The Robustness and Performance of Tree Algorithms in an Unshared Feedback Error Environment*

Technical Report 87-15

Tatsuya Suda and Jungok Bae
Department of Information and Computer Science
University of California, Irvine
Irvine, CA 92717
(phone) 714-856-5474

David C. Baxter
Unisys Corporation
19 Morgan St., Irvine, CA 92718
(phone) 714-380-3387

* This material is based upon work supported by the National Science Foundation under Grant No. DCI-8602052. This research is also in part supported by University of California MICRO program and by the University of California, Irvine, the Academic Senate Committee on Research.

Abstract

This paper discusses in detail the robustness of the Capetanakis Collision Resolution Algorithm (CCRA) for multiple access networks. In particular, we show that the algorithm is robust against errors that cause different stations to see different feedback information on the results of the transmission attempts, and we quantify, through simulations, the performance (throughput/delay) degradation caused by those unshared errors.

1. Introduction

In a multiple-access environment, where a number of geographically separated stations communicate over a single shared communication channel, contention-based protocols provide a relatively efficient way of communication [1]. Contention-based access protocols are characterized by collisions and retransmissions. These protocols simultaneously offer transmission rights to a group of stations in the hope that exactly one of the stations has a packet to send. If, however, two or more stations send packets on the channel at the same time, these messages interfere with each other and none of them will be correctly received by the destination station(s). In such cases, stations retransmit packets according to a collision resolution algorithm until packets are successfully received by the destination station(s).

Among a variety of collision resolution algorithms that have been investigated, a class of tree algorithms [2, 3] is one with the outstanding property to maximize the throughput and to minimize the transmission delay between the arrival of a message at the network and its delivery. Among the various tree algorithms, we investigate the Capetanakis Collision Resolution Algorithm (CCRA) [2], because it is known to be simple and easy to implement, yet to yield good performance and stability. The CCRA is also robust to the type of error where all the stations see the same error. The CCRA lends itself to straightforward analysis and simulation, while retaining the favorable stability and performance properties of tree algorithms.

In most of the tree algorithms including CCRA time is slotted, and all the stations agree on the beginning of each slot. One slot length is equal to transmission time of a packet. All stations have access to a feedback channel, through which they can monitor network channel activity. In particular, each station can determine whether each time slot contained zero, one, or more than one transmission attempts. For the purpose of our discussion this feedback is assumed to be immediate and to require none of the channel bandwidth used for data transmission. Using this feedback information, the CCRA provides some of the benefits of cooperation among the stations without imposing any additional traffic as network overhead.

In most of the studies on tree algorithms, it is assumed that the forward channel is error-free, and thus, all stations in the system can observe the channel activity correctly. After each transmission, all stations are correctly informed of whether the slot was empty (LACK), contained one packet (ACK), or contained a collision (NACK). However, in the presence of fading, jamming, physical obstacles, or channel noise, this assumption does not always correspond to reality.

In modeling an error environment, there are two types of errors. The first type of

error is the uniform error where every station sees the same error. The second type of error is an unshared error where different stations may see different errors. In both types of error environment, there are six possible cases of channel errors. Those are LACK-to-NACK, NACK-to-ACK, ACK-to-NACK, ACK-to-LACK, NACK-to-LACK, and NACK-to-ACK errors. LACK-to-NACK error incorrectly identifies an empty slot as a collision, NACK-to-ACK error identifies an collision as a successful slot, and so forth.

This paper demonstrates the robustness of the CCRA to the second type of error in which different stations see different feedback, even though the errors are not generally recognized by the stations at the time the errors occur. The methods we use are a combination of analysis and simulation.

In section 2 of this paper we describe the CCRA and review previous work, especially in regard to treatment of feedback errors. Section 3 describes the network model used in this paper and section 4 uses that model to analyze the response of the CCRA to errors that cause different stations to see different feedback indications. We show that, in the limit of low error frequency, the network quickly recovers from these errors. Section 5 describes a simulation study that quantifies the effects of the errors on the maximum throughput of the network. The final section reviews our conclusions.

2. The Tree Algorithm and Review of Previous Work

2.1. Capetanakis Collision Resolution Algorithm (CCRA)

We first define a collision resolution interval (CRI) as a particular sequence of time slots. In the first time slot, all stations are enabled, meaning that they may attempt to transmit during the slot if they have at least one message to send. If more than one station attempts to send during the initial slot, the transmissions collide and none of the attempts is successful. The CRI terminates when all of those stations have successfully transmitted their messages. A new CRI begins in the next slot after a CRI terminates. The shortest possible CRI has length one, when the number of stations initially attempting to send is zero or one. There is no upper bound for the length of a CRI in which two or more stations collide in the initial slot.

The CCRA is a distributed algorithm that resolves a collision among simultaneously transmitted packets. In the CCRA, a set of colliding stations partition itself when a collision is recognized. One partition remains enabled and attempts to send in the next slot. The other partition is disabled until all of the enabled stations transmit successfully. A simple calculation, performed separately at each station, reveals the termination of the CRI, after which all stations are enabled. An almost identical calculation determines when the stations in a disabled partition can be enabled. The method of partitioning is relatively unimportant, as long as the partitioning algorithm is fully distributed. Message arrival times or random "coin-tossing" are commonly used to implement the partitioning. (The method by which stations recognize the ends of CRIs is described below in section 2.2.)

As an example, Figure 1 shows a sequence of time slots in a CRI for a three station network. All three stations (a, b and c) have a packet to send. Each slot is labeled with a

slot number and the set of stations that attempt to send during that slot. In slot 1, all three stations attempt to send, resulting in a collision. The set of stations $\{a, b, c\}$ is partitioned into $\{b\}$ and $\{a, c\}$. In slot 2, station b sends (i.e., successful transmission in slot 2). In slot 3, stations a and c send and collide again. After the collision in slot 3, $\{a, c\}$ is partitioned into an empty set and $\{a, c\}$. Slot 4 is empty because the set of enabled stations happens to be empty. In slot 5, a and c send. The partitioning of $\{a, c\}$ after slot 5 produces $\{a\}$ and $\{c\}$. In slot 7, all three stations have successfully transmitted their messages.

Note the that following subsequences of time slots in Figure 1 have (some of) the characteristics of a CRI.

- 2
- 3, 4, 5, 6, 7
- 4
- 5, 6, 7

In each of the subsequences, all the stations enabled at the beginning of the sequence have successfully transmitted at the end of the sequence. The only difference between these subsequences and a CRI is that they begin with fewer than all stations enabled. We refer to these sequences as sub-CRIs.

2.2 Tree Representation of a CRI and Distributed Recognition of a CRI

The CRI in Figure 1 can be represented by the binary tree shown in Figure 2. Each node of the tree is labeled with the number of the corresponding time slot. Slot 1 is the root of the tree, and the four subsequences mentioned in the previous subsection are subtrees. The interior nodes (1, 3, 5) of the tree correspond to the time slots that had collisions, and the leaf nodes (2, 4, 6, 7) of the tree correspond to time slots without collisions (i.e., successful slots and empty slots). Any CRI resulting from a binary partitioning of enabled stations can be represented as a binary tree with the same correspondences. A Q-ary partitioning can likewise be represented as a Q-ary tree.

One property of trees leads to the simple calculation by which stations can recognize the termination of a CRI and recognize the termination of a subsequence (i.e., sub-CRI) corresponding to a subtree. For any complete tree, with branching ratio Q,

$$(Q-1)I - L + 1 = 0 \tag{1}$$

where L is the number of leaf nodes, and I is the number of interior nodes. Each station can count interior nodes and leaf nodes, and determine when Equation 1 is satisfied, signaling the end of a CRI. The proof of the above is inductive and is presented in Appendix A.

For binary tree collision resolution, as each station recognizes collisions, successful transmissions, and empty slots, it calculates the expression $E + S - C$, where E is the number of empty slots, S is the number of successful transmissions, and C is the number of collision. In fact, E and S always appear as a sum, and the stations need only distinguish between

collisionless time slots and time slots with collisions. Because

$$E + S = L, \quad C = I, \tag{2}$$

the tree and the corresponding CRI are complete when

$$E + S - C = 1. \tag{3}$$

All stations calculate $E + S - C$ in parallel and, in the absence of errors, agree on the value; they all recognize satisfaction of equation (3) at the same time.

A temporarily disabled station additionally calculates the same expression for the subtree representing the subsequence during which it is disabled. When the corresponding value is one, the subtree, and the subsequence during which the station is disabled, are complete and the station is enabled. All disabled stations calculate the expression independently but, again, in the absence of errors, agree on the value, so they all become enabled simultaneously.

2.3 Past Work on the Effect of Errors on Tree Algorithms

A lot of studies have been made in the performance of tree-based algorithms in uniform error environment, i.e. error environment where every station sees the same error. Massey [4] has examined the error-robustness of CCRA and MCCRA (Modified CCRA) in this error environment. He has considered all six cases of channel errors (described in introduction section) and found that CCRA is robust to channel error whereas MCCRA is extremely sensitive to such channel errors. In practice, however, due to a sufficiently powerful error-detecting code, it is unlikely that LACK-to-ACK and NACK-to-ACK errors occur. From a signal to noise ratio argument, it is also unlikely that a message would be undetected; namely, NACK-to-LACK and ACK-to-LACK errors are unlikely to happen. Therefore, it is not unrealistic to study tree-based algorithms under the assumption that LACK-to-NACK and ACK-to-NACK errors are the only possible errors, and the probabilities of the other kinds of errors (e.g. LACK-to-ACK, NACK-to-ACK, etc.) are equal to zero. Massey has also examined CCRA and MCCRA under this assumption. It has been found that MCCRA is still sensitive to such channel errors. Simple LACK-to-NACK error can cause deadlock creating infinite number of blank-skip slots. It is concluded that the slightly increased throughput of the MCCRA comes at too great a price in increased sensitivity of channel errors compared to the CCRA. A "cure" of the MCCRA is also suggested; by limiting the number of blank-skip slots, such deadlock can be avoided.

Because of the reasons we have explained above, many studies to investigate the tree-based algorithms have been made under the assumption that LACK-to-NACK and ACK-to-NACK errors are the only possible errors. Vvedenskaya and Tsybakov [5] have studied the effect of a noisy channel on free-access stack algorithms. They have shown that stack algorithms are robust to those channel errors. Ho and Wolf [6] have modified MCCRA using Massey's suggestion; he has modified algorithm such that it limits the number of successive skipped slots. It has been shown that this modified algorithm is robust to channel errors. Ryter [7] and Humblet [8] have examined the Gallager's algorithm. They have found that

Gallager's algorithm is not stable for a noisy channel. Ryter modified the algorithm such that it allows to stop splitting an transmission interval at some point and transmit the packets in the entire interval. Humblet modified Gallager's algorithm in a similar way. Both modified algorithms are found to be error-robust.

Cidon, et. al. [9, 10] examined tree-based algorithms with different channel model. They assumed erasure (the phenomena of failing to detect any packet) and capture (the phenomena of detecting only single packet out of many) as well as noise errors (LACK-to-NACK, ACK-to-NACK), and they proposed algorithms for this channel model. In their model, although every station sees the same error, different stations may have different knowledge about what really happened. For instance, those stations whose packets were transmitted in a certain slot and were acknowledged by a LACK can detect the error, whereas no other station in the system can. They can either retransmit immediately (PERSIST scheme) or wait until the current CRI ends and then retransmit (WAIT scheme).

So far, only one study has been made in the second type of error environment where different stations see the different errors. Kurose, et al. [11] have examined stack algorithms in an unshared error environment. In their model, it is assumed that each station independently sees either the correct feedback (LACK, ACK, or NACK) or no feedback at all (NONE). Because of the asymmetric and independent nature of the feedback error, different stations may take different actions as a result of the observed channel feedback information. According to the possible responses to the receipt of a NONE feedback signal, several modifications to the basic stack algorithm are proposed and shown that those modified algorithms are robust to the channel errors.

The work of Kurose is applied to an stack access algorithm similar to the CCRA. After collisions, the colliding stations are partitioned into enabled and disabled sets. Disabled stations monitor the feedback channel and perform the same calculation as disabled station in the CCRA to decide when to become re-enabled. In the stack algorithm, however, stations with new messages can attempt transmission in the next time slot and need not wait until the beginning of CRI.

3. The Model Analyzed in This Paper

For the analysis and simulation described in this paper, we assume that channel time is slotted, and all the stations agree on the beginning of each slot. One slot length is equal to transmission time of a packet. All stations have access to a feedback channel, through which they can monitor network channel activity. In particular, each station can determine whether each time slot contained zero, one, or more than one transmission attempts. This feedback is assumed to be immediate and to require none of the channel bandwidth used for data transmission.

We investigate the CCRA with the arrival epoch modification. (Using the notion of arrival epochs [2, 4] applied to the CCRA simplifies analysis and allows an extra degree of freedom that can be used to optimize performance.) At stations, time is divided into a contiguous sequence of arrival epochs. We assume that arrival epochs are of equal length,

agreed upon ahead of time by all stations.

Messages that arrive in the i-th arrival epoch are transmitted in the i-th CRI. More precisely, a message that arrives in arrival epoch i at station A will be transmitted in the i-th CRI for station A. Note that unshared errors can cause different stations to see different CRIs and, hence, i-th CRI for some station may be j-th CRI for another station ($i \neq j$).

The i-th CRI cannot begin until after all k-th CRIs ($k < i$) end. In a heavily loaded network, the CRI may be delayed a long time, waiting for the completion of earlier CRIs.

To simplify the analysis, we look at the blocked access variation, in which the i-th CRI cannot begin until after the i-th arrival epoch ends. (A free access version allows the i-th CRI to begin before the termination of the i-th arrival epoch.) In a lightly loaded network, the CRI can start immediately after the end of the corresponding arrival epoch. Every station that has a message arrive in an arrival epoch attempts to transmit the message in the first slot of the corresponding CRI.

CRIs need not be contiguous. A CRI may terminate before the next CRI can begin. In that case, the stations must wait, leaving slots empty before the beginning of the next CRI. The important property of the arrival epochs is that all messages that arrive before the end of i-th arrival epoch are successfully transmitted before the end of the i-th CRI. An equivalent property is that all messages that arrive in the i-th arrival epoch are transmitted in the i-th CRI.

The arrival epochs decouple the number of stations attempting to transmit in consecutive CRIs. The number of stations participating in one CRI has no affect on the number of stations that participate in the next CRI, greatly simplifying the performance analysis of the algorithm.

The ideal number of stations to participate in a CRI is one. In that case, no time slots are wasted in collisions or empty slots. The throughput of those CRIs is 1. The worst number of stations to participate in a CRI is zero, for which the throughput is 0. For CRIs with two or more stations, the average throughput is between 0 and 1. By optimally choosing the lengths of the arrival epochs, the distribution of CRI lengths can be adjusted to maximize throughput.

To summarize, we assume in this paper that the CCRA is used with the arrival epoch modification. When a message arrives at a station, the messages is blocked along with all other messages that arrive during the same arrival epoch, until at least the end of the arrival epoch. If messages from earlier arrival epochs are still being transmitted, the new messages are also blocked until all the messages from previous arrival epochs have been successfully transmitted.

Additionally we assume that for some time slots some number of stations see incorrect feedback on the feedback channel. We show that the effect of any single error is temporary, and assume that the error frequency is low enough that no error occurs before recovery from the effects of the previous error. When such an error occurs, it may be any of the various types of error, ACK-to-LACK, NACK-to-ACK, etc. In the following, LACK to NACK or

ACK to NACK error is referred to as a "leaf to interior" error, NACK to ACK or NACK to LACK error as an "interior to leaf" error, LACK to ACK or ACK to LACK error as a "leaf to leaf" error. Note that, when more than one station errs in a slot, all the erring stations make the same type or error (i.e., either "leaf to interior", "interior to leaf" or "leaf to leaf" type error). Accordingly, an error partitions the network into two sets, those that made no error and those that made the (identical type) error.

4. Robustness against Unshared Errors

All the performance and error analysis of the CCRA assumes that all the stations agree on when CRIs begin and end. If errors could cause permanent disagreement among the stations, then that previous work is in applicable, and would have to be redone to include the effects of the errors. The possibility that even a single error could permanently destroy the cooperation among the stations would have to be included in the analysis. Using arrival epochs to control the transmission of messages would be especially affected because the performance enhancements and simplified analysis assume agreement among the stations on the CRIs. This section shows that errors cause only temporary disagreement among stations about when CRIs begin and end. Thus, if the error rate is low, the network spends almost all of its time operating in accordance with the assumptions.

As noted by Massey [4], a long sequence of time slots without collisions, observed by all stations without errors, causes all stations to become synchronized (i.e., all stations agree on the beginnings and ends of CRIs). This agreement occurs because every CRI without a collision in the first slot has length one. Once a station begins to see a sequence of CRIs with length one, every time slot terminates a CRI. If some other station is in the middle of a longer CRI, the sequence of collisionless time slots will eventually terminate the longer CRI and that station will then be synchronized with the stations seeing single slot CRIs. Complete, single slot, CRIs are either empty or contain a successful transmission. Generally, such a sequence of collisionless time slots would only occur when the network has no messages to send. A fortunate coincidence of several successful transmissions without collisions would have the same effect. Note that we have assumed that the sequence of collisionless time slots is observed correctly by all stations. If one or more stations erroneously see a collision on the feedback channel, then those stations will require two additional collisionless time slots before they again become synchronized with the correct stations. Once all the stations have begun to correctly observe a sequence of unit length collisionless CRIs, then all the stations are operating with the same feedback information, and they are synchronized.

This section demonstrates that a somewhat stronger statement can be made. A single error will always be resolved within two CRIs, even if there is no long sequence of collisionless time slots, provided that no more errors occur in the network before the first error is resolved. In other words, within two CRIs after an error, the erring stations recognize the end of a CRI simultaneously with the correct stations. After that synchronization, the erring stations behave correctly in following the protocol of the CCRA. As shown by the results of our simulation study, in section 5, this short error resolution time sharply limits the effects of errors on throughput and response time when the error rate is low.

The proof depends on the structure of the tree that represents a CRI, and considers single errors in isolation. We do not consider the effects of multiple errors before the synchronization is regained. Our analysis, therefore, applies only in the limit of low error frequency, when the higher order effects of multiple errors can be correctly ignored.

Using the tree construction algorithm, described above in section 2.2, even an erring station implicitly constructs a sequence of trees, but the trees may be different from those of the correct stations. Now we examine the differences that unshared errors cause in the trees that different stations see.

The proof that the CCRA recovers automatically from unshared errors does not require that the tree structure be binary, but applies to arbitrary branching ratios in the partitioning. As an example, assume that the network stations use a 3-ary tree structure to resolve collisions. Figure 3 shows a complete 3-ary tree representing a 16 slot CRI as observed by some of the stations on a network. Collisions occurred at slots 1, 3, 5, 8, and 13. All the other slots were empty or contained a successful transmission.

Now suppose that the other stations on the network observed a successful transmission or an empty slot at slot 5. These stations assign a leaf node to the tree to slot 5 as shown in Figure 4. The original CRI is then resolved at slot six. The next ten slots in Figure 4 constitute three more complete CRIs (and three more complete trees.) We have not specified what actually happened in slot 5, nor have we specified how many stations are in either of the two sets. Clearly one set of stations is correct and the other set has erred, but the proof applies for both cases, because it does not depend on which set was correct. This discrepancy could have been caused by the stations in Figure 3 seeing a "leaf to interior" error (i.e., LACK to NACK, or ACK to NACK), or it could have caused by the stations of Figure 4 seeing an "interior to leaf" error (i.e., NACK to ACK, or NACK to LACK).

In all slots except for slot 5, all the stations on the network agree on the feedback results. In Figures 3 and 4, all nodes other than node 5 are the same. For the stations of Figure 3, the 16 time slots constitute a single CRI and the stations of Figure 3 are ready to begin a new CRI with slot 17. The stations in Figure 4 have constructed four complete 3-ary trees corresponding to four complete CRIs. (The third CRI in Figure 4 contained only one time slot, which corresponds the degenerate tree at node 12.) The stations of Figure 4 are also ready to begin another CRI with slot 17. If no more errors occur, all the stations will construct identical trees and recognize the same CRIs after slot 16.

Note that the trees rooted at nodes 8, 12, and 13 in Figure 4 are identical to the subtrees rooted at the corresponding nodes in Figure 3. This observation leads us to construct a proof that stations will automatically resynchronize, after an error, within Q CRIs, where the branching ratio of the CCRA is Q.

Theorem:

Assume a feedback channel errors causes some stations (partition 1) to assign a leaf node to a particular time slot, while other stations (partition 2) assign an interior node to the same time slot. In other words, the stations of partition 1 had an "interior to leaf" error

or the stations of partition 2 had a "leaf to interior" error. Thus the proof applies to both types of errors. When the stations of partition 2 complete the current CRI, the stations of partition 1 will have observed exactly $Q + 1$ CRIs, where Q is the branching ratio of the collision resolution tree.

Proof:

Assume that, since the beginning of the current time slot, a station has observed I (interior) collision time slots, and L (leaf) time slots that contained either zero or one transmission attempts. Let

$$N = (Q - 1)I - L + 1 \tag{4}$$

where Q is the tree branching ratio, I is the number of observed interior nodes, and L is the number of observed leaf nodes.

In section 2 it was proved that $N = 0$ indicates that the CRI and the corresponding tree are complete. More generally, N can be interpreted as the number of subtrees required to complete the current tree (or CRI). For instance, see Figure 5, which shows a partial tree with 5 interior nodes (1, 2, 5, 6, 8) and 3 leaf nodes (3, 4, 7). Equation 6 indicates that the number of missing subtrees in Figure 5 is $N = (2 - 1) \times 5 - 3 + 1 = 3$. The three missing subtrees are labeled a, b, and c in Figure 5. The missing subtrees may be of any depth. Each subtree could represent a single time slot or many slots.

Each new time slot in the CRI adds a new node to the incomplete tree, changing the value of N. If the time slot adds an interior node to the tree, N becomes $N + Q - 1$. If the time slot adds a leaf node to the tree, then N becomes $N - 1$. Thus, if some stations (partition 1) add a leaf node and some stations (partition 2) add an interior node, the value of N will be greater by Q at the stations of partition 2 than at the stations of partition 1.

That difference, Q, is invariant in later time slots if both sets of stations agree on the feedback from subsequent time slots. When N reaches zero for the stations of partition 1, the value of N will be Q at the stations of partition 2. The next time slot begins a new CRI for the stations of partition 1, and it begins a new sub-CRI corresponding to one of the Q missing subtrees for the stations of partition 2. Each new CRI for the stations of partition 1 will constitute one of the missing subtrees for the stations of partition 2. When the CRI for the partition 2 stations is finally complete, the stations of partition 1 will have seen exactly $Q + 1$ complete CRIs.

Thus the all the stations begin a new CRI together after the partition 2 stations see a single CRI while the partition 1 stations see $Q + 1$ CRIs. In the absence of more errors, the stations will then stay synchronized in the sense that they all begin and end CRIs together.

The use of arrival epochs can introduce a subtle complication to the proof for networks operating below saturation. This complication is discussed in Appendix B.

The above theorem, in fact, shows that there exist renewal points in the CCRA. A renewal point is the termination of a CRI at a time t, such that, there exists an arrival time, $t_0 \leq t$, for which all messages that have arrived at the network in $(0, t_0)$ have been

successfully transmitted in $(0, t)$. In Figure 6, four messages have arrived at the network before the time t_0, and they arrive at stations a, b, c and d. At some later time t, which is the end of a CRI, all four messages have been successfully transmitted, making t a renewal point corresponding to t_0. In the absence of errors, all stations agree on the value of t_0, but errors can cause stations to disagree on the value of t_0 that corresponds to a renewal point at t.

When arrival epochs are used to assign messages to CRIs, the errors can cause stations to permanently disagree on the correspondence between arrival epochs and CRIs. Thus, the j-th CRI may include messages from the j-th arrival epoch at some stations, but from other arrival epochs at other stations. This disagreement is caused by the difference (equal to the tree branching ratio) of the number of CRIs observed by the different stations between the occurrence of the the error and the resynchronization.

In the absence of feedback errors, the CCRA causes renewal points to occur at the end of each CRI. For each renewal interval there is a corresponding arrival interval such that all messages that arrive in the arrival interval are successfully transmitted in the corresponding renewal interval.

In this section, we have shown that renewal intervals can exist even in the presence of errors that cause the stations to see different results on the feedback channel. Because the effects of these errors are temporary, the renewal interval is extended to the time when all stations again agree on the beginning of a new CRI. Thus, a renewal interval may span several CRIs. (Stations may disagree on the arrival times that correspond to renewal points, though.) When the stations again agree on the beginning points of CRIs, the CCRA operates as if the error had never occurred, and the throughput limit is unaffected.

5. Performance of CCRA with Unshared Errors : Simulation Study

In this section we present simulation results and discuss the effect of unshared errors on the performance of the CCRA with arrival epoch modification. In our simulation studies, we assume a finite number of stations N on a network, each generating a new packet with the probability q per slot (i.e., geometric arrivals). We assume constant length packets, and the transmission time of a packet is equal to a slot length. In the following, we measure time in slots. Each station has buffer space only for a packet (per arrival epoch), and thus, it can accept maximum one packet per arrival epoch. Packets arriving at a station which already has a packet are lost due to buffer overflow. The length of an arrival epoch is L slots. Each station sees incorrect feedback with probability P_{error} per slot. In other words, with probability P_{error} a station sees either a "leaf to interior" error or an "interior to leaf" error. A "leaf to leaf" error is not considered, since CCRA is robust to this type of error, and the performance of the algorithm is not affected by this type of error. In our simulation studies, multiple stations may make errors in a slot, but we assume each renewal interval contains one or zero error slot (low error frequency assumption). Note that all the erring stations make the same type of error, either "leaf to interior" error or "interior to leaf" error. Refer to section 3 for more detailed description of the model.

The performance measures we investigate are

- the average packet transmission delay D, i.e., the average time between arrival and departure (end of successful transmission) of a packet
- the throughput S, the probability of successful transmission per slot
- the loss probability P_{loss} of packets at a station due to buffer overflow
- the average length T of renewal intervals; the average time to resynchronize, given that an error has happened.

We also investigate the effect of arrival epoch length on the CCRA performance. We obtain the optimal length L_{opt} of an arrival epoch to maximize the throughput.

For a 20 station network ($N = 20$), Figure 7 shows the throughput S as a function of a total arrival rate $q \times N$ (per slot) for various values of arrival epoch length L. In this figure, stations are assumed to make no error ($P_{error} = 0$). From this figure, it can be seen that the $L = 2$ achieves the highest maximum throughput of 0.486 ($L_{opt} = 2$). This value (0.486) is same as the theoretical upper bound (0.487) for the throughput of the CCRA with arrival epoch modification (i.e., so called part-and-try or interval searching algorithm) in an error free environment [12, 13]. This verifies the accuracy of our simulation model. Figure 8 shows the average transmission delay D for the same network in Figure 7. L is equal to $L_{opt} = 2$ and 10 in Figure 8. As shown in Figures 7 and 8, although $L = 2$ achieves the maximum throughput of 0.486, the delay becomes very large as the input rate approaches to the maximum throughput. Thus, it may be concluded that the CCRA should be used in a light or moderate traffic environment.

In Figures 9 and 10, we assume $P_{error} = 0.2$ and $N = 20$. Figure 9 shows the throughput S for various values of L. Horizontal line represents $q \times N$. L_{opt} is 2, and the corresponding maximum throughput is 0.418. Retransmissions due to feedback error causes increase in channel traffic, and thus, the maximum throughput decreases. Figure 10 shows the transmission delay D for the same network with $L = L_{opt} = 2$. As is the case where there is no feedback error, the transmission delay becomes very large as the traffic approaches to the maximum throughput, and it is not practical to operate the CCRA in such a high traffic environment.

When $L \geq 2$, packets may be lost at a station due to buffer overflow. Figure 11 shows the packet loss probability P_{loss} as a function of an total arrival rate $q \times N$ for a 20 station network ($N = 20$) with $L = L_{opt} = 2$, $L = 3$ and $L = 10$. $P_{error} = 0.2$ in Figure 11. For the same network, Figure 12 shows the average renewal interval T. Both Figures show that $L = L_{opt} = 2$ achieves small loss probability P_{loss} and requires short time to resynchronize.

Table 1 shows the maximum throughput S and the optimal arrival epoch length L_{opt} for various values of feedback error probability P_{error}. It can be seen that $L = 2$ achieves the maximum throughput for all cases.

6. Conclusions

The Capetanakis Collision Resolution Algorithm (CCRA) and its variants provide stability and throughput improvements over random back off algorithms such as ALOHA. These algorithms require that all stations on a multiple access network monitor a feedback channel that tells whether each time slot contained a collision of two or more messages, or no collision. The analysis of the algorithms usually assumes that all stations get the same feedback information, even if the feedback is wrong.

Shared feedback information allows the stations to recognize collectively when all the participants in a collision have successfully transmitted their messages. When all stations get the same feedback information, they all recognize, simultaneously, the beginning and end of each collision resolution interval (CRI).

In this paper we show that unshared errors, which cause stations to see different feedback for the same time slot, cause only temporary disagreement among the stations about when CRIs begin and end. No single error can cause a permanent destruction of the simultaneous recognition of CRIs. We also show the effect of unshared errors on the performance of the CCRA through simulations.

References

[1] J. Kurose, M. Schwartz and Y. Yemini, "Multiple-Access Protocols and Time-Constrained Communication", ACM Computing Surveys, Vol.16, No.1, March 1984.

[2] J. I. Capetanakis, "The multiple access broadcast channel: protocol and capacity considerations", Ph.D. Thesis, Course VI, Mass. Inst. Tech., Cambridge, MA, August, 1977.

[3] B. S. Tsybakov and V. A. Mikhailov, "Free Synchronous Packet Access in a Broadcast Channel with Feedback", Prob. Inform. Transmission, Vol.14, No.4, 1978; translated from Problemy Peredachi Informatsii, Vol.14, No.4, 1978.

[4] J. L. Massey, "Collision-Resolution Algorithms and Random-Access Communications", UCLA Technical Report, UCLA-ENG-8016, 1980.

[5] N. D. Vvedenskaya and B. S. Tsybakov, "Random Multiple Access of Packets to a Channel with Errors", Prob. Inform. Transmission, Vol.19, No.2., 1983.

[6] K. Y. K. Ho, R. Rao and J. K. Wolf, "A Robust Collision Resolution Algorithm for the Random Access System with a Noisy Channel", IEEE Infocom 87.

[7] M. D. Ryter, "A Conflict Resolution Algorithm for Noisy Multiaccess Channels", MIT Technical Report, LIDS-TH-1007, June 1980.

[8] P. A. Humblet, "On the throughput of Channel Access Algorithms with Limited Sensing", IEEE Trans. on Communications, Vol.COM-34, No.4, 1986.

[9] I. Cidon and M. Sidi, "Erasures and Noise in Splitting Multiple Access Algorithms", IEEE Globecom 85.

[10] I. Cidon, H. Kodesh and M. Sidi, "Erasure, Capture and Random Power Level Selection in Multiple-Access Systems", IEEE Infocom 87.

[11] J. F. Kurose, A. Shritvastava and D. Towsley, "Stack Algorithms for Random Multiple Access Networks with Asymmetric Noisy Feedback", to appear in IEEE Globecom 87.

[12] R. G. Gallager, "Conflict Resolution in Random Access Broadcast Networks", Proc. the AFSOR Workshop in Commun. Theory and Applications, Sept., 1978.

[13] B. S. Tsybakov and V. A. Mikhailov, "Random Multiple Packet Access: Part-and-Try Algorithm", Prob. Inform. Transmission, Vol.16, No.4, 1980; translated from Problemy Peredachi Informatsii, Vol.16, No.4, 1980.

Appendix A : Proof of Equation (1)

In this appendix, we present proof of equation (1) for a Q-ary tree.

$$(Q-1)I - L + 1 = 0 \tag{1}$$

As the basis of the induction, consider the degenerate tree, with one node. The node is a leaf because it has no children, and there are no interior nodes. For the degenerate tree, $L = 1$, and $I = 0$, satisfying equation (1).

For the second part of the inductive proof, assume that equation (1) holds for all trees with less than N nodes. We show that equation 1 must also hold for all trees with N nodes.

For the general Q-ary tree, the left hand side of equation 1 becomes

$$(Q-1)I - L + 1 = (Q-1)(\sum_j I_j + 1) - \sum_j L_j + 1 \tag{A1}$$

where L_j is the number of leaves in subtree j, and I_j is the number of interior nodes in subtree j. The right hand side of equation A1 includes the root interior node which is not in either subtree. Figure A1 shows the root of a binary tree ($Q = 2$) with N nodes. Each of its children is the root of a subtree with less than N nodes and the subtrees are labeled 1 and 2. Rearranging the right hand side of equation A1 yields

$$\sum_j((Q-1)(I_j - L_j)) + (Q-1) + 1 = \sum_j(-1) + Q = -Q + Q = 0 \tag{A2}$$

using the induction assumption, $(Q-1)I_j - L_j + 1 = 0$.

Appendix B : Resynchronization with Arrival Epochs

When stations use arrival epochs to assign messages to CRIs, it can occur that the j-th arrival epoch terminates before the (j-1)-st CRI terminates. In this case, the j-th CRI must be delayed. The simplest protocol requires that stations must wait for the end of the j-th arrival epoch before they can transmit in the j-th CRI. This waiting causes one or more empty slots, between CRIs, during which no messages can be transmitted. For the purpose of the analysis of unshared errors, and to distinguish them from the normal sequence of CRIs we will refer to each empty slot as a "ghost" CRI of length one.

In the absence of errors, no station will observe any transmissions or collisions during the ghost CRIs. however, errors can cause stations to disagree temporarily on when CRIs terminate. Thus an error can cause a station to observe a collision or transmission in a time slot that should have been a ghost CRI, because some other station or stations may not have completed a normal CRI. The same effect would be caused by a "LACK to NACK" or "LACK to ACK" feedback error that occurs during a ghost CRI. To allow for errors, the CCRA must be extended to behave correctly when transmissions or collisions are observed during ghost CRIs.

It is sufficient to require stations to monitor ghost CRIs in the same way as normal CRIs. A ghost CRI is just like a normal CRI during which that station had no messages to send. If the ghost CRI is observed to begin with a successful transmission, that ghost CRI has length one, just as if that slot had been empty. Only if a ghost CRI begins with an observed collision, will that ghost CRI have length greater than one. In that case, the observing station must remain blocked until the apparent collision is resolved by satisfying Equation 3.

Note that collisions or transmissions can be observed during ghost CRIs only when errors occur, either before or during the ghost CRI. When the observation is made, however, the observing station does not have enough information to determine whether the error occurred at that time slot or previously. Nor can the station determine whether the feedback error occurred at that station or some other station.

Once we require that each station treat ghost CRIs just like normal CRIs for which it has no messages, the proof of section 4 applies even if some of the CRIs in the proof are ghost CRIs instead of normal CRIs.

Figure B1 shows an example in which some stations observe a collision in a slot that should have been a ghost CRI. Those station see a single ghost CRI of five slots, while the other stations see a single ghost CRI, followed by two normal CRIs.

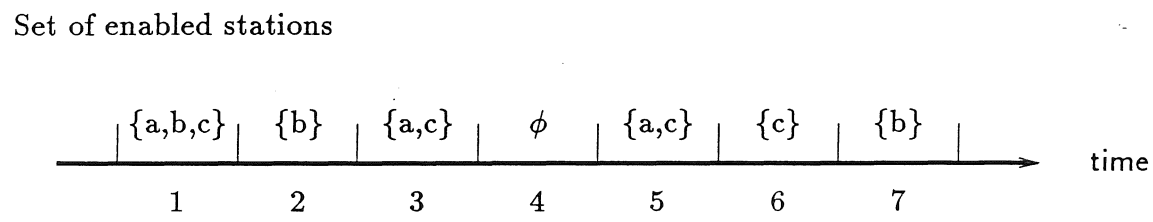

Fig.1 An Example of Collision Resolution Interval (CRI)

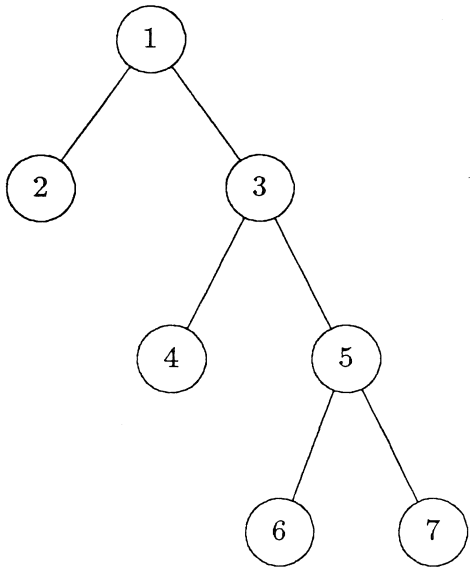

Fig.2 A Binary Tree Representation of the CRI Shown in Fig.1

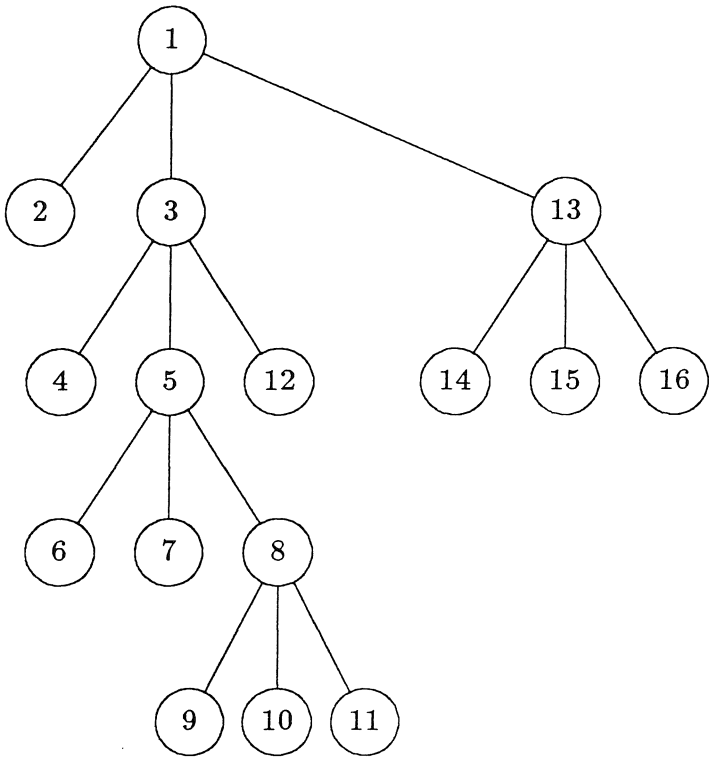

Fig.3 A Tertiary Tree Representation of a Sixteen Slot CRI

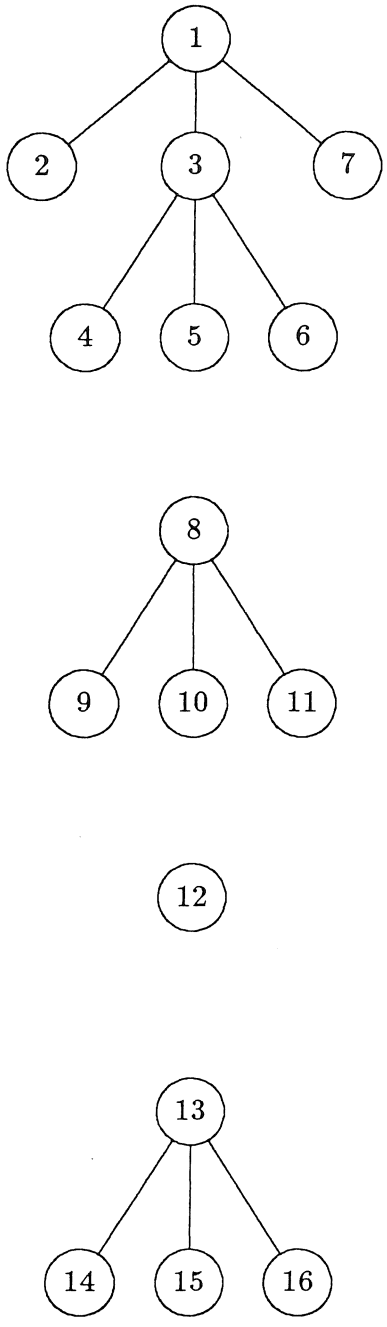

Fig.4 Four Trees, Resulting from a Single Feedback Error in the CRI of Fig.3

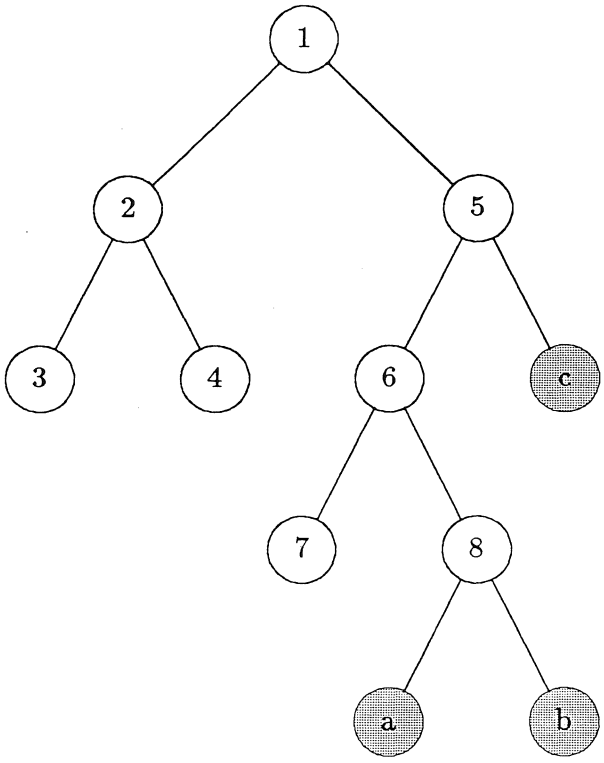

Fig.5 A Partial Tree, Representing an Incomplete CRI

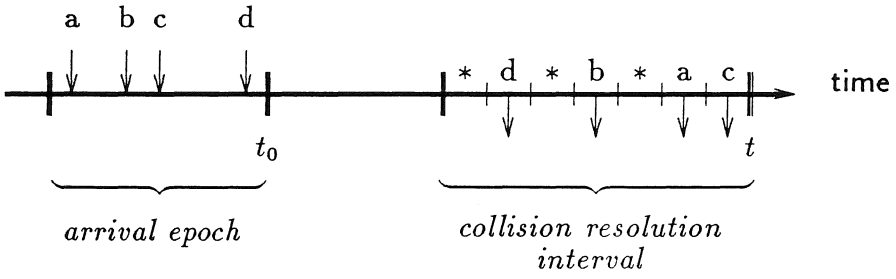

Fig.6 An Arrival Epoch, and its Collision Resolution Interval

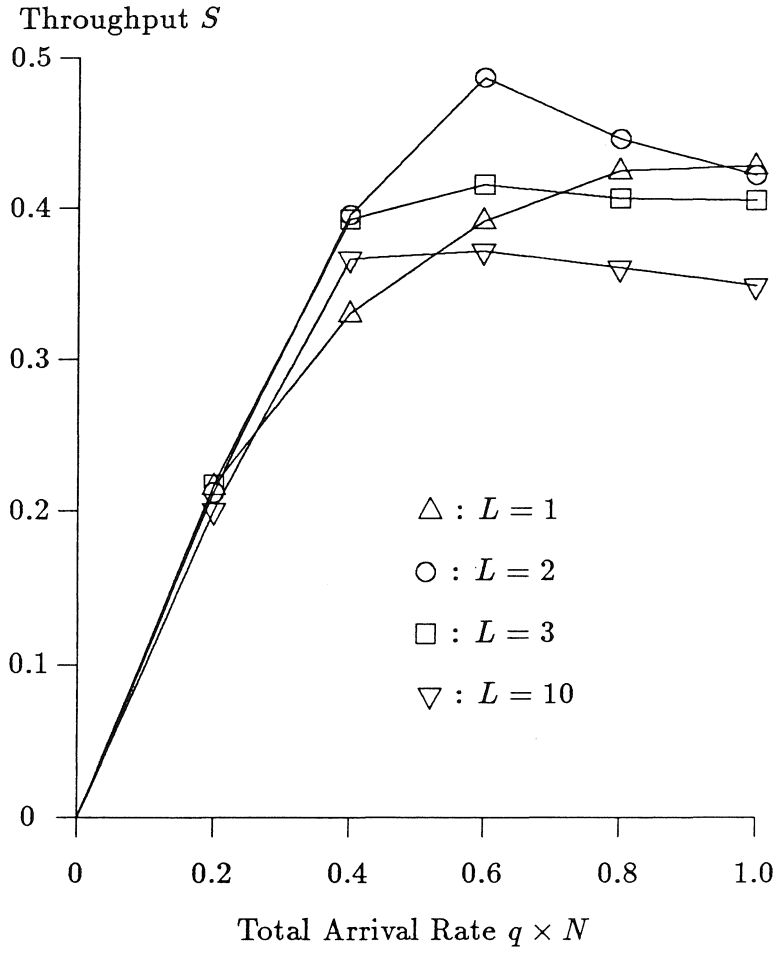

Fig.7 Throughput S ($N = 20$, $P_{error} = 0$)

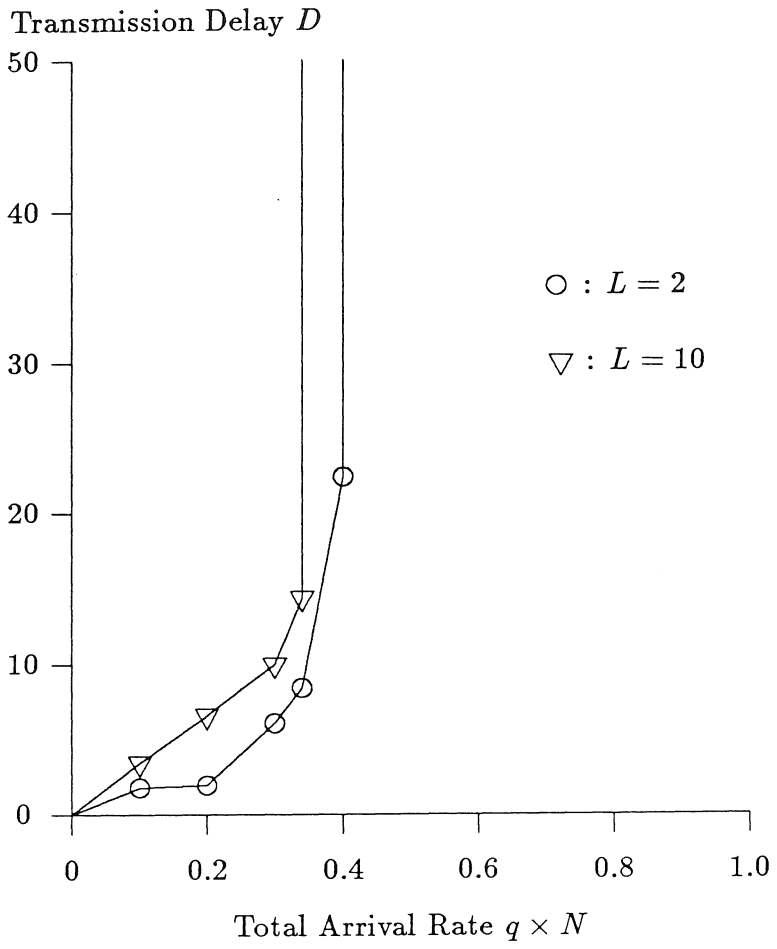

Fig.8 Average Transmission Delay D ($N = 20$, $P_{error} = 0$)

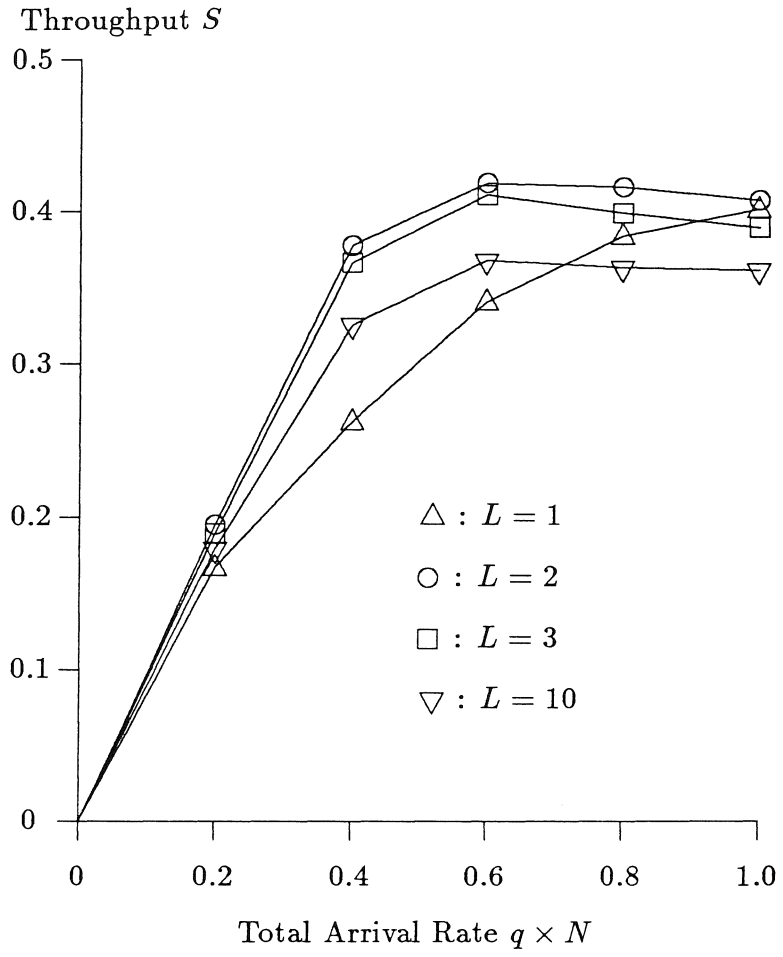

Fig.9 Throughput S ($N = 20$, $P_{error} = 0.2$)

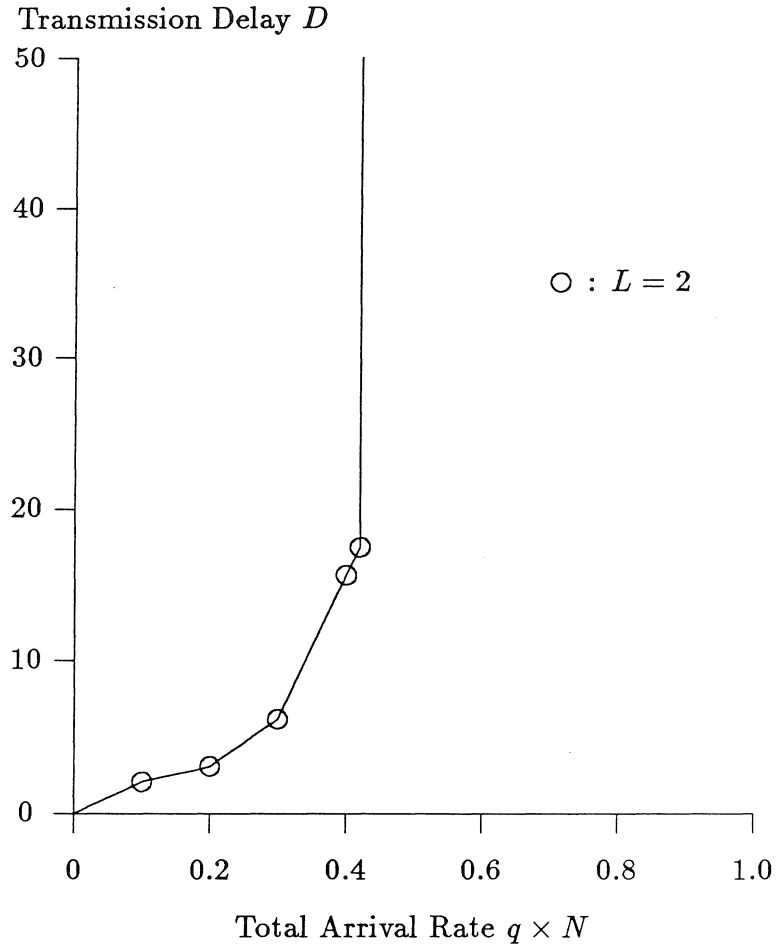

Fig.10 Average Transmission Delay D ($N = 20$, $P_{error} = 0.2$)

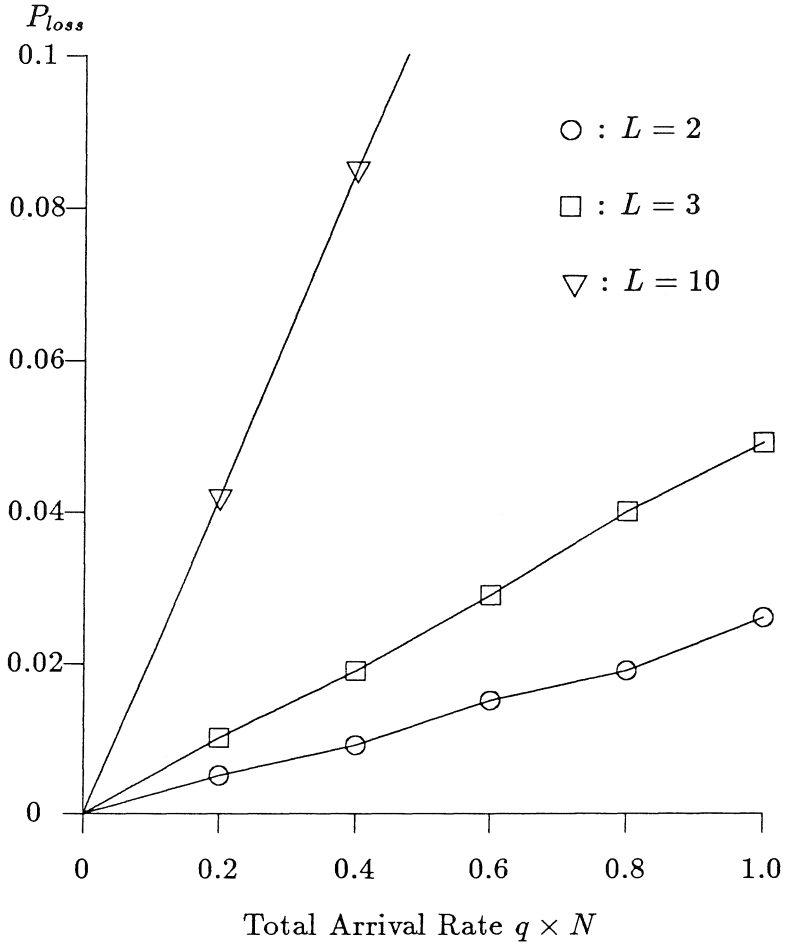

Fig.11 Packet Loss Probablity P_{loss} ($N = 20$, $P_{error} = 0.2$)

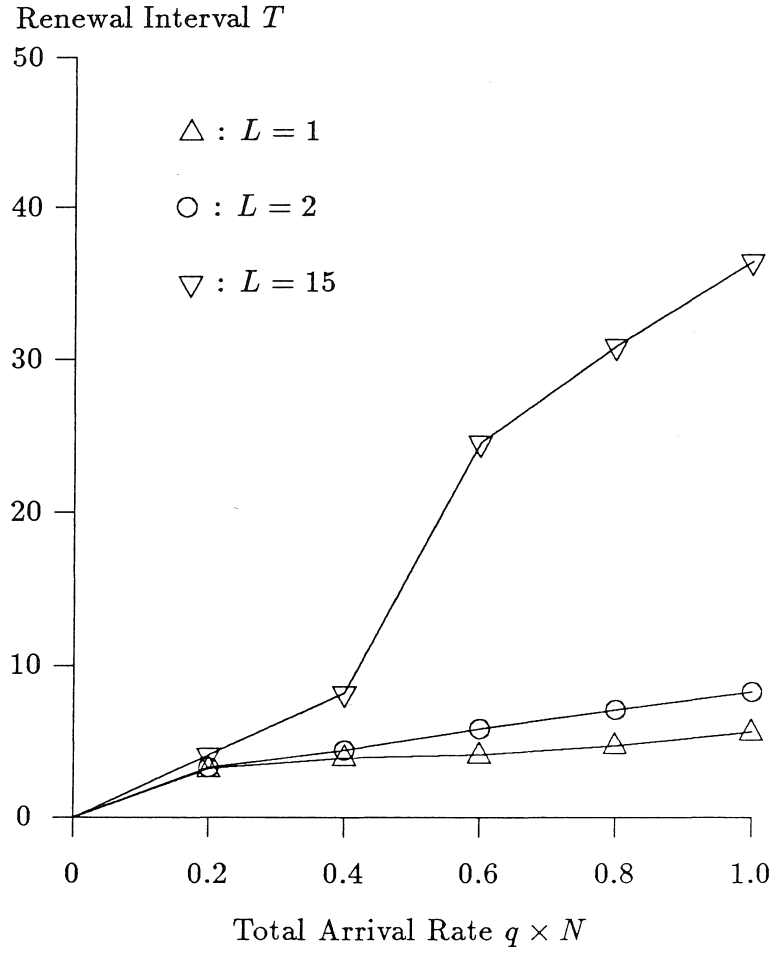

Fig.12 Average Renewal Interval Length T ($N = 20$, $P_{error} = 0.2$)

P_{error}	S	L_{opt}
0	0.486	2
0.1	0.462	2
0.2	0.418	2

Tab.1 Maximum Throughput S and Optimal Arrival Epoch Length L_{opt}

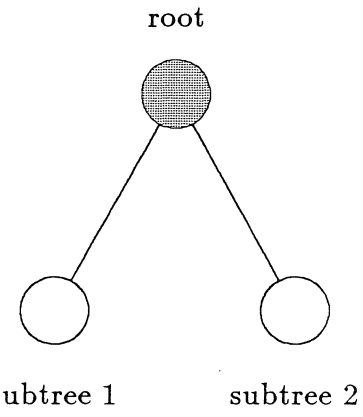

Fig.A1 A Binary Tree, and its Two Subtrees

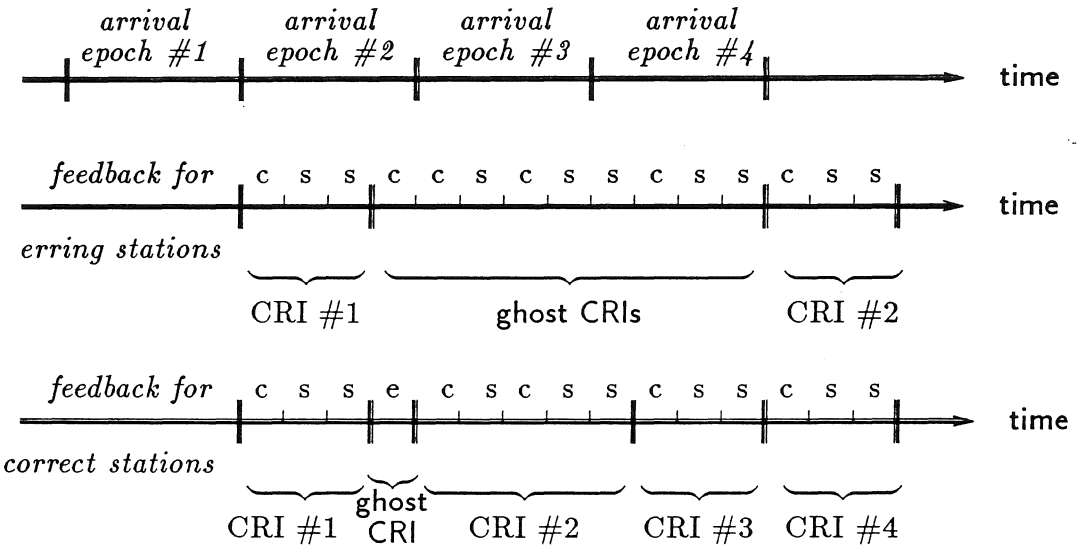

Fig.B1 Error Resolution with "Ghost" CRIs