Erasures and Noise in Splitting Multiple Access Algorithms

ISRAEL CIDON, MEMBER, IEEE, AND MOSHE SIDI, MEMBER, IEEE

Abstract—A system with many nodes accessing a common receiver is considered. The forward channel is a time-slotted collision-type common radio channel. Due to a nonreliable forward channel, the receiver may misinterpret the actual event of a slot. For instance, an idle or a success slot can be interpreted as a conflict and a conflict or a success slot can be interpreted as an idle slot. The former kind of error is called a noise error, while the latter is called an erasure. Splitting multiple-access algorithms are introduced that can handle erasures as well as noise errors. A remarkable feature of the algorithms is that they ensure that, under stable operation, all packets are eventually successfully transmitted, including the erased packets (those packets that were involved in an erasure). The property that is exploited in devising these algorithms is that nodes whose packets were erased can detect that situation as they transmit and acknowledge that the slot was idle. Consequently, they can either retransmit immediately or wait until some agreed point in time (such as the end of a collision resolution interval) and then transmit. The performances of the proposed algorithms are evaluated according to the maximal throughput they can support for Poisson arrival process. The performance degradation due to erasures and noise errors is quantified.

I. Introduction

SPLITTING multiple-access algorithms have been receiving increased attention during recent years. These algorithms are devised for coordinating the access of many nodes to a common receiver. The common assumptions made in studies of splitting multiple-access algorithms are that the forward channel (the channel used by nodes for transmitting information to the common receiver) is a time-slotted collision-type common channel and that each node can transmit one packet at a time at the beginning of a slot. The packet duration is exactly one slot.

The various splitting multiple-access algorithms differ in the assumptions made for the acknowledgment channel (the channel that is used by the common receiver to inform all nodes on the outcome of a slot). The most common assumption is that the receiver is able to distinguish between three different events: *idle slot* (no node is transmitting), *success slot* (exactly one node uses the channel), and *conflict slot* (two or more packets are transmitted and none is correctly received). This is known as the ternary feedback model. Other assumptions have also been investigated. For example, the binary feedback model [6], the assumption that the receiver is able to detect the multiplicity of a conflict [4], [8], the existence of the capture effect [2], [12], etc.

In most of the studies it is assumed that the forward channel is error-free and that the receiver is able to differentiate the exact event occurring during a slot. However, this assumption of error-free discrimination does not always correspond to reality. Therefore, it is of interest to consider a system in which various errors may occur on the forward channel [13]. Due to such errors the receiver is prone to misinterpretations of the actual outcomes of slots. In principle, any kind of misinterpretation is possible. However, in practice, we expect that each transmitted packet would be encoded with a sufficiently powerful error-detecting code, so that it is very unlikely that an idle slot would be interpreted as a packet.

The kind of errors that we expect to find in practical systems may be classified as either 1) erasures, 2) noise errors, or 3) captures. Interpretations of a conflict or a success slot as an idle slot belong to the class of erasures. The reasons for erasures in practical systems are that mobile users (nodes) may occasionally be hidden (for example, because of physical obstacles) from the receiver, or because of fading problems. Interpretations of an idle or a success slot as a conflict slot belong to the class of noise errors. Such errors are mainly due to additive noises that are intrinsic in any physical radio channel. Captures occur whenever two or more nodes are transmitting and the receiver captures one of them, namely, a conflict is interpreted as a success. The reasons for captures are similar to those of erasures. Note that all other packets involved in the conflict, except the single captured packet, are erased.

Splitting multiple-access algorithms that handle noise errors were presented in [5] and in [8]. In [5] the proposed algorithm is based on the tree collision-resolution algorithm (CRA) [1], [9], while that in [8] is based on Gallager's 0.487 algorithm [3]. Erasures are a lot more difficult to handle, and this is the main issue of this paper.

Manuscript received February 7, 1985; revised September 9, 1985. This paper was presented in part at the 1985 IEEE Symposium on Information Theory, Brighton, England, June 1985, and at the IEEE Global Telecommunications Conference (GLOBECOM '85), New Orleans, LA, December 1985.

I. Cidon is with the IBM, Thomas J. Watson Research Center, Yorktown Heights, NY 10598, on leave from the Department of Electrical Engineering, Technion—Israel Institute of Technology, Haifa 32000, Israel.

M. Sidi was with the Department of Electrical Engineering, Technion—Israel Institute of Technology, Haifa 32000, Israel. He is now on leave with IBM, Thomas J. Watson Research Center, Yorktown Heights, NY 10598, USA.

IEEE Log Number 8610197.

0018-9448/87/0100-0132$01.00

The effect of captures has been studied in [2], [11], and [12]. We should note that in [11] erased packets (those packets that are involved in a capture, besides the captured packet itself) are always *lost*.

In this paper we study the effect of erasures and noise errors on splitting multiple-access algorithms. Specifically, we introduce splitting multiple-access algorithms that can handle erasures as well as noise errors. A remarkable feature of our algorithms is that they ensure that all packets are eventually successfully transmitted, including the erased packets, whenever the arrival rate of new packets to the system is less than the maximal throughput that the algorithms can support.

The paper is organized as follows. In Section II we describe the exact model that we use and its basic assumptions. Section III starts with the description of two splitting multiple-access algorithms based on the basic tree CRA [1], with the necessary modifications needed for handling erasures and noise errors. The two algorithms differ in the actions taken by nodes whose packets were erased. Obviously, nodes whose packets were erased can detect that situation as they transmitted and were acknowledged that the slot was idle. Consequently, they can either retransmit immediately or wait until some agreed point in time (such as the end of a collision resolution interval that is globally known) has been reached and then retransmit. In Section III-A the two algorithms are analyzed. Their performances are evaluated according to the maximal throughput they can support for various error patterns. We discuss the results in Section III-B. Section IV contains the description of two versions of a splitting multiple-access algorithm based on Gallager's algorithm [3]. To facilitate the presentation, we present only algorithms that can handle erasures. It is worthwhile to mention that in Gallager's original algorithm a deadlock might be formed if erased packets are lost. Section IV-A contains the analysis of the two versions of the algorithm, and in Section IV-B we discuss the results. Finally, in Section V we summarize the paper and indicate several directions for future research.

II. Model Description

Consider a system with many (effectively infinite) nodes accessing a common receiver. The forward channel (from the nodes to the receiver) is assumed to be a time-slotted collision-type common radio channel. Each node can transmit one packet at a time whose duration is exactly one slot.

Three events are possible in each slot: idle slot (no node is transmitting during the slot), success slot (exactly one node uses the channel), or conflict slot (two or more nodes use the channel, but none of the individual transmitted packets can be reconstructed at the receiver, and they all have to be retransmitted at some later time). The number of packets involved in a conflict is called the *conflict multiplicity*. For uniformity we assume that an idle slot has conflict multiplicity zero, while a success slot has conflict multiplicity one.

At the end of each slot, the common receiver decides whether the slot was an idle, a success, or a conflict and broadcasts that information (LACK, ACK, or NACK, respectively) via an error-free feedback channel to all nodes, instantaneously. Due to a nonreliable forward channel, the receiver may misinterpret the actual outcome of a slot. Specifically, an idle or a success slot can be interpreted as a conflict. These are called *noise errors*, and their respective probabilities are denoted by $\pi_{0,C}$ and $\pi_{1,C}$. Likewise, a conflict or a success slot can be interpreted as an idle slot. As the possibility of such an event might depend on the conflict multiplicity, we denote by $\pi_{i,0} (i \geq 1)$ the probability of misinterpreting a conflict of multiplicity i as an idle slot. The latter misinterpretations are called *erasures*. We assume that noise errors are the dominant event, namely, if exactly one hidden node is transmitting but the receiver hears some noise, it will broadcast NACK. In addition, it is assumed that noise errors and erasures are probabilistically independent.

To summarize, if no node is transmitting, the receiver will broadcast LACK or NACK with probabilities $1 - \pi_{0,C}$ and $\pi_{0,C}$, respectively. If exactly one node transmits, the receiver will broadcast LACK, ACK, or NACK with probabilities $(1 - \pi_{1,C})\pi_{1,0}$, $(1 - \pi_{1,C})(1 - \pi_{1,0})$, and $\pi_{1,C}$, respectively. Finally, if $i \geq 2$ nodes transmit, then the receiver will broadcast LACK or NACK with probabilities $\pi_{i,0}$ and $1 - \pi_{i-0}$, respectively. Other assumptions upon the dominant events are possible, and we discuss this issue in Section V.

One should observe that with this model, though the receiver broadcasts the same information to all nodes, it may happen that different nodes will have different knowledge about what really happened during a particular slot. To see that, assume that some nodes have transmitted a packet in a certain slot and were acknowledged by a LACK. Obviously, these nodes are aware of the error made by the receiver, however, no other node in the system is. Subsequently, those nodes whose packets were "erased" are said to belong to an *erased set* until they retransmit the erased packets. In devising multiple-access algorithms, one can take advantage of the extra information available to erased set nodes. We may note here that in [11], erased set nodes never transmit their packets successfully.

III. Tree-Splitting Algorithms in the Presence of Noise Errors and Erasures

In this section we introduce multiple-access algorithms for channels with noise errors and erasures. The algorithms of this section are based on the tree CRA [1], [5]. If the channel were free of noise errors and erasures, then the tree CRA is as follows. After a collision, all nodes involved flip a binary coin; those flipping zero retransmit in the very next slot; those flipping one retransmit immediately after the collision (if any) among those flipping zero has

been resolved. No new packets may be transmitted until after the initial collision is resolved [5]. It is said that a conflict is resolved precisely when all nodes of the system become aware that all initially colliding packets have been successfully retransmitted. The time elapsed from an initial conflict until it is resolved is called the conflict-resolution interval (CRI). In [5] Massey described a very simple algorithm that can be distributively implemented by the nodes of the system so that each node will know when to transmit and when a CRI ends.

As was pointed out in [5], the presence of noise errors would not require any changes in the tree CRA. However, if erasures occur, the problem that we face is to determine what actions should be taken by nodes that transmitted a packet and were acknowledged by a LACK. Remember that only these nodes know that the receiver made an error. All other nodes continue the basic tree CRA without any changes, as they are not able to distinguish a real LACK from an erasure.

One plausible scheme that can be used by nodes that join the erased set is the PERSIST scheme. In this scheme, whenever a node detects that its packet was erased, it retransmits the erased packet in the next slot. Another plausible scheme that can be used by these nodes is called the WAIT scheme. Here nodes that join the erased set during a CRI wait until that CRI ends and then retransmit the erased packets at the first slot of the next CRI. The points in time in which a CRI ends with the WAIT scheme are identical to those points in the basic tree CRA where erasures are perceived as idle slots.

So far we have not addressed the issue of first-time transmission rule, namely, which packets are transmitted for the first time at the beginning of a CRI. In [1] it was suggested that packets that arrive during a CRI will be transmitted at the beginning of the next CRI. This scheme is not very efficient (its maximal throughput is only 0.346) and is also difficult to analyze, as statistical dependencies exist between the CRI's. An alternate approach, that is the one we adopt in this paper, is to "decouple" transmission times from arrival times [3], [5]. To describe that approach we define an *arrival epoch* of length Δ. The ith arrival epoch is the time interval $(i\Delta, i\Delta + \Delta]$. The rule that is used to transmit a new packet that arrived during the ith arrival epoch in the first utilizable slot following the CRI for new packets that arrived during the $i-1$ arrival epoch [5]. Δ is a fixed-length epoch adjusted to maximize the achievable throughput.

Note that in addition to packets that are transmitted for the first time at the beginning of a CRI (according to the foregoing rule) some *residual* packets are transmitted too. For the PERSIST scheme and the WAIT scheme the residual packets are those packets that join the erased set during the last slot and during all slots of the previous CRI, respectively.

In the following section we present the analysis of the WAIT and the PERSIST schemes. Before doing so, we mention that the algorithms can be trivially modified to allow for lost packets as in [11]. This is done by never retransmitting packets from the erased set.

A. Analysis of the WAIT and the PERSIST Schemes

In this section we analyze the performance of the WAIT and the PERSIST schemes. To facilitate the presentation, we first analyze the performance of the schemes without noise errors (i.e., $\pi_{0,C} = \pi_{1,C} = 0$). At the end of the section we indicate the modifications needed in the analysis to incorporate noise errors as well.

To analyze the performance of the two schemes we first have to characterize the evolution of the CRI's. To that end, the following definitions and notations will be used.

1) Packets that are transmitted for the first time at the beginning of the ith CRI are called "ith new packets."

2) Let $\{A_i, i \geq 1\}$ be a sequence of independently identically distributed (i.i.d.) random variables that represent the number of the ith new packets. Let their probability mass function be denoted by $P_A(m)$, i.e., $\Pr\{A_i = m\} = P_A(m)$, $m \geq 0$, $i \geq 1$, and their expectation be $E[A] = \sum_{m=0}^{\infty} mP_A(m)$.

3) Packets that belong to the erased set at the end of the ith CRI are called "ith residual packets." Note that the ith residual packets are transmitted at the beginning of the $i+1$ CRI.

4) Let $\{Y_i, i \geq 0\}$ be a sequence of random variables that represent the number of the ith residual packets ($Y_0 \equiv 0$).

5) Let $\{X_i, i \geq 0\}$ be a sequence of random variables that represent the *total* number of packets transmitted at the beginning of the ith CRI. Note that X_i is the conflict multiplicity of the first slot of the ith CRI.

From these definitions it is clear that

$$X_i = A_i + Y_{i-1}, \qquad i = 1, 2, \cdots. \tag{1}$$

It is also clear that the sequence of random variables $Y_0, Y_1, Y_2, \cdots$ forms a homogeneous discrete-time discrete-state Markov chain. Let the transition probabilities of this chain be denoted by $p(n_2/n_1)$, i.e.,

$$p(n_2/n_1) = \Pr\{Y_i = n_2/Y_{i-1} = n_1\}. \tag{2}$$

Using (1), we notice that

$$p(n_2/n_1) = \sum_{m=0}^{\infty} \Pr\{Y_i = n_2/Y_{i-1} = n_1, A_i = m\} P_A(m)$$

$$= \sum_{m=0}^{\infty} \Pr\{Y_i = n_2/X_i = n_1 + m\} P_A(m). \tag{3}$$

In (3) we use the fact that new and residual packets are treated identically according to the tree algorithm.

Observe that to compute the transition probabilities $p(n_2/n_1)$, we first need to compute

$$P_n(l) = \Pr\{Y_i = l/X_i = n\}. \tag{4}$$

It is worthwhile to note that the quantities $P_n(l)$ are independent of the distribution of new packets.

Obviously, $P_n(l) \equiv 0$ for $l > n$ and $l < 0$ as the number of residual packets of a CRI cannot be negative, nor can it exceed the total number of packets transmitted at the beginning of that CRI. To continue the computation of $P_n(l)$ for $n \geq 0, 0 \leq l \leq n$, we first need the following. Let $Q_i(n)$ be the probability that i of n nodes will flip zero, i.e., $Q_i(n) = \binom{n}{i} p^i (1-p)^{n-i}$ where p is the probability that a node will flip zero. Then we have for the WAIT scheme (similar expressions for the PERSIST scheme are provided in (11)):

$$P_0(0) = 1 \qquad P_1(0) = 1 - \pi_{1,0} \qquad P_1(1) = \pi_{1,0} \tag{5a}$$

$$P_n(l) = (1 - \pi_{n,0}) \sum_{i=0}^{n} Q_i(n) \sum_{k=0}^{l} P_i(k) P_{n-i}(l-k), \quad 0 \leq l \leq n-1, n \geq 2 \tag{5b}$$

$$P_n(n) = \pi_{n,0} + (1 - \pi_{n,0}) \sum_{i=0}^{n} Q_i(n) \cdot \sum_{k=0}^{n} P_i(k) P_{n-i}(n-k), \qquad n \geq 2. \tag{5c}$$

Equation (5a) is self-explanatory. The reason for (5b) is that, to have l residual packets ($l < n$) in a CRI that started with n transmitted packets, we must have the following. 1) The n packets were not erased in the first slot of the CRI. 2) After the nodes were split into i nodes and $n - i$ nodes according to the distribution $Q_i(n)$, the sum of residual packets from the set of i nodes and the set of $n - i$ nodes should equal l. The reason for (5c) is similar to that of (5b) except that now n packets can be erased in the first slot of the CRI. This occurs with probability $\pi_{n,0}$.

From (5) the probabilities $P_n(l)$, $n = 0, 1, 2, \cdots, 0 \leq l \leq n$, can be computed recursively. Therefore, via (3) the transition probabilities of the chain $\{Y_i, i \geq 0\}$ can be computed.

The Markov chain $\{Y_i, i \geq 0\}$ is a recurrent and aperiodic chain. In the Appendix we provide sufficient conditions for this chain to be ergodic when the WAIT scheme is applied. Similar conditions can be derived for the PERSIST scheme. Under these conditions, the steady-state probabilities $P_Y(i)$, $i = 0, 1, 2, \cdots$, of the chain exist, and they are obtained via

$$P_Y(i) = \sum_{j=0}^{\infty} P_Y(j) p(i/j), \qquad i \geq 0 \tag{6a}$$

$$\sum_{i=0}^{\infty} P_Y(i) = 1. \tag{6b}$$

In addition to the steady-state probabilities $P_Y(i)$, $i = 0, 1, 2, \cdots$, two other quantities play an important role in evaluating the efficiency of the proposed algorithms. The first is the average number of packets that are successfully transmitted during a CRI that starts with m new packets and k residual packets (of the previous CRI). Let this quantity be denoted by $M_{m,k}$. The second is the average length (in slots) of that CRI which is denoted by $L_{m,k}$.

Clearly, if a CRI starts with m new packets and k residual packets and ends with l residual packets, then the number of successfully transmitted packets is $m + k - l$. Consequently,

$$M_{m,k} = m + k - \sum_{l=1}^{m+k} l \cdot P_{m+k}(l). \tag{7}$$

We now observe that, when the steady-state probabilities $P_Y(k)$, $k = 0, 1, 2, \cdots$, exist (i.e., the Markov chain $\{Y_i, i \geq 0\}$ is ergodic), the average number of successful transmissions in a CRI is given by $\sum_{m=0}^{\infty} \sum_{n=0}^{\infty} M_{m,k} P_A(m) P_Y(k)$ which after simple manipulations using (7) reduces to $E[A]$. This shows that the ergodicity of the chain $\{Y_i, i \geq 0\}$ implies that all packets are eventually successfully transmitted, including the erased packets.

In our tree algorithms the average length of a CRI does not depend on m and k individually, but rather on their sum. Consequently, if $n = m + k$, then $L_{m,k} = L_{m+k} = L_n$, and we have for the WAIT scheme (similar expressions for the PERSIST scheme are provided in (12)):

$$L_0 = L_1 = 1 \tag{8a}$$

$$L_n = 1 + (1 - \pi_{n,0}) \sum_{i=0}^{n} Q_i(n)(L_i + L_{n-i}), \qquad n \geq 2. \tag{8b}$$

From (8) the quantities L_n, $n \geq 0$, can be computed recursively. Taking a law-of-large numbers viewpoint, it is easy to see that the ratio of the expected number of successfully transmitted packets per CRI to the expected length (in slots) of a CRI is just the maximum output rate (throughput) possible with the algorithm. Therefore,

$$T = \frac{\sum_{m=0}^{\infty} \sum_{k=0}^{\infty} M_{m,k} P_A(m) P_Y(k)}{\sum_{m=0}^{\infty} \sum_{k=0}^{\infty} L_{m,k} P_A(m) P_Y(k)} = \frac{E[A]}{\sum_{m=0}^{\infty} \sum_{k=0}^{\infty} L_{m,k} P_A(m) P_Y(k)}. \tag{9}$$

To continue, we need to specify the probability distribution of the arrival process of packets to the system. In the following we assume that packets arrive to the system according to a Poisson process with rate λ(packets/slot). Each time a CRI is started, a new epoch of length Δ (in slots) is chosen, so that

$$P_A(m) = \frac{(\lambda \Delta)^m e^{-\lambda \Delta}}{m!}, \tag{10}$$

namely, those packets that arrived during the epoch Δ are transmitted at the first slot of the CRI. When the CRI ends, the consecutive epoch of length Δ is chosen, etc.

If $\lambda < T$, then the system would be stable. We notice that the throughput T depends on both the epoch length Δ and on the coin-flipping probability p. These two parame-

ters can be optimized so that the throughput would be maximized. The procedure that is used is as follows. We first maximize T over the parameters $\mu = \lambda\Delta$ and p. Let μ^* and p^* be the optimal parameters and T^* the maximal throughput. Then for each $\lambda < T^*$, $\Delta^* = \mu^*/\lambda$ is used.

PERSIST Scheme: The analysis of the PERSIST scheme is identical to that already presented except that other recursive relations should be used for computing the quantities $P_n(l)$, $n \geq 0, 0 \leq l \leq n$, and L_n, $n \geq 0$. Specifically, for the PERSIST scheme (5) is replaced by

$$P_0(0) = 1 \qquad P_1(0) = 1 - \pi_{1,0} \qquad P_1(1) = \pi_{1,0} \quad \text{(11a)}$$

$$P_n(n) = \pi_{n,0} + (1 - \pi_{n,0}) \sum_{i=0}^{n} Q_i(n) P_i(i) P_n(n), \qquad n \geq 2 \quad \text{(11b)}$$

$$P_n(l) = (1 - \pi_{n,0}) \sum_{i=0}^{n} Q_i(n) \sum_{k=0}^{i} P_i(k) P_{n-i+k}(l), \qquad 0 \leq l \leq n-1, n \geq 2. \quad \text{(11c)}$$

Equation (11a) is identical to (5a) and is self-explanatory. The reason for (11b) is that for a CRI to start with $n \geq 2$ packets and end with n residual packets, we need that either the n packets are erased at the beginning of the CRI (this occurs with probability $\pi_{n,0}$), or else (with probability $1 - \pi_{n,0}$) a collision occurs, resulting in splitting the colliding nodes into the set of i and $n - i$ nodes. When the CRI of the i nodes ends, all of them must belong to the erased set, so they retransmit with the $n - i$ nodes, resulting in a CRI starting again with n packets, and all these packets must belong to the erased set when this CRI ends.

The explanation for (11c) is similar. Here a CRI starts with $n \geq 2$ packets and ends with $0 \leq l \leq n - 1$ residual packets. Therefore, the first slot of the CRI must be a conflict slot. Then the nodes are split into sets of i and $n - i$ nodes. When the CRI of the i nodes ends, any number $0 \leq k \leq i$ of nodes may belong to the erased set. Consequently, the second CRI starts with $n - i + k$ packets, from which exactly l should belong to the erased set at the end.

Note that $P_n(l)$, $n \geq 0, 0 \leq l \leq n$ can be computed recursively via (11a)–(11c). One first solves the quadratic equation for $P_n(n)$, $n \geq 2$, from (11b) and then uses (11c) for $l = n - 1, n - 2, \cdots, 0$.

To compute the average length of a CRI that starts with n packets (L_n, $n \geq 0$) for the PERSIST scheme, (8) should be replaced by

$$L_0 = L_1 = 1 \quad \text{(12a)}$$

$$L_n = 1 + (1 - \pi_{n,0}) \sum_{i=0}^{n} Q_i(n) \left\{ L_i + \sum_{j=0}^{i} P_i(j) L_{n-i+j} \right\}, \qquad n \geq 2, \quad \text{(12b)}$$

and again L_n, $n \geq 0$, can be computed recursively via (12). To understand (12b), note that if a CRI starts with $n \geq 2$ packets and the packets are not erased, then the n nodes are split into i and $n - i$ nodes according to the distribution $Q_i(n)$. Consequently, we would have a sub-CRI of average length L_i. With probability $P_i(j)$ $(0 \leq j \leq i)$ the other CRI will start with $n - i + j$ nodes.

Modifications Needed to Incorporate Noise Errors: Recall that in the foregoing analysis we assumed that noise errors are absent, i.e., $\pi_{0,C} = \pi_{1,C} = 0$. If $\pi_{0,C} \neq 0$ and $\pi_{1,C} \neq 0$, then some modifications in the analysis are required. Specifically, the quantities $P_1(0)$, $P_1(1)$, L_0, L_1, should be modified. All other quantities remain the same since no distinction is made between noise errors and real collisions.

WAIT Scheme: The analysis of the WAIT scheme in the presence of noise errors is identical to that already presented except that L_0 and L_1 in (8a) should be computed as follows (see [5]):

$$L_0 = 1 + 2\pi_{0,C} L_0 \quad \text{(13a)}$$

$$L_1 = 1 + \pi_{1,C}(L_0 + L_1). \quad \text{(13b)}$$

PERSIST Scheme: For the persist scheme $P_1(0)$, $P_1(1)$ in (11a) and L_0, L_1 in (12a) should be computed as follows:

$$P_1(1) = (1 - \pi_{1,C})\pi_{1,0} + \pi_{1,C} \cdot \left[Q_0(1) P_1(1) + Q_1(1) P_1^2(1) \right] \quad \text{(14a)}$$

$$P_0(1) = 1 - P_1(1) \quad \text{(14b)}$$

$$L_0 = 1 + 2\pi_{0,C} L_0 \quad \text{(14c)}$$

$$L_1 = 1 + \pi_{1,C}\left[Q_0(1)(L_0 + L_1) + Q_1(1) \cdot (L_1 + P_1(0) L_0 + P_1(1) L_1) \right]. \quad \text{(14d)}$$

The explanations for (14a) and (14d) are similar to those of (11b) and (12b), respectively. In the following we discuss some of the computational issues involved with the analysis, and we present some numerical results.

B. Discussion and Results: WAIT and PERSIST Schemes

As we showed before, under the ergodicity condition of the chain $\{Y_i, i \geq 0\}$, specified in the Appendix, both the WAIT and the PERSIST schemes possess the remarkable property that *all* packets (including those that are erased) are eventually transmitted successfully. Devising protocols in which the chain $\{Y_i, i \geq 0\}$ is not ergodic is quite easy. For instance, the protocols described in [11] are of that kind since erased packets are always lost, and therefore the erased set is always increasing. Yet the protocols in [11] provide some throughput. However, the output rate (the throughput) is obviously much smaller than the input rate (the arrival rate). This phenomenon is manifest when the erasure probabilities are increased. With the WAIT and the PERSIST schemes the throughput is the same as the arrival rate when the system is stable.

We now describe some of the numerical results that were obtained for the following three situations: 1) $\pi_{0,C} = \pi_{1,C} = 0$, $\pi_{n,0} = \pi$, $\forall n \geq 1$; 2) $\pi_{0,C} = \pi_{1,C} = 0$, $\pi_{1,0} = \pi_{2,0} = \pi$, $\pi_{n,0} = 0$, $n \geq 3$; 3) $\pi_{n,0} = 0.2$, $\forall n \geq 1$, $\pi_{0,C}$ and $\pi_{1,C}$

vary. We restrict ourselves to regions for which we proved that the chain $\{Y_i, i \geq 0\}$ is ergodic.

Situations 1) and 2) correspond to the absence of noise errors. In Figs. 1 and 2 we plot the maximal throughput that can be supported by the WAIT and the PERSIST schemes for situations 1) and 2), respectively. Note that, for any arrival rate that is under the corresponding curves, the system is stable. Also note that in these situations the WAIT scheme slightly outperforms the PERSIST scheme. The same behavior has been observed in other situations that correspond to absence of noise error and to the "realistic" assumption that $\pi_{n,0}$ does not increase with n. The reason for this behavior is explained by noting that with the WAIT scheme, the packets that are erased during a CRI are accumulated and sent at the beginning of the next CRI, thus increasing the number of packets transmitted at the beginning of a CRI and thus reducing the erasure probability (at least for case 2)).

In situation 3) we fix the erasure probabilities and vary the noise errors probabilities, as depicted in Fig. 3 for the WAIT scheme. The results for the PERSIST scheme are

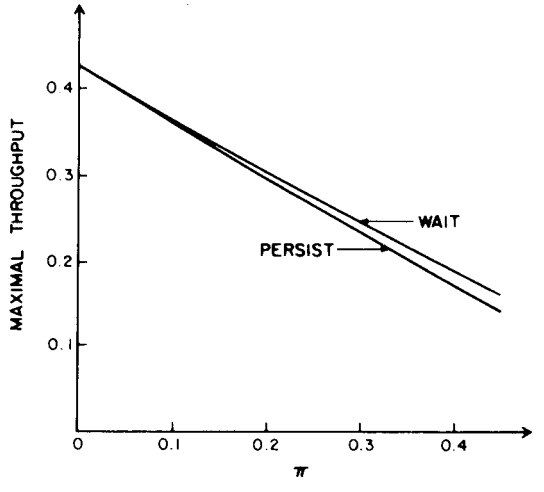

Fig. 1. Maximal throughput: $\pi_{n,0} = \pi, \forall n \geq 1$.

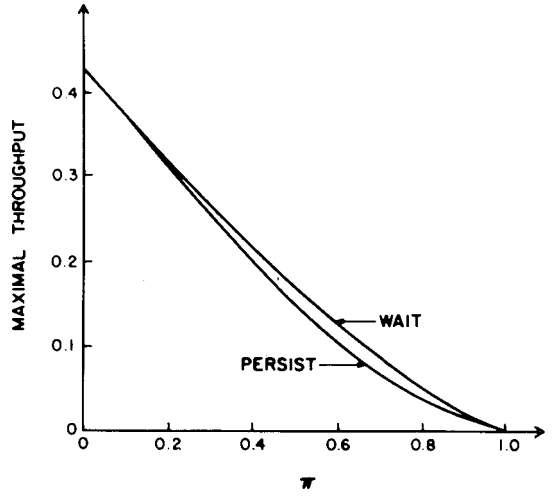

Fig. 2. Maximal throughput: $\pi_{1,0} = \pi_{2,0} = \pi$; $\pi_{n,0} = 0, \forall n \geq 3$.

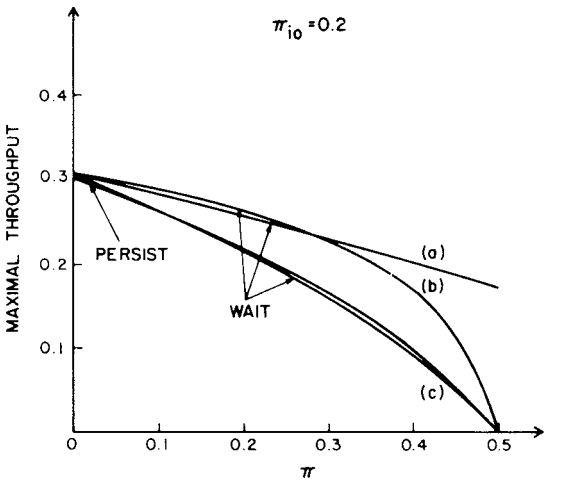

Fig. 3. Maximal throughput. (a) $\pi_{0,C} = \pi$; $\pi_{1,C} = 0$. (b) $\pi_{0,C} = 0$; $\pi_{1,C} = \pi$. (c) $\pi_{0,C} = \pi_{1,C} = \pi$.

similar, and we plot in Fig. 3 only the case $\pi_{0,C} = \pi_{1,C} = \pi$. We notice that, in the presence of noise errors, regions exist in which the PERSIST scheme is slightly better than the WAIT scheme. In this situation, because of the complex behavior of the algorithm, it is difficult to explain the small differences in the observed behavior intuitively.

In the foregoing examples we used $p = 0.5$ (recall that p is the probability that a node will flip 0). As the WAIT scheme is a symmetric scheme, $p = 0.5$ is optimal. For the PERSIST scheme we found that $p = 0.5$ is very close to optimal except for unreasonable high erasure probabilities (beyond 0.8).

IV. Window Algorithms in the Presence of Erasures

In this section we introduce two versions of a multiple-access algorithm that operate in the presence of erasures only. These algorithms are based on Gallager's algorithm [3]. To handle noise errors, one can adopt the ideas presented in [8].

The algorithm that we are going to present is called the *window* algorithm. In this algorithm, each node of the system keeps track of an interval of time which is called a window. The left and right boundaries of the window will be denoted by A and B, respectively. During each slot, all packets that arrived during the current window are transmitted. The system may reside in one of three states S_0, S_1, S_2, depending on the knowledge at the nodes regarding the current window. A CRI is defined to be the interval of time between two successive visits of states S_0.

As our algorithm should operate in the presence of erasures, we need to define two sets of erased packets: the *backlogged* erased packets set and the *residual* erased packets set (backlogged and residual for short). When packets are erased, they always join the backlogged set and remain in that set until they are either successfully transmitted or join the residual set, according to the rules to be described. Packets that join the residual set during a CRI will not be retransmitted during that CRI. They remain in

the residual set until that CRI ends and then join the backlogged set and are retransmitted at the beginning of the consecutive CRI.

We now describe our algorithm (see Fig. 4 for the finite-state machine describing the operation of the algorithm). In state S_0 the system does not have any feedback knowledge regarding the number of packets that arrived within the current window and the number of packets in the backlogged set. This is the point when a CRI starts and all packets of the current window and of the backlogged set are transmitted. If LACK or ACK is received following this transmission, then the system remains in state S_0 (meaning that the current CRI ends) and slides the window forward in time. The new window size is chosen as the smaller between Δ and $T_c - B$ ($A \leftarrow B$, $B \leftarrow \min(\Delta + B, T_c)$). Here Δ is a constant that is chosen so that some performance measure (throughput, for instance) is optimized, and T_c is the current time. In addition, LACK indicates that an erasure might have occurred, in which case all erased packets join the backlogged set.

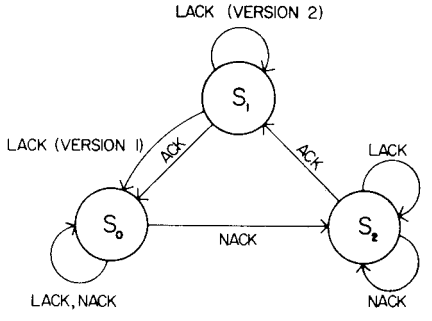

Fig. 4. Window algorithm: finite state machine.

If the system is in state S_0 and NACK is received, then the system enters state S_2 with the knowledge that the window and the backlogged set (together) contain at least two packets. In this transition, the window and the backlogged set are partitioned into left and right parts according to some parameters p_1 and p_2, respectively (these parameters again are chosen so that the performance measure is optimized). The left part of the window becomes the new window ($B \leftarrow A + (B - A)p_1$). The backlogged set is partitioned by flipping a coin with success probability of p_2 by nodes of that set. The nodes that flip success belong to the "left" part of the backlogged set, while all other nodes belong to its "right" part. The "left" part of the backlogged set becomes the new backlogged set. Nodes from both right parts wait until the results of the current transmission is known.

If the system is in state S_2, then LACK or NACK maintain the system in this state. If a NACK is received, the right part of the window is considered as its packets have never been transmitted (this part of the algorithm is the key improvement of Gallager's original algorithm over the tree algorithms). The "right" part of the backlogged set (if any) joins the residual set and will be retransmitted only at the beginning of the next CRI. The new window and the new backlogged set are obtained as in the transition from state S_0 to state S_2, namely, by splitting the current window and the backlogged set into (new) left and right parts.

To see why after receiving a NACK while in state S_2 the right part of the window is statistically indistinguishable from the unexplored portions of the arrival process, let V_L, V_R and W_L, W_R be the left and right parts of the window and backlogged set, respectively. Then for $i \geq 0$, $\Pr\{V_R = i / V_L + V_R + W_L + W_R \geq 2, V_L + W_L \geq 2\} = \Pr\{V_R = i / V_R + W_R \geq 0, V_L + W_L \geq 2\} = \Pr\{V_R = i\}$ where the last equality follows from the fact that V_R and W_R are nonnegative random variables, V_R is independent of V_L (because of the Poisson nature assumed for the arrival process), and W_L contains only packets that arrived prior to the epoch V_R and therefore is independent of V_R.

For LACK while in state S_2, any packet that has been erased (because a misinterpretation has been made) joins the backlogged set. Since the system is already in state S_2, the right parts of the previous window and the newly updated backlogged set clearly contain (together) at least two packets. Therefore, they are partitioned, and a new window and backlogged set are chosen as in the transition from state S_0 to state S_2.

An ACK in state S_2 allows the system to transit to state S_1, where all nodes know that the right parts of the previous window and the previous backlogged set contain (together) at least one packet. These right parts become the new window ($A \leftarrow B$, $B \leftarrow B - A(1 - p_1)/p_1$) and the current backlogged set, respectively.

From state S_1 an ACK causes the system to transit to state S_0 and to start a new CRI by sliding the window forward in time and expanding it to length $\min(\Delta, T_c - B)$. In addition, all packets in the residual set (if any) leave that set and join the backlogged set, so they are retransmitted at the beginning of the new CRI. For a NACK while in state S_1, the same operations are performed as in the transition from state S_0 to state S_2 that is activated by a NACK.

Finally, as erasures may occur, it might be that a LACK will be received while in state S_1. In one version of our algorithm all packets that were erased by this LACK as well as the packets of the residual set (if any) join the backlogged set, the system transits to state S_0, and a new CRI is started. The new window is chosen as in the transition from state S_1 to S_0 caused by an ACK. Alternatively, in our second version of the algorithm we exploit the fact that when the system is in state S_1, it is known that at least one packet has been transmitted, though a LACK has been received. Consequently, in this version all packets that were erased join the backlogged set (the residual set remains unchanged), the system remains in state S_1, and all packets that were transmitted and acknowledged by a LACK retransmit again at the next slot (similarly to our PERSIST scheme).

Before proceeding to the analysis of the two versions of the window algorithm, we should point out that with this algorithm we cannot allow for lost packets. The reason is that, if erased packets are lost, then a deadlock might be formed. To see that, assume that a NACK is followed by a LACK. Because of the NACK the window is partitioned into two parts. Assuming that the left part contains all collided packets, we see that all these packets are erased and, therefore, lost because of the LACK. From this point on, the right part of the window (that is empty) will be indefinitely partitioned, trying to find those packets that originally caused the NACK.

A. Analysis of the Window Algorithm

In this section we analyze the two versions of the window algorithm. The analysis has some similarities to that of the tree algorithms presented in Section III, and we adopt here the same definitions and notations 1)–4) of Section III. In addition, we assume here that the cumulative arrival process of packets to the system is Poisson with parameter λ. Consequently, each time a new window is chosen by the nodes, the number of new packets that are transmitted has Poisson distribution with parameter $\lambda\Delta$. Therefore, we have $P_A(m) = (\lambda\Delta)^m e^{-\lambda\Delta}/m!$.

Recall that $\{Y_i, i \geq 0\}$ is a Markov chain that represents the evolution of the process of residual packets. To compute the transition probabilities of this chain (see (2)), we first consider the following conditional probabilities:

$$P_{m,k}(l) = \Pr\{Y_i = l / Y_{i-1} = k, A_i = m\}, \quad (15)$$

namely, $P_{m,k}(l)$ is the conditional probability of having l residual packets at the end of a CRI, given that the CRI started with m new packets and k residual packets of the previous CRI.

From (2) and (15), using the fact that $\{A_i, i \geq 1\}$ are i.i.d. random variables, we have

$$p(n_2/n_1) = \sum_{m=0}^{\infty} P_{m,n_1}(n_2) P_A(m). \quad (16)$$

To proceed, we first need to compute the probabilities $P_{m,k}(l)$. To that end, we denote by $Q_i(m)$ the probability that i $(0 \leq i \leq m)$ of m nodes whose packets are new will belong to the left part of the window when it is partitioned. Clearly, $Q_i(m) = \binom{m}{i} p_1^i (1-p_1)^{m-i}$ where p_1 is the parameter of the partitioning. Notice that, statistically, $Q_i(m)$ is equivalent to having i of m nodes flipping zero where p_1 is the probability that a node will flip zero. Likewise, we denote by $\hat{Q}_j(k)$ the probability that j $(0 \leq j \leq k)$ of k nodes whose packets belong to the backlogged set will flip zero. Clearly, $\hat{Q}_j(k) = \binom{k}{j} p_2^j (1-p_2)^{k-j}$. Obviously, $P_{m,k}(l) \equiv 0$ for $l > m + k$, $l < 0$ as the number of residual packets of a CRI cannot be negative, nor can it exceed the total number of packets transmitted at the beginning of that CRI. For $m \geq 0$, $k \geq 0$, and $0 \leq l \leq m + k$, the following relations hold for the two versions of the window algorithm:

$$P_{0,0}(0) = 1 \qquad P_{0,1}(0) = P_{1,0}(0) = 1 - \pi_{1,0}$$
$$P_{0,1}(1) = P_{1,0}(1) = \pi_{1,0} \quad (17a)$$

$$\begin{aligned} P_{m,k}(l) = {} & \delta_{m+k}(l)\pi_{m+k,0} \\ & + (1-\pi_{m+k,0})\Big\{ Q_0(m)\hat{Q}_0(k) P_{m,k}(l) \\ & + Q_0(m)\hat{Q}_1(k)[(1-\pi_{1,0}) P^*_{m,k-1}(l) \\ & + \pi_{1,0} P_{m,k}(l)] \\ & + Q_1(m)\hat{Q}_0(k)[(1-\pi_{1,0}) P^*_{m-1,k}(l) \\ & + \pi_{1,0} P_{m-1,k+1}(l)] \\ & + \sum_{\substack{i=0 \\ i+j>1}}^{m} \sum_{j=0}^{k} Q_i(m)\hat{Q}_j(k) \\ & \cdot \big[(1-\pi_{i+j,0}) P_{i,j}(l-k+j) \\ & + \pi_{i+j,0} P_{m-i,k+i}(l)\big] \Big\}, \end{aligned}$$

$$m + k \geq 2, 0 \leq l \leq m + k \quad (17b)$$

where $\delta_{m+k}(l) = 1$ if $l = m + k$ and $\delta_{m+k}(l) = 0$ otherwise. The conditional probabilities $P^*_{m,k}(l)$ are computed differently in the two versions of the window algorithm. In the first version $P^*_{m,k}(l) = P_{m,k}(l)$. For the second version we show in (18) how $P^*_{m,k}(l)$ should be computed.

We now turn to explain the implications of each term in (17). Equation (17a) is self-explanatory. In (17b) the probability of having l residual packets at the end of a CRI that started with m new packets and k packets in the backlogged set $(m + k \geq 2)$ is computed. The term $\delta_{m+k}(l)\pi_{m+k,0}$ corresponds to the case where an erasure occurs at the first slot of the CRI. In this case, if $m + k = l$ $(m + k < l)$, the probability of having l residual packets is one (zero). All other terms correspond to the case where an erasure does not occur at the first slot of the CRI; hence they are multiplied by $1 - \pi_{m+k,0}$. In this case the window and the backlogged set are partitioned, and, therefore, we have the following.

1) If all packets belong to the right parts of the window and the backlogged set after the partitioning $(Q_0(m)\hat{Q}_0(k))$, then the situation is unchanged, and with the same probability $(P_{m,k}(l))$ we would have l residual packets.

2) If all packets except a single one from the backlogged set belong to the right parts of the window and the backlogged set after the partitioning $(Q_0(m)\hat{Q}_1(k))$, that single packet is first transmitted. Then a) if an erasure occurs $(\pi_{1,0})$, we are back at the same situation as in 1) where the nodes are acknowledged by a LACK, hence the term $P_{m,k}(l)$; b) if an erasure does not occur $(1 - \pi_{1,0})$,

then an ACK will be received, and thus we will remain with m packets in the window and $k-1$ in the backlogged set that transmit in the next slot, hence the term $P_{m,k-1}(l)$.

3) If all nodes, except a single one from the new packets, belong to the right parts of the window and the backlogged set ($Q_1(m)\hat{Q}_0(k)$), then the situation is similar to 2) except that if erasure occurs the erased packet joins the backlogged set, and, therefore, we remain with $m-1$ new packets and $k+1$ backlogged packets, hence the term $P_{m-1,k+1}(l)$.

4) If after the partitioning we have i new packets and j backlogged packets in the respective left parts ($Q_i(m)\hat{Q}_j(k)$), then for $i+j>1$ we have the following. a) If an erasure does not occur ($1-\pi_{i+j,0}$), a NACK is received. In this case, all nodes of the corresponding right parts (namely, the $m-i$ and $k-j$ packets) leave the system for the duration of the current CRI. The $m-i$ new packets are considered as if they have never been transmitted while the $k-j$ packets join the residual set. The i and j packets continue to resolve their conflict, hence the term $P_{i,j}(l-k+j)$. b) If an erasure does occur ($\pi_{i+j,0}$), then a LACK is received. In this case the i new packets that were erased join the backlogged set, hence the term $P_{m-i,k+i}(l)$.

This concludes the explanation of (17b).

From (17), $P_{m,k}(l)$ can be computed recursively. We start with computing $P_{0,k}(l)$ for $k=2,3,\cdots, 0\le l\le k$. Then we compute $P_{1,k}(l)$ for $k=1,2,3,\cdots, 0\le l\le k+1$. Then $P_{2,k}(l)$, $P_{3,k}(l),\cdots$, etc., are computed.

As we mentioned before, the conditional probabilities $P_{m,k}(l)$ for the second version of the window algorithm are computed via (17) except that for $P^*_{m,k}(l)$ the following relations hold:

$$P^*_{m,k}(l)=P_{m,k}(l)+\frac{P^*_{0,m+k}(l)\pi_{m+k,0}}{1-\pi_{m+k,0}},$$

$$m+k\ge 2, 0\le l<m+k \quad (18a)$$

$$P^*_{m,k}(l)=P_{m,k}(l)-\pi_{l,0}+\frac{[P^*_{0,m+k}(l)-\pi_{l,0}]\pi_{l,0}}{1-\pi_{l,0}},$$

$$m+k\ge 2, l=m+k. \quad (18b)$$

To understand (18), assume that a single packet (either from the k backlogged packets or the m new packets) belongs to the corresponding left part after the partitioning, and erasure does not occur. Since nodes of the system heard the initial conflict and a following ACK, they know that the right parts of the window and the backlogged set contain together at least one packet. If the transmission of these right parts is erased, then all erased packets join the backlogged set and are retransmitted. This procedure is repeated until erasure does not occur. Notice that since this observation is global, other nodes in the system ignore those subsequent slots, which results in LACK, and do not interfere with the backlogged packets transmission. Using this observation, we can explain (18) as follows.

1) If after the ACK the n new packets and the k backlogged packets are erased, then the m new packets join the backlogged set. Since backlogged sets transmission is repeated until an erasure does not occur, we multiply the probability of the initial erasure ($\pi_{m+k,0}$) by the probability of having l residual packets from a CRI that starts with $m+k$ packets in the backlogged set, given that an erasure does not occur ($P^*_{0,m+k}(l)/(1-\pi_{m+k,0})$ if $l<m+k$) and ($P^*_{0,m+k}(l)-\pi_{m+k,0})/(1-\pi_{m+k,0})$ if $l=m+k$).

2) If an erasure does not occur, then the second version is not activated, and the computation is the same as in the first version.

Using the foregoing analysis, the transition probabilities of the Markov chain $\{Y_i, i\ge 0\}$ can be computed. The Markov chain $\{Y_i, i\ge 0\}$ is clearly recurrent and aperiodic. If it is also ergodic, then the steady-state probabilities $P_Y(i)$, $i=0,1,2,\cdots$ exist and are obtained via

$$P_Y(i)=\sum_{k=0}^{\infty}\sum_{m=0}^{\infty}P_{m,k}(i)P_Y(k)P_A(m) \quad (19a)$$

$$\sum_{i=0}^{\infty}P_Y(i)=1. \quad (19b)$$

As in Section IV, we define $M_{m,k}$ as the average number of packets that are successfully transmitted during a CRI that starts with m new packets and k residual packets, and $L_{m,k}$ as the average length (in slots) of such CRI.

We now turn to present the recursive relations for $L_{m,k}$ and $M_{m,k}$:

$$L_{0,0}=1 \qquad L_{0,1}=L_{1,0}=1 \quad (20a)$$

$$\begin{aligned}L_{m,k}=\pi_{m+k,0}&+(1-\pi_{m+k,0})\Big\{1+Q_0(m)\hat{Q}_0(k)L_{m,k}\\&+Q_0(m)\hat{Q}_1(k)[(1-\pi_{1,0})(1+L^*_{m,k-1})+\pi_{1,0}L_{m,k}]\\&+Q_1(m)\hat{Q}_0(k)[(1-\pi_{1,0})(1+L^*_{m-1,k})+\pi_{1,0}L_{m-1,k+1}]\\&+\sum_{\substack{i=0\\ i+j>1}}^{m}\sum_{j=0}^{k}Q_i(m)\hat{Q}_j(k)\big[(1-\pi_{i+j,0})L_{i,j}+\pi_{i+j,0}L_{m-i,k+i}\big]\Big\},\end{aligned} \quad m+k\ge 2. \quad (20b)$$

Similar expressions are derived for $M_{m,k}$:

$$M_{0,0} = 0 \qquad M_{0,1} = M_{1,0} = 1 - \pi_{0,1} \tag{21a}$$

$$\begin{aligned} M_{m,k} = (1 - \pi_{m+k,0}) \Big\{ & 1 + Q_0(m)\hat{Q}_0(k) M_{m,k} \\ & + Q_0(m)\hat{Q}_1(k)\left[(1 - \pi_{1,0})(1 + M^*_{m,k-1}) + \pi_{1,0} M_{m,k}\right] \\ & + Q_1(m)\hat{Q}_0(k)\left[(1 - \pi_{1,0})(1 + M^*_{m-1,k}) + \pi_{1,0} M_{m-1,k+1}\right] \\ & + \sum_{\substack{i=0 \\ i+j>1}}^{m} \sum_{j=0}^{k} Q_i(m)\hat{Q}_j(k)\left[(1 - \pi_{i+j,0}) M_{i,j} + \pi_{i+j,0} M_{m-i,k+i}\right] \Big\}, \qquad m + k \geq 2. \end{aligned} \tag{21b}$$

The explanation of (20) and (21) is similar to that of (17).

For the first version of the window algorithm, we use $L^*_{m,k} = L_{m,k}$ and $M^*_{m,k} = M_{m,k}$, while for the second version, using the same arguments as in (18), we use for $m + k \geq 1$ the relations $L^*_{m,k} = \pi_{m+k,0} L^*_{0,m+k} + L_{m,k} - \pi_{m+k,0}$ and $M^*_{m,k} = \pi_{m+k,0} M^*_{0,m+k} + M_{m,k}$.

Finally, the throughput is calculated as in (9):

$$T = \frac{\sum_{m=0}^{\infty} \sum_{k=0}^{\infty} M_{m,k} P_A(m) P_Y(k)}{\sum_{m=0}^{\infty} \sum_{k=0}^{\infty} L_{m,k} P_A(m) P_Y(k)}. \tag{22}$$

The maximization of the throughput is done as in Section III.

B. Discussion and Results for the Window Algorithm

Unfortunately, for the window algorithm we were not able to find analytically the ergodicity conditions for the chain $\{Y_i, i \geq 0\}$. Therefore, in the cases that we considered, we checked numerically whether the chain is ergodic or not.

In Fig. 5 we plot the maximal throughput that the system can support with the window algorithms for the case $\pi_{0,C} = \pi_{1,C} = 0$, $\pi_{1,0} = \pi_{2,0} = \pi$, $\pi_{n,0} = 0$, $\forall n \geq 3$. We see that version 2 outperforms version 1. The reason for this behavior is that in version 2 the information available to the nodes is exploited more properly than in version 1. For comparison we also plotted the maximal throughput of the WAIT scheme under the same conditions.

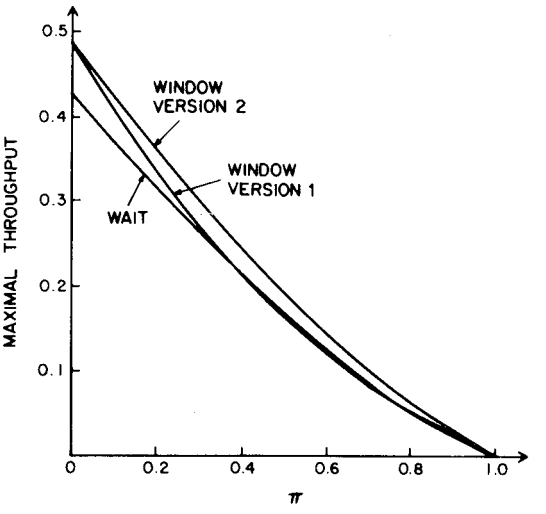

Fig. 5. Maximal throughput: $\pi_{1,0} = \pi_{2,0} = \pi$; $\pi_{n,0} = 0, \forall n \geq 3$.

V. Summary

In this paper we addressed the effects of a class of channel errors, namely, erasures and noise errors, on the operation and performance of a family of splitting multiple-access algorithms. We have shown that it is possible to devise algorithms that ensure that all packets are eventually successfully transmitted, including packets that are erased.

From our analysis we conclude that for many error patterns, the WAIT scheme outperforms the PERSIST scheme in terms of maximal achievable throughput. Yet, we conjecture that when the system is lightly loaded, the PERSIST scheme will yield lower average delay. For the window-type algorithms we have shown how to exploit the information available to the nodes through the feedback channel.

Regarding the underlying model for various types of errors, recall (see Section II) that we assumed that noise errors are dominant, i.e., that when a single hidden node is transmitting and the receiver hears some noise, it will broadcast a NACK. Other assumptions are also possible. For instance, the assumption that erasures are dominant, meaning that when an erasure occurs, it is not important whether or not the packet was first corrupted by noise. Such a change in the assumptions does not affect the derivations in the paper.

Finally, we would mention that it may be of interest to incorporate captures into the proposed algorithms, resulting in multiple access algorithms that operate in presence of noise errors, erasures, and captures. As capture is a positive effect in its nature, it is expected that better performance will be obtained than without captures.

Appendix
Ergodicity of the Chain $\{Y_i, i \geq 0\}$.

In the sequel we derive the conditions for the ergodicity of the chain $\{Y_i, i \geq 0\}$, the number of residual packets from the ith CRI, when the tree algorithm with the WAIT scheme is applied. Let $\{A_i\}$ be the chain of i.i.d. random variables describing the number of new packets transmitted at the beginning of the ith CRI. We assume that these random variables have bounded expectation, i.e., $E[A] < \infty$. As in Section III, we use the notation $P_A(m) = \Pr\{A_i = m\}$. We also recall that $\pi_{n,0} < 1$ for all

$n \geq 1$. The following two theorems provide sufficient conditions for $\{Y_i, i \geq 0\}$ to be ergodic.

Theorem 1: If for some integer $M, \forall n > M$ holds $\pi_{n,0} = 0$, then the chain $\{Y_i, i \geq 0\}$ is ergodic.

Theorem 2: Let $a(n) \equiv 1 - Q_0(n) - Q_n(n)$. If for some integer $M, \forall n > M$ holds $\pi_{n,0} < (a(n)/(1 + a(n))$, then some $\beta^* > 0$ exists such that the chain $\{Y_i, i \geq 0\}$ is ergodic for any arrival process with $E[A] < \beta^*$.

To prove the two theorems we state and prove the following Lemmas. Lemma 1 is due to Pakes [7].

Lemma 1: Let $\{Y_i, i \geq 0\}$, $i \geq 0$, be a discrete-time irreducible aperiodic Markov chain, whose state-space is the set of nonnegative integers. It is ergodic, i.e., a positive stationary probability distribution $\{P_Y(n)\}$ exists such that

$$P_Y(n) = \lim_{k \to \infty} \Pr\{Y_k = n\} > 0, \qquad n \geq 0$$

if

a) $E[Y_{k+1} - Y_k | Y_k = n] < \infty, \qquad \forall n \geq 0,$
b) $\limsup_{n \to \infty} E[Y_{k+1} - Y_k | Y_k = n] < 0.$

Notice that for our chain $\{Y_i\}$,

$$E[Y_{k+1} - Y_k | Y_k = n] = \sum_{m=0}^{\infty} P_A(m) \sum_{l=0}^{n+m} l \cdot P_{n+m}(l) - n. \tag{A1}$$

An important quantity in the following analysis is the average number of residual packets resulting from a CRI which starts with conflict multiplicity n. We denote this quantity by J_n. Clearly,

$$J_n = \sum_{l=0}^{n} l \cdot P_n(l). \tag{A2}$$

We shall compute J_n directly through the following recursive relations:

$$J_0 = 0 \qquad J_1 = \pi_{1,0} \tag{A3a}$$

$$J_n = \pi_{n,0} n + (1 - \pi_{n,0}) \sum_{i=0}^{n} Q_i(n)[J_i + J_{n-i}], \qquad n \geq 2. \tag{A3b}$$

We now show that condition a) of Lemma 1 holds for $\{Y_i, i \geq 0\}$. As $\pi_{i,0} < 1, \forall i \geq 1$, (A3) implies that $J_i < i$ for all $i > 0$ (namely, the number of residual packets is strictly less than the total number of packets). Using (A1) and (A2), we have

$$\begin{aligned} E[Y_{k+1} - Y_k | Y_k = n] &= \sum_{m=0}^{\infty} P_A(m) J_{n+m} - n \\ &< \sum_{m=0}^{\infty} P_A(m)(m + n) - n \\ &= \sum_{m=0}^{\infty} P_A(m) m = E[A] < \infty. \end{aligned} \tag{A4}$$

The following lemmas are used to prove that the conditions of Theorems 1 and 2 are sufficient to fulfill condition b) of Lemma 1.

Lemma 2: If there exist $0 < \alpha < 1$ and an integer M such that for all $n \geq M$, $J_n \leq \alpha n$, then the Markov chain $\{Y_i, i \geq 0\}$ is ergodic.

Proof: For all $n \geq M$

$$\begin{aligned} E[Y_{i+1} - Y_i | Y_i = n] &= \sum_{m=0}^{\infty} P_A(m) J_{n+m} - n \\ &\leq \sum_{m=0}^{\infty} P_A(m) \alpha(n + m) - n \\ &= (\alpha - 1) n + \alpha E[A] \\ &< (\alpha - 1) n + E[A]. \end{aligned} \tag{A5}$$

Since $0 < \alpha < 1$ and $E[A] < \infty$, then $N \geq M$ exist such that for all $n > N$, $E[A] < n(1 - \alpha)$, which ensures that condition b) of Lemma 1 is fulfilled. Since condition a) is always fulfilled, the lemma is proved.

Lemma 3: If some $\beta^* > 0$ and integer M exist such that for all $n \geq M$, $J_n \leq n - \beta^*$, then $\{Y_i, i \geq 0\}$ is ergodic for all processes of new arrivals $\{A_i\}$ with $E[A] < \beta^*$.

Proof: For all $n \geq m$

$$\begin{aligned} E[Y_{i+1} - Y_i | Y_i = n] &= \sum_{m=0}^{\infty} P_A(m) J_{n+m} - n \\ &\leq \sum_{m=0}^{\infty} P_A(m)(n + m - \beta^*) - n \\ &= E[A] - \beta^* < 0 \end{aligned} \tag{A6}$$

which ensures that condition b) of Lemma 1 is fulfilled.

The following two lemmas complete the derivation of sufficient conditions (with respect to the quantities $\pi_{n,0}$ and $E[A]$), ensuring the ergodicity of the Markov chain $\{Y_i, i \geq 0\}$.

Lemma 4: If for some integer M, $\pi_{n,0} = 0$ for all $n \geq M$, then $\alpha, 0 \leq \alpha < 1$, exists such that $J_n \leq \alpha n$ for all n.

Proof: As $\pi_{n,0} = 0$ for $n > M$ (A3) yields

$$J_n = \sum_{i=0}^{n} Q_i(n)[J_i + J_{n-i}], \qquad n > M, \tag{A7}$$

let $\alpha = \max_{1 \leq i \leq M} \{J_i/i\}$. Clearly, $\alpha < 1$ since $J_i < i$ and $J_i \leq \alpha i, \forall 0 \leq i \leq M$.

We now prove by induction that $J_n \leq \alpha n$ for all $n > M$. From (A7),

$$J_n = \frac{\sum_{i=1}^{n-1} Q_i(n)[J_i + J_{n-i}]}{1 - Q_0(n) - Q_n(n)} \tag{A8}$$

using the induction hypothesis:

$$\begin{aligned} J_n &\leq \frac{\sum_{i=1}^{n-1} Q_i(n)[\alpha i + \alpha(n - i)]}{1 - Q_0(n) - Q_n(n)} \\ &= \frac{\alpha n \sum_{i=1}^{n-1} Q_i(n)}{1 - Q_0(n) - Q_n(n)} = \alpha n. \end{aligned} \tag{A9}$$

Lemma 5: If for some integer $M, \forall n > M$, $\pi_{n,0} \leq a(n)/(1 + a(n))$, then some $\beta^* > 0$ exists such that $J_n \leq n - \beta^*$ for all $n > M$.

Proof: Using (A3) and the definition of $a(n)$,

$$J_n = \frac{\pi_{n,0} \cdot n + (1 - \pi_{n,0}) \sum_{i=1}^{n-1} Q_i(n)[J_i + J_{n-i}]}{1 - (1 - \pi_{n,0})(1 - a(n))}. \quad \text{(A10)}$$

We shall prove by induction that, for all $n > M$, $J_n \le n - \beta$ for some $\beta > 0$. Using the induction hypothesis for $M < l \le n - 1$ (namely, $J_l \le l - \beta$) and substituting for $l \le M$, $J_l = l - \beta + J_l - (l - \beta)$, we get

$$\begin{aligned} J_n &\le \frac{\pi_{n,0} \cdot n + (1-\pi_{n,0}) \sum_{i=1}^{n-1} Q_i(n)(n-2\beta) + \sum_{i=1}^{M} [Q_i(n) + Q_{n-i}(n)](J_i - i + \beta)}{1-(1-\pi_{n,0})(1-a(n))} \\ &= \frac{\pi_{n,0} \cdot n + (1-\pi_{n,0})\, a(n)(n-2\beta) + \sum_{i=1}^{M} [Q_i(n) + Q_{n-i}(n)](J_i - i + \beta)}{1-(1-\pi_{n,0})(1-a(n))} \\ &= n - \frac{2\beta(1-\pi_{n,0})\, a(n)}{1-(1-\pi_{n,0})(1-a(n))} + \frac{\sum_{i=1}^{M} [Q_i(n) + Q_{n-i}(n)](J_i - i + \beta)}{1-(1-\pi_{n,0})(1-a(n))} \\ &= n - \beta - \frac{\beta(a(n) - \pi_{n,0}(1+a(n)))}{1-(1-\pi_{n,0})(1-a(n))} + \frac{\sum_{i=1}^{M} [Q_i(n) + Q_{n-i}(n)](J_i - i + \beta)}{1-(1-\pi_{n,0})(1-a(n))}. \end{aligned} \quad \text{(A11)}$$

The induction holds if

$$\frac{-\beta(a(n) - \pi_{n,0}(1 + a(n)))}{1 - (1 - \pi_{n,0})(1 - a(n))} + \frac{\sum_{i=1}^{M} [Q_i(n) + Q_{n-i}(n)](J_i - i + \beta)}{1 - (1 - \pi_{n,0})(1 - a(n))} \le 0 \quad \text{(A12)}$$

for all $n > M$, or

$$-\beta(1 + a(n))\left(\frac{a(n)}{1 + a(n)} - \pi_{n,0}\right) + \sum_{i=1}^{M} [Q_i(n) + Q_{n-i}(n))(J_i - i + \beta) \le 0. \quad \text{(A13)}$$

Since $\pi_{n,0} \le a(n)/(1 + a(n))$, it is easy to see that the left term of (A13) is nonpositive for any $\beta \ge 0$.

Using the fact that $J_i < i$ for all $i \ge 1$, it is easy to show that some $\beta^* > 0$ exists which makes the right term of (A13) nonpositive.

Clearly, choosing $\beta^* = \min_{1 \le i \le M} (i - J_i)$ is sufficient. However, to choose the best β^* derived from (A13), we can select the maximal β which makes (A13) nonpositive for any $n > M$. This yields

$$\beta^* = \inf_{n > M} \left\{ \frac{\sum_{i=1}^{M} [Q_i(n) + Q_{n-i}(n)](i - J_i)}{\sum_{i=1}^{M} [Q_i(n) + Q_{n-i}(n)] + \pi_{n,0}(1 + a(n)) - a(n)} \right\}. \quad \text{(A14)}$$

The proofs of Theorems 1 and 2 follow now from Lemmas 4, 2, 1, and 5, 3, 1, respectively.

In conclusion, if either

1) $\pi_{n,0} < 1 \forall n > 0$, $\pi_{n,0} = 0 \ \forall n > M \ge 0$, $E[A] < \infty$ or

2) $\pi_{n,0} < 1 \forall n > 0$, $\pi_{n,0} \le a(n)/(1 + a(n)) \forall n > M \ge 0$, $E[A] < \beta^*$, where β^* is defined in (A14), then the Markov chain $\{Y_i, i \ge 0\}$ is ergodic.

References

[1] J. I. Capetanakis, "Tree algorithms for packet broadcast channels," *IEEE Trans. Inform. Theory*, vol. IT-25, pp. 505–515, Sept. 1979.

[2] I. Cidon and M. Sidi, "The effect of capture on collision resolution algorithms," *IEEE Trans. Commun.*, vol. COM-33, pp. 317–324, Apr. 1985.

[3] R. G. Gallager, "Conflict resolution in random access broadcast networks," in *Proc. AFOSR Workshop Commun. Theory Appl.*, Sept. 17–20, 1978, pp. 74–76.

[4] L. Georgiadis and P. Papantoni-Kazakos, "A collision resolution protocol for random access channels with energy detectors," *IEEE Trans. Commun.*, vol. COM-30, pp. 2413–2420, Nov. 1982.

[5] J. L. Massey, "Collision resolution algorithms and random-access communications," Univ. of California, Los Angeles, Tech. Rep. UCLA-ENG-8016, Apr. 1980; also in *Multi-User Communications Systems* (CISM Courses and Lectures Series), G. Longo, Ed. New York: Springer-Verlag, 1981, pp. 73–137.

[6] N. Merhavari and T. Berger, "Poisson random multiple-access contention problem with binary feedback," *IEEE Trans. Inform. Theory*, vol. IT-30, Sept. 1984.

[7] Pakes, "Some conditions of ergodicity and recurrence of Markov chains," *Oper. Res.*, vol. 17, 1969.

[8] D. M. Ryter, "A conflict resolution algorithm for noisy multiaccess channels," Mass. Inst. Technol., Cambridge, Rep. LIDS-TH-1007, June 1980.

[9] B. S. Tsybakov and V. A. Mikhailov, "Free synchronous packet access in a broadcast channel with feedback," *Probl. Peredach. Inform.*, vol. 14, pp. 32–59, Oct.–Dec. 1978.

[10] B. S. Tsybakov, "Resolution of a conflict of known multiplicity," *Prob. Inform. Transmission*, vol. 16, no. 2, pp. 134–144 (translated from *Probl. Peredach. Inform.*, vol. 16, pp. 69–82, Apr.–June 1980).

[11] N. D. Vvedenskaya and B. S. Tsybakov, "Random multiple access of packets to a channel with errors," *Prob. Inform. Transmission*, vol. 19, no. 2, pp. 131–147 (translated from *Probl. Peredach. Inform.*, vol. 19, pp. 69–84, Apr.–June 1983).

[12] M. Sidi and I. Cidon, "Splitting protocols in presence of capture," *IEEE Trans. Inform. Theory*, vol. IT-31, pp. 295–301, Mar. 1985.

[13] R. G. Gallager, "A perspective on multiaccess channels," *IEEE Trans. Inform. Theory*, vol. IT-31, pp. 124–142, Mar. 1985.