Appeared in PERFORMANCE EVALUATION, vol. 27 & 28, pp. 331–346, October, 1996.

Centralized Wireless MAC Protocols Using Slotted ALOHA and Dynamic TDD Transmission

Sunghyun Choi and Kang G. Shin

Real-Time Computing Laboratory
Department of Electrical Engineering and Computer Science
The University of Michigan
Ann Arbor, Michigan 48109–2122
E-mail: {shchoi,kgshin}@eecs.umich.edu

ABSTRACT

ALOHA scheme with dynamic Time Division Duplexed (TDD) transmission is designed and analyzed. A centralized (i.e., star) network is adopted as the topology of a cell which consists of a base station and a number of mobile clients. In dynamic TDD transmission mode, a channel is time-shared for downlink and uplink transmissions under the dynamic access control of the base station. We first propose two MAC protocols (called *TDD1* and *TDD2*) depending on how downlink and uplink transmissions are multiplexed. We then analyze throughput and average delay of TDD1 and an alternative ALOHA scheme using Frequency Division Duplexed (FDD) transmission. Finally, we evaluate the performance of these schemes, and compare TDD1 (calculations), TDD2 (simulations), and FDD (calculations). TDD schemes are found to always work as good as, or better than, FDD. TDD2 is observed to outperform TDD1 with respect to the downlink delay in the presence of light uplink and heavy downlink traffic loads while they work almost the same in other cases.

Index Terms — Wireless LANs and MAC protocol, star topology, slotted ALOHA, dynamic Time Division Duplexing (TDD).

The work reported in this paper was supported in part by the US Department of Transportation under Grant No. DTFH61-93-X-00017. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the funding agency.

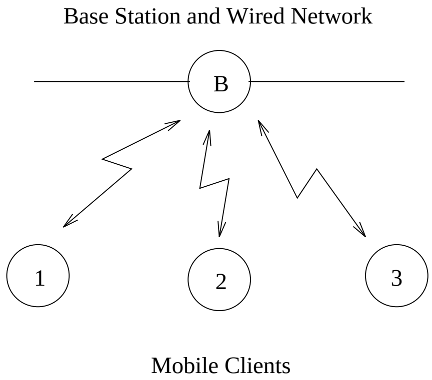


Figure 1: A centralized wireless network with a base station.

1 Introduction

Wireless local area networks (LANs) have recently received considerable attention as an attractive alternative or complementary to wired LANs [1, 2], because they allow us to set up and reconfigure LANs easily without incurring the cost of wiring. Moreover, they can easily meet the growing demand that mobile clients should have access to the existing wired and wireless networks.

This paper is concerned with a centralized (i.e., star) network, called a *cell*, as shown in Figure 1 which consists of a base station (denoted by B) and several mobile clients (denoted by numbers). The base station is connected to a wired network via a bus or a wired link. Examples of using this topology are public telephone networks, cellular mobile telephones, and some of wired and wireless LANs. In this topology, we assume the uplink (mobile-to-base) is not a broadcast channel while the downlink (base-to-mobile) is. Hence, mobile clients are not able to listen directly to other mobiles using the same frequency channel. This assumed situation can occur in real world due to the existence of *hidden* terminals [3, 4]. One important advantage of the star topology is that the network can be designed to use signal-transmission power efficiently. For example, compared to a fully-connected peer-to-peer network, mobile clients in the centralized network can reach clients at twice the distance with the same signal power. Another advantage of this topology is that the mobile unit can be made functionally simple, while more sophisticated control functions are concentrated in the base station. When battery-supported mobile units are to be used, it is very important to conserve power and make the units simple.

However, the centralized topology is susceptible to a single-point (i.e., the base station) failure. Other disadvantages include the indirection delay of the network and a 50% reduction of channel utilization as compared to peer-to-peer networks in the case of intra-cell communications. The store-and-forward delay between two mobiles within the same network is twice that of a fully-connected peer-to-peer network and the channel will be used twice per communication, because all communications must go through the base station. In spite of these drawbacks, the existence of hidden terminals and the other advantages mentioned above and elsewhere [4] justify the adoption of the centralized topology. The entire wireless network may consist of several cells, and mobile clients may move from one cell to another. We will in this paper focus on the communication within a single cell, while considering the incoming and outgoing traffic of the cell.

Dynamic Time Division Duplexed (TDD) transmission is used in the network under consideration, i.e., a wireless channel is time-shared for both downlink and uplink transmissions according to the traffic load under the dynamic control of the base station. We could instead use the Frequency Division Duplexed (FDD) mode, in which two different frequency channels are allocated for uplink and downlink transmissions. FDD, as in AMPS (FDMA), IS-54 (TDMA), and IS-95 (CDMA), is the common duplexing mode in cellular systems [2]. However, dynamic TDD allows for more efficient link utilization in the case of unbalanced uplink and downlink traffic, e.g., non-interactive data transmissions.

ALOHA protocols are probably the richest family of multiple access protocols in both wired LANs [5, 6] and wireless LANs [2, 4]. In this paper, we consider two wireless MAC protocols (called *TDD1* and *TDD2*), in which a slotted ALOHA protocol with dynamic TDD transmission is used to control uplink accesses. The protocols considered here can be used to support the non-real-time communication part of the protocol described in [7]. We analyze the throughput and delay performance of TDD1 using a 2-dimensional Markov chain, while relying on simulations for the evaluation of TDD2. We show, by comparing TDD1 with an FDD-mode protocol, the efficiency/superiority of the dynamic TDD MAC protocols in centralized wireless networks. We also compare TDD1 and TDD2, and find that the latter outperforms the former with respect to the downlink delay under low uplink and high downlink traffic loads while they perform almost equally in other cases.

Throughout the remainder of this paper, we will ignore the packet-propagation delay, since it is usually small relative to the other components like queueing and transmission delays in a cell.[1] For simplicity, we also assume that the transmission channel is error-free if

[1] A cell in this paper refers to a *micro-cell*, which has coverage of the order of a few hundred meters, or a *pico-cell*, which covers small indoor areas [2].

there is no collision among two or more concurrent packets. (How to achieve such error-free transmissions will be reported in a future paper.)

The paper is organized as follows. Section 2 describes the protocols considered. Section 3 analyzes the throughput and average delay of both TDD1 and FDD modes assuming a finite number of mobile clients with no buffering. In Section 4, we present the numerical results of performance evaluation, and compare TDD1 and FDD. We also compare TDD1 and TDD2 using the analytic results of TDD1 and the simulation results of TDD2. Finally, the paper concludes with Section 5.

2 Protocol Description

When a mobile client wants to send a packet, regardless whether the packet is destined for another client in the same cell or in a remote cell, the client must first send the packet to the base station, which will then forward it to the final destination, sometimes via other base stations. Only the downlink channel is the broadcast type: when the base station transmits packets, all but the destination mobiles in the cell ignore them. By contrast, a mobile cannot hear other mobiles' uplink transmissions, and only the base station can determine if a collision has occurred in the uplink channel. Dynamic TDD transmission is used in the network, i.e., a wireless channel is multiplexed for both downlink and uplink transmissions according to the traffic load condition. The multiplexing between uplink and downlink transmissions is controlled dynamically by the base station.

We use the slotted ALOHA to control uplink transmissions. All packets, like ATM cells, are assumed to have the same fixed size, and a packet is transmitted in one time *slot* of duration T_s. In the absence of downlink packets, the base station will issue contention slots. In a contention slot, mobiles with packets to transmit contend for using the uplink based on the slotted ALOHA. Immediately before each contention slot, the base station broadcasts the state (collision, success, or empty/unused) of the previous contention slot using a control packet via a *mini slot* of duration T_{ms}. The control packet is also used to inform mobiles if the next slot is a contention slot.

After each contention slot, the base station may issue another contention slot or transmit downlink packets, depending on the downlink traffic condition. According to how the base station controller multiplexes downlink and uplink transmissions, we define two different transmission schemes: TDD1 and TDD2.

TDD1 Mode: The multiplexing problem under consideration is quite different from the

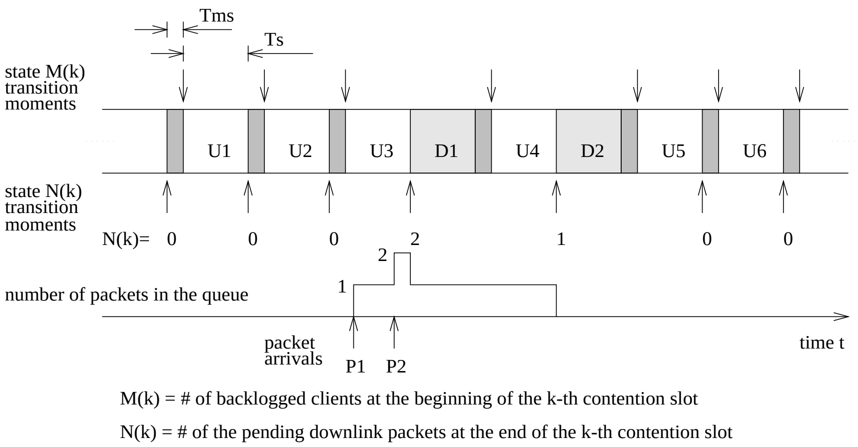


Figure 2: An example of communications using TDD1 mode.

conventional multiplexing in the wired network, in the sense that in the latter, a packet switching node multiplexes the transmission of incoming packets over an outgoing link, while in the former, the base station multiplexes the downlink packets arriving from the wired network with the control packets (to allow for mobiles' uplink transmissions without knowing if the mobiles have packets to transmit). In other words, the information about downlink packets is available to the base station while that about uplink packets is not. A possible remedy is to control the number of uplink slots according to the downlink traffic load while guaranteeing one half of the available slots are used for uplink transmissions.[2]

The base station controller operates as follows: (1) if there is no pending downlink packet, the base station issues contention slots continuously, and hence, a contention slot and a mini slot alternate; (2) if there are pending downlink packets, the base station alternately transmits a packet and issues a contention slot until the downlink packet queue is emptied. Figure 2 shows an example of communications in a cell. After each mini slot, a contention slot (U_i) appears, and is followed by a downlink packet transmission (D_j) if there is a pending packet in the queue. The lower part of this figure shows the number of packets in the queue which increases by one upon arrival of a new packet (P_j) and decreases by one upon transmission of a packet in the queue.

TDD2 Mode: With TDD1, uplink transmissions are guaranteed to use at least one half of the available slots while downlink transmissions can use only up to the other half of the available slots. This strategy is employed due to the fact that the base station has no

[2]With this simple strategy, symmetric applications might have some problem because collisions might take away some of the uplink bandwidth. This kind of problem can be handled with a more complicated reservation-type scheme found in [8].

```
COUNT := 0 and CONT := 1 and COLL := 0;
    ! COUNT is # of consecutive downlink packet transmissions.
    ! CONT is the upper limit of COUNT.
    ! COLL is the estimated # of backlogged clients.
while (time increases) {
    if (downlink packet exists and COUNT < CONT) then
        transmit the packet;
        COUNT := COUNT + 1;
    else then
        issue a uplink contention slot;
        if (idle) then RES := 0;                    ! RES is the index for the state of
        else if (successful) then RES := 1;           a contention slot.
        else if (collision) then RES := 2;
        if (RES = 0 and COLL = 0 and COUNT > 0) then
            if (CONT = MAX_CONT) then CONT := 1;
                ! MAX_CONT is the upper limit of CONT.
            else then CONT := CONT + 1;
        else if (RES = 1) then
            CONT := 1;
            if (COLL > 0) then COLL := COLL − 1;
        else if (RES = 2) then CONT := 1 and COLL := 2;
        COUNT := 0;
}
```

Table 1: A pseudocode of the base station's role for TDD2.

information about the uplink traffic load. We modify TDD1 so that downlink transmissions can use more than one half of the slots in case the uplink traffic load is "estimated" to be low. One possible way to estimate the uplink traffic load is to count the number of contiguous idle uplink slots. The base station controller for TDD2 mode operates as follows: (1) downlink packets can be transmitted consecutively up to $CONT$ slots; (2) $COLL$ is the estimated number of backlogged mobile clients each with a packet to retransmit as a result of collision; (3) $CONT$ increases by 1 whenever $COLL = 0$ and there is an idle uplink slot right after a downlink packet transmission; (4) $COLL = 2$ whenever there is an uplink collision; (5) $COLL$ decreases by 1 whenever a successful uplink slot if $COLL > 0$; (6) $CONT$ is set to 1 if there is a collision or a successful slot; (7) $CONT \leq MAX\_CONT$ and after $CONT$ reaches MAX_CONT, it is again set to 1.

Given in Table 1 is the pseudocode of the base station's role. The reason for setting

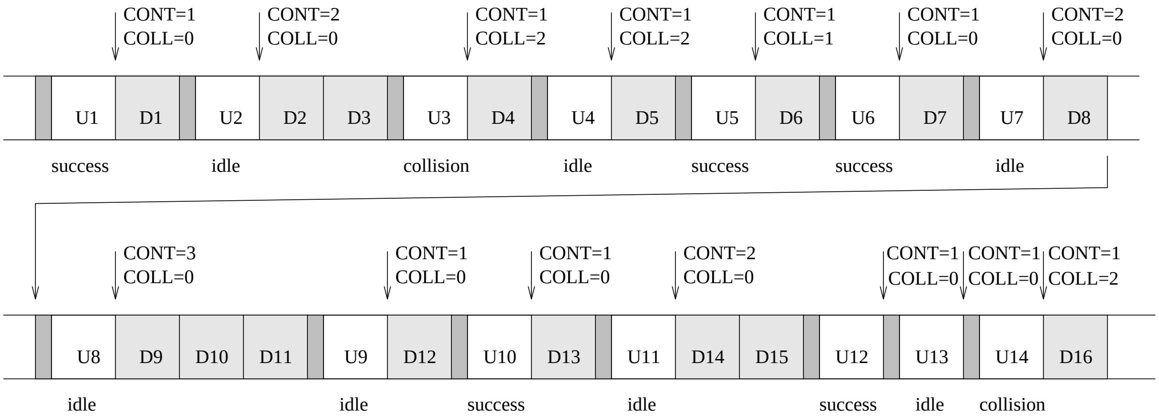


Figure 3: An example of communications using TDD2 mode.

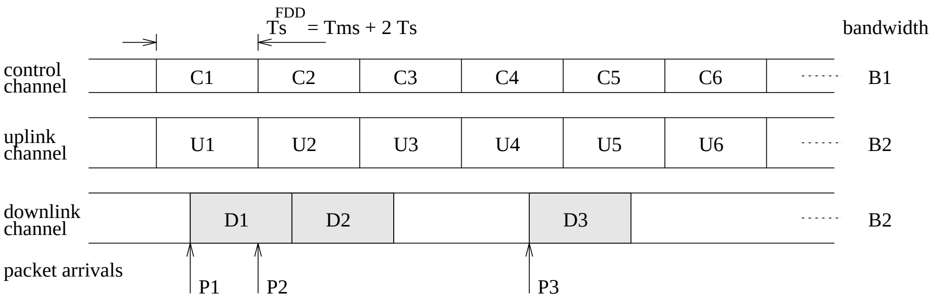


Figure 4: An example of communications using FDD mode.

$COLL := 2$ whenever there is an uplink collision is that the collision among more than two packets is rare, especially under low uplink traffic loads. Note that the basic role of $COLL$ is to avoid increasing $CONT$ when there is a backlogged client. $COLL$ is expected to be a good estimator for the number of backlogged clients under low uplink traffic loads. Under moderate or high uplink traffic loads, $COLL$ will not be able to estimate the number accurately. But, rarely occurs the case when $COLL = 0$ and there is an idle uplink, thus still achieving the purpose of $COLL$. Figure 3 shows an example of communications in a cell with $MAX\_CONT = 3$. We expect TDD2 to work similarly to TDD1 when the uplink is moderately- or heavily- loaded, while it outperforms TDD1 when the uplink is lightly-loaded.

For comparative evaluation of the above two TDD protocols, we consider the following alternative FDD protocol. (The original ALOHA system also used FDD mode [9].)

FDD Mode: As shown in Figure 4, the whole system employs three separate frequency bands for control, uplink, and downlink channels. The uplink is accessed by the slotted

ALOHA scheme where the state of each slot is broadcast by the base station via the control channel.[3] So, we assume here that the base station knows the state of the given slot and then broadcasts it before the next slot. (This can be realized by making the base station check the header of the packet to decide if there was a collision.) The downlink transmission operates like a server with a queue as shown in Figure 4. We assume that FDD mode is assigned the same bandwidth (say B Hz) as TDD modes to make a fair comparison between TDDs and FDD. The control channel is assigned $B_1 = B \cdot T_{ms}/(T_{ms} + 2T_s)$, and both the uplink and downlink channels are assigned $B_2 = B \cdot T_s/(T_{ms} + 2T_s)$. Hence, the slot duration in FDD mode is given by $T_s^{FDD} = T_{ms} + 2T_s$.

3 Throughput and Delay Analysis

3.1 Uplink Slotted ALOHA Access of TDD1 Mode

The analysis in this subsection is based on that of slotted ALOHA in [5,6]. For uplink accesses using slotted ALOHA, we use the model of K_u clients. Our analysis will be based on the following assumptions.

A1. Downlink packets arrive from the wired network (to which the base station is attached) according to a Poisson process. Let λ_d (packets/sec) be the overall packet arrival rate for all connections from the wired network to the mobile clients in the cell.

A2. Packets are generated for transmission at each of the K_u clients according to independent Poisson processes. Let λ_u/K_u be the generation rate at each client, so λ_u (packets/sec) is the overall generation rate by all clients.

A3. A packet is transmitted correctly unless it collides with other packets, i.e., error-free transmission of packets.

A4. Each packet involved in an uplink collision must be retransmitted in a later slot until the packet is successfully received. A client is said to be *backlogged* when it was notified by the base station to have a packet that was not transmitted successfully and hence must be retransmitted.

A5. No buffer at clients, i.e., if one packet at a client is currently waiting for (re)transmission or collided with another packet during transmission, new packets generated at that

[3] We could use a *piggybacking* scheme in which the state of the slot is added into the downlink packets. But, the performance will not be much different from that of the scheme described here.

client are discarded.

Markov Chain Modeling: Since downlink packets are assumed to arrive according to a Poisson process, the number of pending downlink packets, $N(k)$, at the end of the k-th contention slot can be modeled by a Markov chain. The time interval between the k-th and $(k+1)$-th contention slots, $T(n)$, depends on the k-th state, $N(k) = n$:

$$T(n) = \begin{cases} T_{ms} + T_s, & \text{for } n = 0, \\ T_{ms} + 2T_s, & \text{for } n \geq 1. \end{cases} \tag{3.1}$$

The transition probability of moving from state n to $n+i$ is

$$P^d_{n,n+i} = \begin{cases} q^0_a(i), & \text{for } n = 0, \\ q^n_a(i+1), & \text{for } n \geq 1, \end{cases} \tag{3.2}$$

where $q^n_a(i)$ is the probability of i packet arrivals during $T(n)$:

$$q^n_a(i) = e^{-\lambda_d T(n)} \frac{(\lambda_d T(n))^i}{i!}. \tag{3.3}$$

Note that $N(k)$ can decrease by at most one in a single transition. Figure 2 shows state ($=N(k)$) transition times and the values of $N(k)$ at sample moments of $N(k)$. From the state transition probability in Eq. (3.2), we can obtain the steady-state probability:

$$\pi^d_n = \lim_{k \to \infty} Pr\{N(k) = n\}. \tag{3.4}$$

Let $M(k)$ be the number of backlogged clients at the beginning of the k-th contention slot (which is updated with the information in the k-th control packet). Then, the pair $(M(k), N(k-1))$ is modeled by a 2-dimensional Markov chain. (Note that $N(k-1)$ is the number of pending downlink packets at the end of the $(k-1)$-th contention slot.) State ($M(k)$) transitions occur at the beginning of each contention slot as shown in Figure 2. Each of the $M(k)$ backlogged clients will transmit a packet in the k-th contention slot, independently of each other, with probability q_r. Each of the $(K_u - M(k))$ other clients will transmit a packet in the k-th contention slot if one (or more) such packets are generated for the last $T(N(k-1))$ interval. The probability that an unbacklogged client transmits a packet in the k-th contention slot when $N(k-1) = n$ becomes

$$q^n_g = 1 - e^{-\lambda_u T(n)/K_u}. \tag{3.5}$$

Let $Q_g^n(i,m)$ be the probability that i unbacklogged clients transmit packets in the k-th contention slot, and let $Q_r(i,m)$ be the probability that i backlogged clients transmit when $(M(k), N(k-1)) = (m,n)$, then

$$\begin{aligned} Q_g^n(i,m) &= \binom{K_u - m}{i} (1 - q_g^n)^{K_u - m - i} (q_g^n)^i, \\ Q_r(i,m) &= \binom{m}{i} (1 - q_r)^{m-i} (q_r)^i. \end{aligned} \tag{3.6}$$

When $N(k-1) = n$, the conditional state transition probability of moving from state $M(k) = m$ to state $M(k+1) = m+i$ is given by

$$P^u_{m,m+i}(n) = \begin{cases} Q_g^n(i,m), & 2 \leq i \leq K_u - m, \\ Q_g^n(1,m)[1 - Q_r(0,m)], & i = 1, \\ Q_g^n(1,m)Q_r(0,m) + Q_g^n(0,m)[1 - Q_r(1,m)], & i = 0, \\ Q_g^n(0,m)Q_r(1,m), & i = -1. \end{cases} \tag{3.7}$$

Note that $P^u_{m,m+i}(n) = P^u_{m,m+i}(1)$ if $n \geq 1$. The state transition probability of the Markov chain $(M(k), N(k-1))$ from state (m,n) to state $(m+i, n+j)$ is given by

$$P_{(m,n),(m+i,n+j)} = P^u_{m,m+i}(n) P^d_{n,n+j}. \tag{3.8}$$

From this, we can obtain the steady-state probability

$$\pi_{m,n} = \lim_{k \to \infty} Pr\{M(k) = m, N(k-1) = n\}. \tag{3.9}$$

Throughput Analysis: To evaluate the throughput (defined as the long-term fraction of time the channel carries useful information) of the system, we first consider the probability that a given contention slot will be successful (i.e., only one packet is transmitted in that contention slot) for state (m,n),

$$\begin{aligned} P_{suc}(m,n) &\triangleq Pr[Successful\ slot | state(m,n)], \\ &= Q_g^n(1,m)Q_r(0,m) + Q_g^n(0,m)Q_r(1,m). \end{aligned} \tag{3.10}$$

When a packet is transmitted successfully in a contention slot for a given state (m,n), the time used to carry that packet is T_s in $T(n)$ interval from the end of the previous contention slot to the end of the given contention slot. Since the throughput can be calculated by the expected fraction of time spent for transmitting useful information, we define the conditional throughput for a given state (m,n) as

$$P'_{suc}(m,n) = P_{suc}(m,n)T_s/T(n), \tag{3.11}$$

where $P'_{suc}(m,n) = P'_{suc}(m,1)$ for $n \geq 1$. We define two new processes $\hat{N}(t)$ and $\hat{M}(t)$ as $\hat{N}(t) = N(k)$ if $t \in [E_k, E_{k+1})$ and $\hat{M}(t) = M(k)$ if $t \in [B_k, B_{k+1})$, where E_k and B_k are the end and start times of the k-th contention slot, respectively. Note that $\hat{M}(t)$ denotes the number of backlogged clients at time t, and if $t \in [E_k, E_{k+1})$ then $E_{k+1} - E_k = T(\hat{N}(t))$. When the steady-state probability of this process is given by

$$\hat{\pi}_{m,n} = \lim_{t\to\infty} Pr(\hat{M}(t) = m, \hat{N}(t) = n), \tag{3.12}$$

we get

$$\hat{\pi}_{m,0} = \sum_{n_1} \pi_{m,n_1} \frac{\pi_{m,0}T(0)}{\pi_{m,0}T(0) + (\sum_{n_2} \pi_{m,n_2} - \pi_{m,0})T(1)}. \tag{3.13}$$

For a given time t, if $\hat{N}(t) = n$ and $\hat{M}(t) = m$, then the conditional throughput is $P'_{suc}(m,n)$. Now, by averaging this over time, we get the throughput:

$$\begin{aligned} S_u^{TDD1} &= \sum_m \sum_n P'_{suc}(m,n)\hat{\pi}_{m,n}, \\ &= \sum_m \{P'_{suc}(m,0)\hat{\pi}_{m,0} + P'_{suc}(m,1)\hat{\pi}_m^{nz}\}, \end{aligned} \tag{3.14}$$

where $\hat{\pi}_m^{nz} = \sum_n \pi_{m,n} - \hat{\pi}_{m,0}$. Note that since all clients are statistically identical, the individual throughput is given by the value of S_u^{TDD1} in Eq. (3.14) divided by K_u.

Delay Analysis: To derive the average delay, we first need the mean value of $\hat{M}(t)$ at the steady state,

$$E(\hat{M}) = \sum_m \sum_n m\hat{\pi}_{m,n}. \tag{3.15}$$

Since packets are generated according to a Poisson process, the generation time of a packet — generated in $[B_k, B_{k+1})$ for an arbitrary k — will be uniformly distributed in $[B_k, B_k + T(n)]$ if $N(k) = n$ [10]. The probability that a packet is generated in $[B_k, B_{k+1})$ and $N(k) = n$ is $\sum_m \hat{\pi}_{m,n}$. So, the average time from the packet generation to the beginning of next contention slot is given by

$$V = \sum_m \sum_n \frac{T(n)}{2}\hat{\pi}_{m,n}. \tag{3.16}$$

Now, let G_{new} and G_b be the average rate of generating new packets and the average rate at which clients with packets join the backlog, respectively, all measured in packets/sec. Then according to Little's theorem, the average time spent in the backlog is the ratio of the average of backlogged clients to the average join rate or $E(\hat{M})/G_b$. Meanwhile, since G_b is the rate of packets joining the backlog and G_{new} is the rate of new packet generation, a fraction $(G_{new} - G_b)/G_{new}$ of the packets are never backlogged. These packets suffer an average delay of $T_s + V$, where the first term corresponds to the packet-transmission time and the second term corresponds to the time to the next contention slot given in Eq. (3.16). All the others (whose fraction is G_b/G_{new}) suffer the backlog delay mentioned above plus $T_s + V$. The average delay measured is

$$\begin{aligned} D_u^{TDD1} &= \frac{G_{new} - G_b}{G_{new}}(T_s + V) + \frac{G_b}{G_{new}}(T_s + V + \frac{E(\hat{M})}{G_b}), \\ &= T_s + V + \frac{E(\hat{M})}{G_{new}} \quad \text{(seconds)}. \end{aligned} \tag{3.17}$$

For the whole system to be stable, the average rate of new packet generation must equal the average packet departure rate. Since the average packet departure rate measured in packets/slot is equivalent to throughput, we get

$$G_{new} = S_u^{TDD1}/T_s. \tag{3.18}$$

Finally, we get the desired throughput-delay relation under the stable condition

$$D_u^{TDD1} = T_s + V + \frac{E(\hat{M})}{S_u^{TDD1}} T_s \quad \text{(seconds)}. \tag{3.19}$$

3.2 Downlink Packet Broadcast of TDD1 Mode

Since there are no collisions for the downlink, the throughput is given by the long-term fraction of time used for downlink packet transmissions. From the Markov chain, $N(k)$, defined in the previous subsection, we obtain the conditional throughput for given state n as

$$S(n) = \begin{cases} 0, & \text{for } n = 0, \\ \frac{T_s}{T_{ms} + 2T_s}, & \text{for } n \geq 1. \end{cases} \tag{3.20}$$

By adopting the process $\hat{N}(t)$ again and averaging Eq. (3.20) over time, we get the throughput:

$$S_d^{TDD1} = \frac{T_s}{T_{ms} + 2T_s}(1 - \hat{\pi}_0^d), \tag{3.21}$$

where $\hat{\pi}_0^d = \sum_m \hat{\pi}_{m,0}$ or equivalently $\hat{\pi}_0^d = \pi_0^d T(0)/(\pi_0^d T(0) + (1 - \pi_0^d)T(1))$. Note that maximum S_d^{TDD1} is $T_s/(T_{ms} + 2T_s)$. For the downlink transmission to be stable, we must have $\lambda_d \leq 1/(T_{ms} + 2T_s)$, and in this case, the following equation holds:

$$S_d^{TDD1} = \lambda_d T_s. \tag{3.22}$$

Downlink packet transmission can be analyzed by adopting the limited service partially-gated reservation system with a single user in [6], where each mini slot plus contention slot is thought as a fixed-size reservation interval. The average delay is given by

$$D_d^{TDD1} = T_s + \frac{\lambda_d T_s^2 + (1 + \lambda_d T_s)(T_{ms} + T_s)}{2(1 - \lambda_d(T_{ms} + 2T_s))} \quad \text{(seconds)}. \tag{3.23}$$

3.3 Uplink Slotted ALOHA Access of FDD Mode

The analysis of slotted ALOHA access can be found in [5, 6] which is based on the Markov chain with the state being the number of the backlogged clients at the beginning of the k-th slot, $M(k)$. Now, Eq. (3.5) is replaced by

$$q_g = 1 - e^{-\lambda_u T_s^{FDD}/K_u}, \tag{3.24}$$

and the steady-state probability of the Markov chain is given by

$$\pi_m^u = \lim_{k\to\infty} Pr\{M(k) = m\}. \tag{3.25}$$

Similarly, the throughput is obtained by

$$S_u^{FDD} = \sum_m P_{suc}(m)\pi_m^u, \tag{3.26}$$

where

$$P_{suc} = Q_g(1, m)Q_r(0, m) + Q_g(0, m)Q_r(1, m), \tag{3.27}$$

and $Q_g(i, m)$ and $Q_r(i, m)$ are obtained from Eq. (3.6) with q_g instead of q_g^n. The average delay in the stable condition can be shown to be

$$D_u^{FDD} = T_s^{FDD} + V + \frac{E(M)}{S_u^{FDD}} T_s^{FDD} \quad \text{(seconds)}, \tag{3.28}$$

where $V = T_s^{FDD}/2$ and $E(M) = \sum_m m\pi_m^u$.

3.4 Downlink Packet Broadcast in FDD Mode

The throughput S_d^{FDD} can be as high as 1. For the downlink transmission to be stable, the following equation is satisfied:

$$S_d^{FDD} = \lambda_d \cdot T_s^{FDD}, \tag{3.29}$$

where $\lambda_d \leq 1/T_s^{FDD}$. The average delay of the downlink packet can be easily obtained using the M/D/1 queueing system since fixed-size packets arrive according to a Poisson process. It is given by [6]

$$D_d^{FDD} = T_s^{FDD} + \frac{\lambda_d (T_s^{FDD})^2}{2(1-\lambda_d T_s^{FDD})} \quad \text{(seconds)}. \tag{3.30}$$

4 Numerical and Simulation Results

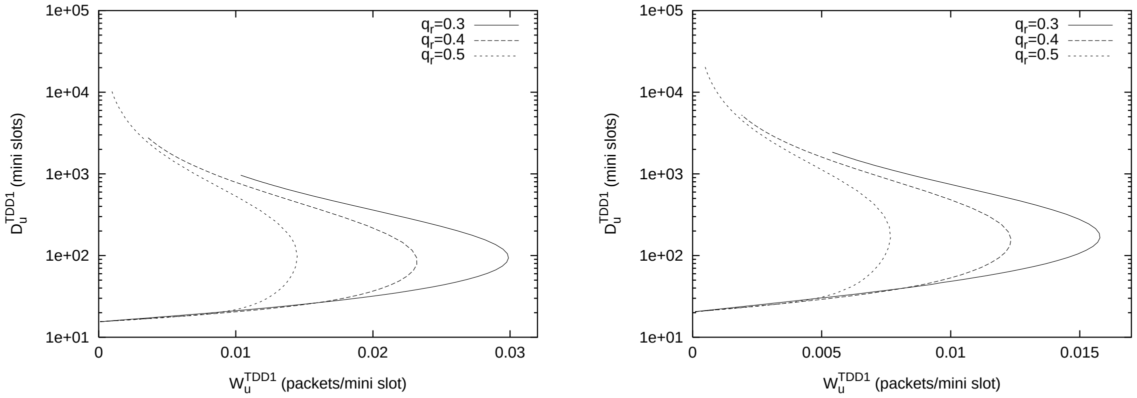
In this section, we show and compare the numerical results from the analysis of TDD1 and FDD in the previous section and the simulation results of TDD2. We will henceforth use T_{ms} as a basic time unit, so the unit of the generation/arrival rates will be packets/mini slot instead of packets/sec, and let $\beta = T_{ms}/T_s$. We consider only the case of $\lambda_d \leq 1/(T_{ms} + 2T_s)$ (e.g., $\approx$ 0.0476 (packets/mini slot) for $\beta = 0.1$) since with this condition, the downlink transmission is stable (more precisely for TDD1 and FDD). To make a fair comparison of throughputs, we define the normalized throughput W in packets/mini slot as S/T where S and T are the throughput and one slot time of the given system, respectively. Hence, we obtain the following relations:

$$\begin{aligned} W_u^{TDD} &= S_u^{TDD}/T_s, \\ W_d^{TDD} &= S_d^{TDD}/T_s = \lambda_d, \\ W_u^{FDD} &= S_u^{FDD}/T_s^{FDD}, \\ W_d^{FDD} &= S_d^{FDD}/T_s^{FDD} = \lambda_d. \end{aligned} \tag{4.1}$$

We also introduce the total throughput, which accounts for both the uplink and downlink, as

$$W_{total} = (S_u + S_d)/T = S_u/T + \lambda_d, \tag{4.2}$$

where S_u and S_d are the uplink and downlink throughputs, respectively, and T is one slot time for the given system. The numerical/simulation results in this section used $\beta = 0.1$ and the number of the clients $K_u = 10$.



(a) $\lambda_d = 0.0001$ (b) $\lambda_d = 0.0476$

Figure 5: Average delay D_u^{TDD1} vs. normalized throughput W_u^{TDD1}; TDD1 mode, while varying λ_u with $\beta = 0.1$, $K_u = 10$.

4.1 Uplink Throughput-Delay Relation of TDD1

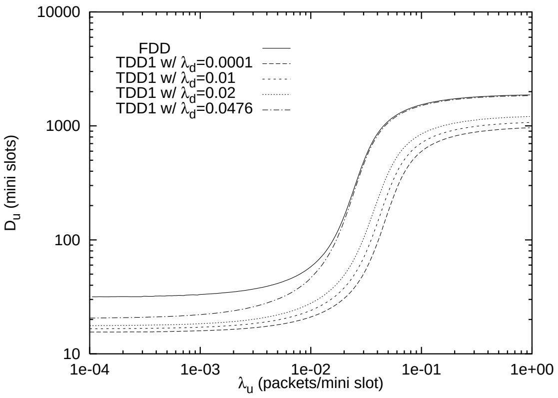
Figure 5 plots the throughput-delay characteristics of uplink access for (a) $\lambda_d = 0.0001$ and (b) $\lambda_d = 0.0476$. Each of the curves in this figure represents one value of q_r with λ_u varying from 0.0001 to 1 along the curve. Note that $\lambda_d = 0.0001$ and $\lambda_d = 0.0476$ are the two (almost) extreme cases in the sense that: (a) for $\lambda_d = 0.0001$, $\pi_0^d \approx 1$, and so downlink packets are rarely transmitted. A control mini-slot and a contention slot alternate with rare downlink packet transmissions. (b) For $\lambda_d = 0.0476$, $\pi_0^d \approx 0$, and so each contention slot is virtually always followed by a downlink packet transmission. A control mini-slot, a contention slot, and a downlink packet transmission appear continuously. Comparison of the two extreme cases indicates that their tendencies are similar. Differences are the capacity (the peak normalized throughput) and the delay (and so the rate λ_u) at which the capacity is attained. It is not difficult to see that

$$\frac{W_{(a)}^*}{W_{(b)}^*} \approx \frac{\lambda_{u,(a)}^*}{\lambda_{u,(b)}^*} \approx \frac{T(1)}{T(0)} = \frac{T_{ms} + 2T_s}{T_{ms} + T_s}, \tag{4.3}$$

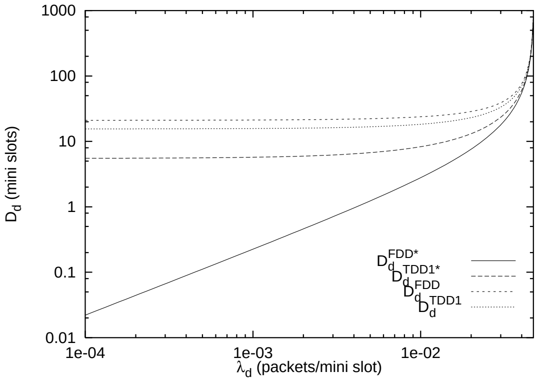
where for each case (i), where $i = a$ or b, $W_{(i)}^*$ is the capacity and $\lambda_{u,(i)}^*$ is the rate at which the capacity is achieved, respectively. It is also notable that for $\lambda_u \approx 0$,

$$D_u^{TDD1} \approx \begin{cases} T_s + \frac{T_{ms}+T_s}{2}, & \text{for } \lambda_d = 0.0001, \\ T_s + \frac{T_{ms}+2T_s}{2}, & \text{for } \lambda_d = 0.0476, \end{cases} \tag{4.4}$$

which are from the $T_s + V$ term in Eq. (3.19), since the backlog delays are negligible.



(a) Uplink; D_u vs. λ_u



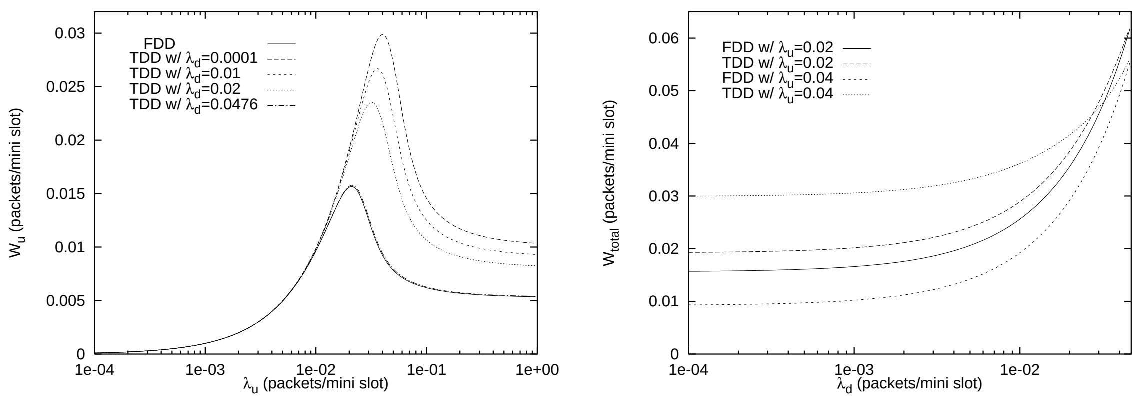
(b) Downlink; D_d vs. λ_d

Figure 6: Average delay D vs. generation/arrival rate λ; TDD1 and FDD modes, $q_r = 0.3$, $\beta = 0.1$, $K_u = 10$.

4.2 Comparison between TDD1 and FDD

Figure 6 (a) plots the uplink delays of FDD and TDD1 modes for various λ_d values. As expected, the smaller λ_d, the smaller delay for TDD1 while $D_u^{TDD1} < D_u^{FDD}$ under all conditions. Note that the difference between FDD and TDD1 with $\lambda_u = 0.0476$ is solely due to the differences in slot duration (or equivalently transmission delay). To observe and understand the downlink delay behavior precisely, we introduce the queueing delays (excluding the transmission delay) as $D_d^{TDD1*} = D_d^{TDD1} - T_s$ and $D_d^{FDD*} = D_d^{FDD} - T_s^{FDD}$. In Figure 6 (b), the various downlink average delays relative to the arrival rate λ_d are plotted: (1) queueing delay for FDD, D_d^{FDD*}; (2) queueing delay for TDD1, D_d^{TDD1*}; (3) whole delay for FDD, D_d^{FDD}; and (4) whole delay for TDD1, D_d^{TDD1}. For a small λ_d, D_d^{TDD1*} is relatively large and different from D_d^{FDD*} since each packet is queued on average for $(T_{ms} + T_s)/2$. As λ_d increases, both D_d^{TDD1*} and D_d^{FDD*} increase while $D_d^{TDD1*} > D_d^{FDD*}$. Now, for the whole delay, we see that $D_d^{TDD1} < D_d^{FDD}$ for all λ_d since the effect of the transmission delay is more than that of queueing delay. Note that all delays approach infinity as λ_d approaches 0.0476.

In Figure 7 (a), the normalized uplink throughput W_u is plotted while varying the generation rate λ_u. The larger λ_d, the less throughput for TDD1. The normalized throughput of FDD and TDD1 with $\lambda_d = 0.0476$ are almost same. We plot the total normalized throughput in Figure 7 (b). As λ_d increases, the increase of W_{total}^{FDD} $(= W_u^{FDD} + \lambda_d)$ is solely due to the λ_d term since W_u^{FDD} is independent of λ_d. Meanwhile, for TDD1, W_u^{TDD1} de-



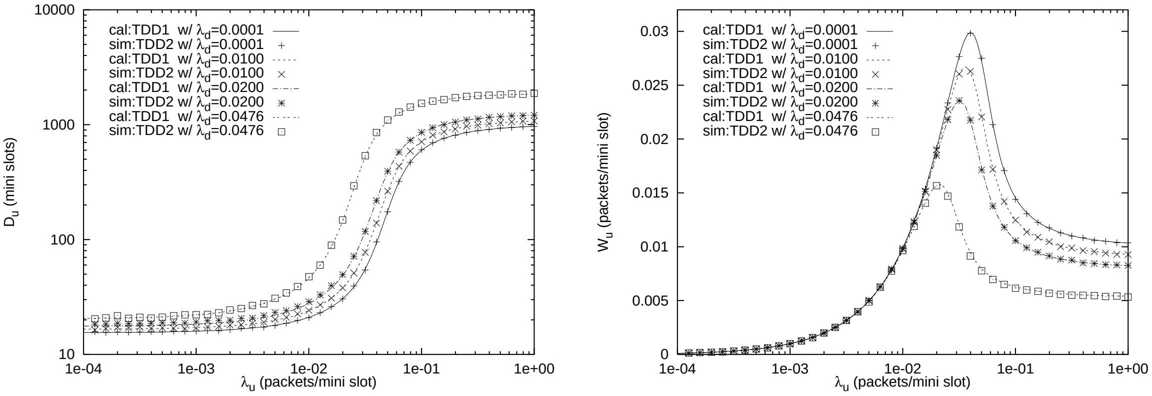
(a) Uplink; W_u vs. λ_u (b) Total; W_{total} vs. λ_d

Figure 7: Normalized throughput W vs. generation/arrival rate λ; TDD1 and FDD modes, $q_r = 0.3$, $\beta = 0.1$, $K_u = 10$.

creases as λ_d increases. Note that W_{total}^{FDD} and W_{total}^{TDD1} meet eventually at $\lambda_d = 0.0476$ while $W_{total}^{FDD} < W_{total}^{TDD1}$ before the meeting point. From the above observations, we conclude that TDD1 always works as well as, or better than, FDD with respect to the packet delay and normalized throughput.

4.3 Comparison between TDD1 and TDD2

We now compare the performances of TDD1 and TDD2 using numeric calculations for TDD1 and simulations for TDD2. For the simulations, we generated Poisson traffic, and followed the assumptions in Section 3.1. Even though we did not plot the simulation results for TDD1, we have observed that the simulation results were very close to the calculated analytical results. (Note that we did not use any approximations for our analysis in the last section.) We use $MAX\_CONT = 5$ for TDD2. In Figure 8 (a), we plot the uplink delays D_u of TDD1 and TDD2 as λ_u increases. We observe that for $\lambda_d = 0.02$ and small λ_u, the uplink delay of TDD2 is slightly larger than that of TDD1 while, for the other cases, it is difficult to differentiate them. In Figure 8 (b), we plot the uplink normalized throughput W_u of TDD1 and TDD2. For all cases, their throughput performances are very close. These results are due to the fact that (1) TDD2 works almost the same as TDD1 under mid and large λ_u values, and (2) under small λ_u, the operation of TDD2 might introduce a slightly larger uplink delay than TDD1, but still rarely affects the throughput because all of the uplink packets are eventually transmitted under small λ_u. Notice that a little larger uplink delay of TDD2 under small λ_u is not a problem since the uplink delays are still relatively



(a) D_u vs. λ_u (b) W_u vs. λ_u

Figure 8: Uplink comparison between TDD1 (calculations) and TDD2 (simulations); $MAX\_CONT = 5$, $q_r = 0.3$, $\beta = 0.1$, $K_u = 10$.

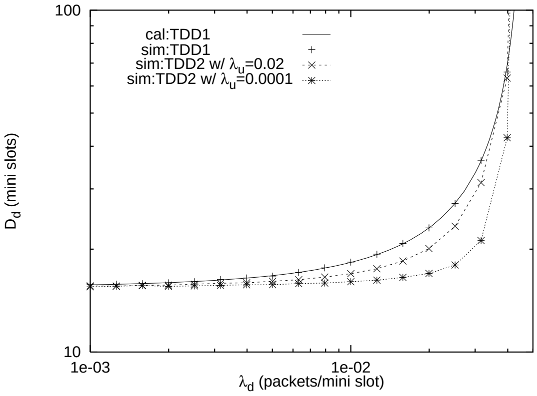


Figure 9: Downlink comparison between TDD1 (with calculation and simulation) and TDD2 (with simulations); $MAX\_CONT = 5$, $q_r = 0.3$, $\beta = 0.1$, $K_u = 10$.

small under this condition.

Figure 9 shows the variations of the downlink delay D_d as the downlink rate λ_d increases for TDD1 and TDD2. First, we observe that the calculation and simulation results for TDD1 are almost the same. For TDD2, we see that there are significant improvements for high λ_d and low λ_u. The higher λ_d and lower λ_u, the larger improvement. But, as λ_d approaches 0.0476, all of the schemes suffer an infinite delay. Even though there is a tradeoff between TDD1 and TDD2, TDD2 is preferable since (1) the downlink delay is reduced notably while the uplink delay is not increased much with TDD2, and (2) the implementation complexity of TDD2 is not much higher than that of TDD1.

5 Conclusion

In this paper, we proposed two centralized wireless MAC protocols (named TDD1 and TDD2) using the slotted ALOHA for uplink control and dynamic TDD transmission. These two protocols differ in how downlink and uplink transmissions are multiplexed. We analyzed the throughput and delay performance of TDD1 using a 2-dimensional Markov chain. TDD1 is analyzed comparatively with an alternative FDD mode protocol. Our numerical results have shown TDD1 to outperform FDD in both throughput and delay. As expected, the uplink performance of TDD1 was notably better than FDD under light downlink traffic loads. We also compared TDD1 (calculations) and TDD2 (simulations): TDD2 outperformed TDD1 with respect to the average downlink delay under light uplink and heavy downlink traffic loads while they performed almost the same in other cases. TDD2 is preferable to TDD1 since the complexity of TDD2 is not much higher than TDD1.

The main contributions of this paper are twofold: (1) analysis of ALOHA-type protocols with dynamic TDD based on a 2-dimensional Markov chain; (2) by comparing with an FDD mode protocol, we showed the efficiency/superiority of the dynamic TDD mode MAC protocols in centralized wireless networks. Recently, we have also investigated the design and analysis of a reservation scheme, which is similar to the reservation ALOHA [11–13], with the dynamic TDD to support the non-real-time communication part of the protocol described in [8]. It was found to be a more attractive protocol with a higher uplink throughput than those schemes considered here due to the nature of the reservation of slots for uplink transmissions.

Acknowledgment

The authors would like to thank N. Song of the University of Michigan for proofreading Section 3.1.

References

[1] K. Pahlavan and A. H. Levesque, "Wireless data communications," *Proceedings of the IEEE*, vol. 82, no. 9, pp. 1398–1430, September 1994.

[2] K. Pahlavan and A. H. Levesque, *Wireless Information Networks*, Wiley-Interscience, New York, NY, 1995.

[3] F. A. Tobagi and L. Kleinrock, "Packet switching in radio channels: Part II-the hidden

terminal problem in CSMA and busy-tone solution," *IEEE Trans. on Communications*, vol. 23, no. 12, pp. 1417–1433, December 1975.

[4] H. Ahmadi, A. Krishna, and R. O. LaMaire, "Design issues in wireless LANs," *Journal of High-Speed Networks*, vol. 5, no. 1, pp. 87–104, 1996.

[5] R. Rom and M. Sidi, *Multiple Access Protocols: Performance and Analysis*, Springer-Verlag, New York, NY, 1990.

[6] D. Bertsekas and R. Gallager, *Data Networks, 2nd ed.*, Prentice-Hall, Englewood Cliffs, New Jersey, 1992.

[7] S. Choi and K. G. Shin, "A wireless local area network for real-time and non-real-time communications," submitted for publication, April 1996.

[8] S. Choi and K. G. Shin, "A cellular local area network with QoS guarantees for heterogeneous traffic," submitted for publication, June 1996.

[9] A. S. Tanenbaum, *Computer Networks, 2nd ed.*, Prentice-Hall, Englewood Cliffs, New Jersey, 1988.

[10] S. M. Ross, *Stochastic Processes*, Wiley, New York, NY, 1983.

[11] S. S. Lam, "Packet broadcast networks—a performance analysis of the R-ALOHA protocol," *IEEE Trans. on Computers*, vol. 29, no. 7, pp. 596–603, July 1980.

[12] S. Tasaka and Y. Ishibashi, "A reservation protocol for satellite packet communication—a performance analysis and stability considerations," *IEEE Trans. on Communications*, vol. 32, no. 8, pp. 920–927, August 1984.

[13] D. Tsai and J.-F. Chang, "Performance study of an adaptive reservation multiple access technique for data transmissions," *IEEE Trans. on Communications*, vol. 34, no. 7, pp. 725–727, July 1986.