This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2007 proceedings.

Towards Optimal Multimedia Packet Bursting for IEEE 802.11 Wireless LANs

Joshua Wall, Jamil Khan
School of Electrical Engineering & Computer Science
University of Newcastle, Australia
joshua.wall@studentmail.newcastle.edu.au, jamil.khan@newcastle.edu.au

***Abstract*— The imminent IEEE 802.11n standard is poised to provide unprecedented throughput experienced at the MAC layer, whilst providing QoS support for the escalating demand in multimedia services and applications. Along with the 802.11e QoS amendment, 802.11n defines advanced ARQ policies designed to increase the channel utilization efficiency by reducing MAC protocol overhead. Although these techniques provide vast improvements over the legacy 802.11 protocol, opportunities exist within this framework to optimize performance based on the type of traffic being serviced. In this paper we investigate the performance optimization of the 802.11 Block Ack ARQ policy, and propose an adaptive block size algorithm for throughput improvement. Using a comprehensive OPNET simulation model we show that the proposed adaptive algorithm achieves higher link layer throughput and improved QoS.**

***Key Terms*— 802.11; ARQ Optimization; Block Ack; WLAN**

I. INTRODUCTION

Current and emerging broadband multimedia applications and services including voice over IP (VoIP), TV-centric home networks, high-definition digital TV (HDTV), and broadband gaming, continue to stretch the capability of current wireless LAN (WLAN) standards. In order to meet these ever increasing needs, current standardization efforts within the 802.11 work group include 802.11n- an advanced, high-rate physical (PHY) and medium access control (MAC) layer specification designed to offer PHY layer throughputs up to 600Mb/s, with in excess of 100Mb/s experienced at the MAC layer [1].

Along with the 802.11e QoS amendment [2], 802.11n defines a number of advanced automatic repeat request (ARQ) policies designed to increase the channel utilization efficiency through techniques such as packet bursting, acknowledgement aggregation (Block Ack), and the elimination of an ARQ policy all together (No Ack). The rules governing the adaptation of ARQ parameters and dynamic switching between the specified ARQ policies is out of the scope of 802.11e/n, and therefore provides an open and challenging research area. Furthermore, the use of static ARQ parameters does not often provide optimal performance for multimedia traffic classes.

Proposed techniques in this area involve the use of suitable ARQ parameters identified through extensive analysis and performance evaluation under varying conditions, as well as the dynamic adaptation of specific ARQ parameters, with a goal of benefiting the type of traffic being serviced. Li and van der Schaar [3] propose an adaptive QoS scheme for the transmission of layered video over an 802.11 WLAN network through real-time adaptation of the ARQ retry limit parameter. Yuan et al. in [4] present a high-performance MAC protocol called ADCA that employs an adaptive block transmission concept within an infrastructure WLAN (centralized control using an Access Point) to reduce overhead and increase aggregate throughput. A comprehensive analysis of 802.11e frame bursting and Block Ack policy is presented in [5,6], with Lee et al. [7] proposing an efficient block size polling scheme that selects efficient static block sizes depending on the traffic class being serviced. In [8], we presented a recipient initiated adaptive ARQ mechanism, which dynamically switched between Ack and No Ack ARQ policies based on link state information. This dynamic switching algorithm serves two main purposes. Firstly, the overhead associated with the Ack transmission is minimized, thus improving the overall throughput. The second advantage is that the algorithm offers protection against packet loss. The algorithm monitors the channel conditions and switches to the use of a No Ack ARQ policy when suitable, so as to reduce unnecessary retransmission. In this paper we propose a distributed, adaptive block size algorithm for multimedia traffic, where the Block Ack protocol's block size is dynamically adjusted based on traffic class requirements, link state information and transmission channel characteristics. We show through a comprehensive simulation that an optimized throughput can be obtained for multimedia traffic.

The rest of the article is structured as follows: Section II provides an overview of the advanced ARQ policies used in 802.11e/n. Section III describes the proposed adaptive block size algorithm. The simulation model used to analyze the proposed algorithm is presented in Section IV, with results shown in section V. A conclusion is drawn in section VI.

1525-3511/07/$25.00 ©2007 IEEE

II. 802.11 ARQ POLICIES

A. 802.11e

The 802.11e standard supports three different ARQ policies, namely Ack (traditionally stop and wait), No Ack, and Block Ack (selective repeat). The legacy 802.11 WLAN standard supports only the Ack policy, which is characterized by the transmission of a positive acknowledgement (Ack) by a recipient node to the originating node upon the successful reception of a data packet, also known as a MAC Protocol Data Unit (MPDU[1]). Acknowledgement of a MPDU occurs after a SIFS duration. If a MPDU is received in error or the transmission of the Ack is unsuccessful, the originating node retransmits the packet. If the packet has not been successfully transmitted after a predefined number of attempts, it is discarded. When operating using a No Ack policy, MPDUs are transmitted once and only once regardless of whether they are received in error or not at all.

The Block Ack mechanism defined in the 802.11e allows a group of MPDUs to be transmitted consecutively within a transmission opportunity (TXOP), while bundling all corresponding acknowledgements into single collective Ack. A TXOP is defined as a bounded duration in which a node may perform packet bursting, where a sequence of SIFS separated packets are exchanged. Block Ack is initialized by an exchange of setup frames that negotiate parameters including buffer size and Block Ack type, where the Block Ack type is either immediate or delayed. Fig. 1 shows the immediate Block Ack policy. After initialization, the originator may transmit a sequence of SIFS separated MPDUs, with the total number of MPDUs (block size) not exceeding the buffer size negotiated at the setup. If the RTS/CTS protection mechanism is not used, an Ack must be transmitted after the first MPDU in the TXOP, as illustrated in Fig. 1a. The originating node requests an Ack of outstanding MPDUs by sending a Block Ack Request (BAR) frame. If immediate Block Ack is used, the recipient responds by transmitting a Block Ack (BA) frame immediately after the BAR, which contains the acknowledgement state of data packets received from the originator. A BA frame can acknowledge up to 64 MSDUs, with each MSDU fragmented up to 16 times. Delayed Block Ack allows a node to acknowledge the BAR with an ACK, then prepare a BA response and send it in the earliest possible subsequent TXOP. The originator acknowledges the receipt of the BA with an ACK. This delayed approach is primarily intended to facilitate inexpensive implementations that use the processing power of a host, as well as to allow existing implementations to use this feature with minimal hardware changes. In the event of unsuccessful transmission, packets are retransmitted in the next available TXOP and acknowledged as soon as possible, as shown in Fig. 1c. This process continues until they are discarded from the transmitter queue upon expiry of an access category (AC) specific delay threshold called a MSDU Lifetime. A timer is started when a MSDU is first passed to the MAC layer, and if it exceeds its associated MSDU Lifetime before successful transmission, all remaining fragments of that MSDU are discarded by the originating node without any further attempt to complete delivery. This is advantageous for multimedia traffic, as if a multimedia data packet is not transmitted within a certain delay, it often exceeds its effective usefulness or ability to contribute at the receiving application, therefore consuming unnecessary bandwidth when transmitted.

[1] A MAC Service Data Unit (MSDU) is a unit of data arriving at the MAC layer from higher layers. Fragmentation partitions a MSDU into smaller MAC level frames called MAC Protocol Data Units (MPDU).

B. 802.11n

Enhancements to Block Ack currently being considered in 802.11n include implicit BAR, and compressed BA [1]. Implicit BAR involves an implicit BA request by asserting a normal Ack policy of an aggregated MPDU. A compressed variant of the 802.11e BA frame is also defined, where the bitmap used to represent the transmission state of individual MPDUs is compressed through the exclusion of fragmentation. Compression parameters are defined during the exchange of the Block Ack setup frames based on the use of fragmentation and the block size. The size of the compressed BA frame is defined by the BA agreement, and is constant throughout the lifetime of the BA agreement.

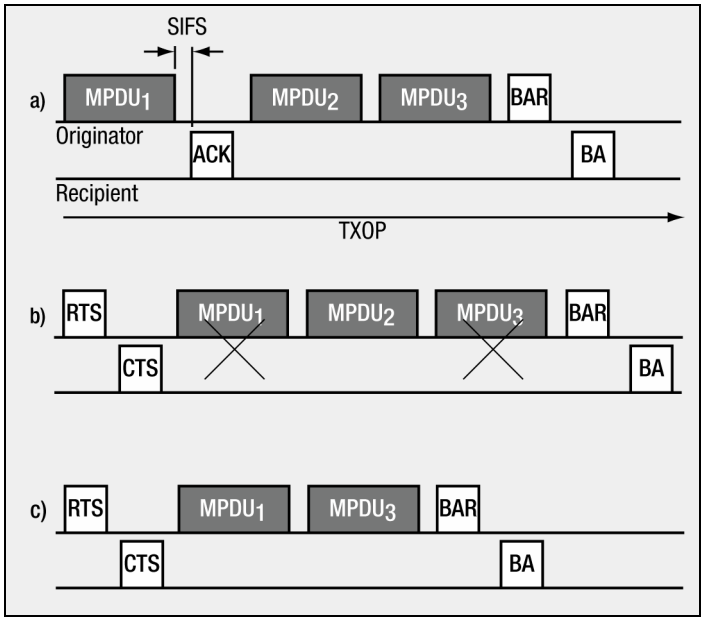

Fig. 1. Immediate Block Ack: a) No protection, b) RTS/CTS protection, c) Retransmission

III. ADAPTIVE BLOCK ACK

The proposed Adaptive Block Size algorithm, here on in referred to as ABS, aims to optimize throughput by dynamically adapting the block size (B_s), based on traffic requirements, link state information, and transmission channel conditions. We define the set of Block Ack (BA) ARQ strategies as

$$ARQ = \{BA_0, BA_1, \ldots BA_{64}, BA_\infty\} \quad (1)$$

where $BA_0 \equiv$ Ack, $BA_\infty \equiv$ No Ack, and $BA_{1..64} \equiv$ Block $\text{Ack}_{1..64}$. The ABS algorithm's aim, as expressed in equation (2), seeks to find the optimal ARQ strategy that maximizes throughput for different traffic categories i, under varying instantaneous channel conditions x = (*SNR*, *contention*).

$$ARQ^{opt}(i,x) = \underset{B_S \in [0,1,\ldots 64,\infty]}{\arg\max} \; Throughput\left(ARQ(i,x)\right) \quad (2)$$

This strategy results in the best link throughput subject to the following delay and packet loss (P_L) constraints:

$$\begin{aligned} &Delay\left(ARQ(i,x)\right) \le MSDU_{Lifetime}, \\ &P_L\left(ARQ(i,x)\right) \le P_{L\,threshold} \end{aligned} \quad (3)$$

Fig. 2 shows a timing diagram of the ABS algorithm, with Fig. 3 showing a pseudo code representation. The algorithm is an originator initiated scheme, in that no additional feedback other than that provided through normal protocol operation is used to provide feedback to the originator. This feedback along with link state information from the originator is used to intelligently steer the block size to provide optimal throughput. A packet loss rate metric is used to determine when to change B_s, calculated using information from the BA and MSDU Lifetime discard rate. Upon reception of a BA, the number of successfully transmitted MSDUs is extracted from the BA and added to a counter. Likewise, when a MSDU is discarded from the transmission queue due to MSDU Lifetime expiry, a counter representing the number of discarded MSDUs is incremented. The originator periodically calculates P_L using these values, and if found to be above or below a traffic class dependant range (Dead-band), the ABS algorithm dynamically adapts B_s accordingly. This control strategy is illustrated in Fig. 4, where B_s is increased for P_L values less than the dead-band, and reduced when P_L is greater than the dead-band.

Channel state information (CSI), such as received signal strength indicator (RSSI) or SNR, is used to determine how much B_s is changed (step size) in each increment. If the channel conditions are found to have diverged by a differential amount (SNR_DIFFERENTIAL) since the last channel assessment, the step size is changed using a binary exponential scheme, i.e. the step size used when increasing the block size is doubled upon improving channel conditions, and halved if the channel is found to be degrading. Conversely, the step size used to reduce the B_s is doubled on channel degradation, and halved on channel improvement. This is best illustrated in Fig. 2a, where B_s has an initial size of 4, and the step size is initially set to 2. If a periodic P_L calculation shows that an increase in B_s is favorable, B_s is increased by the step size to a value of 6 (Fig. 2b). If CSI feedback shows that channel conditions are improving, the step size is doubled to a value of 4, thus causing B_s to be increased to a value of 10 upon the next favorable periodic P_L calculation (Fig. 2c). As well as the normal operating bounds of B_s imposed by the Block Ack ARQ policy, an upper and lower block size limit (BS_MAX, BS_MIN) can be specified, restricting the ABS algorithm from changing the value of B_s beyond these bounds.

The originator is able to gain CSI through mechanisms such as RSSI, closed-loop modulation and coding scheme (MCS) adaptation [1], or customized feedback through control frames such as CTS, ACK and BA. As this feedback can occur more frequently than the packet loss rate calculation, the step size is used as a fine-grained control in the algorithm.

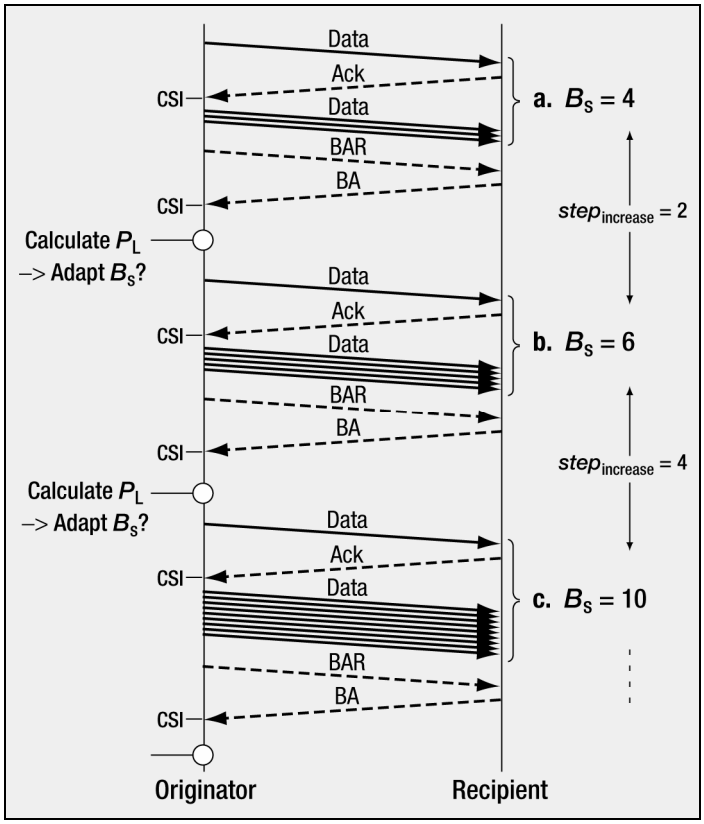

Fig. 2. ABS algorithm timing diagram

```
ON INITIALIZATION
1    P_L = 0.0
2    B_s = BS_MIN_i
3    step_increase = 2, step_reduce = 2
4    n_successful = 0, n_discarded = 0
ON BA RECEPTION
1    /* Determine no. successfully transmitted packets from BA */
2    n_successful = n_successful + BA_successful
ON MSDU LIFETIME EXPIRATION
1    /* Increment no. discarded frames counter */
2    n_discarded = n_discarded + 1
ON PERIODIC PACKET LOSS CALCULATION
1    /* Calculate Packet Loss Rate */
2    P_L = n_discarded / n_successful
3    if P_L > PL_THRESHOLD_i
4        /* Packet Loss Rate exceeds traffic class acceptability level */
5        /* Decrease Block Size */
6        if B_s - step_reduce >= BS_MIN_i
7            B_s = B_s - step_reduce
8        else
9            B_s = BS_MIN_i
10   else if P_L < PL_THRESHOLD_i – PL_DIFFERENTIAL_i
11       /* Packet Loss rate is below traffic class acceptability level */
12       /* Increase Block Size */
13       if B_s + step_increase <= BS_MAX_i
14           B_s = B_s + step_increase
15       else
16           B_s = BS_MAX_i
17   /* Reset Packet Loss variables */
18   n_discarded = 0, n_successful = 0
ON CSI RECEPTION
1    /* Use channel state information to tune block step size */
2    /* NB: A SNR metric is used for ease of simulation */
3    if snr_current < snr_previous
4        if | snr_current – snr_previous | > SNR_DIFFERENTIAL
5            /* SNR is decreasing so adjust step sizes */
6            step_reduce = step_reduce * 2, step_reduce ∈ {2,4,8}
7            step_increase = step_increase / 2, step_increase ∈ {2,4,8}
8    if snr_current > snr_previous
9        if | snr_current – snr_previous | > SNR_ DIFFERENTIAL
10           /* SNR is increasing so adjust step sizes */
11           step_reduce = step_reduce / 2, step_reduce ∈ {2,4,8}
12           step_increase = step_increase * 2, step_increase ∈ {2,4,8}
13   /* Record current SNR for use in next iteration */
14   snr_previous = snr_current
```

Fig. 3. Pseudo code of ABS algorithm

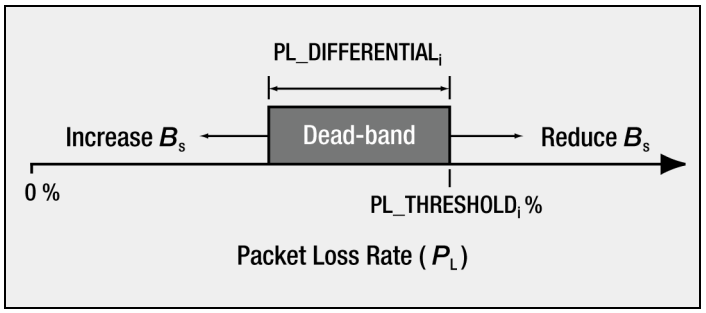

Fig. 4. ABS algorithm control strategy

IV. SIMULATION

A. OPNET model

A comprehensive discrete event simulation model using OPNET Modeler is used to analyze the performance of the proposed ABS algorithm. The model uses the 802.11g PHY layer, which supports PHY layer data rates of up to 54Mb/s. The 2.4GHz wireless channel is simulated in OPNET using a log-distance path loss model of exponent 3.6, and a shadowing variance equal to 5dB with typical NLOS channel characteristics. All stations transmit at a constant power equal to 100mW. The SNR for each data packet is mapped to a bit error rate (BER) value through the use of empirical OPNET modulation curves. To further increase the accuracy of the channel model, separate modulation curves are used for each of the 4 modulation schemes used in 802.11g.

B. Simulation Scenario

Fig. 5 shows the throughput achievable by a single transmitting station operating under various EDCA ARQ policies. A constant packet size of 1500 bytes is used, and it is assumed that there is always a packet available in the transmission queue to be sent over an error free channel. Considering this scenario, it is shown that Block Ack is superior to Ack only for block sizes greater than 3. It can also be seen that a block size of 64 only provides incremental gains compared to a block size of 20. Based on these results, an upper and lower block size limit of 4 and 20 respectively, was selected when employing ABS.

The ABS algorithm is evaluated using a single transmitting station operating with a 54Mb/s transmission data rate. It is assumed that the channel is contention free, with a packet always available for transmission. The 'Interactive Multimedia' AC is used to gain access to the channel, employing a Block Ack ARQ policy with an initial block size of 4. The ABS algorithm uses SNR as the CSI metric for use in block step size calculations, with a SNR differential (SNR_DIFFERENTIAL) of 2dB. A packet loss threshold (PL_THRESHOLD) of 5% is used with a differential (PL_DIFFERENTIAL) of 2%. Packet loss calculation is performed upon reception of every BA frame, with the block size being adjusted accordingly. A MSDU Lifetime of 100ms has been arbitrarily chosen to illustrate the performance of ABS.

V. RESULTS

Fig. 6 shows the single flow throughput achieved when using Ack, Block Ack, and Block Ack with ABS enabled, under degrading SNR conditions. As expected, Block Ack configured with a static block size of 4, out performs the legacy Ack ARQ policy due to reduced control overhead and aggregated

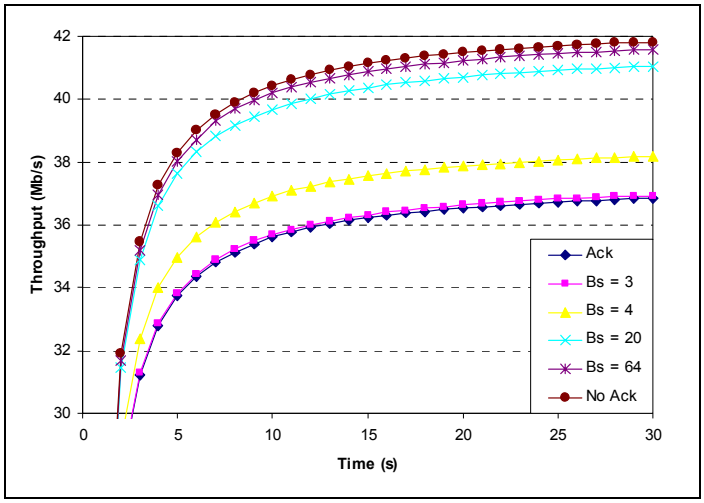

Fig. 5. Throughput under various EDCA ARQ policies

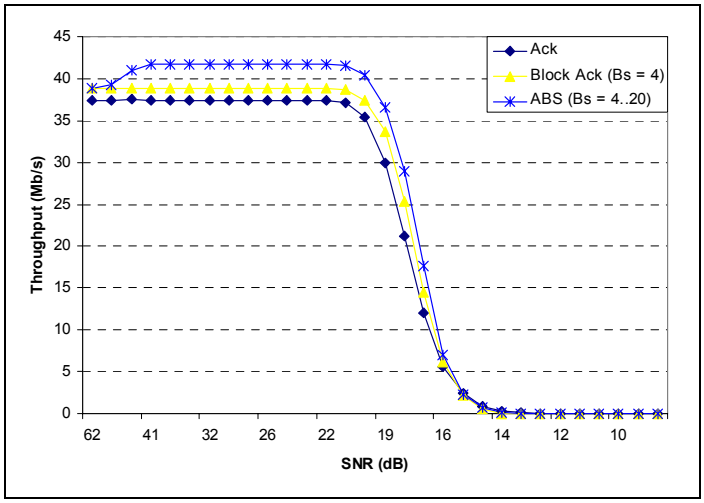

Fig. 6. Single flow throughput comparison

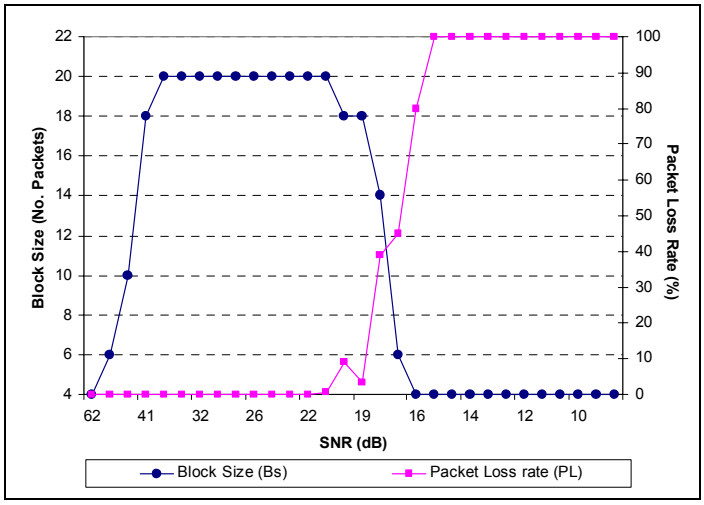

Fig. 7. ABS Packet Loss rate and corresponding Block Size

acknowledgements. As the ABS algorithm is able to dynamically adjust the block size based on channel conditions, larger block sizes are able to be employed when suitable, thus providing superior throughput of around 3Mb/s over a static Block Ack policy, and approximately 4.3Mb/s over the legacy Ack mode. Fig. 7 shows the dynamic adaptation of the block size based on the periodic P_L calculation. As P_L is calculated to be 0% for the first few consecutive instances, B_s is increased exponentially until the upper limit of 20 is reached. As the SNR degrades causing P_L to increase above a traffic class dependant threshold, B_s is reduced following the same binary exponential protocol, eventually reaching the lower bound of 4. It can be seen that at around 19dB, P_L falls within the dead-band, thus B_s remains unchanged.

As the numerous parameters associated with ABS can have an effect on the performance of the algorithm, study into the exact influence of each parameter under varying traffic characteristics and channel conditions provides an opportunity for future analysis, with our work continuing in this area. Although the SNR has been used as a CSI metric in this article due to its availability in the simulation environment, it is often difficult to obtain in a real-world scenario, and therefore a more suitable metric would need to be analyzed.

VI. Conclusion

In this article we propose an adaptive block size algorithm called ABS, designed to optimize throughput for multimedia traffic transmission using the 802.11e/n Block Ack ARQ protocol. As current standards do not define rules for the adaptation of various ARQ parameters and policies, an open and challenging research area is presented.

Although the No Ack ARQ policy is the most efficient in regards to protocol overhead [8], there is a trade-off in reliability, as data packets are transmitted only once regardless of their error status. Fortunately, multimedia traffic can often cope with a small loss rate through techniques such error resilient source coding, forward error correction (FEC), and error concealment schemes at the receiver. The ABS algorithm exploits this characteristic, enabling an opportunity for throughput enhancement and efficient channel utilization when transmitting multimedia traffic.

References

[1] Enhanced Wireless Consortium (EWC), "HT MAC Specification", Version v1.24, http://www.enhancedwirelessconsortium.org

[2] IEEE Std 802.11e -2005, Amendment to IEEE Std 802.11, 1999 Edition (Reaff 2003), Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements.

[3] Q. Li, M. van der Schaar, "Providing Adaptive QoS to Layered Video Over Wireless Local Area Networks Through Real-ime Retry Limit Adaptation", IEEE Transactions on Multimedia, Vol. 6, No. 2, April 2004.

[4] Y. Yuan, G. Daqing, W. Arbaugh, Z. Jinyun,"High-performance MAC for high-capacity wireless LANs", 13th International Conference on Computer Communications and Networks (ICCCN) 2004, Pages: 167 – 172.

[5] Y. Fallah, H. Alnuweiri, "Modeling and performance evaluation of frame bursting in wireless LANs", Proceeding of the International conference on Wireless Communications and Mobile Computing (IWCMC) 2006.

[6] I. Tinnirello, S. Choi, "Efficiency analysis of burst transmissions with block ACK in contention-based 802.11e WLANs", IEEE International Conference on Communications (ICC), 2005. Vol. 5, Pages:3455–3460.

[7] Il-Gu Lee, Sung-Rok Yoon, Sin-Chong Park, "Analysis of throughput and efficient block size based polling scheme for IEEE 802.11e wireless LANs", IEEE International Conference on Communications (ICC) 2005, Vol. 5, Pages:3467 – 3473.

[8] J. Wall, J. Khan, "Adaptive Multimedia Packet Transmission for Broadband IEEE 802.11 Wireless LANs", Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC) 2006.

[9] F.H.R.Fitzek, M. Zorzi, P. Seeling, M. Reisslein, "Video and audio trace files of pre-encoded video content for network performance measurements", Consumer Communications and Networking Conference (CCNC) 2004, Pages 245 – 250.