Performance of Distributed Medium Access Control with an Enhanced Physical-Link Layer Interface

Jie Luo
ECE Dept., Colorado State Univ., Fort Collins, CO 80523
Email: rockey@colostate.edu

***Abstract*—A distributed MAC algorithm is presented to support an enhanced physical-link layer interface with multiple transmission options at each link layer user. Theoretical performance analysis is provided and is shown to match well with the simulated results. Example is given to demonstrate significant throughput improvement of the distributed MAC algorithm compared with the classical 802.11 distributed coordination function (DCF), due to the support of multi-packet reception.**

I. Introduction

Wireless networks are evolving rapidly toward the "information generation" that features massive distributed devices with packet-based bursty messages. Due to the increasing proportion of short messages and the difficulty of quickly coordinating wireless users, there is a growing demand to support efficient distributed communication adaptation without breaking the layered network architecture [1].

In [1] [2], a new channel coding theory was proposed for distributed communication at the physical layer. The coding theory allows each physical layer transmitter to prepare an ensemble of channel codes corresponding to different communication settings. A transmitter can choose an arbitrary code, possibly according to a link layer decision, to encode the message and to transmit the codeword symbols to the receiver. While code ensembles of the users are assumed to be known, actual coding choices are not shared among the transmitters or with the receiver. The receiver, on the other hand, should either decode the messages of interest or report collision, depending on whether a pre-determined error probability requirement can be met. Fundamental limit of the system was characterized using a distributed channel capacity region defined in the vector space of the coding choices of the transmitters [1]. The distributed capacity region was shown to coincide with the classical Shannon capacity region, in a sense explained in [1]. Error performance bounds in the case of finite codeword length were obtained in [3]. The new channel coding theory provided the basic physical layer support for an enhancement to the physical-link layer interface, which allows each link layer user to be equipped with multiple transmission options. The interface enhancement enables data link layer protocols to exploit advanced wireless communication adaptations through the navigation of different transmission options. Layered network architecture is maintained by constraining each link layer user to the provided options for transmission adaptation.

This work was supported by the National Science Foundation under Grant ECCS-2128569. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

In [1], a distributed Medium Access Control (MAC) algorithm was proposed to support the enhanced physical-link layer interface at the data link layer. The MAC algorithm assumes homogeneous users with saturated message queues, and a general link layer channel model that can be derived from the physical layer channel and packet coding details. Each user is associated with a vector of transmission probabilities corresponding to different transmission options. With a carefully designed incremental probability adaptation framework, the distributed MAC algorithm was shown to converge to a unique equilibrium that is close to optimum with respect to a chosen network utility [1].

A practical distributed MAC protocol such as the 802.11 DCF can be regarded as the integration of three key components, namely, a random access scheme, a fast adaptation algorithm, and a random scheduling approach. The "random access scheme" regulates how users with short packets should access the shared channel opportunistically. The "fast adaptation algorithm" enables a user to quickly adapt its short packet transmission probability in response to the sensed communication condition. The "random scheduling approach," on the other hand, is responsible for scheduling undisturbed long message transmissions at the maximum rate. The distributed MAC algorithm proposed in [1] only focused on the component of "random access scheme," due to its assumption of equal-sized short packets and its incremental probability adaptation framework. In this paper, we extend the algorithm toward a practical distributed MAC protocol by adding the components of "fast adaptation algorithm" and "random scheduling approach". Assume that the system contains an unknown number of homogeneous users with saturated message queues. We associate each user with a transmission probability vector, as well as an estimate of the number of users in the system. We develop fast adaptation algorithms by extending the exponential backoff algorithm of 802.11 DCF to adjust the estimated number of users, and then adopting the algorithm proposed in [1] to calculate the transmission probability vector as a function of the estimated number of users. Theoretical performance analysis of the fast adaptation algorithm is carried out under an independence assumption by extending the Markov model presented in [4]. Computer simulations show that actual performance of the fast adaptation algorithm matches well with

the theoretical result. In an example with a Gaussian multiple access channel and equal-sized short packets, the proposed fast adaptation algorithm is shown to achieve a throughput that is consistently above three times that of the 802.11 DCF. We further develop the random scheduling approach by asking each user with a long message to first exchange RTS and CTS handshake messages with the receiver using the random access scheme, similar to the corresponding procedure in 802.11 DCF. With the support of multi-packet reception, it is possible that RTS/CTS handshakes of multiple users can be successful simultaneously. In this case, the random scheduling approach reserves the channel for parallel long message transmissions at the maximum sum rate, which can be much higher than the maximum single user rate in a Gaussian multiple access channel. Simplified theoretical analysis and simulation results are given to characterize the throughput of the scheduled long message transmissions of the distributed MAC algorithm.

II. Review of the Random Access Scheme

In this section, we review the distributed MAC algorithm proposed in [1] for the enhanced physical-link layer interface. Consider a distributed multiple access network with a memoryless channel and K homogeneous users (transmitters). K is known neither to the users nor to the receiver. Each user is backlogged with a saturated message queue. Time is slotted, and the length of each time slot equals the transmission duration of one packet. Each user is equipped with M transmission options plus an idling option. At the beginning of each time slot t, each user, say user k, $k = 1, \ldots, K$, should individually decide whether to idle or to send a message with a randomly chosen transmission option. The corresponding probabilities are specified by an associated M-length probability vector, denoted by $\boldsymbol{p}_k$. We write $\boldsymbol{p}_k = p_k \boldsymbol{d}_k$, with $0 \leq p_k \leq 1$ being the probability that user k transmits a packet, and with vector $\boldsymbol{d}_k$ specifying the conditional probabilities for user k to choose each of the transmission options should it decide to transmit a packet. We term p_k the "transmission probability" of user k, and term $\boldsymbol{d}_k$ the "transmission direction" vector of user k.

In each time slot, the receiver envisions the transmission of a carefully designed "virtual packet". Virtual packets assumed in different time slots are identical. A virtual packet is an assumed packet whose coding parameters are known to the users and to the receiver, but it is not physically transmitted in the system, i.e., the packet is "virtual". We assume that, without knowing the transmission/idling status of the users, the receiver should be able to detect whether the reception of a virtual packet is successful or not [1]. The receiver should maintain an estimate of the success probability of the virtual packet, denoted by $q_v(t)$, and should feed it back to the users. $q_v(t)$ is termed the "channel contention measure" because it is used to measure the contention/availability level of the channel [1]. A high value of $q_v(t)$ reflects a low channel contention level in time slot t.

We require each user to maintain two key functions, $\boldsymbol{p}^*(\hat{K})$ and $q_v^*(\hat{K})$, both are functions of an estimated number of users $\hat{K}$ [1]. The set of functions should be the same across different users. The $\boldsymbol{p}^*(\hat{K})$ function, termed the "theoretical transmission probability vector" function, specifies the designed (or the targeted) transmission probability vector of the user should the number of users in the system equal $\hat{K}$. The $q_v^*(\hat{K})$ function, termed the "theoretical channel contention measure" function, represents the derived theoretical channel contention measure, if the number of users equals $\hat{K}$ and all users have the same transmission probability vector $\boldsymbol{p}^*(\hat{K})$.

Define $K_{\min}$ as the maximum $\hat{K}$ that maximizes $q_v^*(\hat{K})$. With the two key functions, the distributed MAC algorithm operates as follows [1].

Distributed MAC Algorithm:

1) Each user initializes its transmission probability vector.
2) Let $Q > 0$ be a pre-determined integer. Over an interval of Q time slots, the receiver measures the success probability of the virtual packet, denoted by q_v, and feeds q_v back to all users.
3) Upon receiving q_v, each user derives an estimated number of users $\hat{K}$ by solving the following equation.

$$q_v^*(\hat{K}) = q_v, \qquad \text{s.t. } \hat{K} \geq K_{\min}. \tag{1}$$

   If a $\hat{K}$ satisfying (1) cannot be found, a user should set $\hat{K} = K_{\min}$ if $q_v > q_v^*(K_{\min})$, or set $\hat{K} = \infty$ otherwise. Each user then sets the target transmission probability vector at $\hat{\boldsymbol{p}} = \boldsymbol{p}^*(\hat{K})$.
4) Each user, say user k, updates its transmission probability vector by $\boldsymbol{p}_k = (1-\alpha)\boldsymbol{p}_k + \alpha\hat{\boldsymbol{p}}$, where α is the step size parameter for user k.
5) The process is repeated from Step 2 till transmission probability vectors of all users converge.

The distributed MAC algorithm falls into the classical framework of stochastic approximations. Given the actual number of users K, if all users have the same transmission probability vector $\boldsymbol{p}$, the actual channel contention measure $q_v(\boldsymbol{p}, K)$ can be written as a function of $\boldsymbol{p}$ and K. As shown in [1, Theorem 4.6], if the $\boldsymbol{p}^*(\hat{K})$ function is carefully designed such that $q_v^*(\hat{K})$ is continuous and monotonically non-increasing in $\hat{K}$, and $q_v(\boldsymbol{p}^*(\hat{K}), K) = q_v^*(\hat{K})$ only happens when $\hat{K} = K$, then the distributed MAC algorithm should lead the transmission probability vectors of all users to converge to the unique equilibrium corresponding to $\hat{K} = K$. In addition, the design of $\boldsymbol{p}^*(\hat{K})$ should make sure that the system is close to optimal at its equilibrium in terms of maximizing a chosen utility function irrespective of the number of users in the system. Detailed discussions on the design of $\boldsymbol{p}^*(\hat{K})$ and $q_v^*(\hat{K})$ functions can be found in [1].

III. Fast Adaptation Algorithms

The distributed MAC algorithm given in Section II only focused on the component of "random access scheme" due to the assumptions of equal-sized short packets and incremental probability adaption. In this section, we still keep the assumption of equal-sized short packets, but propose a "fast adaptation algorithm" to replace the incremental probability adaptation. The fast adaptation algorithm is developed by extending the exponential backoff approach of the 802.11 DCF protocol.

Note that, in 802.11 DCF, a collision avoidance mechanism is implemented to help reduce the probability of packet collision. More specifically, before transmitting any packet, a user needs to make sure that the channel has been idling for a short duration defined as the Distributed InterFrame Space (DIFS) [4]. Because DIFS is often much shorter in length than a packet, a quick collision detection can help to reduce the chance of a relatively long collision in packet transmission. However, with the assumption that the length and the transmission schedule of each packet should be synchronized to one time slot, it is not difficult to see that collision avoidance, e.g., to make sure channel is available in the leading time slot before each packet transmission, should not reduce the probability of packet collision. Therefore, collision avoidance is not considered in any of the adaptation algorithms to be presented.

We assume that each user, say user k, should maintain an estimate of the number of users, denoted by $\hat{K}_k$. $\hat{K}_k$ should be kept between pre-determined boundaries denoted by $K_{\min}$ and $K_{\max}$. The value of $K_{\max}$ should be set large enough such that the probability of the system having a number of users more than $K_{\max}$ is negligible. $K_{\max}$ should also be chosen to satisfy $K_{\max} = 2^c K_{\min}$ for a positive integer-valued c.

To enable fair performance comparison, we first present a modified 802.11 DCF protocol that fits the time-slotted model with the collision avoidance mechanism being removed. The protocol assumes that each user should only have a single transmission option.

Modified 802.11 DCF:

1) Each user, say user k, initializes its estimated number of users at $\hat{K}_k = K_{\min}$.
2) User k sets its "backoff window," denoted by $W_k(\hat{K}_k)$, at $W_k(\hat{K}_k) = 2\hat{K}_k$. User k then initializes its random "backoff counter" uniformly between 0 and $W_k(\hat{K}_k)-1$.
3) In each time slot, if the "backoff counter" of user k equals 0, user k should transmit a packet.
4) The receiver feeds packet reception status back to the users.
5) At the end of each time slot, user k should take the following actions.
    a) Assume that the "backoff counter" of user k equals 0. If the packet transmitted by user k is received successfully, user k should update its backoff window by $\hat{K}_k = K_{\min}$. If the packet reception failed, user k should update its backoff window by $\hat{K}_k = \min\{K_{\max}, 2\hat{K}_k\}$. The process continues from Step 2.
    b) Assume that the "backoff counter" of user k is positive, user k should decrease its backoff counter by 1. The process continues from Step 3.

Next, we present the proposed fast adaptation algorithm.

Proposed Fast Adaptation Algorithm:

1) Each user, say user k, initializes its estimated number of users at $\hat{K}_k = K_{\min}$.
2) Let $\boldsymbol{p}^*(\hat{K}_k) = p^*(\hat{K}_k)\boldsymbol{d}^*(\hat{K}_k)$ where $p^*(\hat{K}_k)$ is the transmission probability and $\boldsymbol{d}^*(\hat{K}_k)$ is the transmission direction vector of user k. Let $\lfloor 2/p^*(\hat{K}_k)\rfloor$ be the maximum integer no larger than $2/p^*(\hat{K}_k)$. User k first sets its "backoff window" randomly, denoted by $W_k(\hat{K}_k)$, at $W_k(\hat{K}_k) = \lfloor 2/p^*(\hat{K}_k)\rfloor - 1$ with probability $1 + \lfloor 2/p^*(\hat{K}_k)\rfloor - 2/p^*(\hat{K}_k)$ and at $W_k(\hat{K}_k) = \lfloor 2/p^*(\hat{K}_k)\rfloor$ with probability $2/p^*(\hat{K}_k) - \lfloor 2/p^*(\hat{K}_k)\rfloor$. User k then initialize its random "backoff counter" uniformly between 0 and $W_k(\hat{K}_k) - 1$.
3) In each time slot, if the "backoff counter" of user k equals 0, user k should transmit a packet by randomly choosing a transmission option according to the conditional probabilities specified in the transmission direction vector $\boldsymbol{d}^*(\hat{K}_k)$.
4) The receiver judges whether virtual packet reception in each time slot should be regarded as successful or not, and updates the users with an estimated virtual packet failure probability p.
5) At the end of each time slot, user k should take the following actions.
    a) Assume that the "backoff counter" of user k equals 0. User k should update its estimated number of users randomly by $\hat{K}_k = \max\{K_{\min}, \hat{K}_k/2\}$ with probability $1-p$, and by $\hat{K}_k = \min\{K_{\max}, 2\hat{K}_k\}$ with probability p. The process continues from Step 2.
    b) Assume that the "backoff counter" of user k is positive, user k should decrease its backoff counter by 1. The process continues from Step 3.

We purposely presented the two adaptation algorithms using similar description terms and organizations to enable their step-by-step comparison, and to help illustrating the key extensions implemented in the proposed fast adaptation algorithm.

Performance of the proposed fast adaptation algorithm can be analyzed by following the framework presented in [4]. Let us assume that, when the process of probability adaptations of the users become stationary, transmission activities of the users should be mutually independent. Consequently, virtual packet receptions in different time slots should be independent, each having a constant failure probability, denoted by p. From a single user's perspective, behavior of the user can be modeled using a Markov chain illustrated in Figure 1, where state transition of the Markov chain happens in each time slot.

The Markov chain has $c+1$ rows of sates. On the ith row, $i = 0, 1, \ldots, c$, State (i, j) for $j = 0, 1, \ldots, \lfloor 2/p_i^*\rfloor - 1$ corresponds to an estimated number of users equaling $2^i K_{\min}$ and a "backoff counter" equaling j. In the figure, $p_i^* = p^*(2^i K_{\min})$, for $i = 0, \ldots, c$, is the theoretical transmission probability when the estimated number of users equal $2^i K_{\min}$. Also in the figure, r_{ij} for $i = 0, 1, \ldots, c$ and $j = 0, 1, \ldots, \lfloor 2/p_i^*\rfloor - 1$ represents the probability that, when the estimated number of users equals $2^i K_{\min}$, the user initialize its "backoff counter" at j. Because the backoff window can take two sizes randomly, the values of

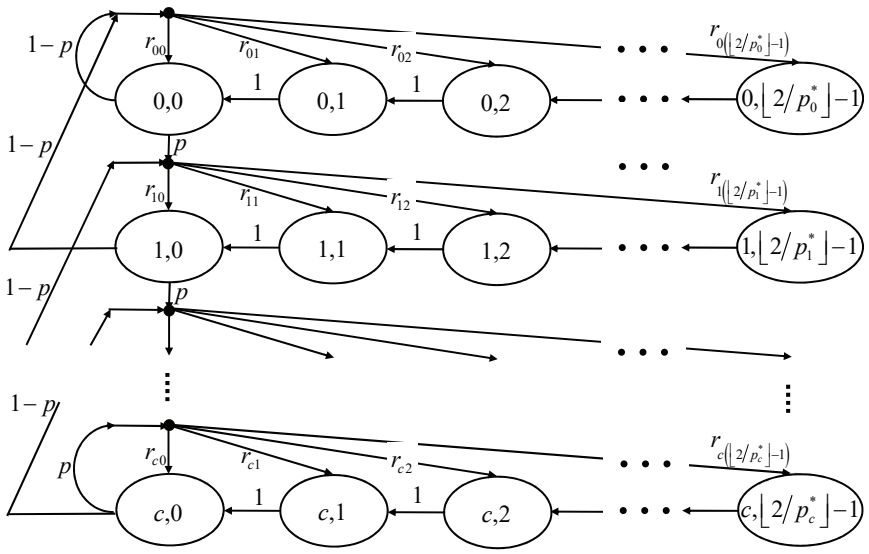

Fig. 1. Markov chain model of the proposed fast adaptation algorithm.

r_{ij} are given by

$$r_{ij} = \begin{cases} \frac{2/p_i^*}{\lfloor 2/p_i^* \rfloor} - 1 & \text{for } j = \lfloor 2/p_i^* \rfloor - 1 \\ \frac{2\lfloor 2/p_i^* \rfloor - 2/p_i^*}{\lfloor 2/p_i^* \rfloor (\lfloor 2/p_i^* \rfloor - 1)} & \text{for } j < \lfloor 2/p_i^* \rfloor - 1 \end{cases}. \quad (2)$$

From the state transition diagram, we can see that

$$\begin{aligned} & pb_{0,0} = (1-p)b_{1,0}, \\ & b_{i,0} = pb_{i-1,0} + (1-p)b_{i+1,0}, \quad \text{for } i = 1, \ldots, c-1, \\ & (1-p)b_{c,0} = pb_{c-1,0}. \end{aligned} \quad (3)$$

This gives

$$b_{i,0} = \frac{p}{1-p} b_{i-1,0} = \left(\frac{p}{1-p}\right)^i b_{0,0} \quad \text{for } i = 1, \ldots, c. \quad (4)$$

Consequently,

$$b_{i,j} = \left(\sum_{k=j}^{\lfloor 2/p_i^* \rfloor - 1} r_{ik} \right) \left(\frac{p}{1-p}\right)^i b_{0,0} \quad \text{for } i = 0, \ldots, c \text{ and } j \le \lfloor 2/p_i^* \rfloor - 1. \quad (5)$$

Note that

$$\sum_{j=0}^{\lfloor 2/p_i^* \rfloor - 1} \sum_{k=j}^{\lfloor 2/p_i^* \rfloor - 1} r_{ik} = \frac{1}{p_i^*}, \quad \text{for } i = 0, \ldots, c, \quad (6)$$

which means that conditional transmission probability given $\hat{K}_k = 2^i K_{\min}$ equals p_i^*. Because $\sum_{i=0}^{c} \sum_{j=0}^{\lfloor 2/p_i^* \rfloor - 1} b_{i,j} = 1$, according to (6) and (5), $b_{0,0}$ can be obtained by

$$b_{0,0} = \frac{1}{\sum_{i=0}^{c} \frac{1}{p_i^*} \left(\frac{p}{1-p}\right)^i}. \quad (7)$$

Finally, given the number of users K, virtual packet failure probability p can be written as a function of $b_{i,0}$ for $i = 0, \ldots, c$ using the link-layer channel model, which we will illustrate soon in an example. Consequently, p, stationary probabilities of the Markov chain, as well as the performance of the system, can be obtained.

Example: Consider a multiple access system over a Gaussian channel. Each user is equipped with two rate options termed the "high" and "low" rates. The length of all packets, irrespective of their data rates, remains that of one time slot. Assume that, if a single user transmits, irrespective of the rate option, the signal should arrives at the receiver with a receiving signal to noise ratio (SNR) of SNR $= 15$ dB. Let us define single user rate $r_s = \frac{1}{2}\log(1 + \text{SNR})$, high rate $r_h = \frac{1}{16}\log(1 + 8\text{SNR})$, and low rate $r_l = \frac{1}{128}\log(1 + 64\text{SNR})$, all in bits per symbol. We assume that packets transmitted in the modified 802.11 DCF protocol have a rate of r_s, while packets transmitted in the fast adaptation algorithms have a rate of r_h when the high rate option is used, or a rate of r_l when the low rate option is used. Let N_h and N_l be respectively the number of high rate and low rate packets being transmitted in parallel. We assume that all packets should be received successfully if

$$N_h r_h + N_l r_l \le \frac{1}{2}\log(1 + (N_h + N_l)\text{SNR}). \quad (8)$$

Otherwise, no packet will be received. We assume that a virtual packet should be equivalent to the joint transmissions of 3 high rate packets. Virtual packet reception should be regarded as successful if and only if

$$(N_h + 3)r_h + N_l r_l \le \frac{1}{2}\log(1 + (N_h + N_l + 3)\text{SNR}). \quad (9)$$

Assume that users intend to maximize the symmetric throughput. For the proposed fast adaptation algorithm, we design the theoretical transmission probability vector function $\boldsymbol{p}^*(\hat{K})$ by following the guideline illustrated in [1]. More specifically, we partition the range of $\hat{K}$ into 3 regions. We define $\{\hat{K} | \hat{K} \le 12\}$ as the "Head" region, and define

$\{\hat{K}|\hat{K} \geq 58\}$ as the "Tail" region. Assuming that users should only use the "high" rate option in the Head region and should only use the "low" rate option in the Tail region. By following the guideline presented in [1, Section 4.3], $\boldsymbol{p}^*(\hat{K})$ in the two regions are designed as

$$\boldsymbol{p}^*(\hat{K}) = \begin{cases} \frac{5.804}{\max\{5,\hat{K}\}+1.01} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} & \hat{K} \leq 12 \\ \\ \frac{52.28}{\hat{K}+12.29} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} & \hat{K} \geq 58 \end{cases}. \quad (10)$$

Next, under the assumption that all users should have the same transmission probability vector, we define $\boldsymbol{p}^*_{\text{opt}}(\hat{K}) = p^*_{\text{opt}}(\hat{K})\boldsymbol{d}^*_{\text{opt}}(\hat{K})$ as the optimal transmission probability vector that maximizes the sum throughput of the system. For $\{\hat{K}|12 < \hat{K} < 58\}$, we first set transmission direction vectors $\boldsymbol{d}^*(\hat{K})$ at $\boldsymbol{d}^*(\hat{K}) = \boldsymbol{d}^*_{\text{opt}}(\hat{K})$, for $\hat{K} = 13, 14, 15$. We also choose $\boldsymbol{d}^*(\hat{K})$ such that $\boldsymbol{d}^*(\hat{K})$ transits linearly in $\hat{K}$ for $15 \leq \hat{K} \leq 58$. After that, we choose transmission probability $p^*(\hat{K})$ such that the resulting "theoretical channel contention measure" function $q^*_v(\hat{K})$ (whose definition can be found in Section II) should transit linearly in $\hat{K}$ for $12 \leq \hat{K} \leq 58$. Note that, while we skipped the reasoning of the $\boldsymbol{p}^*(\hat{K})$ function design, a detailed explanation of the design in a similar example can be found in [1, Section 4.3].

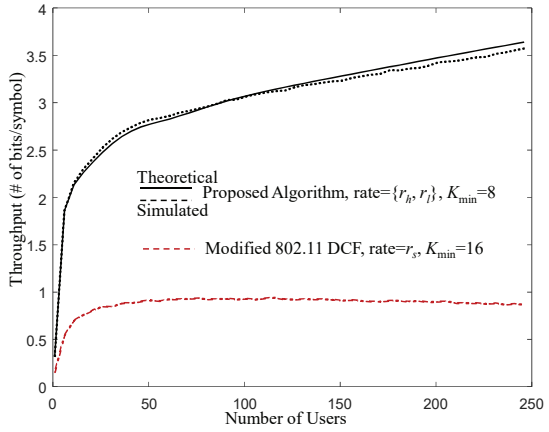

Fig. 2. Throughput as a function of the number of users. SNR = 15 dB.

In Figure 2, we plotted the throughput performances of the proposed fast adaptation algorithm and the modified 802.11 DCF protocol, with $K_{\max} = 512$ and with appropriately chosen $K_{\min}$ values[1]. We can see that the theoretical performance of the proposed fast adaptation algorithm matches well with its simulated result. Due to the support of multi-packet reception, which not only reduces packet collision probability but also increases the maximum sum throughput, the proposed fast adaptation algorithm achieved a throughput that is consistently above three times of the throughput of the modified 802.11 DCF.

[1]Note that $K_{\min} = 16$ and $K_{\max} = 512$ for the modified 802.11 DCF are set to reflect typical window sizes of the IEEE 802.11 standard.

IV. Random Scheduling Approach

In 802.11 DCF, when a long message becomes available at a user, the user should first send a control message RTS to the receiver. If the message is received successfully, the receiver should reply with a CTS message. All other users hearing the RTS/CTS exchange should remain idle, and the channel is then reserved exclusively for the long message transmission. We term this mechanism the "random scheduling approach" because, while it schedules undisturbed long message transmissions, such scheduling activities are initiated randomly by distributed users without coordinated multi-user planning. In this section, we further extend the distributed MAC algorithm to include a random scheduling approach.

For 802.11 DCF, the random scheduling approach enabled long messages to be transmitted at the maximum single user rate. Therefore, if communication is dominated by long message transmissions, overall rate of the system becomes close to the single user rate. When multi-packet reception is supported at the receiver, however, maximum sum rate of the channel can increase in the number of participating users. Since long message transmissions are scheduled randomly, the number of participating users in each scheduled transmission is a random variable. Consequently, even when communication is dominated by scheduled transmissions, calculating the overall sum rate of the system can still be a tricky task.

Let us assume that short packets of the users are still transmitted using a distributed MAC protocol such as those introduced in Section III. Assume that an RTS message can be embedded in any short packet, irrespective of the chosen transmission option. Because the distributed MAC protocol may support parallel multi-packet reception, it is possible that multiple RTS messages can be received simultaneously. In the following, we present a random scheduling approach that can be inserted as an intermediate Step 4.5 into any of the distributed MAC algorithms introduced in Section III.

Proposed Random Scheduling Approach:

Step 4.5) In each time slot, the receiver checks whether an RTS message is received. If so, the receiver should respond with a CTS message using a feedback channel. The CTS message should include the total number of RTS messages received in parallel, and the corresponding sizes of long messages involved in the transmission requests.

Upon hearing a CTS message, users should calculate the length of the scheduled long message transmission assuming that messages should be sent in parallel at the supported maximum sum rate. Scheduled long message transmission should then begin with appropriately designed channel codes. Other users hearing the RTS/CTS exchange should pause their communication activities, and should remain idle during this process.

Once the long messages are received successfully, the receiver should send an ACK message to confirm the reception.

It is not difficult to carry out theoretical performance analysis of the random scheduling approach if we focus on the throughput of the long message transmissions only. Such a focus helps to remove from the results the impact of the sizes of messages, and therefore can make the results relatively easy to understand. Let R_s be the expected sum throughput of a scheduled long message transmission, in bits per symbol. Denote by P_N the probability of scheduling the parallel transmissions of N long messages. Denote the sum throughput in bits per symbol of a scheduled transmission of N messages by $R_N = \frac{1}{2}\log(1 + N \times \text{SNR})$. Let S_m be the expected size of a long message. R_s can be obtained by

$$R_s = \frac{E[\text{\# of message bits}]}{E[\text{\# of channel symbols}]} = \frac{\sum_N P_N N S_m}{\sum_N P_N \frac{N S_m}{R_N}} = \frac{\sum_N P_N N}{\sum_N P_N \frac{N}{R_N}}. \quad (11)$$

Given a particular distributed MAC algorithm, P_N can be further calculated using the channel model and the stationary transmission probabilities of the users, as we will illustrate in the following example.

Example (Continued): Let us consider the example presented in Section III. We assume that each user has a constant probability, denoted by p_s, to embed an RTS message in a short packet, irrespective of the chosen transmission option.

For the modified 802.11 DCF protocol, because rate of the scheduled long message transmission is fixed at the single user rate, we have $R_S = \frac{1}{2}\log(1 + \text{SNR})$.

For the proposed fast adaptation algorithm, stationary short packet transmission probability of each user is given by $p_t = \sum_{i=0}^{c} b_{i,0}$. Let d_h and d_l be the conditional probabilities that a user chooses the high rate option and the low rate option, respectively. Let N_h and N_l be respectively the number of high rate short packets and low rate short packets being transmitted in a time slot. Define $I(N_h, N_l)$ be the indicator function that $I(N_h, N_l) = 1$ if and only if N_h and N_l satisfy (8). Given the number of users K, we can calculate P_N, which is the probability that N RTS messages are received successfully in parallel, as follows.

$$P_N = \sum_{\substack{N_h,\, N_l \\ I(N_h,N_l)=1 \\ N \le N_h + N_l \le K}} \binom{N_h}{K} \binom{N_l}{K - N_h} \times (p_t d_h)^{N_h} (p_t d_l)^{N_l} (1 - p_t)^{(K - N_h - N_l)} \binom{N}{N_h + N_l} \times p_s^N (1 - p_s)^{(N_h + N_l - N)}. \quad (12)$$

In Figure 3, we set $p_s = 0.4$, and illustrated the throughput of the scheduled long message transmissions in bits per symbol for the proposed distributed MAC algorithm as well as the modified 802.11 DCF. Here throughput gain of the proposed distributed MAC algorithm comes purely from its capability of scheduling multiple long message transmissions in parallel. Note that, to further exploit such throughput gain, one can revise the RTS/CTS random scheduling approach to group long message transmissions requested in multiple successive time slots, and hence to increase the number of participating users in each scheduled transmission event. Investigations on such a revision, however, is beyond the scope of this paper.

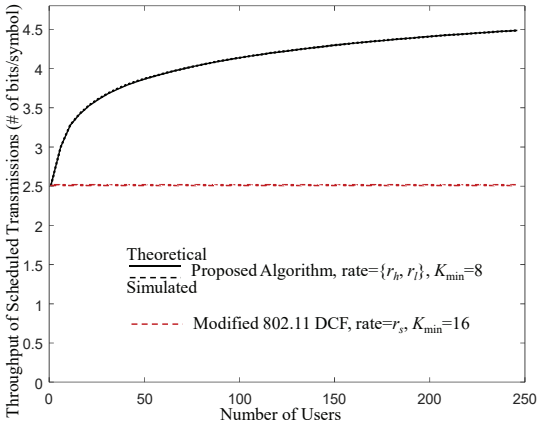

Fig. 3. Throughput of scheduled transmissions as a function of the number of users. SNR = 15 dB, $p_s = 0.4$.

In a practical environment, depending on the relative sizes of long and short messages, overall throughput performance achieved by a MAC algorithm should lie between the throughput of its scheduled transmissions and its short packet transmissions. The probability p_s of requesting a scheduled long message transmission depends on the traffic load. There are other factors such as transmission delay, sizes of the control messages, sizes of the headers of packets, etc, should be taken into consideration. These factors are ignored in this paper to give a relatively clean image about the performance of the MAC algorithms.

V. Conclusion

We extended the distributed MAC algorithm of [1] toward a practical MAC protocol with added components of "fast adaptation algorithm" and "random scheduling approach". We provided theoretical throughput analysis and showed that it matches well with the simulated results. In an example with a Gaussian multiple access channel, we demonstrated significant throughput improvement of the MAC algorithm against that of the classical 802.11 DCF, due to the support of parallel multi-user transmissions, both in opportunistic transmissions of short packets as well as in scheduled transmissions of long messages.

References

[1] Y. Tang, F. Heydaryan, J. Luo, *Distributed Coding in A Multiple Access Environment,* Foundations and Trends in Networking, Vol. 12, pp. 260-412, Now Publishers, 2018.

[2] J. Luo and A. Ephremides, *A New Approach to Random Access: Reliable Communication and Reliable Collision Detection,* IEEE Trans. Inform. Theory, Vol. 58, pp. 379-423, Feb. 2012.

[3] Z. Wang and J. Luo, *Error Performance of Channel Coding in Random Access Communication,* IEEE Trans. Inform. Theory, Vol. 58, pp. 3961-3974, Jun. 2012.

[4] G. Bianchi, *Performance Analysis of the IEEE 802.11 Distributed Coordination Function,* IEEE J. Sel A. in Commun., Vol. 18, pp. 535-547, Mar. 2000.