Opportunistic Splitting Algorithms For Wireless Networks

Xiangping Qin and Randall Berry
Dept. of Electrical and Computer Engineering
Northwestern University
2145 Sheridan Rd., Evanston IL 60208
Email: sandra@ece.nwu.edu, rberry@ece.nwu.edu

***Abstract*— In this paper, we develop medium access control protocols to enable users in a wireless network to opportunistically transmit when they have favorable channel conditions, without requiring a centralized scheduler. We consider approaches that use splitting algorithms to resolve collisions over a sequence of mini-slots, and determine the user with the best channel. First, we present a basic algorithm for a system with i.i.d. block fading and a fixed number of backlogged users. We give an analysis of the throughput of this system and show that the average number of mini-slots required to find the user with the best channel is less than 2.5 independent of the number of users or the fading distribution. We then extend this algorithm to a channel with memory and also develop a reservation based scheme that offers improved performance as the channel memory increases. Finally we consider a model with random arrivals and propose a modified algorithm for this case. Simulation results are given to illustrate the performance in each of these settings.**

I. Introduction

Recently, "opportunistic scheduling" approaches have received much attention as a means for exploiting the "multiuser diversity" inherent in a wireless setting (e.g., [5–9]). These approaches attempt to schedule transmissions during periods when a user's channel is "good" and hence can support a larger transmission rate. This has a theoretical basis in work such as [2], which shows that to maximize the ergodic capacity of a multiple-access fading channel, only a single user with the best channel state should transmit at any time. Such approaches have been integrated into many recent standards, such as Qualcomm's High Data Rate (HDR) system (1xEV-DO) [4].

In this work, as in [2], we consider an uplink (multiple access) model where a group of mobile users are communicating to a single receiver at base station or access point. The approach in [2] requires a centralized scheduler with knowledge of each user's channel gain. This requires the scheduler to acquire estimates of each users' channel state before making the scheduling decision; the overhead and delay incurred in doing this may limit the system's performance, particularly if the number of active users is large or the channels change rapidly. For example, suppose that each user transmits an orthogonal signal (e.g. via TDMA), which the base station uses to estimate their channel. As shown in the top portion of Fig. 1, the time it takes to measure all the users' channels will grow linearly with the number of users. When the number of the users is large or the channel changes fast, the time required to measure each channel and feedback the scheduling decision may exceed the coherence time of the channel, which will degrade the resulting performance.[1] On the other hand, consider the distributed approach shown in the lower part of Fig. 1. Here, the base station broadcasts a pilot signal to all users, and each user measures its own channel using this pilot signal (here we are assuming that the up and downlink channels are symmetric, as in a time division duplex (TDD) system).[2] This approach only requires one-half a round trip time and scales as the number of users increases. Moreover, for a model with random arrivals, the centralized scheduling scheme requires the base station to know when a new user arrives and when an existing user leaves each time in order to make the right decision. While with a distributed scheduling approach, users can adapt themselves to changing traffic patterns.

In this work, we consider distributed approaches, where each user has knowledge of its own channel conditions, but no knowledge of the other users' channels. The transmission decisions are individually made by each user based on their local channel information. In prior work [3], we have shown that multi-user diversity can still be exploited in a distributed setting by using a simple variation of the slotted Aloha random access protocol, called *channel-aware Aloha*. In this approach, as in Aloha, users randomly transmit packets, but now the transmission probabilities are based on the user's channel statistics. This is related to the decentralized power control approach presented in [10] and the "opportunistic Aloha" protocol studied in [11], [12]. For an all backlogged model, the throughput using the channel-aware Aloha approach increases at the same rate as the optimal centralized scheme which transmits to the best user at any time. Asymptotically the ratio

This research was supported in part by the Motorola-Northwestern Center for Telecommunications and NSF CAREER award CCR-0238382.

[1] If users use a different approach, such as CDMA, to transmit the pilot signals, we still need to ensure enough degrees of freedom are available to support the number of users. Therefore similar scaling problems still exist.

[2] Also, this model is more appropriate when the channel variation is determined primarily by multi-path fading and shadowing effects, and not by interference from other users. For example, this will be the case in a wireless LAN with a single access point or a cellular system with a sufficiently large re-use factor.

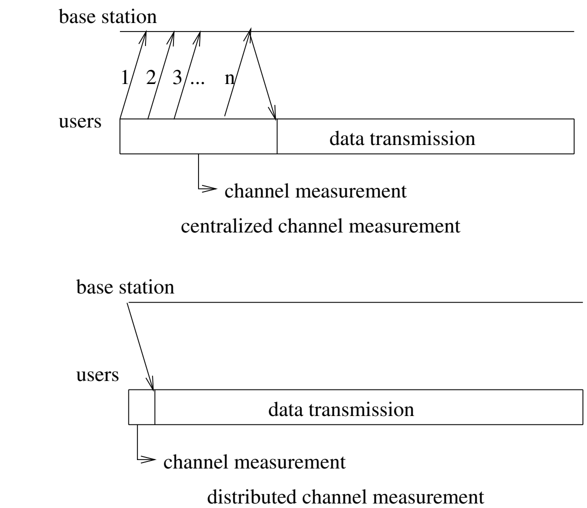

Fig. 1. Time scale of centralized and distributed channel measurement

of these two approaches is equal to $\frac{1}{e}$. In other words, the only penalty incurred due to distributed channel knowledge is due to the contention inherent in the Aloha protocol. In this paper, we consider a distributed splitting algorithm to reduce this contention. A splitting algorithm is an approach that divides the users involved in a collision into several subsets using some tree-like mechanism [1], [13]. Only the user or users in one of the subsets will transmit at the next time slot so that the probability of collision is reduced. Some recent work on splitting algorithms for wireless channels can be found in [15], [16]. The splitting algorithm in this paper differs from traditional splitting approaches in that the goal is not just to resolve a collision but to find the user with the best channel gain out of all backlogged users. By doing this, we show that the throughput is improved and approaches the optimal value as the channel's coherence time increases. We also show that with random arrivals, a splitting approach can improve the delay and stability over the channel aware Aloha approach studied in [3].

The rest of the paper is organized as follows. First, a splitting algorithm is developed for a block fading channel, where the round-trip delay between each transmitter and receiver is less than the channel's coherence-time and all users are backlogged. In Sect. III, we analyze the performance of this algorithm and give upper and lower bounds on the resulting throughput. Next, in Sect. IV, we consider a more realistic channel model where the channel gain changes between each time-slot according to a Markov chain model. For this case, a modified version of the splitting algorithm is introduced and simulation results are given to illustrate the effect of channel memory on the system's performance. This modified splitting algorithm is extended to a reservation scheme similar to the RTS/CTS (request to send/clear to send) handshake used in IEEE 802.11. For sufficiently slow fading, this is shown to improve the overall performance. Finally, in Sect. VI, we study a model with random arrivals and a channel with memory. A

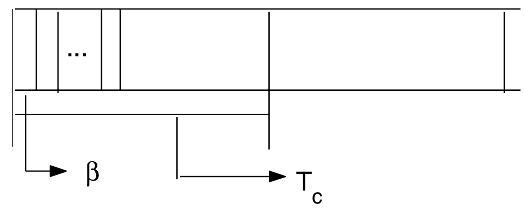

Fig. 2. A time-slot made up of several mini-slots.

further modification to the splitting algorithm is presented. Compared to the Aloha approach from [3]; this algorithm is shown to significantly reduce the average delay when the channel is slowly fading.

II. Splitting Algorithm

We consider a model of the uplink in a wireless network with n users all transmitting to a common receiver. The channel between each user and the receiver is modeled as a time-slotted, block-fading channel; if only the ith user transmits in a given time-slot, the received signal, $y_i(t)$ is given by

$$y_i(t) = \sqrt{H_i} x_i(t) + z(t),$$

where $x_i(t)$ is the transmitted signal, H_i is the fading channel gain, and $z(t)$ is additive white Gaussian noise. Each user has a short-term power constraint that requires the transmission power to be less than P_m during each time-slot.[3] Hence, if only the ith user transmits using this power, the received power level is given by $P_r = H_i P_m$. The resulting transmission rate is a function of P_r. The channel gain is assumed to be fixed during each time-slot and to randomly vary between time-slots. Initially we model the gains of each user at each time as independent (both across users and time) and identically distributed (i.i.d.) random variables with a continuous probability density $f_H(h)$ on $[0, \infty)$. We assume that at the start of each time-slot, each transmitter has knowledge of its own channel gain during the slot, but not the gain of any of the other transmitters. For example, this knowledge could be gained by having the receiver broadcast a pilot signal at the start of each slot as shown in Fig. 1.

Assume the time-scale over which the channel varies is larger than the round-trip time between each transmitter and the receiver. As shown in Fig. 2, at the beginning of each slot, we consider using several mini-slots with length β to communicate with the base station and find the best user. Here β is equal to the round-trip time required for a user to transmit a small reservation packet and detect if a collision occurs. Let T_c denote the length of one time slot within which a user's channel is stable, i.e. this is less than the coherence time of the channel. We begin by considering an idealized model where each time-slot contains K of these mini-slots (i.e., $T_c = K\beta$), and there are n backlogged users in the

[3]Much of the following can also be extended to the case where users have a long-term average power constraint as in [3].

system that always have packets available to send. We assume that n is known by each user.[4] Given these assumptions we describe the splitting algorithm first, and then analyze its performance in the following section. We then proceed to relax these assumptions and provide extensions of this basic algorithm for more realistic channel models and a system with random arrivals.

The purpose of the splitting algorithm is to determine two thresholds, H_l and H_h for each mini-slot, such that at each time only users whose channel gains, h, that satisfy $H_l < h < H_h$ are allowed to transmit. At the end of each mini-slot, each user receives a $(0,1,e)$ feedback, indicating if the mini-slot was idle (0), contained a successful transmission (1), or contained a collision (e). We denote the received feedback by m. If $m = 1$, this means that only the user with the best channel gain transmitted in the mini-slot. In this case, that user will continue to transmit through the remainder of the time slot. If $m = 0$ or $m = e$ then the users will adjust their thresholds and repeat the algorithm until either a success occurs or the time-slot ends. The exact manner in which this is done is given by the following pseudo-code. Here k is the number of mini-slots used so far, and H_{ll} is largest value of H_l used in a prior mini-slot such that it is known that there are some users with channel gains greater than H_{ll}. An example of these quantities is shown in Figure 3.

Basic Splitting algorithm:
initialize: $H_l = F_H^{-1}(\frac{1}{n})$, $H_h = \infty$ and $H_{ll} = 0$
while $m \neq 1$ and $k \leq K$ **do**
  m = (0,1,e) feedback from last slot.
  if $m = e$ **then**
    $H_{ll} = H_l$; $H_l = \text{split}(H_l, H_h)$;
  else if $m = 0$ **then**
    $H_h = H_l$;
    if $H_{ll} \neq 0$ **then**
      $H_l = \text{split}(H_{ll}, H_h)$;
    else
      $H_l = \text{lower}(H_l)$
    end if
  end if
  k = k+1
end while

Here, $F_H(h) = \text{Pr}(H > h)$ denotes the complimentary cumulative distribution function of the channel gains. At the start of a time-slot, the thresholds are initialized to $H_l = F_H^{-1}(\frac{1}{n})$, and $H_h = \infty$, so that the probability that one user's channel gain is above H_l is $1/n$. This choice minimizes the probability of a collision in the first mini-slot. At any time, if a collision occurs ($m = e$), the range $H_l < h < H_h$ is split into two parts (denoted by the function "split"); users in the upper part will transmit in the next mini-slot. If an idle mini-slot occurs ($m = 0$), there are two possibilities: One, as shown in Figure 3 is that there has been a collision before, *i.e.* $H_{ll} \neq 0$. This means that the best channel gain lies between $H_{ll} < h < H_l$. In this case we again split $H_{ll} < h < H_l$ into two parts; the new transmission range will be the upper part. The other possibility is that there has never been a collision before, *i.e.* $H_{ll} = 0$. This means all the users' channel gains are all below H_l, therefore the threshold H_l is lowered (denoted by the function "lower").

[4]In practice n would need to be estimated. This could be done, for example, using a pseudo-Bayesian algorithm [1].

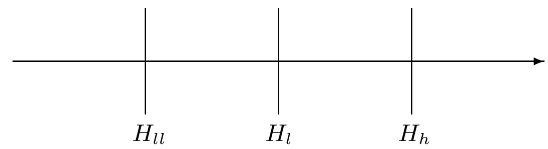

Fig. 3. Example of a split range: H_{ll} is largest value of H_l used in the prior mini-slots such that there are some users above H_{ll}. $H_l < H < H_h$ is the transmission range.

When a collision occurs, the most likely scenario is that two users were involved in this collision [1]. If exactly two users are involved, to maximize the probability of a success in the next mini-slot, the new range should be chosen so that each user transmits with probability 0.5. Therefore, the new splitting threshold, H_t, should be chosen so that

$$\text{Prob}(H > H_t | H \in [H_l, H_h]) = 0.5.$$

Based on this observation, we define the function split(H_l, H_h) as follows:

$$\text{split}(H_l, H_h) = F_H^{-1}\left(\frac{F_H(H_l) + F_H(H_h)}{2}\right),$$

Given that two users are involved in the collision, this can be shown to have the desired properties.

If a mini-slot is idle and there have been no other collisions, then the threshold is lowered using the function lower(H_l). Given n backlogged users all with channel gains less than H_l, we chose this function to maximize the probability of a success in the next mini-slot. Assume the probability that the channel gain is above the current threshold is p_k. After lowering the threshold, the probability that channel gain is above the new threshold is p. Then the probability of a success in the next slot, given an idle feedback is received in the current slot, is given by

$$Q(p) = \frac{n(p - p_k)(1 - p)^{n-1}}{(1 - p_k)^n}. \tag{1}$$

Let p^*_{k+1} be the value of p that maximizes $Q(p)$. Setting $\frac{d}{dp}Q(p) = 0$, we have

$$p^*_{k+1} = p_k + \frac{1}{n} - \frac{p_k}{n}. \tag{2}$$

Therefore, given $p_k = F_H(H_l)$, the desired function is

$$\text{lower}(H_l) = \begin{cases} F_H^{-1}\left(F_H(H_l)(1 - \frac{1}{n}) + 1/n\right) & H_l > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

This completely describes the splitting algorithm. In the next section we present an analysis of its performance.

III. Throughput analysis of the splitting algorithm

We denote the throughput of a system with n users using the splitting algorithm by $s_s(n)$. Compared to a centralized scheduler as in ([2]), the loss in throughput with the splitting algorithm will be the number of mini-slots required to find the user with the best channel. In other words, if each time-slot has a length of T_c seconds, then the throughput ratio of the splitting algorithm to the optimal centralized scheduler is given by

$$\frac{s_s(n)}{s_{ct}(n)} = 1 - \frac{m\beta}{T_c},$$

where m is the average number of mini-slots used per time-slot to find the user who has the best channel gain. Clearly, as n increases, m should increase, yielding a poorer performance for the splitting algorithm. However, it can be shown that though m is increasing, it is bounded, and in the limit of many users, on average only a small number of mini-slots are needed to find the user with the best channel gain. This problem is related to the problem of "partitioning a sample with binary type questions" studied in [14].[5] In this problem, one tries to find the maximum of a sample by asking binary questions. For example, suppose there are n people in a room, and the goal is to find the one who is the oldest by asking 'yes/no' questions such as "is your age greater than 30". The number of questions required to find the oldest is the same as the number of mini-slots required to find the user with the best channel in our problem. In [14], it is shown that the average number of questions required converges to 2.4278 as the sample size increases to infinity. However, in [14], the number of people who answer 'yes' to each question is known, but in our setting, the number of users involved in a collision is unknown. Therefore the number of questions required in [14] provides a lower bound to the number of mini-slots required provided that $\frac{\beta}{T_c} \to 0$. Hence, as $n \to \infty$ and $\frac{\beta}{T_c} \to 0$, the expected number of mini-slots must be greater than 2.4278. Next we will upper-bound the average number mini-slots required by the splitting algorithm. First, given that k users are involved in a collision, the following lemma gives upper and lower bounds on the number of mini-slots required to resolve that collision.

Lemma 1: Let EX_k denote the expected number of mini-slots required to resolve a collision with k users involved. This quantity satisfies

$$\log_2(k) \leq EX_k \leq \log_2(k) + 1,$$

for all k.

Proof: See Appendix I.

Before a collision occurs, some mini-slots may be required in order to find a non-idle range. In other words, we have to take into account the number of times the lower(H_l) function is called at the start of the algorithm.

[5] The relationship of this problem to multiple access issues was also noted in [18].

To simplify our analysis, we modify the lower algorithm according to the following definition:

$$\hat{\text{lower}}(H_l) = \begin{cases} F_H^{-1}\left(F_H(H_l) + 1/n\right), & H_l > 0, \\ 0, & \text{otherwise.} \end{cases} \quad (4)$$

Initially the lower threshold is set so that the probability each user's channel gain is greater than the threshold is $\frac{1}{n}$. Lowering the threshold $l \leq n$ times using this rule results in the (unconditional) probability of a user having a channel gain greater than the new threshold being $\frac{l}{n}$. After lowering the threshold n times, we have $H_l = 0$ and hence there is no need to further lower it. This way of setting the threshold is not optimal (in terms of maximizing the probability of a successful transmission). However, note that from (1) and (2), the probabilities p_k^* corresponding to using lower(H_l) in (2) satisfy:

$$p_{k+1}^* = p_k(1 - \frac{1}{n}) + \frac{1}{n}.$$

Thus, starting with $p_0 = \frac{1}{n}$, then

$$p_1^* = \frac{2}{n} - \frac{1}{n^2} = \frac{2}{n} + O(1/n^2).$$

Iteratively, it follows that $p_k^* = \frac{k}{n} + O(1/n^2)$. When using $\hat{\text{lower}}(H_l)$, the corresponding probabilities are $\hat{p}_k = \frac{k}{n}$. Therefore, $\lim_{n\to\infty} \frac{p_k^*}{p_k} = 1$. Also,

$$Q(p_{k+1}^*) = \frac{(1 - p_k - \frac{1}{n} + \frac{p_k}{n})^{n-1}}{(1-p_k)^{n-1}},$$

and using that $p_k^* = \frac{k}{n} + O(1/n^2)$, we have

$$\lim_{n\to\infty} \frac{Q(p^*)}{Q(\hat{p})} = 1.$$

In other words, $\hat{\text{lower}}(H_l)$ is asymptotically optimal as $n \to \infty$.

The number of mini-slots required by using $\hat{\text{lower}}(H_l)$ is an upper bound for the algorithm using lower(H_l) in previous section. Using this modified algorithm, we have the following upper bound on the average number of mini-slots required to find the best user.

Proposition 1: The average number of mini-slots required, $m(n)$, satisfies

$$m(n) < 2.5070.$$

Proof: See Appendix II.

This bound is independent of the actual fading distribution (assuming a continuous density) and holds for any value of K (the number of mini-slots per time-slot). Also, note that this upper bound is quite close to the lower bound of 2.4278 discussed above; however this lower bound is only valid when $K \to \infty$. For finite K, the algorithm will stop after K mini-slots even if a success is not achieved, which is different from the assumption in [14].

From Prop. 1, it follows that the throughput ratio of the splitting algorithm to the centralized scheme is lower bounded by

$$\lim_{n\to\infty} \frac{s_s(n)}{s_{ct}(n)} > 1 - \frac{2.5070\beta}{T_c}.$$

Obviously, the throughput depends on the ratio of β/T_c. If the round-trip time is much smaller than the coherence time, β/T_c will approach 0 and the throughput will approach that obtained by the centralized scheduler.

Next, suppose that the base station is able to detect the number of users involved in a collision, k. In this case the problem becomes identical to that in [14], and the new range after a collision can be chosen so that each user involved in the collision transmits with probability $1/k$ to maximize the probability of success. The new threshold H_t satisfies:

$$F_H(H_h) - F_H(H_t) = \frac{1}{k}\left(F_H(H_h) - F_H(H_l)\right).$$

The function split(H_l, H_h) is changed to

$$\text{split}_k(H_l, H_h) = F_H^{-1}\left(\frac{F_H(H_l)}{k} + (1-\frac{1}{k})F_H(H_h)\right).$$

We motivated the above splitting algorithms by attempting to maximize the probability of a successful transmission in each time-slot given the information available prior to that slot. Another reasonable criterion would be to minimize the average number of mini-slots required for a success. This approach is also discussed in [14]; however, the difference between the average number of mini-slots under these two criteria is very small. Moreover, using the criterion of maximizing the probability of success in each time-slot results in a much simpler algorithm to analyze; so we focus on this case here.

Simulation results for a Rayleigh fading channel with $K = 40$ mini-slots per time-slot is shown in Figure 4. This figure shows the average number of mini-slots required per time-slot as a function of the number of users in the system. Two sets of results are shown. Let k denote the number of users involved in a collision. The curve labeled 'without knowledge of k' is the number of mini-slots required using the splitting algorithm we described in Sec. II. The other curve is the number of mini-slots required for the above modified splitting algorithm based on having knowledge of k. Both curves are upper bounded by the bound of 2.5070 given by Prop. 1. We can see there is little difference between the two curves. In other words, knowing the number of users involved a collision does not improve the throughput much. The asymptotic lower bound of 2.4278 is also shown; note that since $K < \infty$, this bound does not strictly apply here.

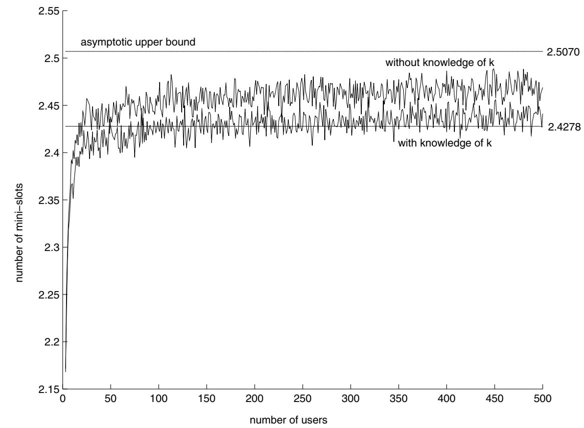

Fig. 4. The average number of mini-slots versus the number of users, with and without knowledge of k (the number of users involved in each collision)

IV. Channel with Memory and Adaptive Splitting Algorithm

In the previous sections, we assumed that each user's channel was independent from one time-slot to another and that the channels stayed fixed for each mini-slot within a time-slot. In this section, we consider a more realistic continuously changing channel. We model the channel as having short-length time-slots, where each user's channel changes slowly from one slot to the other and stays the same during a slot. Specifically, each channel will change independently in next slot with probability r, and stay the same with probability $1-r$. The parameter r will be small when the length of the time slot is small which indicates a large memory; r becomes larger with a longer time slot and indicates a channel that changes faster. Note the length of the time slot has to be less than the coherence time. For simplicity, we consider this idealized channel model with memory; similar results can be shown for more realistic Markov channel models, such as the models in [17] or a first order Gauss-Markov model. In this Markov channel model, instead of transmitting requests first then transmitting data after a successful request, we assume that data packets are transmitted directly in each slot, i.e., there are no mini-slots. If collision happens, the packet gets retransmitted.

First consider using the basic channel-aware Aloha protocol from [3], in which there is fixed threshold and users whose channel gains are above the threshold will transmit in each slot. Here the threshold will not change according to the feedback of the previous slots. In [3], we have shown that for the backlogged model, channel memory has no effect on the total throughput and the throughput ratio of the channel-aware Aloha to the optimal centralized scheme remains $\frac{1}{e}$ regardless of the channel memory.

However, by utilizing the feedback information, channel memory can be used to further improve the throughput. To illustrate this, we first present an adaptive splitting algorithm, which is a variation of the splitting algorithm from Section II. In Section V we discuss a reservation scheme that also takes advantage of increased channel memory. The adaptive splitting algorithm is specified as follows:

initialize: $H_l = F_H^{-1}(\frac{1}{n})$, $H_h = \infty$ and $H_{ll} = 0$
$m =$ (0,1,e) feedback from last slot.
if $m = e$ **then**
  $H_{ll} = H_l$; $H_l = \text{split}(H_l, H_h)$;
else if $m = 0$ **then**
  $H_h = H_l$;
  if $H_{ll} \neq 0$ **then**
    if $H_l \neq H_{ll}$ **then**

$H_l = H_{ll}$
else
$H_h = \infty$ and $H_{ll} = 0$
end if
else
$H_l = \text{lower}(H_l)$
end if
else if $m = 1$ **then**
Transmit a packet
end if

The functions split(H_l, H_h) and lower(H_l, H_h) are the same as in the original splitting algorithm from Section II.

Because the channel is now changing from slot to slot, the (1,0,e) feedback from the previous slot may not truly indicate the channel states during the current slot. For example, in an interval (H_l, H_h) in which previously a collision occurred, there may now be no users due to changes in the channel gains. Therefore the splitting algorithm introduced in previous section can not be applied directly. The splitting algorithm is modified as follows: if $m = 0$, and $H_{ll} = H_l$, then it is known that the users' channels have changed and the current range is no longer meaningful. In this case, the algorithm is reinitialized and the splitting starts over again. As a result, the efficiency of the splitting is lowered, *i.e.* more slots on average are required than in the original splitting algorithm to have a success. This is reasonable because the original splitting algorithm is designed for a more idealized channel model.

Simulation results for this algorithm are shown in Figure 5. This figure shows the ratio of the throughput of the adaptive splitting algorithm to that of the optimal centralized scheduler versus the number of users, once again for a Rayleigh fading channel. We can see when r is small, *i.e.* the channel's memory is long, the adaptive splitting algorithm has a higher throughput. When the memory decreases, the throughput also decreases, and when $r > 0.5$, the throughput ratio is less than $1/e$, *i.e.* less than the throughput of the channel-aware Aloha protocol. The reason is when channel changes fast, the feedback information is not as reliable as when channel changes slowly. Therefore when channel becomes memoryless, the channel-aware Aloha protocol is more suitable. Also as noted previously, if short-length slots are used, the channel memory will be larger and a higher throughput can be achieved, but, of course, additional overhead will be incurred. This can also be viewed as the result of more frequent feedback. Note that the length of a slot must be greater than the round-trip time.

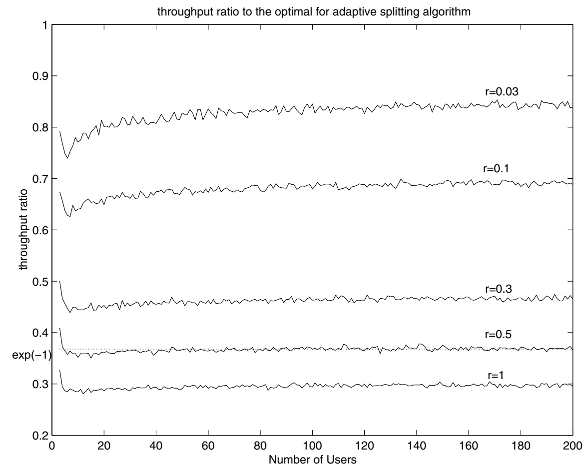

Fig. 5. Ratio of the throughput of the adaptive splitting algorithm to the optimal centralized approach vs. the number of users in a Rayleigh fading channel

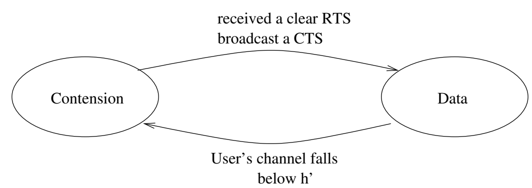

Fig. 6. State transition for the base station

V. Reservation Scheme

When channel has memory, once a user succeeds in one slot, it is reasonable to let the same user continue to transmit in the following slots until its channel becomes bad. We next introduce a reservation scheme based on the adaptive splitting algorithm for this situation. This reservation protocol is illustrated in Figures 6, 7 and 8. In Figure 6, it is shown that the base station has two states: the "contention" state, when all users request to reserve the channel and the "data" state, when one of the users is transmitting data. Figure 7 shows three states which each user may be in. Besides the contention and data states, there is also an idle state; this corresponds to when the user's channel gain is below the threshold or some other user has reserved the channel. Similar to the CSMA-CA technique in IEEE 802.11, this scheme is based on a RTS (Request To Send) and CTS (Clear To Send) handshake. As shown in Figure 8, at the beginning of each slot, the base station transmits a pilot, from which the users measure their channels. According to the adaptive splitting algorithm, those users whose channel gains fall within the current range will transmit RTS packets to the base station. The adaptive splitting algorithm keeps running until the base station receives a collision-free RTS. Then the base station sends out the CTS signal. This CTS signal acts as an inhibiting signal to other users. After the CTS signal is sent out, the data state begins and all other users enter an idle state. Both RTS and CTS contain the requesting user's ID. At the beginning of the data state, the transmitting user will transmit the current range from the adaptive splitting algorithm to the base station. The base station will then monitor the user's channel until the transmitting user's channel drops out of the current range (H_l, H_h). At this time, the base station will transmit a release signal to all other users. This release signal releases the other user's inhibition and the contention state begins again. Fundamentally, there is a trade-off: on one hand, using reservations reduces overhead; on the other hand, users

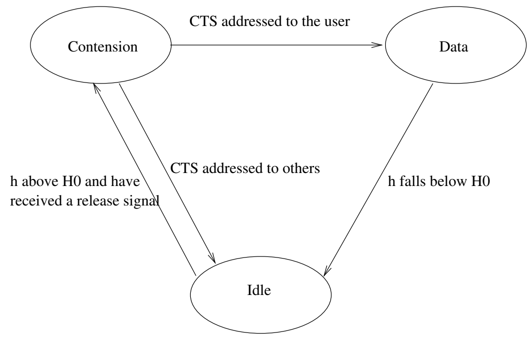

Fig. 7. State transition for the users

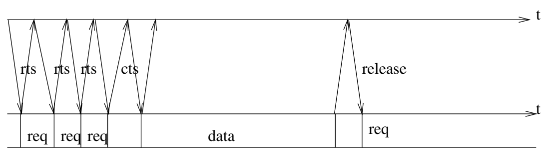

Fig. 8. Timing diagram for the reservation scheme

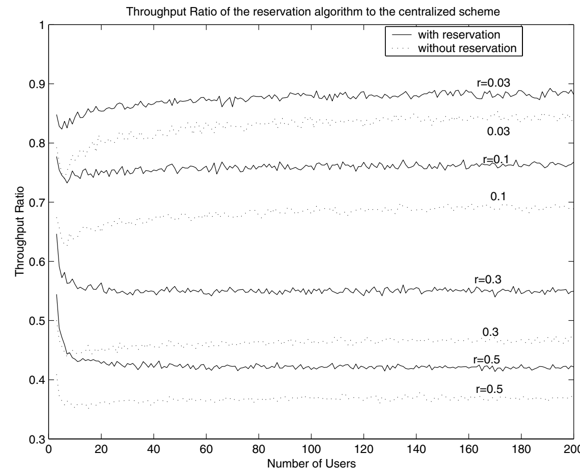

Fig. 9. Ratio of the throughput of the reservation scheme to the centralized scheme versus the number of users in a Rayleigh fading channel with different channel memories.

with a better channel may have no chance to transmit when some other user reserves the channel. This trade-off can be managed by adapting the thresholds out of which the data state stops and anther contention period begins. For example, one modification to reduce the overhead and to keep the data state longer is to let the user continue transmitting until its channel gain $H < H_l$, instead of dropping out of the range (H_l, H_h). However, our simulation results show this has a worse performance, because as mentioned before, users with a better channel have no chance to transmit when another user is transmitting. Other designs based on these ideas are a topic of future research..

Simulation results for this reservation algorithm is shown in Figure 9. The dotted lines are the ratio of throughput of the adaptive splitting algorithm (without reservation) to that of a centralized scheduler; these are the same as in Figure 5. The solid lines represents the throughput ratio of the reservation scheme under the same conditions. We can see that the use of reservation improves the throughput for $r \leq 0.5$. When channel becomes memoryless, the performance advantage of the reservation scheme decreases. As in previous section, in a memoryless channel, the channel aware Aloha approach is a more suitable scheme.

VI. Random Arrivals in Channel with Memory

Next, we consider a model with Poisson arrivals and the same Markov channel model as in previous sections.

First, we consider the channel-aware Aloha approach in this setting. In [3], we have shown that the channel-aware Aloha protocol performs well with random arrivals in a memoryless channel, assuming that users can accurately estimate the number of backlogged users. It is stable for any total arrival rate λ for an infinite user model, and the total delay decreases for a fixed total arrival rate as the number of users in the network increases. This is accomplished by exploiting the increased multiuser diversity present with more users. However, in a channel with memory, the transmission of the packets becomes more bursty and hence the queuing delay becomes larger. Figure 10 shows simulation results for the channel-aware Aloha protocol under different values of the memory parameter r. For each choice of r, the average delay versus the number of users is shown. These simulations are for an infinite user model where packets arrive according to a Poisson process with a total arrival rate of 0.5 packets/second. Each packet has a length of $L = 1000$ bits. The transmission rate of a packet is given by $R = W \log(1 + \frac{P_m H_l}{N_0 W})$, where the bandwidth $W = 1$KHz, the product of the transmission power and the average channel gain $P_m \mathrm{E}(H) = 1$, and the Gaussian noise power is $N_0 W = 1$. Once again the channel gain experience Rayleigh fading. We assume that the length of the time-slot can be adjusted according to the different transmission rate, as in [3]. It can be seen from Fig. 10 that when r is large, *i.e.* when there is less memory, the delay decreases as the number of users increases; when r is small, *i.e.* the memory is large, the delay increases with the number of users. Similar results are shown in Figure 11; this shows the average delay versus the channel memory for different numbers of users, again assuming a total arrival rate of 0.5 packets/second. It can be seen from both Figure 10 and Figure 11 that the delay increases as r decreases.

When the users' channel has memory, the probability that a collision occurs given the previous feedback is a collision is high in the Aloha approach. Therefore, to improve performance, we consider a variation of the adaptive splitting algorithm. Similar to the adaptive splitting algorithm in section IV, the threshold used in the next slot is adjusted according to the feedback of the current slot. Now we only adjust the lower threshold H_l and let all users whose channel gains are above H_l transmit. Because of the dynamics of the system (i.e. both new arrivals may occur and the channels may change), it is

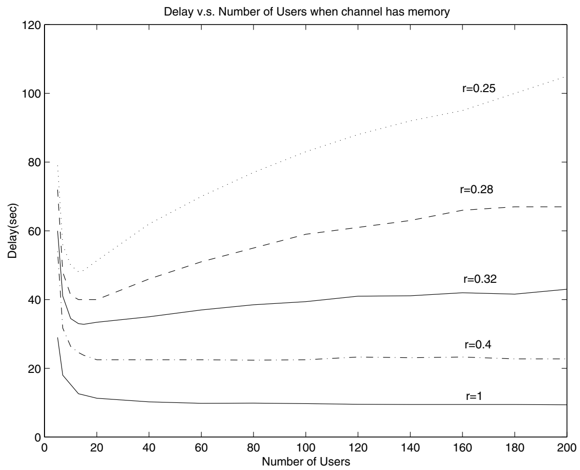

Fig. 10. Delay vs. the number of users for the channel aware Aloha protocol in a Rayleigh fading channel with memory and a total arrival rate of 0.5 packets/second.

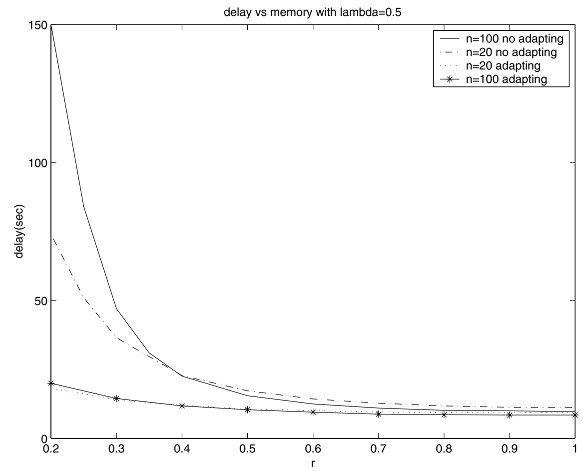

Fig. 12. Average delay vs. the channel memory for different number of users in a Rayleigh fading channel.

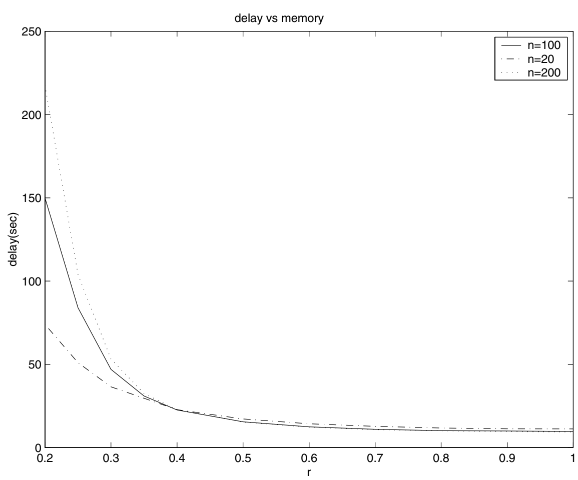

Fig. 11. Delay as a function of channel memory in a Rayleigh fading channel with a total arrival rate of 0.5 packets/second.

hard to estimate the upper threshold H_h. The complexity of this adjusting algorithm with only adjusting H_l is low and our simulation results show that it results in better performance than using the pure Aloha approach. We still assume that the number of backlogged users in the current slot n is known, as well as the number in the last slot n_{-1}. The modified splitting algorithm is given by the following:

initialize: $H_l = F_H^{-1}(\frac{1}{n})$
$m =$ (0,1,e) feedback from last slot.
if $m = e$ **then**
  if $n > n_{-1}$ **then**
    $H_l = F_H^{-1}(\frac{1}{n})$
  else
    $H_l = \text{split}(H_l)$;
  end if
else if $m = 0$ **then**
  if $n < n_{-1}$ **then**
    $H_l = F_H^{-1}(\frac{1}{n})$
  else
    $H_l = \text{lower}(H_l)$
  end if
else if $m = 1$ **then**
  Transmit a packet
end if

In this case

$$\text{split}(H_l) = F_H^{-1}\left(\frac{F_H(H_l)}{2}\right).$$

As mentioned before, only the lower threshold H_l is adjusted and the range is split by increasing H_l, so that the probability that the channel gain is above the new H_l is half of the probability that the channel gain is above the original H_l. The function lower(H_l) is the same as before. Because users arrive randomly, when a collision happens, there could be two reasons. One reason is that there are new arrivals; in other words, the number of backlogged users in the system increases. In this case, the threshold is adjusted according to the change of the users as what we did for the channel-aware Aloha model. The other reason is that more than one user's channel gain stays above H_l. In this case, the new threshold is adjusted to the value of split(H_l). Similarly, when an idle feedback is received, the reason could be that some users departed, or it could be that all users' channel gains stayed below H_l. In the later case the new threshold value is adjusted to lower(H_l).

Figures. 12 and 13 show the performance of the modified splitting algorithm compared to the channel-aware Aloha. Once again this is for a Rayleigh fading channel with the same parameters as before. Figure 12 shows delay versus memory for different number of users. It can be seen from Figure 12 that the adaptive algorithm lowers the delay when memory is large, *i.e.* r is small. Also notice that for small r, the delay in

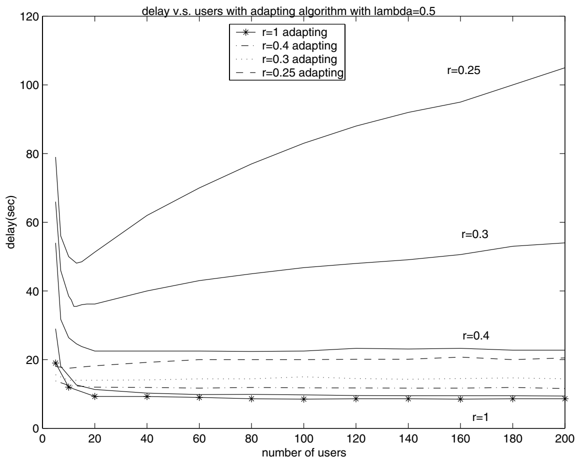

Fig. 13. Average delay vs. the number of users for different values of channel memory, r in a Rayleigh fading channel.

the channel-aware Aloha approach becomes much larger when the number of users increases, while for the splitting algorithm, the delay stays relatively constant as the number of users grows. Figure 13 shows delay versus the number of users for different values of channel memory. The adaptive algorithm reduces the delay significantly for large memories, especially when the number of users is large. These simulations suggest that using the modified splitting algorithm to solve collisions for a channel with memory is effective and a higher throughput is achieved with the same delay constraint.

VII. Conclusion

In this paper, we presented several medium access control algorithms based on splitting for distributed opportunistic transmission in a wireless network. We provided a throughput analysis of basic splitting algorithm in a simplified setting and showed that when the number of mini-slots is large the throughput can approach that achieved by a centralized system. An adaptive splitting algorithm and an reservation scheme are proposed for a channel model with memory and simulation results are given that show improved performance in slow fading environment. For a model with random arrivals, a modified splitting algorithm is applied and performance improvement is shown by simulations. There are many issues that still need to be addressed in future research, such as the study of asymmetric models and the consideration of fairness issues.

Appendix I
Proof of the Lemma.1

Proof: First, we show that $EX_n \le \log_2(n)+1$. $E(X_n)$ can be written as

$$EX_n = (.5)^n \left[\left(\binom{n}{0} + \binom{n}{n} \right)(EX_n+1) + \binom{n}{1}1 + \binom{n}{2}(EX_2+1) + ... + \binom{n}{n-1}(EX_{n-1}+1) \right], \quad (5)$$

where $(.5)^n\binom{n}{i}$ is the probability that after one split there are still i users with channel gains in the upper part of the interval. Therefore, we still need EX_i mini-slots on average to find the best user after the first splitting. Notice when there is no one in the upper part, it means there are n users in the lower part, therefore we will continue to split the lower part.

After simplification, (5) becomes

$$0.5^n \left(\sum_{k=2}^{n-1} \binom{n}{k} EX_k \right) + 1 = (1-0.5^{n-1})EX_n. \quad (6)$$

We then use induction to complete the proof. Initially, $EX_0 = EX_1 = 0$, $EX_2 = 2$, $EX_3 = \frac{7}{3}$ and $EX_4 = \frac{8}{3}$, therefore $EX_k \le \log_2(k)+1$ holds for all $0 \le k \le 4$. Assuming $EX_k \le \log_2(k)+1$, for all $k \le n-1$ and $n > 4$, we prove $EX_n \le \log_2(n)+1$. Using the induction hypothesis in (6), we have

$$\begin{aligned} &0.5^n \left(\sum_{k=2}^{n-1} \binom{n}{k} (\log_2(k)+1) \right) + 1 \\ &\ge (1-0.5^{n-1})EX_n. \end{aligned} \quad (7)$$

Let

$$c = \sum_{k=2}^{n-1} \binom{n}{k} (0.5)^n = 1 - 2(0.5)^n - n(0.5)^n.$$

Using Jenson's inequality, for all $n > 1$, we have

$$\begin{aligned} &0.5^n \left(\sum_{k=2}^{n-1} \binom{n}{k} \log_2(k) \right) \\ &= c \left(\sum_{k=2}^{n-1} \frac{0.5^n \binom{n}{k}}{c} \log_2(k) \right) \\ &\le c \log_2 \left(\frac{0.5^n \left(\sum_{k=1}^{n} \binom{n}{k}(k) - 2n \right)}{c} \right) \\ &= c \log_2 \left(\frac{\frac{n}{2} - 2n0.5^n}{c} \right) \\ &< \log_2 \left(\frac{\frac{n}{2} - 2n0.5^n}{1 - 2(0.5)^n - n(0.5)^n} \right) \\ &< \log_2 \left(\frac{n}{2} \right). \end{aligned}$$

Substituting this into (7) yields

$$\begin{aligned} (1-0.5^{n-1})EX_n &\le \log_2\left(\frac{n}{2}\right) + \sum_{k=2}^{n-1} \binom{n}{k} (0.5)^n + 1 \\ &\le \log_2(n) + 1 - \frac{2}{2^n} - \frac{n}{2^n}. \end{aligned} \quad (8)$$

To complete the induction step, we need to show that

$$\frac{\log_2(n) + 1 - \frac{2}{2^n} - \frac{n}{2^n}}{1 - 0.5^{n-1}} < \log_2(n) + 1. \quad (9)$$

This is equivalent to $\log_2(n) < \frac{n}{2}$, which is true for all $n > 4$. Therefore, $EX_n < \log_2(n) + 1$ as desired.

Next we prove that $X_n > \log_2(n)$. Assume Z_k is the number of users left in the next mini-slot after splitting k times, thus $X_n = \inf\{k : Z_k = 1 | Z_0 = n\}$. Note that given

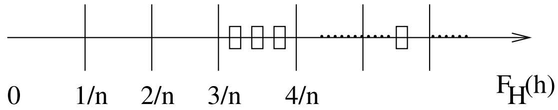

Fig. 14. An example of the splitting sequence with $i = 4$ and $k = 3$. The line corresponds to the value of the complementary distribution $F_H(h)$.

Z_k, Z_{k+1} is independent of Z_i, for $0 < i < k$. Therefore, given Z_k, the expected value of Z_{k+1} is given by

$$E\{Z_{k+1}|Z_k = z\} = \sum_{k=1}^{z} \binom{z}{k} k(.5)^z + z(.5)^z. \tag{10}$$

It follows that

$$E\{Z_{k+1}|Z_k = z\} > \sum_{k=1}^{z} k \binom{z}{k} (.5)^z, \tag{11}$$

and so $E\{Z_{k+1}|Z_k = z\} > \frac{z}{2}$.

Because $Z_0 = n$, $E\{Z_1|Z_0 = n\} > n/2$, and iterating we have,

$$E\{Z_{X_n}|Z_0 = n\} > n/(2^{X_n}).$$

Again using Jenson's inequality, we find

$$\frac{n}{2^{E\{X_n\}}} < E\left\{\frac{n}{2^{X_n}}\right\} < E\{Z_{X_n}\} = 1. \tag{12}$$

Therefore $E\{X_n\} > \log_2(n)$, as desired. ■

Appendix II
Proof of Proposition 1

Proof: To simplify our analysis, we assume the function $\hat{\text{lower}}(H_l)$ defined in (4) is used in the algorithm. To upper bound the average number of mini-slots required, $m(n)$, we first make the pessimistic assumption that there an infinite number of mini-slots in a time-slot. With this assumption, $m(n)$ satisfies

$$m(n) = \sum_{i=1}^{n} \sum_{k=1}^{n} \binom{n}{k} \left(\frac{1}{n}\right)^k \left(1 - \frac{i}{n}\right)^{n-k} (EX_k + i)\,. \tag{13}$$

Here, $\binom{n}{k}(\frac{1}{n})^k(1 - \frac{i}{n})^{n-k}$ is the probability that the first non-idle slot occurs in the ith mini-slot and $k \geq 1$ users are involved. Notice $k = 1$ corresponds to a success and $k > 1$ corresponds to a collision. Figure 14 shows an example with $i = 4$ and $k = 3$, *i.e.* the 4th mini-slot is non-idle and 3 users are involved in a collision. It can be seen that there are k users' whose channel gains are within the corresponding range $F_H^{-1}(\frac{i-1}{n})$ and $F_H^{-1}(\frac{i}{n})$, and all others' gains are less than $F_H^{-1}(\frac{i}{n})$. Let EX_k denote the expected number of mini-slots required to resolve a collision with k users involved, and define $EX_1 = 0$. Lemma 1 gives bounds on EX_k. Using this lemma we have

$$m(n) \leq \sum_{i=1}^{n} \sum_{k=2}^{n} \binom{n}{k} (\frac{1}{n})^k (1 - \frac{i}{n})^{n-k} \times (\log(k) + 1 + i) + \sum_{i=1}^{n} (1 - \frac{i}{n})^{n-1} i.$$

Here the last term corresponds to the case where only $k = 1$ user is in the first non-idle mini-slot. Since,

$$\binom{n}{k} \left(\frac{1}{n}\right)^k = \frac{n!}{(n-k)!k!n^k} < \frac{1}{k!},$$

we have

$$m(n) < \sum_{i=1}^{n} \sum_{k=2}^{n} \frac{1}{k!} (1 - \frac{i}{n})^{n-k} (\log(k) + 1 + i) + \sum_{i=1}^{n} (1 - \frac{i}{n})^{n-1} i. \tag{14}$$

We show that the right-hand side of this expression is bounded by 2.5070 as $n \to \infty$. First we show that for any $\epsilon > 0$, there exists an N large enough, so that for any $n > N$,

$$\sum_{i=N}^{n} \sum_{k=N}^{n} \frac{1}{k!} (1 - \frac{i}{n})^{n-k} (\log(k) + 1 + i) + \sum_{i=N}^{n} (1 - \frac{i}{n})^{n-1} i < \epsilon. \tag{15}$$

This is equivalent to showing that the right-hand side of 14 converges as $n \to \infty$.

Note that since $\lim_{n\to\infty}(1 - \frac{i}{n})^{n-1} = e^{-i}$, then

$$\sum_{i=1}^{n} (1 - \frac{i}{n})^{n-1} i < \sum_{i=1}^{n} M e^{-i} i,$$

for some constant $M > 1$. Therefore $\lim_{n\to\infty} \sum_{i=1}^{n} (1 - \frac{i}{n})^{n-1} i$ converges, and thus for a large enough N, the second term on the left-hand side of (15) satisfies

$$\sum_{i=N}^{n} (1 - \frac{i}{n})^{n-1} i < \frac{\epsilon}{2}. \tag{16}$$

Next we show that the first term on the left-hand side of (14) can also be made arbitrarily small by choosing a large enough N. Let $m_1(n)$ denote this term and let $\alpha \in (0.5, 1)$ be a constant such that αn is an integer and $\alpha n > N$. Then we have

$$\begin{aligned}
m_1(n) &= \sum_{i=N}^{n} \sum_{k=N}^{n} \frac{1}{k!} (1 - \frac{i}{n})^{n-k} (i + \log(k) + 1) \\
&= \sum_{i=N}^{n} \sum_{k=N}^{\alpha n} \frac{1}{k!} (1 - \frac{i}{n})^{n-k} (i + \log(k) + 1) \\
&\quad + \sum_{i=N}^{n} \sum_{k=\alpha n}^{n} \frac{1}{k!} (1 - \frac{i}{n})^{n-k} (i + \log(k) + 1) \\
&< \sum_{i=N}^{n} \sum_{k=N}^{\alpha n} \frac{1}{k!} (1 - \frac{i}{n})^{(1-\alpha)n} (i + \log(k) + 1) \\
&\quad + \sum_{i=N}^{n} \sum_{k=\alpha n}^{n} \frac{1}{(\alpha n)!} (1 - \frac{i}{n})^{n-k} (i + \log(k) + 1).
\end{aligned}$$

Next, since $(1-\frac{i}{n})^{n-k}) < 1$, and $\log(k) < k$,

$$m_1(n) < \sum_{i=N}^{n}\sum_{k=N}^{\alpha n}\frac{1}{k!}(1-\frac{i}{n})^{(1-\alpha)n}(i+\log(k)+1) + \sum_{i=N}^{n}\sum_{k=\alpha n}^{n}\frac{1}{(\alpha n)!}(i+k+1).$$

For large enough N, $(1-\frac{i}{n})^{(1-\alpha)n} < Me^{-(1-\alpha)i}$ for any constant $M > 1$. Thus, for N large enough,

$$\begin{aligned} m_1(n) <& \sum_{i=N}^{n}\sum_{k=N}^{\alpha n}\frac{M}{k!}e^{-(1-\alpha)i}(i+\log(k)+1) + \sum_{i=N}^{n}\sum_{k=\alpha n}^{n}\frac{1}{(\alpha n)!}(i+1+k) \\ <& \sum_{i=N}^{n}\sum_{k=N}^{\alpha n}\frac{M}{k!}e^{-(1-\alpha)i}(i+\log(k)+1) \\ &+ \frac{(n+N+2)(n-N+1)}{2}(1-\alpha)n\frac{1}{(\alpha n)!} \\ &+ \frac{(1+\alpha)n[(1-\alpha)n-1]}{2}(n-N)\frac{1}{(\alpha n)!} \\ <& \epsilon/2. \end{aligned}$$

The last step follows because $\lim_{n\to\infty}\sum_{k=1..n}\log(k)/k!$ and $\lim_{n\to\infty}\sum_{k=1..n}1/k!$ converge.

Combining the above results we have that for N large enough,

$$\sum_{i=N}^{n}\sum_{k=N}^{n}\frac{1}{k!}(1-\frac{i}{n})^{n-k}(\log(k)+1+i) + \sum_{i=N}^{n}(1-\frac{i}{n})^{n-1}i < \epsilon.$$

The rest of the sum in (14) satisfies

$$\begin{aligned} &\lim_{n\to\infty}\sum_{i=1}^{N}\sum_{k=2}^{N}\frac{1}{k!}(1-\frac{i}{n})^{n-k}(\log(k)+1+i) + \sum_{i=1}^{N}(1-\frac{i}{n})^{n-1}i \\ =& \sum_{i=1}^{N}\sum_{k=2}^{N}\frac{1}{k!}\left(\lim_{n\to\infty}(1-\frac{i}{n})^{n-k}\right)(\log(k)+1+i) + \sum_{i=1}^{N}\left(\lim_{n\to\infty}(1-\frac{i}{n})^{n-1}\right)i \\ =& \sum_{i=1}^{N}\sum_{k=2}^{N}\frac{1}{k!}e^{-i}(\log(k)+1+i) + \sum_{i=1}^{N}e^{-i}i \\ <& \sum_{i=1}^{\infty}\sum_{k=2}^{\infty}\frac{1}{k!}e^{-i}(\log(k)+1+i) + \sum_{i=1}^{\infty}e^{-i}i \\ =& 2.5070. \end{aligned}$$

And so, we have $m(n) < 2.5070$ as desired. ■

REFERENCES

[1] D. Bertsekas and R. Gallager, '*Data Networks*, 2nd Ed., Prentice Hall, 1992.

[2] R. Knopp and P. A. Humblet, "Information capacity and power control in single-cell multiuser communications," *Proc. IEEE ICC '95*, Seattle, WA, June 1995.

[3] X. Qin and R. Berry, "Exploiting Multiuser Diversity for Medium Access Control in Wireless Networks," *Proc. IEEE Infocom '03*.

[4] P. Bender., *et al.*, "CDMA/HDR: a bandwidth efficient high speed wireless data service for nomadic users," *IEEE Communications Magazine*, Vol. 38, No. 7, pp. 70-77, July 2000.

[5] X. Liu, E. K. P. Chong, and N. B. Shroff, "Opportunistic Transmission Scheduling with Resource Sharing Constraints in Wireless Networks", *IEEE JSAC*, vol. 19, pp. 2053-2064, Oct. 2001.

[6] S. Borst, CWI, " User Level Performance of Channel-Aware Scheduling Algorithms in Wireless Data Networks", *Proc. IEEE INFOCOM'03* April, 2003

[7] Y. Liu, E. Knightly, "Opportunistic Fair Scheduling over Multiple Wireless Channels", *Proc. IEEE INFOCOM'03* April, 2003

[8] V. Tsibonis, L. Georgiadis, L. Tassiulas, "Exploiting Wireless Channel State Information for Throughput Maximization", *Proc. IEEE INFOCOM'03* April, 2003

[9] P. Viswanath, D. Tse and R. Laroia, "Opportunistic Beamforming using Dumb Antennas", submitted *IEEE Trans. on Information Theory.*

[10] S. Shamai and E. Telatar, "Some Information Theoretic Aspects of Decentralized Power Control in Multiple Access Fading Channels," *1999 Information Theory and Networking Workshop*, June 1999.

[11] P. Venkitasubramaniam, S. Adireddy, and L. Tong, "Opportunistic Aloha and Cross-layer Design for Sensor Networks," *Proc. of IEEE MILCOM*, Oct. 2003, Boston, MA.

[12] S. Adireddy and L. Tong, "Exploiting Decentralized Channel State Information for Random Access," Submitted to *IEEE Transactions on Information Theory,* Nov. 1, 2002.

[13] J. Capetanakis, "The Multiple Access Broadcast Channel: Protocol and Capacity Considerations", IEEE Trans. Infom. Theory, IT-25:505-515, 1979.

[14] K. Arrow, L. Pesotchinsky and M. Sobel, "On partitioning a Sample With Binary-Type Questions in Lieu of Collecting Observations",*Journal of the American Statistical Association*, June 1981.

[15] Garces, R. and Garcia-Luna-Aceves, J.J., "Collision avoidance and resolution multiple access for multichannel wireless networks", Proc. of IEEE INFOCOM 2000, 2000

[16] Van Houdt, B., Blondia, C., "Analysis of an identifier splitting algorithm combined with polling (ISAP) for contention resolution in a wireless access network", IEEE Journal on Selected Areas in Communications, Vol. 18, No. 11 , Nov 2000

[17] "Performance of a Wireless Access Protocol in Correlated Rayleigh Fading Channels with Capture", Chockalingam, Zorzi, Milstein, Venkataraman, IEEE Trans. Comm., 1998.

[18] V. Anantharam and P. Varaiya, "An Optimal Strategy For a Conflict Resolution Problem," in *Open Problem in Communication and Computation*, T. Cover and B. Gopinath(Ed's), Springer-Vorlag, 1987.