# Delay Analysis for Statistical Real-Time Channels in Mobile Ad-Hoc Networks*

**Min-Gu Lee and Sunggu Lee**
*Electrical and Computer Engineering Division*
*Pohang Univ. of Science and Technology (POSTECH)*
*San 31 Hyoja Dong*
*Pohang 790-784, South Korea*

## Abstract

*Wireless devices that communicate using the IEEE 802.11 protocol can be used to create mobile ad-hoc networks (MANETs). Many interesting applications using MANETs are realizable if real-time channels can be supported on such networks. However, frames are commonly dropped (and thus must be retransmitted) in wireless networks due to interference from other wireless devices and transmission range problems. To provide real-time communication in such networks, statistical real-time channels can be used. This paper proposes an analysis of the end-to-end network delay for two nodes in a MANET using the IEEE 802.11 DCF mode. This delay analysis can serve as the basis for the creation of statistical real-time channels in IEEE 802.11 MANETs.*

## 1. Introduction

IEEE 802.11 [1] is an international standard for wireless networks that has been widely used in most commercial products available in the market. IEEE 802.11 has two different medium access control (MAC) schemes based on contention-based and polling-based protocols. The former is referred to as called the *Point Coordination Function (PCF)* and the latter is referred to as the *Distributed Coordination Function (DCF)*. Current IEEE 802.11 systems yield unpredictable delay and do not support service differentiation. In a wireless network, if two devices send out packets at the same time, there will be a packet collision, garbled data will appear in the transmission medium, and neither packet will be decipherable. Thus, the two devices must retransmit their packets, hopefully after different delay intervals. However, there can still be packet collisions with other transmitting devices. This leads to unpredictable, possibly unbounded network delays. Although IEEE Task Group E has been working on a new 802.11e standard [2] that supports applications with quality-of-service (QoS) requirements, this standard has yet to be widely adopted.

Devices with IEEE 802.11 network interfaces can be interconnected together to form ad-hoc networks. This type of network, referred to as a *mobile ad-hoc network (MANET)*, is an interesting network with many potential interesting applications. In addition, if *real-time computing* can be supported on such networks, many additional possibilities arise. Example applications requiring real-time computing on MANETs include computer-assisted automobile control, on-line music concerts involving musicians at different sites, and distributed health monitoring systems.

To support real-time computing on a MANET, the network has to be able to support some form of real-time communication, in which there is some type of guarantee on the delivery times of messages. Real-time communication can be supported through the creation of a real-time channel [3], in which a set of network resources are reserved along a path from the source to the destination in order to be able to guarantee timely delivery of messages along that path. However, in a MANET, the network nodes are *mobile*, and will thus move around over time. This implies that network connections will be broken and re-established numerous times during the lifetime of a typical long-running application. Thus, the type of real-time channel described above cannot be used in a MANET.

This paper proposes the use of *statistical* real-time channels, defined in [4, 5], for the support of real-time communication in MANETs. In a statistical real-time channel, the *probability* of a message being delivered within a fixed number of frame transmission attempts is guaranteed to be within a fixed probability level. In order to work with statistical real-time channels in MANETs, this paper presents a detailed analysis of the end-to-end network delay in an IEEE 802.11 DCF mode MANET. This problem is complicated by the fact that frequent packet collisions and retransmissions can occur in such networks. In the next section, related work is presented. Then the delay analysis and

* This work was supported by the Brain Korea 21 Project in 2005.

Proceedings of the 10th IEEE International Workshop on Object-Oriented Real-Time Dependable Systems (WORDS'05)
0-7695-2347-1/05 $20.00 © 2005 IEEE

numerical results and simulated results are presented in Section 3. Next, in Section 4, the requirements for middleware used to support statistical real-time channels are discussed. Finally, the paper concludes with Section 5.

## 2. Related Work

There are several terms associated with the data sent over an IEEE 802.11 network. First, a *message* generated by an application program is partitioned into a set of fixed-size *packets*. Then, each such packet is combined with control packets and other data packets (that are part of other messages originating from the same source node) into a *frame*. The frames can be classified into *best-effort* frames, for non-real-time applications, and *real-time* frames, for real-time applications. These frames are the objects that are actually transmitted in the physical communication medium.

A number of approaches have been proposed to support service differentiation in networks. Most of these approaches require changes in the actual MAC functions. However MAC functions are normally hard-coded in the Network Interface Card (NIC) firmware. Furthermore, such an approach is not very portable. For this reason, software-level support of QoS in an IEEE 802.11 network may be a more desirable approach.

Jain *et. al.* [6] proposed an approach for supporting QoS in an IEEE 802.11 network by modifying the network device driver. Their method implements a real-time message queue that prioritizes messages using an Earliest-Deadline-First (EDF) algorithm and a non-real-time message queue using a *traffic smoother*. A traffic smoother serves the all-important role of regulating the transmission of best-effort frames to an NIC in order to guarantee sufficient transmission time for real-time messages. This type of approach was also previously created for the LINUX operating system in order to support statistical real-time channels in 802.11 DCF mode networks. However, most device drivers are provided in only binary format. Therefore it is difficult to adopt this type of approach for general applications and devices. In addition, this approach only works well under low traffic conditions.

Ahn *et. al.* [7] proposed a similar approach to regulate best-effort frames. This algorithm also provides source-based admission control. The source node sends a probing request packet to the destination node. Each intermediate node checks the available bandwidth and updates the bottleneck bandwidth field of the packet. After the destination receives the probing message, it replies to the probe with a response packet, which includes a copy of the bottleneck bandwidth field, to the source node. After the source node receives the response packet, it performs an admission control test. A drawback of this algorithm is the fact that it suffers from the problem of false admissions.

To enable some form of real-time communication in multi-access networks, Chou and Shin [4] defined the concept of a *statistical* real-time channel. Kweon and Shin [5] applied the statistical real-time channel concept to IEEE 802.3 Ethernet networks. Although the IEEE 802.11 DCF mode is very similar to Ethernet, an important difference is that a transmitting station cannot detect packet collisions.

## 3. Delay Analysis

A statistical real-time channel, as defined by Kweon and Shin [5] for the Ethernet, is a communication channel that satisfies the following condition:

$$P(r \le K) \le 1 - Z \quad (1)$$

where $r$ is the number of trials taken to transmit the frame successfully. In IEEE 802.11 DCF mode, the MAC delay depends on the number of trials required before successful transmission. Equation (1) can be rewritten in delay form as follows:

$$P(D \le D_K) \le 1 - Z \quad (2)$$

where $D$ is the network delay experienced and $D_k$ is the worst-case delay experienced by a frame when its transmission is successful during the $K$ th trial. Equation (2) is guaranteed to hold due to Equation (1).

The IEEE 802.11 DCF mode supports two types of channel contention mechanisms: the basic scheme and the request-to-send/clear-to-send (RTS/CTS) scheme. Unlike Ethernet, a transmitting station in a wireless LAN cannot detect collisions of multiple transmitting signals because the signal strength of its own outgoing signal overwhelms any signal received from other stations. Therefore, a transmitting station must wait for an acknowledgement (ACK) packet from the receiving station in order to be ensured of successful transmission. This basic scheme results in significant additional network delay whenever a packet collision occurs. This in turn may result in significant performance degradation if there are a large number of transmitting stations.

In the RTS/CTS scheme, a station that wishes to transmit a data frame needs to reserve the transmission medium. Reservation is accomplished by exchanging RTS/CTS packets between transmitting and receiving stations. Collisions may still occur during the RTS/CTS exchange. However, these collisions result in much lower network overhead than collisions between frames containing large data payloads.

### 3.1 Analysis of 802.11 DCF mode operation

Before delving into admission control, we need to analyze the IEEE 802.11 DCF functions. The analysis of the DCF mode is complicated by the fact that there is no central controller of the network. This fact, in turn, makes it more difficult to support real-time communication.

Related studies were previously undertaken by Bianchi [8] and Ziouva and Antonakopoulos [9]. However, Bianchi's model [8] does not take into account the busy medium conditions for invoking the backoff procedure. Ziouva and Antonakopoulos' model [9] accounts for the busy medium condition using a channel-busy probability. Adopting Ziouva and Antonakopoulos' model [9], we also assume ideal channel conditions in order to analyze the IEEE 802.11 DCF mode.

In the IEEE 802.11 MAC protocol, no frame can be transmitted without a backoff stage. However, Ziouva and Antonakopoulos' model *can* transmit a frame without backoff a stage. Therefore, we modify Ziouva and Antonakopoulos' model to fit the IEEE 802.11 MAC protocol. This new model is shown in Figure 1, and its transition probabilities are as follows:

$$\begin{cases} P\{i,k \mid i,k+1\} = 1-P_b & k \in (0, W_i-2), i \in (0,m) \\ P\{i,k \mid i,k\} = P_b & k \in (1, W_i-1), i \in (0,m) \\ P\{i,k \mid i-1,0\} = P_c / W_i & k \in (0, W_i-1), i \in (1,m) \\ P\{m,k \mid m,0\} = P_c / W_m & k \in (0, W_i-1) \\ P\{0,k \mid i,0\} = (1-P_c)/W_0 & k \in (0, W_i-1), i \in (0,m) \end{cases}$$

As in Ziouva and Antonakopoulos' model, let $b(t)$ be the stochastic process representing the backoff time counter for a given station and $s(t)$ be the stochastic process representing the backoff stage of the station at time $t$. $m$ is the maximum stage of the backoff procedure. In IEEE 802.11, the *binary exponential backoff* method is used. In this method, the backoff window size (the amount of time that a station waits before transmitting its frame in order to attempt to avoid collisions) of a station during stage $i$ is $W_i = 2^i W_{\min}$ if $0 \le i \le m$ and $W_i = 2^m W_{\min}$ if $m < i$. Two constant probabilities are defined. $P_c$ is the collision probability; i.e., the probability that a transmitted frame collides with at least one other frame. $P_b$ is the probability that the channel is busy at a given slot time. $n$ is the number of active stations that have frames to transmit in their NIC buffers.

Let $b_{i,k} = \lim_{t\to\infty} P\{s(t)=i, b(t)=k\}$ be the stationary distribution of the Markov chain, where $i$, $k$ are integers and $0 \le i \le m, 0 \le k \le W_i - 1$. Then,

$$b_{i,0} = P_c^{\,i} b_{0,0} \tag{3}$$

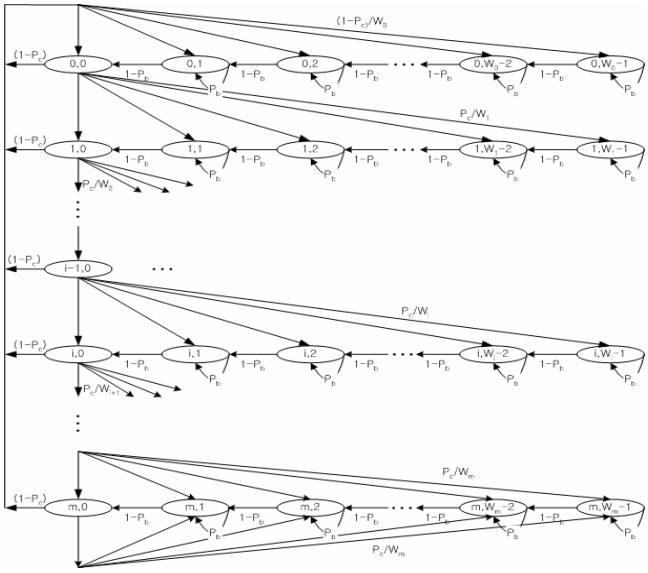


**Figure 1. State transition diagram**

$$b_{m,0} = \frac{P_c^{\,m}}{1-P_c} b_{0,0} \tag{4}$$

$$b_{i,k} = \frac{W_i - k}{W_i} \frac{1}{1-P_b} b_{i,0} \tag{5}$$

By the probability conservation relation, we have

$$\sum_{i=0}^{m} \sum_{k=0}^{W_i-1} b_{i,k} = 1 \tag{6}$$

By substituting Equations (3) ~ (5) into Equation (6), we can derive the following equation:

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

$$= \frac{1}{1-P_b} \left( \sum_{i=0}^{m-1} P_c^{\,i} b_{0,0} \frac{W_i+1}{2} + \frac{W_m+1}{2} \frac{P_c^{\,m}}{1-P_c} \right) = 1$$

Thus,

$$b_{0,0} = \frac{2(1-2P_c)(1-P_c)(1-P_b)}{(1-2P_c)(W+1) + P_c W (1-(2P_c)^m)} \tag{7}$$

Let $\tau$ be the probability that a station transmits during a given slot time. Because a retransmission occurs only when the backoff timer becomes zero, $\tau$ can be expressed as

$$\tau = \sum_{i=0}^{m} b_{i,0} = \frac{2(1-2P_c)(1-P_b)}{(1-2P_c)(W+1) + P_c W (1-(2P_c)^m)} \tag{8}$$

The channel is detected as being busy when at least one of the stations transmits during a slot time. Therefore, the channel busy probability $P_b$ is given by

$$P_b = 1 - (1-\tau)^n \tag{9}$$

A transmitted frame collides when two or more stations transmit during a slot time. Therefore, the collision probability $P_c$ is given by

$$P_c = 1 - (1-\tau)^{n-1} \quad (10)$$

Let $S_i$ be the probability that a packet is transmitted successfully during the $i$th backoff stage. Then

$$S_i = (1 - P_c)(1 - \prod_{j=1}^{i-1} S_j) \quad (11)$$

The probability that packet is successfully transmitted during the $K$th attempt is $S_K$. Therefore we need to control the number of station that share the channel in order to satisfy Equation (1). Figure 2 shows the accumulated success probability graph when $W_{\min} = 32$ and $m = 5$; these are parameter values used in the IEEE 802.11 Direct Sequence Spread Spectrum (DSSS) mode. Other parameter values used in the IEEE 802.11 DSSS specification are shown in Table 1. As expected, if the number of stations is increased, the expected number of collisions also increases. Thus, in order to satisfy Equation (1), the number of stations that share the channel needs to be carefully controlled.

### 3.2 Backoff Delay Analysis

In order to analyze the delay of IEEE DCF mode operations, we need to investigate the backoff delay that a frame experiences. The backoff delay of a frame depends on its backoff counter value and the duration of the freezing counter. The delay $D$ that a frame may experience is given by

$$D = \sum_{i=0}^{N_c} BD_i + N_c(T_c + T_o) + T_s \quad (12)$$

where $N_c$ is the number of collisions that a station experiences, $BD_i$ is the backoff delay that a frame experiences during the $i$th backoff stage, $T_c$ is the time taken when the channel is occupied by packets

**Table1. Parameters used in the IEEE 802.11 DSSS specification**

| Parameters | Values |
|---|---|
| $L^{MTU}$ | 2312 octets |
| $L^{RTS}$ | 20 octets |
| $L^{CTS}$ | 14 octets |
| $L^{ACK}$ | 14 octets |
| $L^{MAC\_OH}$ | 34 octets |
| $L^{PHY\_OH}$ | 16 octets |
| $T^{DIFS}$ | 50μs |
| $T^{SIFS}$ | 20μs |
| $T^{SlotTime}$ | 10μs |
| $W_{\min}$ | 32 |
| $m$ | 5 |
| $T^{ACK\_timeout}$ | 300μs |
| $T^{CTS\_timeout}$ | 300μs |

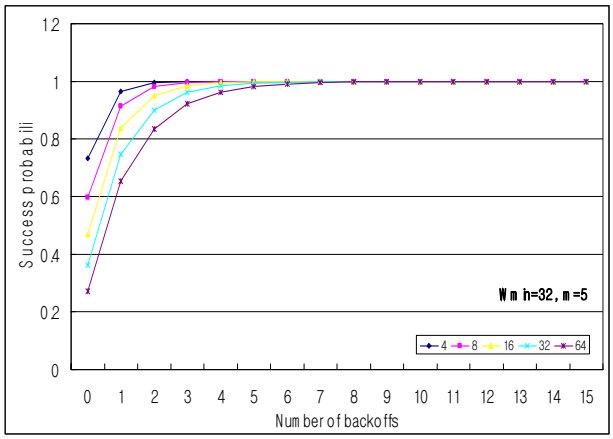


**Figure 2. The accumulative success probability given $i$ backoffs with 4, 8, 16, 32, and 64 transmitting stations.**

undergoing collisions, $T_o$ is the time that a station has to wait before sensing the channel again to detect whether a collision occurs or not, and $T_s$ is the time required for a station to successfully transmit the frame once it has succeeded in acquiring the channel.

In the basic access scheme, $T_c^{basic}$ is given by $T_c^{basic} = H + L + \delta + T^{DIFS}$, where $H$ is the time needed to transmit the PHY and MAC headers of the packet, $L$ is the packet size, $\delta$ is the propagation delay, and $T^{DIFS}$ is the DCF inter-frame delay. A station must wait $T_o^{basic}$ time before sensing the channel again. This time can be computed as $T_o^{basic} = T^{SIFS} + T^{ACK\_timeout}$ where $T^{SIFS}$ is a short inter-frame delay and $T^{ACK\_timeout}$ is the timeout delay, after which the transmitting station gives up waiting for an ACK from its receiving station. Finally, the time required for successful transmission $T_s^{basic}$ is given by

$$T_s^{basic} = H + L + \delta + T^{SIFS} + T^{ACK} + \delta + T^{DIFS} \quad (13)$$

where $T^{ACK}$ is the ACK packet transmission time including the time required to send the headers.

In the RTS/CTS scheme, $T_c^{RTS/CTS}$ is given by $T_c^{RTS/CTS} = T^{RTS} + \delta + T^{DIFS}$ where $T^{RTS}$ is the RTS packet transmission time. $T_o^{RTS/CTS} = T^{SIFS} + T^{CTS\_timeout}$ where $T^{CTS\_timeout}$ is the timeout delay, after which a station that transmits an RTS packet gives up waiting for the corresponding CTS packet from the receiving station. Finally, $T_s^{RTS/CTS}$ is given by

$$T_s^{RTS/CTS} = T^{RTS} + \delta + T^{SIFS} + T^{CTS} + \delta + H + L + \delta + T^{SIFS} + T^{ACK} + \delta + T^{DIFS} \quad (14)$$

where $T^{CTS}$ is the CTS packet transmission time. Note that $T_c$, $T_o$ and $T_s$ cannot be controlled. The

only term that can be controlled is the data load size $L$.

There are two components to the backoff delay. One is the time required to decrement the backoff counter itself, and the other is the time spent with the counter frozen when the channel is captured by other stations. Therefore, $BD_i$ is given by

$$BD_i = B_i + FR_i \quad (15)$$

where $B_i$ is the value of the backoff counter during the $i$th backoff stage and $FR_i$ is the time spent with the counter frozen when the transmission medium is captured by other stations during the $i$th backoff stage.

The worst-case backoff counter value is $W_i$ during the $i$th stage. Therefore, the worst-case total backoff counter value during stage $i$ is given by

$$B_i^{worst} = \begin{cases} 2^i W_{\min} & 0 \le i \le m \\ 2^m W_{\min} & m < i \end{cases}$$

The backoff counter is frozen when a station senses that the medium is busy. Unlike other delay components, the time spent with the counter frozen is affected by other stations. If a station transmits a frame during the $i$th trial, then its backoff counter value is $B_i$. During $B_i$ slot times, $P_b B_i$ slot times are used for transmitting frames sent by other stations and $(1-P_b)B_i$ slot times remain idle. Therefore, other stations may transmit $P_s B_i (1-P_b)$ frames successfully and $(1-P_s)B_i(1-P_b)$ frames experience collisions, where $P_s$ (the probability that a transmission is successful) is given by

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

because a frame transmission is successful when only one of $n$ stations attempts to transmit. Therefore, the total time spent with the clock frozen during the $i$th trial is

$$FR_i = (P_s T_s + (1-P_s)T_c)B_i(1-P_b) \quad (13)$$

The worst-case delay for $FR_i$ occurs when $P_s$ is 1, $P_b$ is 0, and $B_i$ attains its worst-case value. In this case, all of the other stations' frames are successfully transmitted after waiting for the maximum backoff delay. However, the probability of this case is extremely low.

To compute the expected delay for a frame, Equation (12) should be converted into expected value format. Equation (12) can be rewritten as follows:

$$E[D] = \{E[N_c](E[BD]) + E[N_c](T_c + T_o)\} + E[BD] + T_s \quad (14)$$

where $E[N_c]$ is the average number of collisions that a station experiences and $E[BD]$ is the average backoff delay of a station. The average number of retransmissions is $1/P_s$. Therefore the average number of collisions $E[N_c] = 1/P_s - 1$. To calculate the expected backoff delay, we need to know the expected freezing time and the expected backoff counter value. The expected backoff counter value $E[B]$ is given by

$$E[B] = \sum_{i=0}^{m} \sum_{k=1}^{W_i - 1} k b_{i,k} = \frac{W^2(1-P_c-3P_c(4P_c)^m+4P_c-1)}{6(1-P_b)(1-4P_c)(1-P_c)} b_{0,0}$$

The expected freezing time can be calculated from Equation (14) as follows:

$$E[FR] = (P_s T_s + (1-P_s)T_c)E[B](1-P_b)$$

Finally, the expected backoff delay $E[BD]$ is given by

$$E[BD] = E[B] + E[FR]$$

### 3.3 Numerical Results

Let us now compute the expected end-to-end delays for use with statistical real-time channels under various conditions, using the parameter values given in Table 1 as a basis. First, we need to investigate $D_k$, which represents the worst-case delay experienced when a frame is transmitted during the $k$th attempt. Figure 3 shows the worst-case delay $D_k$ versus the number of retries for a fixed frame size with $\alpha = 1$, where $\alpha$ is a constant that is used when specifying the maximum frame size as $L = \alpha * L^{MTU}$. Note that the $D_k$ value for the RTS/CTS scheme is slightly larger than the corresponding value for the basic scheme because the worst-case situation requires that all backoff slots are expended (which implies that $P_b$ is zero) before successful transmission (which implies that $P_s$ is one) of other stations. This is because $T_s^{basic} \le T_s^{RTS/CTS}$ by Equations (13) and (14).

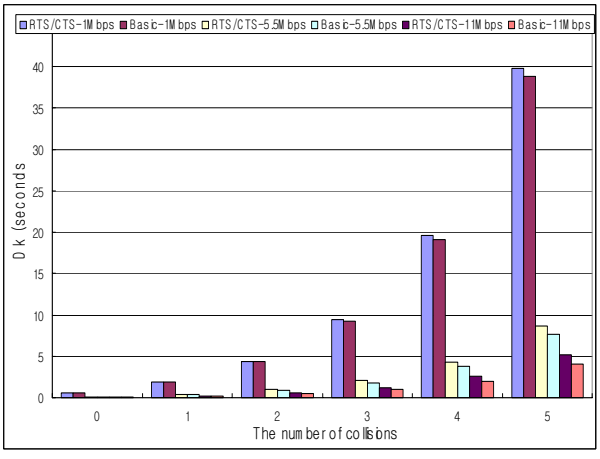


**Figure 3. $D_k$ for small numbers of collisions with 1, 5.5, and 11Mbps data rates.**

Figure 4 shows $D_k$ with $k$ values ranging from 0 to 15. Unlike Figure 3, when $k$ is greater than 5, $D_k$ increases linearly. This is because the IEEE 802.11 DCF mode uses the binary backoff scheme, which fixes the maximum backoff window at $W_i = 2^m W_{\min}$ when the number of retries exceeds a certain threshold $m$.

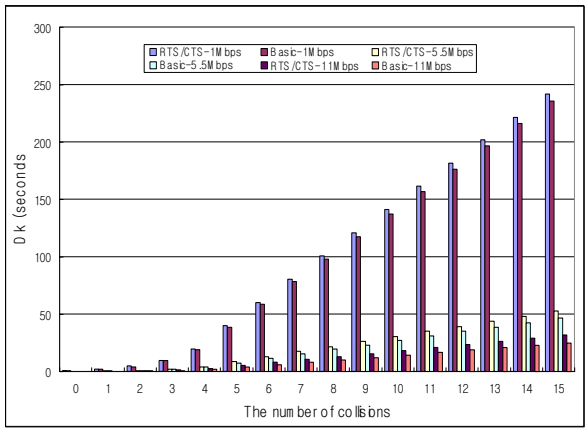


**Figure 4. $D_k$ for large numbers of collisions with 1, 5.5, and 11Mbps data rates.**

Figure 5 shows the relationship between $\alpha$ and $D_k$. $D_k$ is a monotonically and linearly increasing function of $\alpha$. $D_k$ for the RTS/CTS scheme is also larger than $D_k$ for the basic scheme for the same reason stated above. This figure shows that it may be desirable to fragment a large frame into several smaller ones in order to reduce the overall delay.

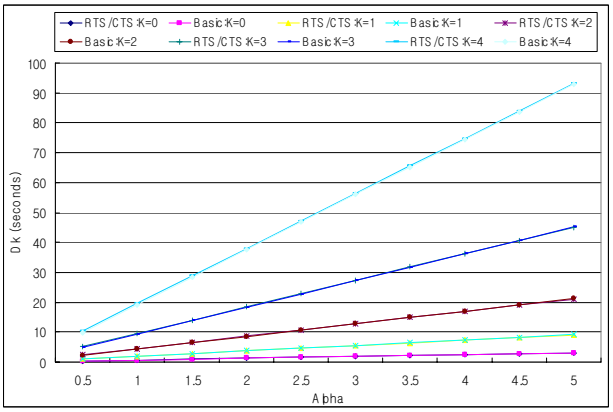


**Figure 5. $D_k$ for given $\alpha$ with K=0, 1, 2, 3, and 4.**

The results that are shown above are the upper bounds of the delays that of a frame may experience. The main delay analysis that is required for the creation of a statistical real-time channel is the computation of the probability that a frame will be transmitted within a given deadline. Note that, at any given time, not all stations have frames ready to transmit. Let us assume that at any given slot time, the frame arrival rate follows a Poisson distribution. Then, at any given slot time, the number of stations that have frames waiting to be transmitted also follows a Poisson distribution. Therefore, the expectation of the number of active stations that have frames to transmit is $n = N\lambda$, where $N$ is the number of stations that share the channel with real-time traffic and $\lambda$ is the mean value of the real-time frame arrival rate.

Figure 6 shows the relationship between $\lambda$ and the average delay for a 1 Mbps data rate with $\alpha$ equal to 1. This figure shows that the average delay is proportional to $N$ and $\lambda$. Unlike the graphs for $D_k$, Figure 6 shows that the RTS/CTS scheme increasingly outperforms the basic scheme as $\lambda$ and $N$ are increased. As the number of active stations is increased, the packet collision rate also increases. When a collision occurs, RTS/CTS scheme only loses $T_c^{RTS/CTS}$ time. However, the basic scheme loses all of the time spent transmitting the entire data payload as shown in $T_c^{basic}$. Note, however, that as shown in Figure 2, the basic scheme has a smaller worst-case delay $D_k$ than the RTS/CTS scheme.

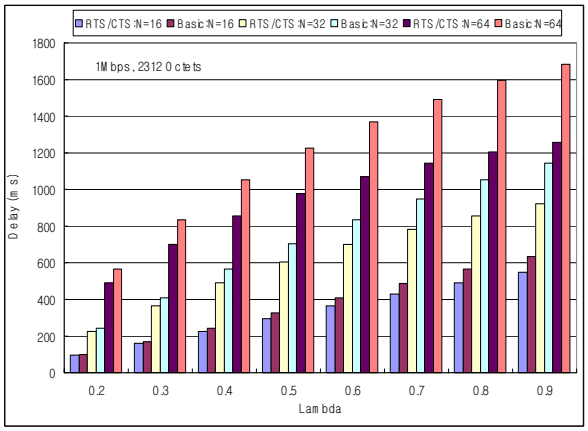


**Figure 6. Average delay for a given $\lambda$ with $N$=16, 32 and 64.**

Figure 7 shows the average delay, given various $\lambda$ values, for data rates of 1, 5.5, and 11 Mbps. As expected, a higher data rate yields lower average delay. This figure also shows that the average delay is linearly proportional with $\lambda$ for a given data rate. Note that the differences between the RTS/CTS and basic schemes increase as $\lambda$ is increased.

Figure 8 shows the average expected delay that a frame may experience given varying numbers of active stations and data rates with the RTS/CTS scheme. In this figure, comparisons are shown between calculated and simulated results. Simulated results were obtained

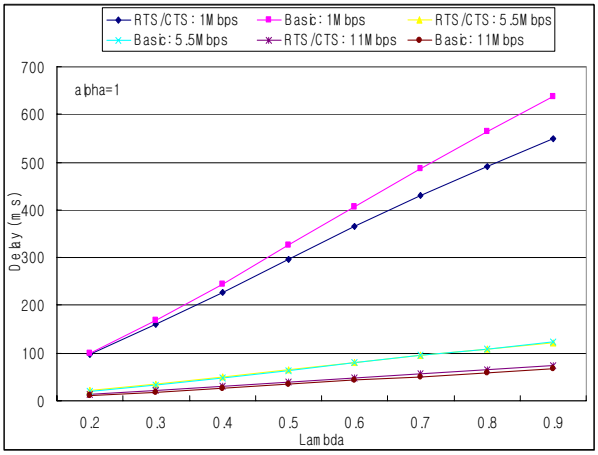


**Figure 7. Average delay for a given $\lambda$ with 1, 5.5, and 11Mbps data rates.**

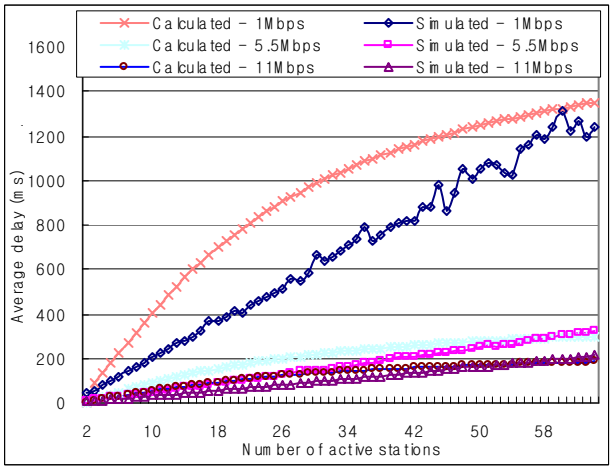


**Figure 8. Average delay (RTS/CTS scheme) vs. number of active stations with 1, 5.5, and 11 Mbps data rates (calculated and simulated).**

from experiments performed using *ns-2* [10], which is a discrete event simulator tool commonly used in networking research. Comparisons were only made with the RTS/CTS scheme since this was the only scheme supported in the current version of ns-2. The total simulation time was set to 70 seconds. However the results of the first 10 seconds were discarded in order to avoid anomalous startup effects.

As expected, as the number of active stations is increased, the average frame-transmission delays of the active stations also increase. With large numbers of active stations, the calculated expected delays and simulation results are found to match fairly well. However, the differences between calculated and simulated results are relatively large with low to medium numbers of active stations (e.g., the simulated results are lower than the calculated results by as much as 38% at the 1Mbps data rate). Nevertheless, since the calculated results overestimate the simulated results most of the time, the largest difference is about 38% at the lowest data rate with 28 active stations, and the differences are smaller at higher data rates and with larger numbers of active stations, the proposed delay computation method is deemed to be a useful tool for the implementation of statistical real-time channels in mobile ad-hoc networks.

Although the differences between the computed and simulated results could be due to many factors, the main contributing factors are conjectured to be the collision probability ( $P_c$ ) and the channel busy probability ( $P_b$ ) parameters used in our model. In our analysis, it was assumed that $P_c$ and $P_b$ are constant probabilities. This assumption is reasonable for large numbers of active stations. However, $P_c$ and $P_b$ are not constant probabilities for small numbers of stations. Therefore, the average delay values for small numbers of active stations show relatively large differences between calculated and simulated results.

## 4. Middleware Architecture

In order to support statistical real-time channels, there must be a method for controlling the injection of frames into the network. Networks designed for real-time computing, such as the FieldBus [5], typically operate in a deterministic manner that permits strict control of the rate at which packets are inserted into the network. General-purpose network protocols such as IEEE 802.11 normally do not include such mechanisms. However, without a source packet-insertion control mechanism, the statistical analysis required for the creation of statistical real-time channels becomes meaningless.

It is proposed that special middleware be created to control the insertion of packets into an IEEE 802.11 DCF mode network. This type of middleware was previously created for the LINUX operating system in order to support statistical real-time channels in Ethernet networks [5]. As part of our future research work in this area, we propose to create the corresponding type of middleware for IEEE 802.11 DCF mode MANETs. The details of the operation of this proposed middleware are described below.

The middleware maintains frame queues that an application uses to make message transmission requests. The middleware schedules the requests in the frame queues and passes them to the wireless network interface card (NIC). Only frames that are passed to the wireless NIC will contend for transmission. The middleware will have two separate queues: a real-time class queue and a best-effort class queue. Frames in the real-time queue may be serviced using the earliest-deadline-first (EDF) or other real-

time scheduling method. If EDF is used, frames in the real-time class queue are maintained in increasing order of their deadlines. The middleware passes the first frame in the real-time class queue to the NIC only when the transmission buffer in the NIC becomes empty; once the frame has been moved to the NIC's transmission buffer, the NIC sends out the frame using the binary exponential backoff procedure.

In contrast, frames in the best-effort queue are serviced in a first-in first-out (FIFO) manner and passed to the NIC only when the real-time class queue becomes empty. This ensures that local best-effort frames do not hinder real-time frames, except possibly one frame that has already been passed to the NIC. A best-effort frame should attempt to transmit only after the channel has become idle for a certain amount of time. This is so that pending real-time frames in other stations will have plenty of opportunity to transmit. Therefore we can assume that no best-effort frame disturbs real-time frames.

All components of Equation (12) include the $L$ term, which is the size of the frame. If a frame that captures the communication medium has very a large $L$, then other stations must wait $T_s$ time units, where this term is computed using Equation (13) for the basic scheme and (14) for the RTS/CTS scheme. Therefore, we should bound $L$ as $L = \alpha * L^{MTU}$, where $L^{MTU}$ is the maximum transfer unit defined in Table 1. If a real-time stream requires a larger data size, then the middleware should partition the data across multiple packets, which are then sent to the NIC in order. By limiting the size of each packet, we can estimate the delay that a given packet will experience using Equation (12).

To support statistical real-time channels, the middleware must be able to control the admission of real-time streams. For a given $\lambda$, which is the arrival rate of real-time frames, we need to adjust the number of stations that share the channel with real-time traffic. In order to satisfy (1), $S_K \geq 1 - Z$ should be satisfied. First, we can obtain the maximum permitted $K$ value and data rate that satisfies the specified real-time requirements. After that, we can calculate the value of $P_c$ that satisfies $S_K \geq 1 - Z$ using Equation (11). Then, we can obtain $\tau$ using Equation (8). Finally we can calculate the maximum value of $n$, the number of active stations. The expected value of the number of active stations is $n = N\lambda$. From this equation, we can then determine the maximum possible value of $N$.

## 5. Conclusion

Due to the characteristics of the IEEE 802.11 DCF mode, it is very difficult to support real-time communication in mobile ad-hoc networks (MANETs) that use this communication protocol. This paper proposes the use of *statistical real-time channels* for the support of real-time communication in MANETs. In particular, this paper provides an analysis, supported by comparison with simulations results, of the statistical end-to-end delay properties for IEEE 802.11 DCF mode networks. The analysis provided in this paper can serve as the basis for the creation of statistical real-time channels in MANETs built from IEEE 802.11 devices.

Finally, we propose the use of special middleware to control the rate at which packets are inserted into the network. Such a mechanism is necessary for the support of statistical real-time channels. This middleware is the topic of proposed future research work in this area.

## 6. References


[1] *IEEE 802.11 WG, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification*, IEEE Press, 1999.

[2] *IEEE 802.11 WG, Draft Supplement to Part 11: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications: QoS*, IEEE 802.11e/D2.0, IEEE Press, Nov. 2001.

[3] D. Ferrari and D. C. Verma, "A scheme for real-time channel establishment in wide area networks," *IEEE J. Selected Areas in Communications*, pp. 368-379, 1990.

[4] C.C. Chou and K.G. Shin, "Statistical Real-Time Channels on Multiaccess Networks," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 8, No. 8, pp. 769-780, Aug. 1997.

[5] S. Kweon and K. G. Shin, "Statistical Real-Time Communication over Ethernet," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 14, No. 3, pp. 322-335, March, 2003.

[6] A. Jain, D. Qiao, K. G. Shin, "RT-WLAN: A Soft Real-Time Extension to the ORiNOCO Linux Device Driver," in *The 14th IEEE Int'l Symp. On Parallel on Personal Indoor and Mobile Radio Communications*, Beijing, pp.1434-1440, Sep. 2003.

[7] G. Ahn, A. T. Campbel, A. Veres, L. Sun, "Supporting Service Differentiation for Real-Time and Best-Effort Traffic in Stateless Wireless Ad Hoc Networks (SWAN)", *IEEE Transactions on Mobile Computing*, Vol. 1, No. 3, July-Sep. 2002.

[8] G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function", *IEEE Journal on Selected Areas in Communications*, Vol. 18, No. 3, pp. 535-547, March 2000.

[9] E. Ziouva and T. Antonakopoulos, "CSMA/CA performance under high traffic conditions: throughput and delay analysis", *Computer Communications*, Vol. 25, pp. 313-321, 2002.

[10] NS-2 homepage, http://www.isi.edu/nsnam/ns