Medium Access Control Protocols With Memory

Jaeok Park, *Member, IEEE*, and Mihaela van der Schaar, *Fellow, IEEE*

***Abstract*—Many existing medium access control (MAC) protocols utilize past information (e.g., the results of transmission attempts) to adjust the transmission parameters of users. This paper provides a general framework to express and evaluate distributed MAC protocols utilizing a finite length of memory for a given form of feedback information. We define protocols with memory in the context of a slotted random access network with saturated arrivals. We introduce two performance metrics, throughput and average delay, and formulate the problem of finding an optimal protocol. We first show that a time-division multiple access (TDMA) outcome, which is the best outcome in the considered scenario, can be obtained after a transient period by using a protocol with $(N-1)$-slot memory, where N is the total number of users. Next, we analyze the performance of protocols with one-slot memory using a Markov chain and numerical methods. Protocols with one-slot memory can achieve throughput arbitrarily close to 1 (i.e., 100% channel utilization) at the expense of large average delay by correlating successful users in two consecutive slots. Finally, we apply our framework to wireless local area networks (WLANs).**

***Index Terms*—Access control, access protocols, communication systems, distributed decision-making, multiaccess communication.**

I. Introduction

In MULTIACCESS communication systems, multiple users share a communication channel and contend for access. Medium access control (MAC) protocols are used to coordinate access and resolve contention among users. We can categorize MAC protocols[1] into two classes, centralized and distributed protocols, depending on the existence of a central entity that coordinates the transmissions of users. An example of centralized protocols is time-division multiple access (TDMA), where a scheduler assigns time slots to users. Centralized control can achieve a high level of channel utilization by avoiding collisions, but it requires large overhead for the communication of control messages. Examples of distributed protocols include slotted Aloha and distributed coordination function (DCF). In the slotted Aloha protocol [1], users transmit new packets in the next time slot while retransmitting backlogged packets with a fixed probability. In the IEEE 802.11 DCF [2], carrier-sense multiple access with collision avoidance (CSMA/CA) and binary slotted exponential backoff (EB) are used for users to determine their transmission times. These distributed protocols can be implemented without explicit control messages, but coordination is limited in that collisions may occur or the channel may be unused when some users have packets to send.

In this paper, we aim to improve the degree of coordination attainable with distributed protocols by introducing *memory* into the MAC layer. Under a protocol with memory, a node dynamically adjusts its transmission parameters based on the history of its local information. The idea of utilizing histories at the MAC layer can be found in various existing protocols. For example, the slotted Aloha protocol in [1] and its generalized version in [3] adjust the transmission probabilities of nodes depending on whether the current packet is new or backlogged. The pseudo-Bayesian algorithm in [4] utilizes channel feedback to update the estimated number of backlogged packets in the system, based on which the transmission probability is determined. The EB protocols in [5] use the results of transmission attempts to adjust the contention window and current window sizes of nodes or their transmission probabilities.

The above protocols, however, utilize available past information in a limited way. We can consider the operation of the above protocols as the current state of a node determining its transmission parameters and the state transition occurring based on its local observations. Although this structure makes implementation simple in that nodes can simply keep track of their states in order to make transmission decisions, there may be many possible paths that lead to the same state, and important information may be lost by aggregating different histories into a single state. For example, in the slotted Aloha protocol in [1], a node with a backlogged packet uses the same transmission probability following a slot in which it waited and following a slot in which it transmitted and collided. However, it can distinguish these two outcomes, and a significant performance improvement may be achieved by using different transmission probabilities following the two outcomes. Another limitation of the above protocols is that they are designed assuming a particular form of feedback information. In case that more informative feedback is available, utilizing the additional information may result in performance gains. For example, the EB protocols in [5] prescribe that a node should not update its parameters following a waiting slot. If a node can sense the channel while waiting, utilizing the information obtained from sensing may improve the performance of the EB protocols.

In order to overcome the limitations of the existing protocols utilizing memory, we provide a systematic framework to express and evaluate protocols with memory in the context of a slotted multiaccess system with *saturated arrivals* where nodes make transmission decisions based on their transmission probabilities. The assumption of saturation with a fixed number of users allows us to analyze the utilization of past information in a setting where all nodes in the system collect information constantly. Our framework allows us to formally express a protocol utilizing memory of any finite length and operating under any

Manuscript received June 02, 2009; revised January 05, 2010 and May 10, 2010; accepted May 11, 2010; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor E. Modiano. Date of publication June 03, 2010; date of current version December 17, 2010.

The authors are with the Electrical Engineering Department, University of California, Los Angeles, CA 90095 USA (e-mail: jaeok@ee.ucla.edu; mihaela@ee.ucla.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNET.2010.2050699

[1]Since we deal with MAC protocols exclusively in this paper, we use the term "protocol" to represent "MAC protocol" hereafter.

form of feedback information. Also, we introduce two performance metrics, throughput and average delay, based on which we can evaluate protocols with memory. The two main results of this paper can be summarized as follows.

1) In the considered scenario with saturated arrivals, TDMA is the best protocol in the sense that there is no other protocol that achieves a higher throughput or a smaller average delay than TDMA. A TDMA outcome can be obtained after a transient period by a protocol with $(N-1)$-slot or N-slot memory, where N is the total number of nodes in the system.
2) A protocol with one-slot memory can achieve throughput arbitrarily close to 1 (i.e., 100% channel utilization) at the expense of large average delay, by correlating successful nodes in two consecutive slots (i.e., a node that has a successful transmission in a slot has a high probability of success in the next slot).

The proposed protocols with memory can be related to splitting algorithms [6] and reservation Aloha [7]. In splitting algorithms such as tree algorithms [8], backlogged nodes are divided into groups, one of which transmits in the next slot. Protocols with memory use histories to split nodes into groups. As nodes randomly access the channel using transmission probabilities, histories will evolve differently across nodes as time passes. The probability of a successful transmission can be made high by choosing transmission probabilities in a way that the expected size of the transmitting group is approximately one most of the time. In reservation Aloha, nodes maintain frames with a certain number of slots, and a successful transmission serves as a reservation for the same slot in the next frame. Reservation Aloha can thus be expressed as a protocol with memory whose length is equal to the number of slots in a frame, provided that all nodes can learn successful transmissions in the system. Protocols with memory are more flexible than reservation Aloha in that protocols with memory can specify different transmission probabilities in nonreserved slots, can make reservations in a probabilistic way, and can be implemented with an arbitrary form of feedback information.

The rest of this paper is organized as follows. In Section II, we describe the considered slotted multiaccess model and define protocols with memory. In Section III, we introduce performance metrics and formulate the problem of finding an optimal protocol. In Section IV, we show that a protocol with $(N-1)$-slot or N-slot memory can achieve the performance of TDMA. In Section V, we analyze the properties of protocols with one-slot memory using numerical methods. In Section VI, we show that protocols with memory can be applied to a wireless local area network (WLAN) environment and can achieve a performance improvement over DCF. We conclude the paper in Section VII.

II. System Model

A. Setup

We consider a slotted multiaccess system as in [3]. The system has total N contending users, or transmitter nodes, and the set of users is denoted by $\mathcal{N} \triangleq \{1,\ldots,N\}$. We assume that the number of users is fixed over time and known to users. Users share a communication channel through which they transmit packets. Time is slotted, and users are synchronized in their slot transmission times. We label slots by $t = 1,2,\ldots$. A user always has packets to transmit and can attempt to transmit one packet in each slot. The set of actions available to a user in a slot is denoted by $A \triangleq \{T,W\}$, where T stands for "transmit" and W for "wait." We denote the action of user i by $a_i \in A$ and an action profile, or *a transmission outcome*, by $\mathbf{a} \triangleq (a_1,\ldots,a_N)$. The set of (transmission) outcomes is denoted by $\mathcal{A} \triangleq A^N$. A transmission is successful if it is the only transmission in the slot, and two or more transmissions in the same slot result in a collision. The major notation used in this paper is summarized in Table I.

TABLE I
Major Notation

$\mathcal{N} = \{1,\ldots,N\}$	set of users
$A = \{T,W\}$	set of actions (T: transmit, W: wait)
$a_i \in A$	action of user i
$\mathcal{A} = A^N$	set of action profiles
$\mathbf{a} \in \mathcal{A}$	action profile (or outcome)
$\mathbf{a}^i$	outcome such that $a_i = T$ and $a_j = W\ \forall\ j \neq i$
ρ	feedback technology
Z_ρ	set of feedback information under ρ
$z_i \in Z_\rho$	feedback information of user i
τ_i	throughput of user i
τ	total throughput
D_i	average delay of user i
D	average delay of a user under a symmetric protocol
$\tilde{D}_i$	average interpacket time

B. Feedback Information

After a user transmits a packet, it learns whether the packet is successfully transmitted or not using an acknowledgement (ACK) response. If the user receives an ACK from the receiver node, it learns that its transmission was successful. Otherwise, it concludes that its packet has collided. We assume that there is no error in the transmission and the reception of ACK responses so that a user always learns the correct results of its transmission attempts. Formally, we represent ACK feedback to user i by $ACK_i(\mathbf{a})$, which takes the value *yes* if $\mathbf{a} = \mathbf{a}^i$, where $\mathbf{a}^i$ is the outcome in which only user i transmits, and *no* otherwise.

At the end of each slot, users obtain channel feedback about the number of transmissions in that slot. Let $\mathcal{K} \triangleq \{0,1,\ldots,N\}$ be the set of all possible numbers of transmissions in a slot. We represent a channel feedback model by an information partition H of $\mathcal{K}$. An information partition H assigns $h(k) \in H$ to each $k \in \mathcal{K}$ in such a way that $k \in h(k)$ for all k [9]. The set $h(k)$ consists of the numbers of transmissions that users regard as possible when the actual number of transmissions is k. When outcome $\mathbf{a}$ occurs, users receive channel feedback $h(k(\mathbf{a}))$, where $k(\mathbf{a})$ is the number of transmissions in $\mathbf{a}$.

Consider a partition of $\mathcal{K}$, $\{\{0\},\{1\},e\}$, where $e \triangleq \{2,\ldots,N\}$. Each element of the partition represents a type of channel feedback, and users can potentially learn whether there has been no packet (idle), one packet (success), or more than one packet (collision) transmitted in the slot. With an abuse of notation, we will use 0 and 1 to represent channel feedback that corresponds to $\{0\}$ and $\{1\}$, respectively [6]. We say that the channel possesses ternary feedback, or $(0,1,e)$ feedback, if all the three types of feedback are available. Binary feedback is also possible, and given the three possible types of channel feedback, we can consider three kinds of binary feedback: success/failure (S/F) feedback, which informs users

whether there was a successful transmission (1) or not $(0 \cup e)$; collision/no collision (C/NC) feedback, which informs users whether there was a collision (e) or not $(0 \cup 1)$; and empty/not empty (E/NE) feedback, which informs users whether the current slot was empty (0) or not $(1 \cup e)$ [10]. We can also consider no channel feedback, which does not give any channel information to users and corresponds to information partition $\{\mathcal{K}\}$, and $(N+1)$-ary feedback, which informs users of the exact number of transmissions and corresponds to information partition $\{\{0\},\{1\},\ldots,\{N\}\}$, the finest partition of $\mathcal{K}$. (See [11] for a similar list of channel feedback models in a multiple-reception scenario.)

The *feedback information* of user i consists of ACK feedback and channel feedback, i.e., $(ACK_i(\mathbf{a}), h(k(\mathbf{a})))$. We define a *feedback technology* as a rule that generates feedback information for each user depending on transmission outcomes. We can represent a feedback technology by a mapping $\rho : \mathcal{S} \rightarrow 2^{\mathcal{K}}$, where $\mathcal{S} \triangleq \cup_{\mathbf{a}\in\mathcal{A}}\{(a_i, k(\mathbf{a}))\}$, and write the feedback information of user i more compactly as $\rho(a_i, k(\mathbf{a}))$. Since $ACK_i(\mathbf{a}) = no$ if $a_i = W$, we set $\rho(W, k(\mathbf{a})) = h(k(\mathbf{a}))$. Since user i can distinguish the outcomes in 1 from those in e using ACK feedback whenever it transmits, we set $\rho(T, k(\mathbf{a})) = 1$ if $ACK_i(\mathbf{a}) = yes$ and $h(k(\mathbf{a})) \cap e$ if $ACK_i(\mathbf{a}) = no$. Let $Z_\rho \triangleq \cup_{\mathbf{a}\in\mathcal{A}}\{\rho(a_i, k(\mathbf{a}))\}$, which is independent of i since users receive the same channel feedback. Z_ρ represents the set of feedback information that a user can obtain with feedback technology ρ. The feedback information of user i is denoted by $z_i \in Z_\rho$. We use $\mathcal{R}$ to denote the set of all feedback technologies, which is equivalent to the set of all partitions of $\mathcal{K}$.

Remark: Using an information partition to represent a channel feedback model in fact leads to several simplifications. For analytic convenience, we use the simplified channel feedback structures to obtain the results in this paper. However, it is possible to carry out a similar analysis with more general channel feedback structures. First, we can allow asymmetry in channel feedback across users by specifying user-dependent information partitions and having user i receive channel feedback $h_i(\mathbf{a}) \in H_i$ when outcome $\mathbf{a}$ is chosen, where H_i is an information partition for user i. Second, we can admit errors in channel feedback by allowing the possibility that $k(\mathbf{a}) \notin h_i(\mathbf{a})$. This possibility can model situations such as the hidden terminal problem (also called "erasures" in [10]), where users may interpret a success or a collision slot as an idle slot, and noise errors [10], where users may interpret an idle or a success slot as a collision slot. Lastly, we can consider stochastic channel feedback, in which case the channel feedback of user i is determined according to a probability distribution on H_i, $\triangle(H_i)(\mathbf{a})$.

C. Protocols With Memory

A user decides whether to transmit or not in each slot using a transmission probability, which lies in [0,1]. A protocol is a rule based on which users determine their transmission probabilities. We assume that control or coordination messages cannot be used in the system. Then, the transmission action and the feedback information of user i are all the information that it obtains in a slot. The *M-slot history* for user i in slot t is given by

$$L_i^t = \left(a_i^{t-M}, z_i^{t-M}; \ldots; a_i^{t-1}, z_i^{t-1}\right)$$

for $M = 1,2,\ldots$ and $t = 1,2,\ldots$.[2] We set $(a_i^{t'}, z_i^{t'}) = (W, \rho(W,0))$ for $t' \leq 0$ as initialization. Let $\mathcal{L}_\rho$ be the set of all possible action–feedback pairs under feedback technology ρ, i.e., $\mathcal{L}_\rho \triangleq \cup_{\mathbf{a}\in\mathcal{A}}\{(a_i, \rho(a_i, k(\mathbf{a})))\} \subset A \times Z_\rho$. Then, the set of all M-slot histories is given by $\mathcal{L}_\rho^M$. A stationary decision rule based on M-slot histories is defined by a mapping

$$f : \mathcal{L}_\rho^M \rightarrow [0,1]$$

where $f(L)$ represents the transmission probability for a user whose M-slot history is $L \in \mathcal{L}_\rho^M$.

Since the set of M-slot histories $\mathcal{L}_\rho^M$ is affected by feedback technology ρ, the set of stationary decision rules based on M-slot histories depends on the feedback technology of the system. In particular, as the feedback technology is more informative in the sense that the corresponding information partition is finer, users can distinguish more outcomes, and thus more decision rules can be deployed. We use $\mathcal{F}_{M,\rho}$ to denote the set of all stationary decision rules based on M-slot histories under feedback technology ρ. We define a *protocol with M-slot memory* as a profile of stationary decision rules based on M-slot histories given a feedback technology ρ, i.e., $\mathbf{f} \triangleq (f_1, \ldots, f_N) \in \mathcal{F}_{M,\rho}^N$. We say that a protocol $\mathbf{f}$ is *symmetric* if it prescribes the same decision rule to every user, i.e., $f_1 = \cdots = f_N$. We will sometimes use f to represent a symmetric protocol $\mathbf{f}$ with a common decision rule f when there is no confusion.

D. Automaton Representations of Protocols With Memory

A protocol with memory can be described by a finite automaton, which consists of a finite set of states, an initial state, an action rule, and a state transition rule [12]. We can represent a symmetric protocol $f \in \mathcal{F}_{M,\rho}$ as a finite automaton by defining states as possible M-slot histories, the initial state as the M-slot history obtained from M idle slots, the action rule as the decision rule f, and the state transition rule as to specify the new state as the M-slot history updated based on the transmission action and the feedback information in the current slot.

Fig. 1(a) shows the automaton representation of a protocol with memory in the simplest case of one-slot memory and no channel feedback (i.e., ACK feedback only). With ACK feedback only, there are three possible action–feedback pairs, or one-slot histories: $(W, \emptyset)$, $(T, 1)$, and (T, e), where $\emptyset$ corresponds to channel feedback $\mathcal{K}$. Hence, there are three states in the automaton, corresponding to the three possible action–feedback pairs, with the initial state specified as $(W, \emptyset)$. The transmission probability in each state (a, z) is given by $f(a, z)$. The state in the next slot is determined by the action–feedback pair in the current slot.

[3] proposes generalized slotted Aloha protocols, which we call two-state protocols. Under a two-state protocol, users choose their transmission probabilities depending on whether their current packets are new (a free state) or have collided before (a backlogged state). Thus, having ACK feedback suffices to implement a two-state protocol. The automaton representation of a two-state protocol is shown in Fig. 1(b). In

[2] In addition to actions and feedback, a user knows the transmission probabilities it has used. We do not include past transmission probabilities in histories because we focus on protocols that do not depend on past transmission probabilities directly.

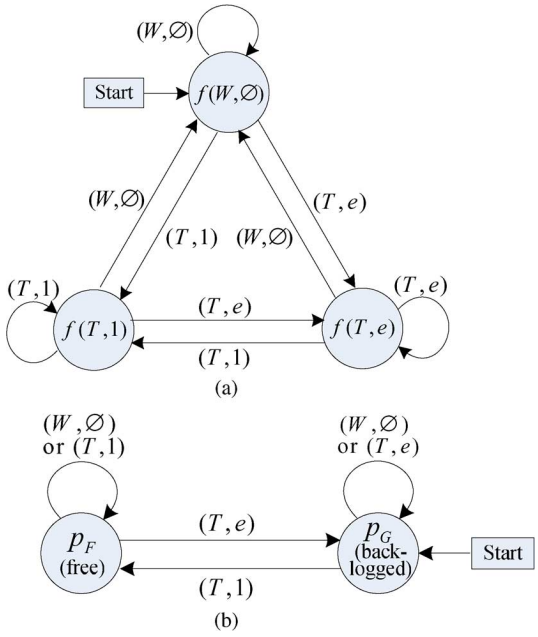

Fig. 1. Automaton representations of protocols: (a) a protocol with one-slot memory, and (b) a two-state protocol.

general, a two-state protocol cannot be expressed as a protocol with finite memory. Since the state remains the same following a waiting slot, the transmission probability for the current slot is determined by the result of the most recent transmission attempt, which may have occurred arbitrarily many slots ago. However, [3] shows that under saturated arrivals, two-state protocols achieve the maximum throughput when $p_{\mathrm{F}} = 1$ and $p_{\mathrm{G}} \to 0$, where p_{F} and p_{G} are transmission probabilities in the free and backlogged states, respectively. When a two-state protocol specifies $p_{\mathrm{F}} = 1$, which is also the case in the slotted Aloha protocol in [1], the action–feedback pair $(W, \emptyset)$ cannot occur in the free state, and thus we can find an equivalent protocol with one-slot memory such that $f(T,1) = 1$ and $f(W,\emptyset) = f(T,e) = p_{\mathrm{G}}$. In effect, a two-state protocol with $p_{\mathrm{F}} = 1$ aggregates the two different action–feedback pairs, $(W,\emptyset)$ and (T,e), into one state, the backlogged state, and assigns the same transmission probability following these two action–feedback pairs. On the contrary, a protocol with one-slot memory can fully utilize all the available information from the previous slot in that three different transmission probabilities can be assigned following the three action–feedback pairs.

As will be shown later in Section V-D, the limited utilization of past information by two-state protocols results in performance limitations. In particular, throughput, or the fraction of success slots, is bounded from above by $N/(2N-1)$ under two-state protocols [3, Theorem 2], while protocols with one-slot memory can achieve throughput arbitrarily close to 1. When memory longer than one slot or channel feedback is available, there are more distinguishable histories for each user. Two-state protocols aggregate different histories into just two states, whereas protocols with memory can assign as many transmission probabilities as the number of possible histories given the length of memory and the feedback technology. Thus, the performance gap between the two classes of protocols will be larger when longer memory or more informative feedback is available.

III. Problem Formulation

A. Performance Metrics

1) Throughput: We define the *throughput* of a user as the fraction of slots in which it has a successful transmission, and *total throughput* as the fraction of slots in which there is a successful transmission in the system, which is equal to the sum of the individual throughput of users. When users follow a protocol with memory $\mathbf{f}$, throughput can be computed using a Markov chain. Let M be the length of memory used by protocol $\mathbf{f}$, and ρ be the associated feedback technology. Then, we can consider a Markov chain whose state space is given by $\mathcal{A}^M$. We write an element of $\mathcal{A}^M$ as $\bar{\mathbf{a}} \triangleq (\mathbf{a}_1, \ldots, \mathbf{a}_M)$. Let $L_i(\bar{\mathbf{a}})$ be the M-slot history for user i when the outcomes in the recent M slots are $\bar{\mathbf{a}}$, i.e.,

$$L_i(\bar{\mathbf{a}}) = (a_{i,1}, \rho(a_{i,1}, k(\mathbf{a}_1)); \ldots; a_{i,M}, \rho(a_{i,M}, k(\mathbf{a}_M)))$$

where $a_{i,m}$ is the ith element of $\mathbf{a}_m$. Given $L_i(\bar{\mathbf{a}})$ as its M-slot history, user i transmits with probability $f_i(L_i(\bar{\mathbf{a}}))$. The transmission probabilities of users yield a probability distribution on the outcome in the current slot. The probability that the current outcome is $\mathbf{a}'$ under $\mathbf{f}$ when the outcomes in the recent M slots are $\bar{\mathbf{a}}$ is given by

$$P(\mathbf{a}'|\bar{\mathbf{a}};\mathbf{f}) = \prod_{i=1}^{N} \left[\mathbf{1}_{\{\mathbf{a}:a_i=T\}}(\mathbf{a}') f_i\left(L_i(\bar{\mathbf{a}})\right) + \mathbf{1}_{\{\mathbf{a}:a_i=W\}}(\mathbf{a}')\left(1 - f_i\left(L_i(\bar{\mathbf{a}})\right)\right)\right] \quad (1)$$

where $\mathbf{1}_{\mathcal{B}} : \mathcal{A} \to \{0,1\}$ is an indicator function such that $\mathbf{1}_{\mathcal{B}}(\mathbf{a}) = 1$ if $\mathbf{a} \in \mathcal{B}$, and 0 otherwise. The transition probability from $\bar{\mathbf{a}} \in \mathcal{A}^M$ to $\bar{\mathbf{a}}' \triangleq (\mathbf{a}'_1, \ldots, \mathbf{a}'_M) \in \mathcal{A}^M$ under protocol $\mathbf{f}$ is given by

$$Q(\bar{\mathbf{a}}'|\bar{\mathbf{a}};\mathbf{f}) = P\left(\mathbf{a}'_M|\bar{\mathbf{a}};\mathbf{f}\right)$$

if $\mathbf{a}'_m = \mathbf{a}_{m+1}$, for all $m = 1, \ldots, M-1$, and 0 otherwise.

Let $\mathbf{v}_t(\mathbf{f})$ be the probability distribution on the state space $\mathcal{A}^M$ in slot t induced by protocol $\mathbf{f}$. By the initialization in the definition of protocols with memory, the initial distribution $\mathbf{v}_0(\mathbf{f})$ has element 1 for $\bar{\mathbf{a}} = (\mathbf{a}^0, \ldots, \mathbf{a}^0)$, and 0 elsewhere, where $\mathbf{a}^0$ denotes the idle outcome, $(W, \ldots, W)$. Let $\mathbf{Q}(\mathbf{f})$ be the transition matrix of the Markov chain under protocol $\mathbf{f}$. Then, the probability distribution on $\mathcal{A}^M$ in slot t can be computed by

$$\mathbf{v}_t(\mathbf{f}) = \mathbf{v}_0(\mathbf{f})\mathbf{Q}(\mathbf{f})^t$$

for $t = 1, 2, \ldots$. Let $\mathcal{S}^i$ be the subset of $\mathcal{A}^M$ with user i's success as the most recent outcome, i.e., $\mathcal{S}^i \triangleq \{\bar{\mathbf{a}} \in \mathcal{A}^M : \mathbf{a}_M = \mathbf{a}^i\}$. Then, the probability of user i's success in slot t is given by $\tau_{i,t}(\mathbf{f}) \triangleq \sum_{\bar{\mathbf{a}} \in \mathcal{S}^i} v_t(\bar{\mathbf{a}};\mathbf{f})$. The fraction of slots with user i's success, or the throughput of user i, is given by

$$\tau_i(\mathbf{f}) \triangleq \lim_{J\to\infty} \frac{1}{J} \sum_{t=1}^{J} \tau_{i,t}(\mathbf{f})$$

assuming that the limit exists. If $\mathbf{f}$ is chosen so that the induced Markov chain has only one closed communicating class, then there exists a unique stationary distribution $\mathbf{v}(\mathbf{f})$, independent of the initial distribution $\mathbf{v}_0(\mathbf{f})$, which satisfies

$$\mathbf{v}(\mathbf{f}) = \mathbf{v}(\mathbf{f})\mathbf{Q}(\mathbf{f}) \quad \text{and} \quad \mathbf{v}(\mathbf{f})\mathbf{e} = 1 \tag{2}$$

where $\mathbf{e}$ is the column vector of length $|\mathcal{A}^M|$ whose elements are all 1 [13]. Then, the expression for the throughput of user i is reduced to $\tau_i(\mathbf{f}) = \sum_{\bar{\mathbf{a}} \in \mathcal{S}^i} v(\bar{\mathbf{a}}; \mathbf{f})$. Finally, the total throughput of the system under protocol $\mathbf{f}$ is given by $\tau(\mathbf{f}) = \sum_{i=1}^{N} \tau_i(\mathbf{f})$.

2) Average Delay: The *average delay* of a user is defined as the average waiting time, measured in the unit of slots, until the beginning of its next successful transmission starting from an arbitrarily chosen time. Average delay under a protocol with memory $\mathbf{f}$ can be computed using a Markov chain. Consider a slot to which an arbitrarily chosen time belongs, and let $\bar{\mathbf{a}} \in \mathcal{A}^M$ be the outcomes in the recent M slots. Given $\bar{\mathbf{a}}, \mathbf{f}$ yields a probability distribution on the outcome in the next slot, $P(\mathbf{a}'|\bar{\mathbf{a}}; \mathbf{f})$, given by Equation (1). Using $P(\mathbf{a}'|\bar{\mathbf{a}}; \mathbf{f})$, we can compute the probability that user i succeeds for the first time after r slots when the outcomes in the recent M slots are given by $\bar{\mathbf{a}}$ and users follow $\mathbf{f}$

$$\mu_i(r; \bar{\mathbf{a}}, \mathbf{f}) \triangleq \Pr\{\mathbf{a}_{t+r} = \mathbf{a}^i \quad \text{and} \quad \mathbf{a}_{t+r'} \neq \mathbf{a}^i, \quad \text{for all} \quad r' = 1, \ldots, r-1 | \bar{\mathbf{a}}_t = \bar{\mathbf{a}}; \mathbf{f}\} \tag{3}$$

for $r = 1, 2, \ldots$. For example

$$\mu_i(1; \bar{\mathbf{a}}, \mathbf{f}) = P(\mathbf{a}_{t+1} = \mathbf{a}^i | \bar{\mathbf{a}}_t = \bar{\mathbf{a}}; \mathbf{f})$$

and

$$\mu_i(2; \bar{\mathbf{a}}, \mathbf{f}) = \sum_{\mathbf{a} \in \mathcal{A} \setminus \{\mathbf{a}^i\}} P\left(\mathbf{a}_{t+2} = \mathbf{a}^i | \bar{\mathbf{a}}_{t+1} = (\bar{\mathbf{a}}_t, \mathbf{a}); \mathbf{f}\right) \times P(\mathbf{a}_{t+1} = \mathbf{a} | \bar{\mathbf{a}}_t = \bar{\mathbf{a}}; \mathbf{f})$$

where $(\bar{\mathbf{a}}_t, \mathbf{a}) \in \mathcal{A}^M$ is obtained by deleting the first outcome in $\bar{\mathbf{a}}_t$ and adding $\mathbf{a}$ as the most recent outcome. Using (3), we can compute the average number of slots until user i's next success starting from a slot with the recent M outcomes $\bar{\mathbf{a}}$

$$d_i(\bar{\mathbf{a}}; \mathbf{f}) \triangleq \sum_{r=1}^{\infty} \mu_i(r; \bar{\mathbf{a}}, \mathbf{f}) r.$$

Since the distribution on the recent M outcomes in slot t is given by $\mathbf{v}_t(\mathbf{f})$, the average delay of user i under protocol $\mathbf{f}$ can be computed as

$$D_i(\mathbf{f}) \triangleq \lim_{J \to \infty} \frac{1}{J} \sum_{t=1}^{J} \sum_{\bar{\mathbf{a}} \in \mathcal{A}^M} v_t(\bar{\mathbf{a}}; \mathbf{f}) d_i(\bar{\mathbf{a}}; \mathbf{f}) - 0.5$$

where 0.5 is subtracted to take into account that the average time staying in the initial slot is 0.5 starting from an arbitrarily chosen time. When the Markov chain has a unique stationary distribution $\mathbf{v}(\mathbf{f})$, the expression for the average delay of user i is reduced to

$$D_i(\mathbf{f}) = \sum_{\bar{\mathbf{a}} \in \mathcal{A}^M} v(\bar{\mathbf{a}}; \mathbf{f}) d_i(\bar{\mathbf{a}}; \mathbf{f}) - 0.5.$$

3) Discussion on Throughput and Average Delay: The throughput of the system can be considered as an efficiency measure of a protocol as it gives the channel utilization over time. However, as its definition suggests, throughput reflects the performance averaged over a long period of time and does not contain much information about the short-term and medium-term performance. As an illustration, consider two sequences of outcomes generated by repeating $(\mathbf{a}^1, \mathbf{a}^2, \ldots, \mathbf{a}^N)$ and $(\mathbf{a}^1, \mathbf{a}^1, \mathbf{a}^1, \mathbf{a}^1, \mathbf{a}^1, \mathbf{a}^2, \mathbf{a}^2, \mathbf{a}^2, \mathbf{a}^2, \mathbf{a}^2, \ldots, \mathbf{a}^N, \mathbf{a}^N, \mathbf{a}^N, \mathbf{a}^N, \mathbf{a}^N)$. In both sequences, each user succeeds in one out of N slots over a long period of time, as measured by throughput $1/N$. However, users may prefer the first sequence to the second one as the first exhibits the steadier performance over a short period of time. For example, suppose that a user counts its successes in N consecutive slots from an arbitrary slot. The first sequence guarantees one success in N consecutive slots regardless of the initial slot. On the contrary, in the second sequence, N consecutive slots contain no success with a high probability and many successes with a low probability. The variation in the performance over a short period of time is captured by average delay.

A widely used measure of delay in queueing theory is the interpacket interval. In our model, we can compute the *average interpacket time* of user i, i.e., the average number of slots between two successes of user i, using

$$\tilde{D}_i(\mathbf{f}) = \sum_{\bar{\mathbf{a}} \in \mathcal{S}^i} \frac{v(\bar{\mathbf{a}}; \mathbf{f})}{\sum_{\bar{\mathbf{a}} \in \mathcal{S}^i} v(\bar{\mathbf{a}}; \mathbf{f})} d_i(\bar{\mathbf{a}}; \mathbf{f}). \tag{4}$$

We can interpret $d_i(\bar{\mathbf{a}}; \mathbf{f})$ as the average time to enter one of the states in $\mathcal{S}^i$ starting from the state $\bar{\mathbf{a}}$ under protocol $\mathbf{f}$. Then, $D_i(\mathbf{f})$ can be interpreted as the average time to enter $\mathcal{S}^i$ starting from an arbitrary state, where the starting state is chosen following the stationary distribution. Similarly, $\tilde{D}_i(\mathbf{f})$ can be interpreted as the average time to return $\mathcal{S}^i$ starting from a state in $\mathcal{S}^i$.

Note that $d_i(\bar{\mathbf{a}}; \mathbf{f})$ satisfies

$$d_i(\bar{\mathbf{a}}; \mathbf{f}) = 1 + \sum_{\bar{\mathbf{a}}' \in \overline{\mathcal{S}^i}} Q(\bar{\mathbf{a}}' | \bar{\mathbf{a}}, \mathbf{f}) d_i(\bar{\mathbf{a}}'; \mathbf{f}) \tag{5}$$

for all $\bar{\mathbf{a}} \in \mathcal{A}^M$, where $\overline{\mathcal{S}^i}$ is the complement of $\mathcal{S}^i$, i.e., $\overline{\mathcal{S}^i} \triangleq \mathcal{A}^M \setminus \mathcal{S}^i$. Using (2) and (5), we can show that $\sum_{\bar{\mathbf{a}} \in \mathcal{S}^i} v(\bar{\mathbf{a}}; \mathbf{f}) d_i(\bar{\mathbf{a}}; \mathbf{f}) = 1$ for any protocol with memory $\mathbf{f}$, and thus (4) can be rewritten as

$$\tilde{D}_i(\mathbf{f}) = \frac{1}{\tau_i(\mathbf{f})} \tag{6}$$

which can be regarded as a version of Little's Theorem [6]. Therefore, the average interpacket time is completely determined by throughput, and thus it provides no additional information beyond that provided by throughput.

The waiting time paradox [14] suggests that not only the mean of the interpacket time but also its variance matters for the average waiting time measured from an arbitrarily chosen time. Let X_i be the random variable that represents the interpacket time of user i with support $\mathbb{N} \triangleq \{1, 2, \ldots\}$. By definition, we have $E(X_i) = \tilde{D}_i$. The Pollaczek–Khinchine (P–K) formula

[15] gives the average residual time until the next success, or average delay, by

$$D_i = \frac{E\left(X_i^2\right)}{2E(X_i)} = \frac{1}{2}\left(1+\kappa_i^2\right)\tilde{D}_i \tag{7}$$

where κ_i represents the coefficient of variation of X_i, i.e., $\kappa_i = \sqrt{Var(X_i)}/E(X_i)$. The P–K formula shows that average delay is increasing in the variance of the interpacket time for a given level of throughput (or equivalently, for a given mean of the interpacket time).

Consider the following three examples with a fixed level of the average interpacket time $\tilde{D}_i$ and different levels of the coefficient of variation κ_i. 1) Suppose that user i has periodic successful transmissions. Then, it transmits a packet successfully once in $\tilde{D}_i$ slots, and its average delay is $\tilde{D}_i/2$ since $\kappa_i = 0$. Note that $\tilde{D}_i/2$ is the smallest average delay achievable with protocols that yield the average interpacket time $\tilde{D}_i$. 2) Suppose that successes for user i are completely random in the sense that X_i follows a geometric distribution. Such a random variable can be generated by a memoryless protocol that prescribes a fixed transmission probability for each user. In this case, $\kappa_i^2 = 1 - 1/\tilde{D}_i$, and thus $D_i = \tilde{D}_i - 0.5$. 3) Suppose that the successes of user i are highly correlated over time. Then, $\kappa_i \gg 1$, and thus $D_i \gg \tilde{D}_i$. In this case, user i has frequent successes for a short period of time, but sometimes has to wait for a long period of time until its next success.

It is reasonable to assume that users prefer to have a steady stream of transmissions as well as a high transmission rate. As the above examples illustrate, the coefficient of variation of the interpacket time measures the volatility in successful transmissions over time. Since average delay reflects the coefficient of variation of the interpacket time, we use it as a second performance metric to complement the long-term performance metric, throughput.

B. Protocol Design Problem

We formulate the problem faced by the protocol designer as a two-stage procedure. First, the protocol designer chooses the length of memory, M slots, and the feedback technology, ρ. Next, the protocol designer chooses a protocol from the class of available protocols given the stage-one choice (M, ρ). For simplicity, we assume that the protocol designer considers only symmetric protocols. For a symmetric protocol f, we have $\tau_1(f) = \cdots = \tau_N(f)$ and $D_1(f) = \cdots = D_N(f)$. Total throughput is given by $\tau(f) = N\tau_i(f)$ for any $i \in \mathcal{N}$, and we use $D(f)$ to denote the average delay of each user.

The protocol design problem can be written formally as

$$\max_{M\in\mathbb{N},\rho\in\mathcal{R}} \left\{ \left(\max_{f\in\mathcal{F}_{M,\rho}} U\left(\tau(f), D(f)\right) \right) - C(M,\rho) \right\}. \tag{8}$$

U represents the utility function of the protocol designer, defined on total throughput and average delay. We assume that U is increasing in total throughput and decreasing in average delay so that the protocol designer prefers a protocol that yields high total throughput and small average delay. C represents the cost function of the protocol designer, defined on the length of memory and the feedback technology. We assume that C is increasing as memory is longer and as the feedback technology is more informative. Note that from a practical point of view, the cost of expanding memory is vanishingly small compared to the cost associated with the feedback technology, which implies that the system is more likely to be constrained by the available feedback technology rather than by the size of memory.[3] Hence, it is more natural to interpret the cost associated with memory as the cost of implementing protocols with a certain length of memory. For instance, the protocol designer may prefer protocols with short memory to protocols with long memory because the former is easier to program and validate than the latter. We say that a protocol f is *optimal given* (M, ρ) if it attains $\max_{f\in\mathcal{F}_{M,\rho}} U(\tau(f), D(f))$. We say that f is an *optimal protocol* for the protocol designer if f solves (8).

In order to analyze the performance of protocols with memory, we approach the protocol design problem from two different directions. In Section IV, we first find the feasible throughput-delay pair most preferred by the protocol designer. Then, we show that the most preferred throughput-delay pair can be achieved by a protocol with memory. In Section V, we focus on the simplest class of protocols with memory, namely protocols with one-slot memory, and investigate their properties using numerical methods.

IV. Most Preferred Protocols

Suppose that the protocol designer can choose *any* protocol at no cost. Then, the protocol design problem becomes

$$\max_{f\in\mathcal{F}} U\left(\tau(f), D(f)\right) \tag{9}$$

where $\mathcal{F}$ denotes the set of all symmetric protocols including those that cannot be expressed as protocols with memory. We say that f is a *most preferred protocol* if f solves (9). Consider a fixed level of total throughput τ, which implies individual throughput τ/N by the symmetry assumption on protocols. Note that the P–K formula (7) was obtained without imposing any structure on protocols, and it shows that for a given level of throughput, average delay is minimized when there is no variation in the interpacket time of a user, i.e., when $\kappa_i = 0$. Hence, combining (6) and (7), we can express the minimum average delay given total throughput τ as

$$D_{\min}(\tau) = \frac{\tilde{D}_i}{2} = \frac{N}{2\tau}.$$

Since $D_{\min}(\tau)$ is decreasing in τ, setting total throughput at the maximum level 1 yields the minimum feasible average delay $D_{\min} = D_{\min}(1) = N/2$. In other words, there is no protocol $f \in \mathcal{F}$ that attains $D(f)$ less than $N/2$. Hence, if there exists a protocol f that achieves the maximum total throughput 1 and the minimum average delay $N/2$ at the same time, then f is a most preferred protocol.

In order to obtain the most preferred throughput-delay pair $(1, N/2)$, a protocol needs to provide each user with a successful transmission in every N slots. TDMA is a protocol that achieves such a sequence of outcomes. Since the indices for users are arbitrary, we can describe the TDMA protocol as having user i transmit in slot t if $(t \bmod N) = (i \bmod N)$ and wait in all other slots. Then, each user has one successful transmission in every N slots, and thus the TDMA protocol is a most preferred protocol. However, the TDMA protocol requires coordination

[3]Our formulation includes the case where there is only one available feedback technology, say $\bar{\rho}$, in the system. In this case, we have $C(M,\rho) = +\infty$ for all $\rho \neq \bar{\rho}$.

messages to be sent to users in order to assign time slots, and thus it does not belong to the class of protocols with memory defined in Section II-C. The following theorem establishes the existence of a most preferred protocol in the class of protocols with memory.

Theorem 1: Assume $(N-1)$-slot memory and success/failure (S/F) binary feedback ρ_{SF}. Denote an $(N-1)$-slot history by $L=(a^1,z^1;\ldots;a^{N-1},z^{N-1})$, and let $n(L)$ be the number of successes in L, i.e., $n(L)=\left|\{m: z^m=1, m=1,\ldots,N-1\}\right|$. Define a protocol with memory $f\in\mathcal{F}_{(N-1),\rho_{\mathrm{SF}}}$ by

$$f(L)=\begin{cases}0, & \text{if } L \text{ contains } (T,1)\\ 1/\left(N-n(L)\right), & \text{otherwise.}\end{cases}$$

Then, $\tau(f)=1$ and $D(f)=N/2$.

Proof: Given $(z^1,\ldots,z^{N-1})$, define a state $(\tilde{z}^1,\ldots,\tilde{z}^{N-1})$ by setting $\tilde{z}^m=S$ if $z^m=1$ and $\tilde{z}^m=F$ otherwise, for $m=1,\ldots,N-1$. Let $\tilde{n}$ be the count of S in $(\tilde{z}^1,\ldots,\tilde{z}^{N-1})$. Note that $\tilde{n}$ takes an integer value between 0 and $N-1$. Under the protocol f in the theorem, $N-\tilde{n}$ users transmit with probability $1/(N-\tilde{n})$, while remaining $\tilde{n}$ users wait following a history that yields $(\tilde{z}^1,\ldots,\tilde{z}^{N-1})$. Hence, the state in the next slot is given by $(\tilde{z}^2,\ldots,\tilde{z}^{N-1},S)$ with probability $(1-1/(N-\tilde{n}))^{N-\tilde{n}-1}$ and by $(\tilde{z}^2,\ldots,\tilde{z}^{N-1},F)$ with probability $1-(1-1/(N-\tilde{n}))^{N-\tilde{n}-1}$. This defines a Markov chain with state space $\{S,F\}^{N-1}$. Note that S occurs with probability 1 when $\tilde{n}=N-1$ while both S and F occur with positive probability when $0\le\tilde{n}\le N-2$. Hence, state $(S,\ldots,S)$ is an absorbing state while all the other states are transient. By [13, Theorem 11.3], the absorbing state $(S,\ldots,S)$ is reached with probability 1. Once the absorbing state is reached, a TDMA outcome is obtained, and thus f is a most preferred protocol. ∎

Theorem 1 shows that with $(N-1)$-slot memory and S/F feedback, N users can determine their transmission slots in a distributed way without need of explicit coordination messages, thereby achieving the same outcome as TDMA after a transient period. By trial and error, users find their transmission slots in a self-organizing manner and stabilize to transmit once in every N slots. Note that $(N-1)$ is the minimum length of memory required to emulate TDMA using a protocol with memory. A cycle of successes by N users cannot be generated with memory shorter than $(N-1)$ slots. Also, S/F feedback is necessary to guarantee a success after $(N-1)$ consecutive successes.

The expected duration of a transient period can be computed using the fundamental matrix for the Markov chain constructed in the proof of Theorem 1 with the initial state specified as $(F,\ldots,F)$ (see [13, Theorem 11.5]). We can shorten the expected duration of a transient period by using N-slot memory as follows. Consider a protocol with N-slot memory, $\tilde{f}\in\mathcal{F}_{N,\rho_{\mathrm{SF}}}$, defined by

$$\tilde{f}(L)=\begin{cases}1, & \text{if } (a^1,z^1)=(T,1)\\ 0, & \text{if } (a^1,z^1)=(W,1)\\ 0, & \text{if } z^1\neq 1 \text{ and } (a^m,z^m)=(T,1)\\ & \quad \text{for some } m=2,\ldots,N\\ 1/\left(N-\tilde{n}(L)\right), & \text{if } z^1\neq 1 \text{ and } (a^m,z^m)\neq(T,1)\\ & \quad \text{for all } m=2,\ldots,N\end{cases} \tag{10}$$

where $\tilde{n}(L)$ is the number of successes in recent $(N-1)$ slots. The first and the third lines of (10) state that once a user succeeds, it waits for $(N-1)$ slots before the next transmission. The second and the fourth lines state that a user with no success in recent N slots waits if the current slot is already "reserved" by some user that succeeded N slots ago and contends with other such users if no user succeeded N slots ago. Hence, under $\tilde{f}$ in (10), once a user succeeds in slot t, it is guaranteed to succeed in slots $t+N$, $t+2N$, and so on, whereas under f in Theorem 1, a user that succeeded N slots ago has to compete with other users with no success in recent $(N-1)$ slots during a transient period. This additional feature of $\tilde{f}$ results in a reduced expected duration of a transient period. Using a similar approach as in the proof of Theorem 1, we can construct a Markov chain with state space $\{S,F\}^N$ to show that $\tilde{f}$ is a most preferred protocol and to compute the expected duration of a transient period under $\tilde{f}$. We obtain that, by using $\tilde{f}$ instead of f, the expected number of slots until obtaining $(N-1)$ consecutive successes for the first time is reduced from 6.38 to 4.92 when $N=3$, from 16.12 to 8.52 when $N=4$, and from 38.39 to 12.67 when $N=5$. Again, S/F feedback is necessary to let users know whether the current slot is reserved or not. $\tilde{f}$ can be considered as a generalization of reservation Aloha with a frame of N slots, where users can adjust their transmission probabilities depending on the number of contending users for nonreserved slots.

In practice, the number of users may vary over time as users join and leave the system. When ternary feedback is available, users can find the exact number of users in the system using the protocol $\tilde{f}$ as long as the number of users does not change too frequently. The initial estimate on the number of users is set to be the maximum number of users that the system can allow. When users follow $\tilde{f}$ based on their initial estimate, there will be empty slots in a cycle obtained after a transient period. Once a cycle is repeated, users can construct a new cycle by deleting the empty slots, adjusting the number of slots in a frame equal to the actual number of users. If some users leave the network, empty slots will appear in a frame, and the remaining users can reduce the length of a frame by deleting the empty slots. If some users join the network, we require them to transmit immediately. Then, a collision will occur, and it serves as a signal to notify the existing users that a new user arrived in the system. Once a collision occurs after a transient period, users reset their estimates to the maximum number of users and repeat the procedure from the beginning in order to increase the length of a frame by the number of arriving users.

V. Protocols With One-Slot Memory

A. Structure of Delay-Efficient Protocols

We now examine protocols with one-slot memory, which are the simplest among protocols with memory. Under protocols with one-slot memory, users determine their transmission probabilities using their action–feedback pairs in the previous slot. In Appendix A, we explain in detail how to compute throughput and average delay under symmetric protocols with one-slot memory using a Markov chain. Since the protocol designer prefers a protocol with small average delay for a given level of throughput, we consider the following reduced protocol design problem:

$$D^*(\tau)\triangleq\min_{f\in\mathcal{F}_{1,\rho}} D(f) \quad \text{subject to} \quad \tau(f)=\tau \tag{11}$$

for $\tau\in[0,1]$ and some feedback technology ρ. We say that a protocol f is *delay-efficient* if it solves (11) for $\tau=\tau(f)$, i.e., if

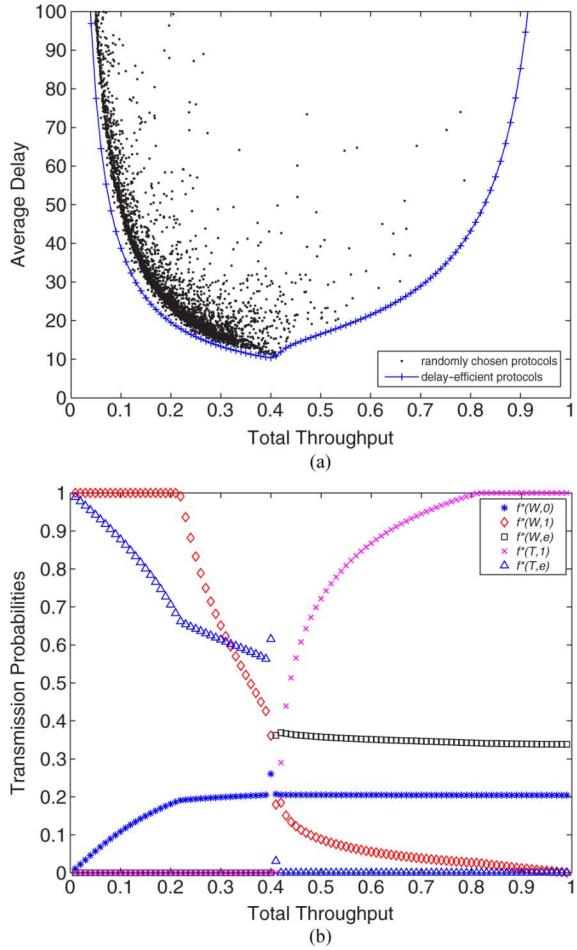

Fig. 2. Delay-efficient protocols with one-slot memory: (a) total throughput and average delay under delay-efficient protocols and randomly chosen protocols, and (b) transmission probabilities of delay-efficient protocols.

$D(f) = D^*(\tau(f))$. Also, we call the set of points $\{(\tau, D^*(\tau)) : \tau \in [0,1]\}$ the *delay-efficiency boundary* of protocols with one-slot memory. Since computing $\tau(f)$ and $D(f)$ for a given protocol $f \in \mathcal{F}_{1,\rho}$ involves solving matrix equations, it is in general difficult to solve (11) analytically, and thus we rely on numerical methods.

To obtain numerical results, we consider ternary feedback, denoted by ρ_3, and five users, i.e., $N = 5$. In order to guarantee the existence of a stationary distribution for transition matrix $\mathbf{Q}(f)$, we restrict the range of f (i.e., the set of possible transmission probabilities) to be $[10^{-4}, 1-10^{-4}]$, instead of [0,1]. We use the function *fmincon* of MATLAB to solve the restricted version of (11).[4] Fig. 2(a) depicts the delay-efficient boundary of protocols with one-slot memory, shown as a U-shaped curve. Since the optimization problem to find delay-efficient protocols is not necessarily convex, it is possible that the numerical results locate local minima instead of global minima. To validate the delay-efficiency of the protocols obtained by the *fmincon* function, we generate 5000 symmetric protocols with one-slot memory where transmission probabilities are randomly chosen on [0,1] and plot total throughput and average delay under those protocols as dotted points in Fig. 2(a) (in the figure, only 3631 points with average delay less than 100 are shown). The results in Fig. 2(a) suggest that the numerically computed protocols are indeed (at least approximately) delay-efficient.[5] Fig. 2(b) plots the transmission probabilities under the delay-efficient protocols, denoted by f^*, as τ varies. It shows that the structure of f^* changes around $\tau = 0.41$, which is also the turning point of the delay-efficiency boundary. There is numerical instability for τ between 0.41 and 0.48 in that the solution depends highly on the initial protocol. In fact, for τ in that region we can find protocols that yield average delay slightly smaller than those in Fig. 2(a) by specifying different initial protocols.

In the following, we provide an explanation for the shape of the delay-efficiency boundary by investigating the structure of the delay-efficient protocols. In the low throughput region ($\tau \leq 0.21$), the structure of f^* is given by

$$f^*(W,0) \in [0, 0.2], \quad f^*(W,1) = 1, \quad f^*(W,e) = 0$$
$$f^*(T,1) = 0, \quad f^*(T,e) \in [0.65, 1].$$

Since $f^*(T,1) = 0$ and $f^*(W,1) = 1$, a success lasts for only one slot and is followed by a collision. Since $f^*(T,e)$ is high, a collision is likely to be followed by another collision. Similarly, an idle slot is likely to be followed by another idle slot since $f^*(W,0)$ is low. This means that total throughput is kept low by inducing many idle or collision slots between two successes. Hence, as total throughput increases in the low throughput region, the expected number of idle or collision slots between two successes is reduced, and as a result users transmit their packets more frequently. According to the P–K formula (7), average delay is determined by the mean and the coefficient of variation of the interpacket time. On the left-hand side of the delay-efficiency boundary ($\tau \leq 0.4$), the average interpacket time is reduced while the coefficient of variation of the interpacket time remains about the same as τ increases, resulting in an inverse relationship between throughput and average delay.

In the high total throughput region ($\tau \geq 0.82$), the structure of f^* is given by

$$f^*(W,0) \approx \frac{1}{N}, \quad f^*(W,1) \approx 0, \quad f^*(W,e) \approx \frac{1}{N-2}$$
$$f^*(T,1) = 1, \quad f^*(T,e) = 0. \tag{12}$$

In a slot following an idle slot, users transmit with probability close to $1/N$. In a slot following a success, the successful user transmits with probability 1, while other users wait with a high probability. The transmission probability of other users approaches 0 as τ becomes close to 1. In a slot following a collision, users that transmitted in the collision wait, while other users transmit with probability close to $1/(N-2)$. Since a collision involving two transmissions is most likely among all kinds of collisions, setting $f^*(W,e) = 1/(N-2)$ maximizes the probability of success given that colliding users wait. $f^*(T,1) = 1$ and $f^*(W,1) \approx 0$ is the key feature of protocols with one-slot memory that achieve high total throughput. By correlating successful users in two consecutive slots, protocols

[4]We vary τ from 0.99 to 0.01 with a step size of 0.01. We choose the initial protocol for *fmincon* as $(f(W,0), f(W,1), f(W,e), f(T,1), f(T,e)) = (1/N, 0, 1/(N-2), 1, 0)$ for $\tau = 0.99$ and use the solution for $\tau = n \times 0.01$ as the initial protocol for $\tau = (n-1) \times 0.01$ from $n = 99$ to $n = 2$.

[5]Another interesting point to notice from Fig. 2(a) is that most of the randomly chosen protocols yield throughput-delay pairs close to those achieved by memoryless protocols. It suggests that protocols with one-slot memory need to be designed carefully in order to attain total throughput above the maximum level achievable with memoryless protocols.

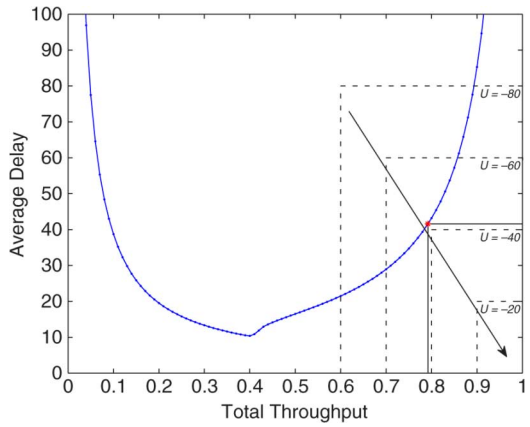

Fig. 3. Illustration of an optimal protocol.

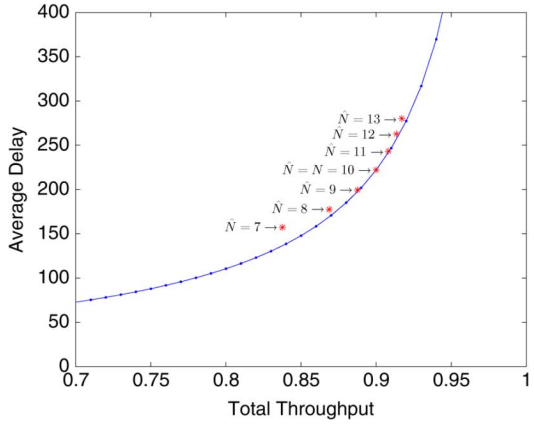

Fig. 4. Total throughput and average delay under delay-efficient protocols based on the estimated numbers of users when the actual number of users is ten.

with one-slot memory can yield total throughput arbitrarily close to 1. However, higher throughput is achieved by allowing a successful user to use the channel for a longer period, which makes other users wait longer until they transmit next time. On the right-hand side of the delay-efficiency boundary ($\tau > 0.4$), the coefficient of variation of the interpacket time increases without bound as τ approaches 1, resulting in a positive relationship between throughput and average delay.

Since the protocol designer prefers a protocol that yields high total throughput for a given level of average delay, optimal protocols must lie on the right-hand side of the delay-efficiency boundary. As an illustrative example, suppose that the utility function of the protocol designer is given by

$$U(\tau, D) = -\max\{200(1-\tau), D\}.$$

Then, an increase in total throughput by 0.1 has the same utility consequence as a decrease in average delay by 20 slots. In Fig. 3, dashed curves depict the indifference curves of the utility function, each of which represents the throughput-delay pairs that yield the same level of utility, while the arrow shows the increasing direction of the utility function. The throughput-delay pair that maximizes the utility of the protocol designer is (0.792, 41.6), marked with an asterisk in Fig. 3. The protocol designer can determine the optimal protocol f^{o} given $(M, \rho) = (1, \rho_3)$ by finding a protocol $f \in \mathcal{F}_{1,\rho_3}$ that yields $(\tau(f), D(f)) = (0.792, 41.6)$, which is

$$\begin{aligned} &f^{\mathrm{o}}(W,0) = 0.20, \quad f^{\mathrm{o}}(W,1) = 0.03, \quad f^{\mathrm{o}}(W,e) = 0.34 \\ &f^{\mathrm{o}}(T,1) = 0.99, \quad f^{\mathrm{o}}(T,e) = 0. \end{aligned} \tag{13}$$

B. Robustness Properties of Delay-Efficient Protocols

1) Unknown Number of Users: We relax the assumption that users know the exact number of users in the system. Instead, we assume that each user has an estimate on the number of users and executes the prescribed protocol based on the estimate.[6] We consider a scenario where there are 10 users and ternary feedback is available. For simplicity, we assume that the objective of the protocol designer is to achieve total throughput 0.9 and that all users have the same estimate on the number of users, which will be the case when users use the same estimation method based on past channel feedback. Fig. 4 shows a portion of the delay-efficiency boundary with $N = 10$ and plots the throughput-delay pairs when the 10 users follow delay-efficient protocols that are computed to achieve total throughput 0.9 based on the estimated number of users, denoted by $\hat{N}$, between 7 and 13. The results from Fig. 4 suggest the robustness of delay-efficient protocols with respect to variations in the number of users in the sense that as the estimated number of users is close to the actual one, the protocol designer obtains a performance close to the desired one.

Based on the observations from Fig. 4, we can consider the following procedure to dynamically adjust the estimates of users. Users update their estimates periodically by comparing the actual total throughput since the last update with the desired total throughput.[7] Users increase (resp. decrease) their estimates by one if the actual total throughput is lower (resp. higher) than the desired total throughput by a certain threshold level. When designed carefully, this estimation procedure will make the estimated number of users converge to the actual number of users because the actual total throughput is lower than (resp. equal to, higher than) the desired total throughput if the estimated number of users is smaller than (resp. equal to, larger than) the actual number. This procedure can be regarded as an extension of the pseudo-Bayesian algorithm of [4] in which users adjust their estimates in every slot based on the channel feedback of the previous slot. In the proposed estimation procedure, memory is utilized not only to coordinate transmissions but also to estimate the number of users.

2) Errors in Feedback Information: So far, we have restricted our attention to deterministic feedback technologies. We relax this restriction and consider stochastic feedback technologies. As discussed in Section II-B, stochastic channel feedback for user i can be represented by a mapping h_i from $\mathcal{A}$ to $\triangle(H_i)$. If the structure of channel feedback, $(h_1, \ldots, h_N)$, is known to the protocol designer, it can be modeled in the transition matrix $\mathbf{Q}(f)$ in Section III by extending the state space from $\mathcal{A}^M$ to $(\mathcal{L}_\rho^N)^M$. Then, the protocol designer can find an optimal protocol taking into account randomness in feedback information.

[6] In other words, the protocol designer specifies not a fixed protocol but a protocol as a function of the numbers of users.

[7] Note that users can compute the actual total throughput using ternary feedback.

TABLE II
TOTAL THROUGHPUT AND AVERAGE DELAY WITH A STOCHASTIC FEEDBACK TECHNOLOGY

		Total throughput	Average delay
Analysis		0.7920	41.5935
Simulation	$\epsilon = 0$	0.7910	41.2375
	$\epsilon = 0.01$	0.7667	37.4377
	$\epsilon = 0.02$	0.7441	33.4907
	$\epsilon = 0.03$	0.7235	31.4114
	$\epsilon = 0.05$	0.6844	28.0600
	$\epsilon = 0.07$	0.6467	25.2149
	$\epsilon = 0.10$	0.6049	22.9282
	$\epsilon = 0.20$	0.4996	19.0503

Here, we introduce random errors in channel feedback that are not modeled by the protocol designer, and we examine the performance of an optimal protocol in the presence of random errors. We assume that ternary feedback is available but subject to random errors. In particular, a user obtains the correct feedback signal with probability $1-2\epsilon$ and each of the two incorrect signals with probability ϵ, for a small $\epsilon > 0$. For example, if there is no transmission in the system, then a user receives feedback 0 with probability $1-2\epsilon$ and each of feedback 1 and e with probability ϵ. We assume that the feedback signals of users are independent. We continue to assume that ACK feedback is perfect, and thus transmitting users always learn the correct results of their transmission attempts, regardless of the realization of channel feedback.

Table II shows the performance of the optimal protocol f^{o} in (13) at the various levels of ϵ when $N = 5$. To obtain the simulation results, we generate transmission decisions and feedback information for 100 000 slots, for each level of ϵ. Table II suggests that delay-efficient protocols have a robustness (or continuity) property with respect to random errors in feedback information, since the obtained performance is close to the desired one when the error level is small. Note that an error occurring to a waiting user following a success induces the user to transmit with a higher probability (i.e., to transmit with probability $f^{\mathrm{o}}(W,0)$ or $f^{\mathrm{o}}(W,e)$ instead of $f^{\mathrm{o}}(W,1)$), making consecutive successes last shorter. Thus, as the error level increases, both total throughput and average delay decrease. The obtained throughput-delay pairs remain close to the delay-efficiency boundary, suggesting that errors following an idle or a collision slot cause little performance degradation.

C. Comparison of Channel Feedback

We now analyze the impact of the different forms of channel feedback on the performance of protocols with one-slot memory. As mentioned in Section II-C, the set of available protocols $\mathcal{F}_{M,\rho}$ expands as the feedback technology ρ becomes more informative (in other words, a protocol $f \in \mathcal{F}_{M,\rho}$ can always be replicated by another protocol $f' \in \mathcal{F}_{M,\rho'}$ if ρ' is more informative than ρ). This relationship implies that a more informative feedback technology yields a lower delay-efficiency boundary for a given length of memory. We consider six feedback technologies with ACK feedback and different channel feedback models: no channel feedback, S/F binary feedback, C/NC binary feedback, E/NE binary feedback, ternary feedback, and $(N+1)$-ary feedback, as introduced in Section II-B. Fig. 5 depicts the delay-efficiency boundaries of protocols with one-slot memory under the six feedback technologies. As expected, for a given level of throughput, the minimum average delay becomes smaller as we move from no feedback to binary feedback, to ternary feedback, and to $(N+1)$-ary feedback.[8]

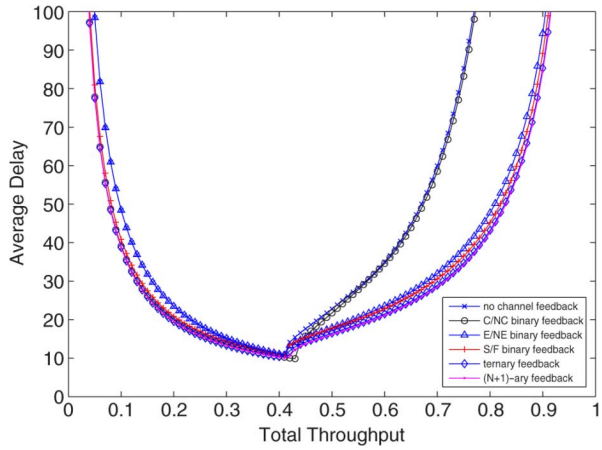

Fig. 5. Delay-efficiency boundaries under various channel feedback.

Fig. 5 shows that in the operating region of the protocol designer (i.e., the right-hand side of the delay-efficiency boundaries), protocols with one-slot memory perform worse under no channel feedback and C/NC binary feedback than under other considered channel feedback. In order to obtain high throughput with one-slot memory, we need a high correlation between successful users in two consecutive slots, which requires $f(W,1) \approx 0$ and $f(T,1) \approx 1$. However, under no channel feedback or C/NC binary feedback, a waiting user cannot distinguish between idle slots and success slots, leading to $f(W,0) = f(W,1) \approx 0$. This makes idle slots last long once one occurs, resulting in large average delay. Under S/F binary feedback, a user is constrained to use $f(W,0) = f(W,e)$. As can be seen in Fig. 2(b), the values of $f(W,0)$ and $f(W,e)$ are not much different in delay-efficient protocols under ternary feedback. Thus, using a single probability $f(W,0 \cup e)$ instead of two different probabilities, $f(W,0)$ and $f(W,e)$, causes only minor performance degradation. Under E/NE binary feedback, a delay-efficient protocol with one-slot memory that achieves high throughput has the following structure:

$$f(W,0) \approx \frac{1}{N}, \quad f(W,1 \cup e) = 0$$
$$f(T,1) \approx 1, \quad f(T,e) \approx \frac{1}{2}. \tag{14}$$

Following a collision, only colliding users transmit under (14) whereas only noncolliding users transmit under (12). Fig. 5 shows that the restriction that E/NE binary feedback imposes compared to ternary feedback has a small impact on performance. Fig. 5 also shows that the improvement in performance from having $(N+1)$-ary feedback over ternary feedback is only marginal. The ability of users to distinguish the exact numbers of transmissions in collisions does not help much because collisions involving three or more transmissions rarely occur under ternary feedback in the high throughput region.

[8]In Fig. 5, average delay is smallest under C/NC binary feedback for $\tau = 0.42$ and 0.43, which is a result of numerical instability around the turning point of the delay-efficiency boundaries, as pointed out in Section V-A.

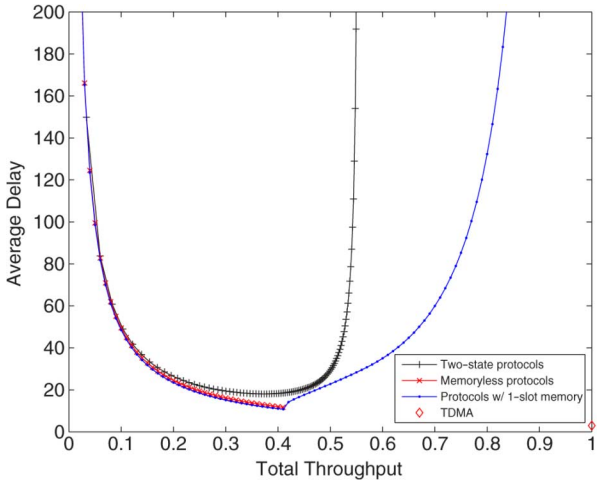

Fig. 6. Total throughput and average delay under two-state protocols, memoryless protocols, protocols with one-slot memory, and TDMA.

D. Comparison of Protocols

We consider $N = 5$ and compare the performance of four different kinds of protocols: protocols with one-slot memory, two-state protocols in [3], memoryless protocols, and TDMA. In Fig. 6, the delay-efficiency boundary of protocols with one-slot memory is shown assuming no channel feedback (i.e., ACK feedback only) because two-state protocols can be implemented without using channel feedback. [3] proposes two-state protocols of the form $p_{\mathrm{F}} = 1$ and $p_{\mathrm{G}} = 1 - \sqrt[N-1]{1 - 1/\eta}$, where η is called a short-term fairness parameter, which measures the average duration of consecutive successes. We vary $1/\eta$ from 0.01 to 1 with a step size of 0.01 to generate the throughput-delay pairs under two-state protocols plotted in Fig. 6. The result confirms that total throughput achievable with a two-state protocol is bounded from above by $N/(2N-1) = 5/9$, whereas protocols with one-slot memory can attain any level of total throughput between 0 and 1. Moreover, in the range of total throughput achievable with a two-state protocol, delay-efficient protocols with one-slot memory yield smaller average delay than two-state protocols. Since two-state protocols with $p_{\mathrm{F}} = 1$ can be considered as imposing a restriction of $f(W, \emptyset) = f(T, e)$, these results suggest that there is a significant performance degradation by assigning the same transmission probability following the two action–feedback pairs, $(W, \emptyset)$ and (T, e). Moreover, by combining Fig. 6 with Fig. 5, we can see that the performance degradation from using a two-state protocol instead of a protocol with one-slot memory is severer when channel feedback is available.

A symmetric memoryless protocol can be represented by a single transmission probability, which is used regardless of past histories. The throughput and average delay of user i under memoryless protocol $f \in [0, 1]$ are given by $\tau_i(f) = f(1-f)^{N-1}$ and $D(f) = 1/f(1-f)^{N-1} - 0.5$, respectively. The optimal memoryless protocol is thus the transmission probability that maximizes $\tau_i(f)$ and minimizes $D(f)$ at the same time, which is given by $f_0 = 1/N$ [16]. The throughput-delay pairs under memoryless protocols in Fig. 6 are obtained by varying f from 0 to 1. The lower-right point of the throughput-delay curve corresponds to the throughput-delay pair under the optimal memoryless protocol, $(\tau(f_0), D(f_0)) = ((1-1/N)^{N-1}, N/(1-1/N)^{N-1} - 0.5) = (0.41, 11.71)$. Protocols with one-slot memory also outperform memoryless protocols in that protocols with one-slot memory support a wider range of achievable total throughput than memoryless protocols do. In Section IV, we have seen that TDMA achieves the most preferred throughput-delay pair, $(1, N/2) = (1, 2.5)$. Protocols with one-slot memory cannot achieve the most preferred throughput-delay pair because memory of length at least $(N-1)$ slots is necessary to obtain it.

VI. Application to Wireless Local Area Networks

In the idealized slotted multiaccess system considered so far, all packets are of equal size, the transmission of a packet takes the duration of one slot, and users receive immediate feedback information. We relax these assumptions to apply protocols with memory to WLANs. In particular, packet sizes may differ across packets, and we take into consideration propagation and detection delay as well as overhead such as a packet header and an ACK signal. We consider a WLAN model where users follow a random access scheme using transmission probabilities based on CSMA/CA. In the WLAN model, the duration of slots depends on the channel state (idle, success, or collision). Let σ_0, σ_1, and σ_2 be the duration of a slot when the channel state is idle, success, and collision, respectively. Expressions for σ_1 and σ_2 can be found in (14) and (17) of [17], depending on whether the RTS/CTS mechanism is disabled or not. Total throughput is expressed as

$$\tau = \frac{P_1 E[P]}{P_0\sigma_0 + P_1\sigma_1 + P_2\sigma_2} \tag{15}$$

where $E[P]$ is the average packet transmission duration and P_0, P_1, and P_2 are the fractions of idle, success, and collision slots, respectively. In the idealized slotted model, we assume that the size of each packet is equal to the slot duration and ignore overhead so that $\sigma_0 = \sigma_1 = \sigma_2 = E[P]$, and thus the expression for total throughput in (15) is reduced to P_1, the fraction of success slots. [17] shows that the IEEE 802.11 DCF protocol can be approximated by a protocol that prescribes a single transmission probability, i.e., a memoryless protocol. The transmission probability corresponding to DCF is determined as a function of the minimum and maximum contention window sizes by solving (7) and (9) of [17] simultaneously. In Appendix B, we derive the expressions for throughput and average delay under symmetric protocols with one-slot memory and memoryless protocols in the WLAN model.

Fig. 7 depicts the delay-efficiency boundaries of protocols with one-slot memory under ternary feedback and E/NE binary feedback. It also plots the throughput-delay pairs under memoryless protocols as well as the throughput-delay pair achieved by the memoryless protocol corresponding to DCF. To obtain numerical results, we consider $N = 5$ and use parameters specified by IEEE 802.11a PHY mode-8 [18], which can be found in Table I of [19]. The values of $E[P]$, σ_0, σ_1, and σ_2 are 341.33, 9, 419.56, and 400.48, respectively, in μs. To find the transmission probability corresponding to DCF, we use the minimum and maximum contention window sizes $CW_{\min} = 16$ and $CW_{\max} = 1024$. An upper bound on total throughput can be obtained by setting $P_1 = 1$, which yields $\bar{\tau} = E[P]/\sigma_1 \approx 0.8136$. We compute delay-efficient protocols

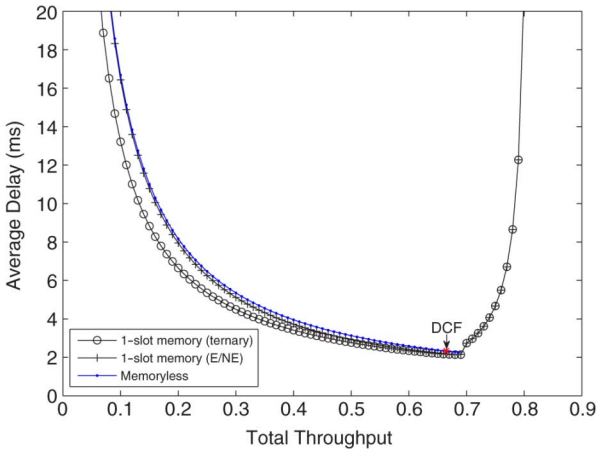

Fig. 7. Total throughput and average delay under delay-efficient protocols with one-slot memory and memoryless protocols in the WLAN model.

with one-slot memory for τ between 0.01 and 0.81 with a step size of 0.01 using the same numerical method as in Section V.

Comparing Fig. 7 to Fig. 6, we can see that by utilizing a carrier sensing scheme, memoryless protocols resolve contention more efficiently in the WLAN model than in the slotted multiaccess model. Since idle slots are much shorter than collision slots in WLAN, idle slots have much smaller effects on total throughput and average delay than collision slots have. Thus, in WLAN, transmission probabilities can be set low to achieve a success without experiencing many collisions. Carrier sensing also induces the turning point of the delay-efficiency boundaries in WLAN to occur around $\tau = 0.69$, yielding the narrower and steeper right-hand side of the delay-efficiency boundaries. The main findings in Section V-D—that protocols with one-slot memory can achieve smaller average delay for a given level of total throughput and a wider range of total throughput compared to memoryless protocols—remain valid in the WLAN model. On the right-hand side of the delay-efficiency boundaries, i.e., for $\tau \geq 0.7$, the delay-efficient protocols with one-slot memory under ternary feedback have the structure of $f(W,1) = f(W,e) = 0$, and thus the performance is not affected by having E/NE binary feedback instead of ternary feedback.

VII. Conclusion

In this paper, we have investigated how memory can be utilized in MAC protocols to achieve coordination without relying on explicit control messages. With $(N-1)$-slot memory, N users can share the channel as in TDMA. With one-slot memory, high throughput can be obtained by correlating successful users in two consecutive slots, which results in large average delay. Generalizing these results, with M-slot memory, where $M \leq N-2$, we can have the first M successful users use the channel alternatingly, while a collision created by a nonsuccessful user with a small probability leads to a potential change of hands for the collision slot.

Our framework can be extended in several directions. First, we can consider asymmetric protocols to provide quality-of-service differentiation across users. Second, as the literature on repeated games suggests, memory can also be used to sustain cooperation among selfish users. By utilizing memory, users can monitor the behavior of other users and punish misbehavior. Third, we can analyze protocols with memory in a multipacket reception model, which includes channels with capture as a special case [20]. Lastly, the basic idea of this paper can be carried over to a general multiagent scenario where it is desirable to have one agent behave in a different way than others.

Appendix A
Derivation of Throughput and Average Delay Under Symmetric Protocols With One-Slot Memory in the Slotted Multiaccess System

Exploiting the symmetry of protocols, we consider a Markov chain for a representative user whose state is defined as a pair of its transmission action and the number of transmissions, instead of a Markov chain with the state space $\mathcal{A}$ as in Section III. Let user i be the representative user. The state of user i in the current slot when the outcome of the previous slot was $\mathbf{a}$ is given by $s = (a_i, k(\mathbf{a}))$. We use $\mathcal{S}$ to denote the set of all states, as defined in Section II-B. There are total $2N$ states, which we list as

$$(T,1),\ldots,(T,N),(W,0),\ldots,(W,N-1). \tag{16}$$

Suppose that users follow a symmetric protocol $f \in \mathcal{F}_{1,\rho}$. Then we can express the transition probabilities across states in terms of f. If user i is in state (T,k), then k users including user i transmit with probability $f(T,\rho(T,k))$, and $(N-k)$ users with probability $f(W,\rho(W,k))$. Hence, a transition from (T,k) to (T,k') occurs with probability as shown in (17) at the bottom of the page, for $k' = 1,\ldots,N$, and to (W,k') as shown in (18) at the bottom of the next page, for $k' = 0,\ldots,N-1$, where $\mathbb{N}_0$ is the set of nonnegative integers. Similarly, if user i is in state (W,k), then k users transmit with probability $f(T,\rho(T,k))$, and $(N-k)$ users including user i with probability $f(W,\rho(W,k))$. A transition from (W,k) to (T,k') occurs with probability as shown in (19) at the bottom of the next page,

$$Q\left((T,k')|(T,k);f\right) = f\left(T,\rho(T,k)\right) \sum_{\left\{(x,y)\in\mathbb{N}_0^2 : x+y=k'-1, x\leq k-1, y\leq N-k\right\}} \left[\binom{k-1}{x} f\left(T,\rho(T,k)\right)^x \times \left(1-f\left(T,\rho(T,k)\right)\right)^{k-1-x} \binom{N-k}{y} f\left(W,\rho(W,k)\right)^y \left(1-f\left(W,\rho(W,k)\right)\right)^{N-k-y}\right] \tag{17}$$

for $k' = 1, \ldots, N$, and to (W, k') as shown in (20) at the bottom of the page, for $k' = 0, \ldots, N-1$.

Using (17)–(20), we obtain a $(2N \times 2N)$ matrix $\mathbf{Q}(f)$. The (j, j')-entry of $\mathbf{Q}(f)$ is the transition probability from state j to state j', where the states are numbered in the order listed in (16). If f is chosen so that $f(a, \rho(a,k)) \in (0,1)$ for all $(a,k) \in \mathcal{S}$, then the Markov chain is irreducible and there exists a unique stationary distribution $\mathbf{v}(f)$ on $\mathcal{S}$, represented by a row vector of length $2N$, that satisfies

$$\mathbf{v}(f) = \mathbf{v}(f)\mathbf{Q}(f) \quad \text{and} \quad \mathbf{v}(f)\mathbf{e}_{2N} = 1$$

where $\mathbf{e}_{2N}$ is a column vector of length $2N$ whose elements are all 1 [13]. Let $s^* \triangleq (T, 1)$ be the state of a successful transmission. Then, the throughput of user i is given by the first element of $\mathbf{v}(f)$, i.e.,

$$\tau_i(f) = v(s^*; f).$$

By symmetry, total throughput is given by $\tau(f) = N\tau_i(f)$.

Using (5), we obtain the relationship

$$d_i(s; f) = 1 + \sum_{s' \in \mathcal{S} \setminus \{s^*\}} Q(s'|s; f) d_i(s'; f) \tag{21}$$

for all $s \in \mathcal{S}$. Let $\mathbf{d}_i(f)$ be the column vector consisting of $d_i(s; f)$ and $\mathbf{Q}_0(f)$ be a matrix obtained by replacing all the elements in the first column of $\mathbf{Q}(f)$ by 0. Then, (21) can be expressed as the following matrix equation:

$$\mathbf{d}_i(f) = \mathbf{Q}_0(f)\mathbf{d}_i(f) + \mathbf{e}_{2N}. \tag{22}$$

Solving (22) for $\mathbf{d}_i(f)$, we obtain

$$\mathbf{d}_i(f) = (\mathbf{I}_{2N} - \mathbf{Q}_0(f))^{-1} \mathbf{e}_{2N}$$

assuming that the inverse exists, where $\mathbf{I}_{2N}$ is the $(2N \times 2N)$ identity matrix. Since the long-run frequency of each state is given by the stationary distribution $\mathbf{v}(f)$, the average delay of user i can be expressed as

$$D(f) = \mathbf{v}(f)\mathbf{d}_i(f) - 0.5.$$

Also, the average interpacket time is given by the first element of $\mathbf{d}_i(f)$, i.e., $\tilde{D}_i(f) = d_i(s^*; f)$.

APPENDIX B
DERIVATION OF THROUGHPUT AND AVERAGE DELAY UNDER SYMMETRIC PROTOCOLS WITH ONE-SLOT MEMORY AND MEMORYLESS PROTOCOLS IN THE WLAN MODEL

Let f be a symmetric protocol with one-slot memory. The long-run fractions of idle, success, and collision slots are given by

$$P_0(f) = v((W,0); f) \tag{23}$$
$$P_1(f) = v((T,1); f) + v((W,1); f) \tag{24}$$
$$P_2(f) = 1 - P_0(f) - P_1(f).$$

Using (15), total throughput under f in the WLAN model can be written as

$$\tau(f) = \frac{P_1(f)E[P]}{P_0(f)\sigma_0 + P_1(f)\sigma_1 + P_2(f)\sigma_2}. \tag{25}$$

$$Q((W,k')|(T,k); f) = (1 - f(T, \rho(T,k))) \sum_{\{(x,y) \in \mathbb{N}_0^2 : x+y=k', x \le k-1, y \le N-k\}} \left[\binom{k-1}{x} f(T, \rho(T,k))^x \times (1 - f(T, \rho(T,k)))^{k-1-x} \binom{N-k}{y} f(W, \rho(W,k))^y (1 - f(W, \rho(W,k)))^{N-k-y} \right] \tag{18}$$

$$Q((T,k')|(W,k); f) = f(W, \rho(W,k)) \sum_{\{(x,y) \in \mathbb{N}_0^2 : x+y=k'-1, x \le k, y \le N-k-1\}} \left[\binom{k}{x} f(T, \rho(T,k))^x \times (1 - f(T, \rho(T,k)))^{k-x} \binom{N-k-1}{y} f(W, \rho(W,k))^y (1 - f(W, \rho(W,k)))^{N-k-1-y} \right] \tag{19}$$

$$Q((W,k')|(W,k); f) = (1 - f(W, \rho(W,k))) \sum_{\{(x,y) \in \mathbb{N}_0^2 : x+y=k', x \le k, y \le N-k-1\}} \left[\binom{k}{x} f(T, \rho(T,k))^x \times (1 - f(T, \rho(T,k)))^{k-x} \binom{N-k-1}{y} f(W, \rho(W,k))^y (1 - f(W, \rho(W,k)))^{N-k-1-y} \right] \tag{20}$$

In the WLAN model, we define the average delay of a user as the average waiting time (measured in a time unit) until the beginning of its next successful transmission starting from an arbitrarily chosen time. We define $d_i(s;f)$ as the average waiting time of user i starting from the beginning of a slot whose outcome yields state s to user i. Let $\sigma(s)$ be the duration of a slot yielding state s, i.e., $\sigma(W,0)=\sigma_0, \sigma(a,1)=\sigma_1$ for $a=T,W$, and $\sigma(a,k)=\sigma_2$ for $a=T,W$ and $k\geq 2$. Then, (21) can be modified as

$$d_i(s;f)=\sigma(s)+\sum_{s'\in\mathcal{S}\setminus\{s^*\}} Q(s'|s;f)d_i(s';f). \tag{26}$$

Let $\mathbf{b}$ be the column vector of the durations of slots, i.e., $\mathbf{b}\triangleq(\sigma(T,1)\,\sigma(T,2)\,\cdots\,\sigma(W,N-1))^\top$. Then, (26) can be written as a matrix equation

$$\mathbf{d}_i(f)=\mathbf{Q}_0(f)\mathbf{d}_i(f)+\mathbf{b}. \tag{27}$$

Solving (27) for $\mathbf{d}_i(f)$, we obtain

$$\mathbf{d}_i(f)=(\mathbf{I}_{2N}-\mathbf{Q}_0(f))^{-1}\mathbf{b}.$$

The probability that an arbitrarily chosen time belongs to a slot yielding state s is given by

$$y(s;f)=\frac{v(s;f)\sigma(s)}{\sum_{s'\in\mathcal{S}}v(s';f)\sigma(s')}.$$

Note that a user stays in the initial slot for the half of its duration on average. Let $\mathbf{y}(f)$ be the row vector consisting of $y(s;f)$. Then, the average delay of user i under protocol f can be computed by

$$D(f)=\mathbf{y}(f)\left[\mathbf{d}_i(f)-0.5\mathbf{b}\right].$$

Now, let f be a symmetric memoryless protocol, which is simply a single transmission probability. Then, (23) and (24) can be expressed as $P_0(f)=(1-f)^N$ and $P_1(f)=Nf(1-f)^{N-1}$, respectively, and we can use (25) to compute total throughput. When users follow a memoryless protocol f, the average waiting time starting from the next slot, $d_i(s;f)-\sigma(s)$, is independent of s, and thus we write the value as $\tilde{d}_i(f)$. Manipulating (26) yields

$$\tilde{d}_i(f)=\frac{N\left[P_0(f)\sigma_0+P_2(f)\sigma_2\right]+(N-1)P_1(f)\sigma_1}{P_1(f)}.$$

The average waiting time until the next slot is given by

$$r(f)=\frac{1}{2}\mathbf{y}(f)\mathbf{b}=\frac{P_0(f)\sigma_0^2+P_1(f)\sigma_1^2+P_2(f)\sigma_2^2}{2\left[P_0(f)\sigma_0+P_1(f)\sigma_1+P_2(f)\sigma_2\right]}.$$

Hence, average delay under memoryless protocol f can be computed by

$$D(f)=r(f)+\tilde{d}_i(f).$$

REFERENCES

[1] L. G. Roberts, "Aloha packet system with and without slots and capture," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 5, no. 2, pp. 28–42, Apr. 1975.

[2] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE 802.11, Aug. 1999.

[3] R. T. Ma, V. Misra, and D. Rubenstein, "An analysis of generalized slotted-Aloha protocols," *IEEE/ACM Trans. Netw.*, vol. 17, no. 3, pp. 936–949, Jun. 2009.

[4] R. L. Rivest, "Network control by Bayesian broadcast," *IEEE Trans. Inf. Theory*, vol. IT-33, no. 3, pp. 323–328, May 1987.

[5] J.-W. Lee, A. Tang, J. Huang, M. Chiang, and A. R. Calderbank, "Reverse-engineering MAC: A non-cooperative game model," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 6, pp. 1135–1147, Aug. 2007.

[6] D. Bertsekas and R. Gallager, *Data Networks*, 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 1992.

[7] W. Crowther, R. Rettberg, D. Walden, S. Ornstein, and F. Heart, "A system for broadcast communication: Reservation-ALOHA," in *Proc. 6th Hawaii Int. Conf. Syst. Sci.*, Jan. 1973, pp. 371–374.

[8] J. Capetanakis, "Tree algorithms for packet broadcast channels," *IEEE Trans. Inf. Theory*, vol. AP-25, no. 5, pp. 505–515, Sep. 1979.

[9] D. Fudenberg and J. Tirole, *Game Theory*. Cambridge, MA: MIT Press, 1991.

[10] B. Bing, *Broadband Wireless Access*. New York: Springer, 2002.

[11] N. Mahravari, "Random-access communication with multiple reception," *IEEE Trans. Inf. Theory*, vol. 36, no. 3, pp. 614–622, May 1990.

[12] G. Mailath and L. Samuelson, *Repeated Games and Reputations: Long-Run Relationships*. Oxford, U.K.: Oxford Univ. Press, 2006.

[13] C. M. Grinstead and J. L. Snell, *Introduction to Probability*. Providence, RI: AMS, 1997.

[14] W. Feller, *An Introduction to Probability Theory and Its Applications*. New York: Wiley, 1971.

[15] E. D. Lazowska, J. Zahorjan, G. S. Graham, and K. C. Sevcik, *Quantitative System Performance: Computer System Analysis Using Queueing Network Models*. Englewood Cliffs, NJ: Prentice-Hall, 1984.

[16] J. L. Massey and P. Mathys, "The collision channel without feedback," *IEEE Trans. Inf. Theory*, vol. AP-31, no. 2, pp. 192–204, Mar. 1985.

[17] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 3, pp. 535–547, Mar. 2000.

[18] *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: High-Speed Physical Layer in the 5 GHz Band*, IEEE 802.11a, Sep. 1999, Supplement to IEEE 802.11 Standard.

[19] D. Qiao and K. G. Shin, "Achieving efficient channel utilization and weighted fairness for data communications in IEEE 802.11 WLAN under the DCF," in *Proc. IWQoS*, 2002, pp. 227–236.

[20] S. Ghez, S. Verdú, and S. C. Schwartz, "Stability properties of slotted Aloha with multipacket reception capability," *IEEE Trans. Autom. Control*, vol. 33, no. 7, pp. 640–649, Jul. 1988.

Jaeok Park (M'10) received the B.A. degree in economics from Yonsei University, Seoul, Korea, in 2003, and the M.A. and Ph.D. degrees in economics from the University of California, Los Angeles (UCLA), in 2005 and 2009, respectively.

He is currently a Post-Doctoral Scholar with the Electrical Engineering Department, UCLA. From 2006 to 2008, he served in the Republic of Korea Army. His primary research interests include game theory, mechanism design, network pricing, and wireless communication.

Mihaela van der Schaar (F'10) received the M.S. and Ph.D. degrees from the Eindhoven University of Technology, Eindhoven, The Netherlands, in 1996 and 2001, respectively.

She is currently an Associate Professor with the Electrical Engineering Department, University of California, Los Angeles. Her research interests are in multiuser communication networks, multimedia communications, processing and systems, online learning, network economics, and game theory.