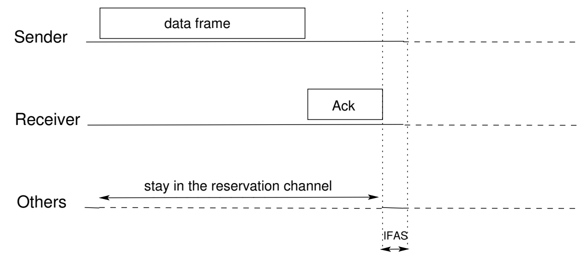

Fig. 2. Exchange frames between source and destination

When an authorized transmission successively ends[4], the authorized source and the destination(s) remain in the data channel until sensing that a new transmission starts or that the list of reservations becomes empty. After that, if all of them are neither the source nor the destination of the new authorized transmission[5], they have to switch to the reservation channel in order to update their lists of reservations by taken into account the reservations requests sent while they were in the data channel. They will have also the opportunity to send their waiting reservations requests.

C. Updating the list of reservations

To update its list of reservations (for example after transmitting or receiving a data frame), a node broadcasts a LREQ (List REQuest) message which contains the last reservation request that it has received. The node with the lowest ID among those in the reservation list (expect the source or the destination(s) of the last frame that has been sent) sends back an LREP (List REPly) message containing the reservations requests that this node was asked for.

Note that a station may not know in advance that it will be the destination of the data frame to be sent by the authorized node. This situation may occur if this station does not receive the corresponding reservation request because it was receiving or sending the previous data frame(s) which occupied the data channel.

It may also happen that frames are not sent according to their arrival times, even when using a FIFO scheduler, in the

[3]Others techniques could be applied to choose the node that should send an AREP message. The lowest ID method is used by several Internet protocols such as IGMP (Internet Group Management Protocol).

[4]Assume that ARQ stop-and-wait is used at the MAC layer for reliable transfer, this means that the source station receives an ACK from the destination station

[5]It is possible to not allow the new transmission if the destination is the same as that of the old authorized transmission when there is no reservation entry in the list where it is the source and it has a frame waiting to be sent. This policy allows a fair share of bandwidth between competing stations and prevent that a node is flooding with other nodes and it cannot send its own frames.

case for example if some reservation requests were delayed because the sender was sending or receiving a frame when its upper layer sent a packet to the MAC layer. On the other hand, other nodes in the reservation channel could send immediately their reservation requests of new data frames sent by the upper layer to the MAC layer during the occupation of the data channel by another node.

D. Initial startup

A node becomes active when it has a packet to send or to receive. When a node becomes active it senses the data channel. If it is busy for more than IFAS Time, this node sends its frame without sending a reservation request because it assumes that there is no concurrent transmission. Otherwise, it switches to the reservation channel. It sends a List REQuest (LREQ) message using the 802.11 basic DCF mechanism indicating the last reservation entry it has in its list of reservations. The node having the lowest identifier among the nodes that receive this request and they are in the reservation channel sends back a List REPly (LREP) message containing the current content of the queue. If this node does not receive a LREP after `LREQMaxRetry`, it assumes that there is no other active node and decides to send its data without waiting for more time. When a node receives a LREP message, it updates its list of reservations.

E. Access scheduling mechanism

This policy could be based on requests arrival time (i.e, waiting time to send a packet) or on QoS parameters of their waiting packets to be sent.

For QoS support, different scheduling mechanisms could be used. DiffServ-like architecture could be also supported. Buffer management mechanism such as WFQ, DRR, etc. could also be used. QAMP's operations are independent of the used scheduling mechanism, the only requirement is that the same scheme should be used in every node in the network.

Due to space limitations, we cannot provide more details about the scheduling mechanism

F. An illustration example

To understand how QAMP operates, an illustration example is given in this section. Assume four competing stations: A, B, C, and D and let us denote the initial time T_0. Assume that node A has one waiting packet to be send: (A,B,6) at $T_0 + 2$, C has one packet to send (C,D,6) at $T_0 + 3$, node D has one packet to send: (D,A,6) at $T_0 + 1$, and node B does not have any packet to send.

As shown in Figure 3, assume that at the beginning all nodes are in the reservation channel. When the MAC layer of node D receives a frame from upper layers to be sent

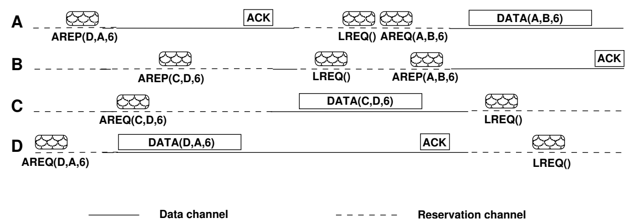

Fig. 3. An example of a four-nodes network.

to node A, it broadcasts a AREQ(D,A,6) message and it receives an AREP(D,A,6) from A which has the lowest identifier. After that nodes, all nodes switch to the data channel as a unicast data transmission between A and D will occur. Nodes C and B switch back to the reservation channel after sensing the data channel busy. After switching to the reservation channel, node C broadcasts its reservation request AREQ(C,D,6). Node B is the only node that receives this request, it sends back a AREP(C,D,6) to node C. When node A finishes sending its frame to D, nodes B and C switch back again to the data channel. At this stage all nodes are in the data channel. Node C starts sending its packet (C,D,6) to D. Only nodes A and B switch to the reservation channel to update their list of reservations as they were in the data channel. Updating the list of reservation is down by sending a LREQ message, but in this example there is no LREP message sent back to nodes A and B as there is no other node in the reservation channel. The same procedure is applied for the data transmission between nodes A and B: DATA(A,B,6).

G. MAC-level fragmentation/packing

In 802.11 MAC-level fragmentation is used to decrease the collision probability especially in high loaded networks. In QAMP we also make use of MAC-level fragmentation but not for the same purpose as 802.11. Fragmentation in QAMP could be enforced to share fairly the bandwidth between competing nodes. In other words, it could be possible that the scheduler enables frames fragmentation in order to achieve a good share of bandwidth between competing nodes. For the same purpose, assembling (packing) several frames to be sent to the same destination within the same frames is also possible.

H. Deployment issues

QAMP uses at least two channels: a reservation channel and at least one data channel. The physical layer is assumed to support efficiently the management of several channel. OFDM (Orthogonal Frequency Dynamic Multiplexing) allows to use different channels.

Another constraint is the rapid switching between channels. To reduce this time, it is possible to use multi-antenna to manage the reservation and data channels differently: one antenna is attributed for the reservation channel and the other antenna for data channel. This allows to reduce the complexity of deploying QAMP. Indeed a node could be able to send/receive a data or a reservation frame at the same time.

III. Analytical analysis

A. Average delay

Denote by $N(t)$ the number of entries in the reservation list at time t and assume the use of a FIFO scheduler. Assume, without loss of generality, that all data packets have the same size s. Let's T^s_{data} be the transmission time of a data frame of size s.

To send a data frame, a node has first to send an AREQ and wait to receive an AREP while another data frame is sent in its neighborhood.

The maximum time needed to send a reservation is $T_{res} = T_{areq} + IFAS + T_{arep}$. The necessary time to send all waiting packets is $T_{N(t)} = N(t) * (T^s_{data} + IFAS)$. The required delay to send a packet is then $D(t) = T_{res} + T_{N(t)} + T^s_{data} = (N(t)+1) * T^s_{data} + N(t) * IFAS + T_{res}$.

The maximum delay is obtained for $N(t) = N_{max}$, where N_{max} is the maximum number of neighbors that a node can have. The extreme case which is not practical is when N_{max} is equal to the total number N of nodes in the network. The maximum delay is then $D_{max} == (N+1) * T^s_{data} + N * IFAS + T_{res}$.

B. Saturation throughput

The saturation throughput is that obtained when each node has a packet to send, which means that each node has sent a reservation request. The size of the list of reservations of each node is then equal to the number of its neighbors.

The saturation throughput is defined as the ratio of the time required to send a frame over the transmission time of a frame. Let's C be the link capacity. The saturation throughput is then $\frac{T^s_{data}}{T^s_{data}+IFAS} = \frac{s*C}{s*C+IFAS}$.

In Figure 4, the saturation throughput is plotted for different values of the packet size and for link capacity equal to 11Mbps. From the figure's curves, it's obvious that even for small data frames of ten or twenty bytes, QAMP is able to achieve a good saturation throughput which is more than 85% of the link capacity. Measurement studies in the Internet have shown that more than 95% of packets have an IP packet size around 500 bytes. QAMP achieves a saturation throughput very close to 100% of the link capacity when the packet size is more than 100 bytes.

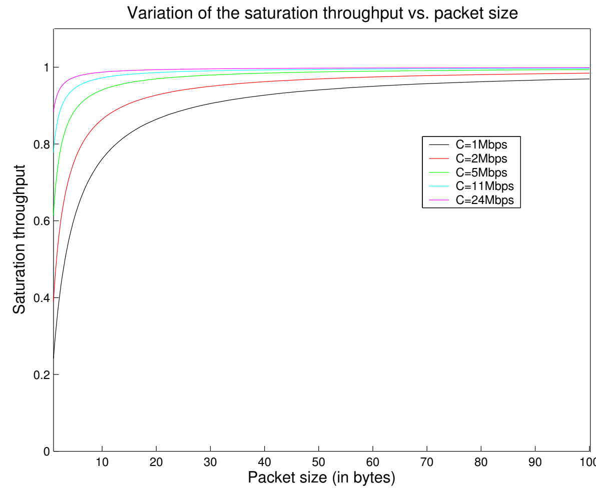

Fig. 4. Variation of the saturation throughput as a function of the packet size.

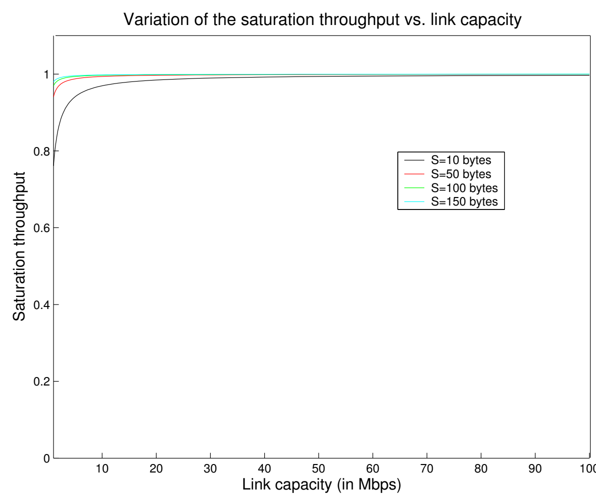

Fig. 5. Variation of the saturation throughput as a function of link capacity.

Figure 5 shows the variation of the saturation throughput as a function of the link capacity going from 1Mbps to 100Mbps. For different values of the packet size, the saturation throughput remains close to 100% of the link capacity.

IV. Preliminary simulation analysis

We implemented a first version of QAMP protocol in ns2 network simulator and we compared QAMP and 802.11 protocols performance for a basic scenario which consists of two nodes: $n1(100, 100, 0)$ and $n2(200, 100, 0)$ and we set up two CBR traffic: one from $n1$ to $n2$ and the other one from $n2$ to $n1$ in order to reflect a real situation with collisions. The simulations time was set to $50s$ and the packet size of both flows was set to 1000bytes. Both flows start at $1s$ remain alive during the rest of the simulation. The physical basic rate and data rate of QAMP and 802.11 are set to $1Mbps$ and $2.2Mbps$, respectively. The current ns2's QAMP implementation uses the same physical layer as 802.11, this allows to have an accurate comparison of both protocols.

The variation of the network utilization as a function of both CBR rates is shown in Figure 6.

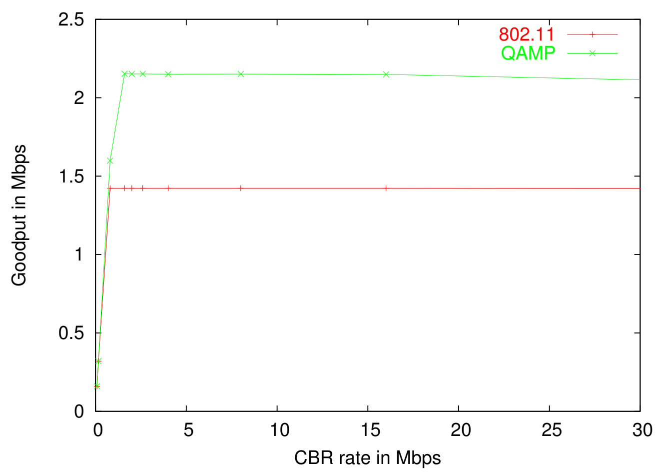

Fig. 6. Variation of the saturation throughput as a function of link capacity.

It's obvious that QAMP outperforms 802.11 in terms of the achieved goodput that defines the throughput received by the receiver. Indeed, it is able to reach a throughput equal to $2.15Mbps$ when the network is highly loaded which is around 97% of the physical capacity. This does not change even when the network load increases very much.

V. Conclusion and Future Works

In this paper, we presented QAMP (Quality of Service MAC Protocol), a medium access mechanism designed for ad hoc networks. QAMP uses a distributed reservation mechanism and efficiently avoids collisions. QAMP tries to maximize the network throughput while maintaining a fair allocation thanks to the use of two different channels: one channel for the data channel and another one for the reservation channel. In summary, QAMP's characteristics are: (1) the network utilization does not decrease with network load, (2) the fairness between competing nodes, (3) QoS support through the use of a pre-configurable QoS-based scheduling mechanism of the reservation requests.

Another QAMP-derived MAC protocol could be used which is adequate when it is impossible to have more than one channel is to use a single channel with two periodic time frames: one for reservation requests and the other one for the data frames.

Due to space limitations, different features of QAMP have not been presented as the avoiding of hidden and exposed node problems which are explicitly solved thanks to the list of reservation knowledge to two neighborhood. Moreover, mobility does not affect QAMP's performance given that the list of reservations is updated periodically using LREQ and LREP messages.

Preliminary performance analysis and simulations show that QAMP provides significant improvements compared to 802.11 MAC protocol. Future works would include more analytic results and comparison of QAMP with other MAC protocols using NS-2 simulations.

References

[1] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, An Architecture for Differentiated Service, RFC 2475, December 1998.

[2] G. Bianchi, Performance Analysis of the IEEE 802.11 Distributed Coordination Function, IEEE Journal on Selected Areas in Communications, Vol. 18, No. 3, March 2000.

[3] R. Braden, D. Clark, S. Shenker, Integrated Services in the Internet Architecture: an Overview, IETF RFC 1633, June 1994.

[4] S. Deering, Host Extensions for IP Multicasting, IETF, RFC 1112, August 1989.

[5] IEEE, Part 11, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, ANSI/IEEE Std 802.11, 1999 Edition, (ISO/IEC 8802-11:1999(E)).

[6] IEEE, Part 11, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Medium, Access control (MAC) Enhancements for Quality of Service (QoS), ANSI/IEEE Std 802.11e, Draft 5.0, July 2003.

[7] IEEE, Part 3, Medium Access Control (MAC) Bridges, ANSI/IEEE Std 802.1D, 1998 Edition, (ISO/IEC 15802-3:1998).

[8] C. H. Nam, S. C. Liew, and C. P. Fu, An experimental study of ARQ protocol in 802.11b Wireless LAN, In Proc. of Wireless Personal Multimedia Communications (WPMC'02), October 2002.

[9] M. A. Visser and M. El Zarki, Voice and data transmission over an 802.11 wireless networks, In Proc. of PIMRC'95, Toronto, Canada, September 1995.

[10] X. Yang and N. Vaidya, Pipelined Packet Scheduling in Wireless LANs, Technical Report, March 2002 (revised August 2002).