Controlling Window Protocols for Time-Constrained Communication in a Multiple Access Environment*

James F. Kurose** Mischa Schwartz*** Yechiam Yemini**

Columbia University
New York, New York 10027

CUCS-75-83

ABSTRACT

For many time-constrained communication applications, such as packetized voice, a critical performance measure is the percentage of messages which are transmitted within a given amount of time after their arrival at a sending station. We examine the use of a group random access protocol based on time windows for achieving time-constrained communication in a multiple access environment. First, we formulate a policy for controlling protocol operation in order to minimize the percentage of messages with waiting times greater than some given bound. A semi-Markov decision model is then developed for protocol operation and three of the four optimal control elements of this policy are then determined.

Although the semi-Markov decision model can also be used to obtain performance results, the procedure is too computationally expensive to be of practical use. Thus, an alternate performance model based on a centralized queueing system with impatient customers is developed. Protocol performance under the optimal elements of the control policy shows significant improvements over cases in which the protocol is not controlled in this manner. Simulation results are also presented to corroborate the analytic results.

---

*This work partially supported by the Defense Research Projects Agency Project N00039-82-C-0247, NSF grant ECS-81-08776 and ONR grant N000014-82-K-0508. This paper will appear in *Proceedings of the 8th Data Communications Symposium*, Oct. 1983.

**Department of Computer Science

***Department of Electrical Engineering and Department of Computer Science

1. Introduction

During the past ten years, numerous research efforts have been directed towards developing protocols for efficiently sharing a single multiple access communication channel among a community of distributed users; a survey of these efforts can be found in [Tobagi 80]. A primary performance objective of these protocols has been to minimize average message delay, i.e., the average time between a message's arrival at the sending station and its reception at the destination station.

Recently, there has been considerable interest in supporting *real-time* or *time-constrained* communication applications such as packetized voice [Cohen 77] [Gitman 81] and distributed sensor networks [DSN 82] in a multiple access environment. The communication requirements of time-constrained applications differ significantly from those of traditional multiple access communication in two respects. First, a certain amount of message loss is usually tolerable. Secondly, a message which is not successfully transmitted within a certain amount of time after its arrival at the sending station is considered lost, regardless of whether or not it is eventually received at the destination station. One consequence of these differences is that the primary performance objective is no longer to minimize average message delay but rather to maximize the percentage of messages successfully transmitted within the time constraint. These fundamentally different communication characteristics and performance objectives suggest that previously developed multiple access protocols may not be well suited for time-constrained applications.

In earlier work [Kurose 83] it was noted that in addition to its traditional role as an arbiter of channel sharing, a multiple access protocol also serves as a *distributed scheduling mechanism* by (implicitly or explicitly) imposing a global transmission order on all messages distributed among the stations in the network. This scheduling function was shown to critically affect the distribution of message delays and thus the time-constrained performance of the protocol as well. A random access protocol, based on a generalization of time window protocols [Gallager 78] [Towsley 82] was proposed which provided a family of scheduling disciplines based on message arrival times. The performance of this protocol, which required that *all* messages be sent, was examined for the cases in which messages in the network were sent on a global FCFS, LCFS and RANDOM basis.

In this paper, we relax the constraint that all messages be transmitted, and examine one policy for controlling protocol operation in order to maximize the percentage of messages with delays below a given time constraint. In the following section we review the basic operation of the time window protocol and define the different elements which comprise a protocol control policy. In section 3 a semi-Markov decision model of protocol operation is developed and three of the four optimal elements of the control policy are then determined. A heuristic is presented for the final policy element. Although the semi-Markov decision model can also be used to obtain performance results, the procedure is too computationally expensive to be of practical use. Thus, an alternate performance model based on a centralized queueing system with impatient customers is developed in section 4 and protocol performance is examined.

2. Group Polling and the Time-Window Protocol

Many of the protocols which have been proposed for sharing a multiple access channel belong to the class of group polling [Hayes 78] or tree protocols [Tanenbaum 81]. Their operation is based on the principle of granting transmission rights to a group of stations (the *enabled* stations) in the hope that only a single station in the group is *ready*, i.e., has a message to send. Depending on the manner in which the enabled group is determined, group polling can often allocate the channel more efficiently than standard hub polling [Schwartz 77]. Group polling protocols operate by selecting some criterion, e.g., a range of station addresses [Hayes 78] [Capetanakis 79] or an interval of time [Gallager 78] [Towsley 82], such that all stations satisfying the given criteria (e.g., having an address or a message arrival within specified range) are in the enabled group. The manner in which this criteria is determined and possibly modified in order to isolate a single ready station in the enabled group will be discussed shortly. For time-constrained communication, since the distribution of message delay is of primary importance, it is natural to base protocol operation on group polling techniques that grant transmission rights based on message arrival times. Let us now briefly review the operation of such a protocol.

The protocol operates in a synchronous fashion. All stations continuously monitor the channel and begin by selecting some interval or *window* of time in the past according to some policy. In the following sections, a policy for selecting this initial window will be formulated, but for now let us simply assume that some policy exists, that all stations follow this policy, and thus all stations select the same window. Once a window has been selected, one of three possibilities can then occur. One possibility is that no stations have a message arrival in the selected initial time window. This situation is shown in figure 1a, in which message arrival times at stations throughout the network are shown below the time axis. In this case, no stations are enabled, no messages are transmitted and the channel remains silent. Once the channel remains silent for an amount of time equal to the end-to-end propagation delay of the channel, τ, all stations know that no other stations have arrivals in this time interval. The stations can then determine a new initial time window and repeat the channel access procedure.

The second possibility is that exactly one station has a message arrival in the initial window. In this case, this station begins transmitting its message without interference. The above process is then repeated upon termination of the message transmission. Finally, if more than one station has an arrival in the initial window, then two or more stations transmit messages (figure 1b) and a collision occurs. If there is a collision, then it is detected by all stations within τ time units. Stations then continue to monitor the channel and attempt to resolve this collision by splitting the initial window in half and selecting one of the two halves (figure 1c). Since all stations follow the same policy, they select the same half of the split window to be the new time window and the group polling procedure is then repeated. If no station has an arrival in the selected half of a split window, then the channel remains idle and all stations then select the other half of the split window, immediately split this window (since it is known to contain two or more message arrivals), choose one of the halves of the newly split window and repeat the above procedure using this half of the window. This splitting process continues until a single message is finally transmitted.

Thus, the result of each windowing process, i.e., the selection of an initial window and its subsequent splitting (if any), always eventually results in either the start of a single message transmission or, if the initial window contains no message arrivals, the start of a new windowing process. Note that each step (choosing or splitting a window and transmitting any arrivals in the window) in the process takes r units of time, the time required for all stations to determine whether a single message transmission, no transmission or a message collision has occurred.

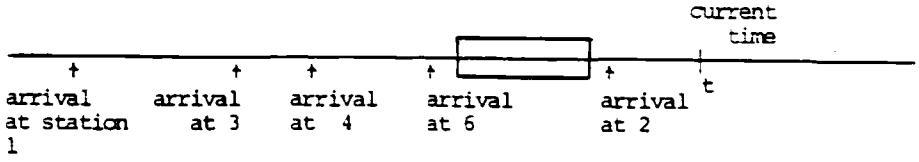


fig. 1a: the initial window contains no arrivals

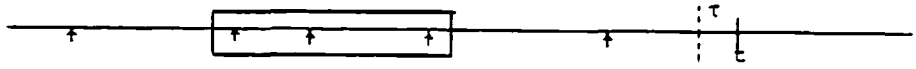


fig 1b: a new window is chosen and collisions occur

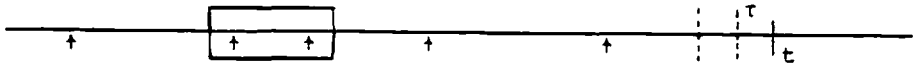


fig. 1c: window is split in half and another collision occurs

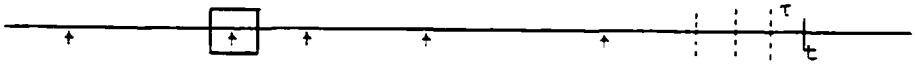


fig. 1d: window is split again and station 3 begins successful transmission of its message

Figure 1: Operation of the time window protocol

All stations perform this windowing process and thus all stations know the width and starting time of every interval of time in the past which is known to contain either no message arrivals or arrivals that have aready been successfully transmitted. Since these intervals are known to contain no untransmitted message arrivals, the stations can remove these intervals of time from further consideration. Thus, the stations might view the time axis as shown below in figure 2. The shaded regions in figure 2 indicate those intervals of time which are known to contain no arrivals of untransmitted messages.

Note that the opportunity for controlling protocol operation arises only at those times when an initial window must be selected. Assuming the protocol maintains some state information (such as a record of its past history as in the time axis in figure 2), then a selection of

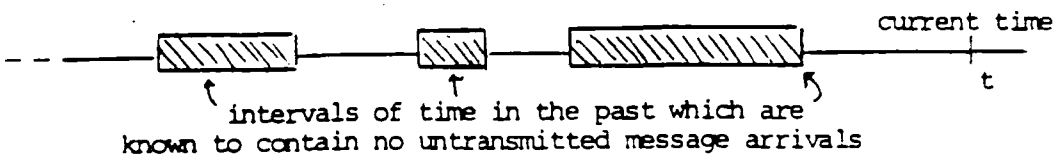


Figure 2: A station's view of the time axis

- the position of the initial window

- the length of the initial window

- a procedure for splitting the window should collisions occur

must be made, based on the current state, from some set of alternatives. Once an alternative for operation has been selected, the probabilistic evolution of the protocol is determined until the next time an initial window must be chosen. The selection of an alternative for action (i.e., specifying (1) through (3) above) given the current state is known as a *decision* and the set of all decisions is known as a *policy*.

The problem we now want to address is how to select those alternatives for operation that will maximize the percentage of messages with delays below the given time constraint. A message's delay (or waiting time) will be defined as the amount of time between its arrival at the sending station and the beginning of the windowing process immediately preceding and resulting in, its own successful transmission. Thus, a message's delay does not (by definition) include the time required for the single windowing process resulting in its transmission. However, the delay *does* include the windowing time for the other message transmissions since its arrival; this point will be further discussed in section 4.

3. Controlling the Time Window Protocol

In this section we address the problem of selecting policy elements (1) - (3) in order to maximize the percentage of messages with a delay below a given bound. First, we develop a state space representation suitable to encode the necessary past history of the protocol. This state space description is then used as a basis for a semi-Markov decision model [Howard 71] of protocol operation. The usual approach for finding the optimal policy is to choose some initial policy and then to iteratively obtain better and better policies. Unfortunately this iterative process can be computationally quite expensive and moreover may provide little insight into the operation of the protocol itself. Our approach here will be to use our understanding of the protocol to infer elements (1) and (3) of the optimal policy and then to prove that no policy iteration would yield a better policy. A simple closed form characterization of (2) does not appear possible; this problem will be further discussed in section 4.

3.1 A State Space Description and Pseudo Time

Let us assume that time is discrete in units of Δ (where Δ can be arbitrarily small but finite) and is small enough so that the probability of more than one message arrival (anywhere in the network) can be assumed to be zero. One straightforward state space approach is simply to encode for each Δ unit of time in the past, whether or not it may still contain an untransmitted message arrival. However, this approach leads to a complicated state space which also grows exponentially in size with each Δ.

An alternative approach is to introduce the notion of *pseudo time*, determine the optimal policy elements (1) and (3) within a state space based on pseudo time and then to relate these results back to actual time. The relationship between actual time and pseudo time is shown below in figure 3. Pseudo time is defined such that each unit of pseudo time in the past is associated with a unit of actual time in the past which may still contain an untransmitted message arrival. With no loss of generality, we can require that if t_1 precedes t_2 in actual time then the pseudo time associated with t_1 precedes the pseudo time associated with t_2. In this case, the introduction of pseudo time essentially compresses the actual time axis by removing those intervals of time known to contain no untransmitted message arrivals.

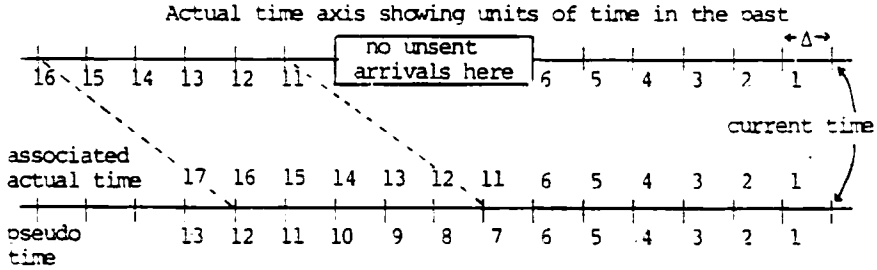


Figure 3: Actual time and pseudo time

Each state in pseudo time thus simply corresponds to the total amount of time which may still contain message arrivals. One possible pseudo time state space description is thus:

$$S = \{ 0, 1, 2, 3 \ldots \} \tag{3.1}$$

where a particular state, i, indicates that i units of time may still contain untransmitted message arrivals.

This state space description can be further refined as follows. Let K (in units of Δ) represent the imposed time constraint. Clearly, transmitting a message with a delay greater than K (in actual time) is useless work since the message will be lost at the receiver with probability 1. Thus the protocol should never send a message with a delay greater than K. When an initial window is

chosen, all messages with a delay greater than K can therefore be discarded *by the sending station*. Thus, in addition to policy elements (1), (2) and (3) of section 2, the optimal policy will also:

4. discard any messages with a delay greater than K. These messages can be effectively discarded by marking the actual time intervals containing possible arrivals which would have a delay greater than K as if the intervals were known to contain no untransmitted arrivals.

As a result of policy element (4), there are never more than K units of actual time (and hence pseudo time) possibly containing untransmitted message arrivals. Thus the pseudo time state space can be further refined to:

$$S = \{ 0, 1, 2, \ldots, K-1, K\} \tag{3.2}$$

3.2 Optimal Elements of the Window Control Policy

In this section we establish the following theorem which states that the window control policy elements (1) and (3) which maximize the percentage of messages with delays less than K, result in successfully transmitted messages being sent on a global (network-wide) FCFS basis:

Theorem 1: In the case that all message lengths are identically distributed and given policy element (4), the optimal selection of policy elements (1) and (3) is independent of policy element (2) and can be characterized as follows:

1. The beginning of an initial window should be placed at the point in time closest to, but not exceeding, K units of time in the past (where K is the imposed time constraint) which may contain untransmitted messages.

3. the older half of a split window is always selected first.

Note that a message is lost unless the windowing process which results in its successful transmission begins within K units of time after its arrival. Intuitively, since all the message lengths are identically distributed, it would seem reasonable to transmit that message with a delay closest to, but not exceeding, the time bound; this is exactly what (1) and (3) above specify. A similar scheduling policy, minimum slack time scheduling [Coffman 76], can be easily proven optimal in the deterministic case that all arrival times are known in advance. The present case is somewhat more complicated since only probabilistic information concerning arrival times is known and since any decision influences the future evolution of the system.

In establishing Theorem 1, the following definitions will be useful:

- actual delay of a message. The amount of time between the current time and the arrival time of the message.

- pseudo delay of a message. The amount of time between the current time and the pseudo time associated with the actual arrival time of a message. Note that while a message's

actual delay always increases with time, the pseudo delay may both increase and decrease in time, depending on the behavior of the window mechanism.

- actual loss: fraction of messages not successfully transmitted with an actual delay less than K. Since the loss will be a function of the policy, we will write actual loss(P) to indicate the actual loss under policy P.

- pseudo loss: fraction of messages not successfully transmitted with a *pseudo delay* less than K. Pseudo loss(P) indicates the pseudo loss under policy P.

- one-step pseudo loss: the expected number of messages which have a pseudo delay less than K when a decision is made, but have a pseudo delay greater than K (and hence be lost under policy element 4) when the next decision is made.

In order to establish Theorem 1, we will first establish the following lemmas, which will require the following assumption:

Assumption 1 : The distribution of arrival times measured in actual time and in pseudo time is the same, i.e., the removal of intervals of time and the concommitant shifting of actual arrival times (to get the pseudo arrival times) preserves the distribution of interarrival times. For the case of Poisson arrivals, the removal of an initial window containing 0 or 1 message arrivals preserves this property exactly. In general, however, if one half of a split window is removed, then the interarrival time distribution may not be exactly preserved.

Lemma 1 : For any policy P: actual loss(P) $\geq$ pseudo loss(P)

Proof: By the definition of pseudo time, the pseudo delay of a message is always less than or equal to the actual delay of a message. Thus if a message's pseudo delay exceeds K, then its actual delay also exceeds K.

Lemma 2 : Given policy element (4), any policy with policy elements (1) and (3) as in Theorem 1 preserves the following property of all messages which are not lost:

pseudo delay of a message = actual delay of a message

Proof: Lemma 2 can be inductively established. Suppose the above property holds when a decision is made. If this property does not hold when the next decision is made, then the selection of the first window must have been such that there was a message which arrived between K units of time in the past and the start of the first initial window. However, by hypothesis, the policy contains elements (1) and (3) as in Theorem 1 and thus no such message can exist. Thus the above property is preserved by each decision.

Lemma 3 : Let $\{P^w\}$ be the set of all policies which choose the same *size* window when in the same state (i.e., the set of all policies with the same second element). Let $P^w_{ms} \in \{P^w\}$ be the single policy with elements (1) and (3) as in Theorem 1. Then P^w_{ms} minimizes the one-step pseudo loss over all policies in $\{P^w\}$.

Proof: Let us define the *critical messages* associated with a decision as follows. When a decision is made at time t' in state i, a probabilistic amount of time, σ, is required for the windowing process and message transmission (if any). The critical messages associated with the decision made at t' are those messages with a pseudo arrival time after $t'+i$ but before $t'+\sigma-K$. That is, a critical message is one with a pseudo delay less than i at t' which would have a pseudo delay greater than K at $t'+\sigma$ (when the next decision is made) if it is not transmitted. The one-step pseudo loss can thus be expressed by the following expected value:

one-step pseudo loss =
E [number of critical messages · prob. that a critical message is transmitted] (3.3)

By assumption 1, all units of pseudo time are statistically identical with respect to message arrival times. Thus, for a given window size, the time between two consecutive decisions is independent of both the *position* of the first initial window and the procedure for choosing halves of a split window. Thus, for a given window size, the number of critical messages associated with a decision is independent of policy elements (1) and (3). Now, since P^w_{ms} always selects the message with a pseudo delay closest to, but not exceeding K (i.e., the message that will be critical if *any* messages are critical), P^w_{ms} maximizes the second term in equation 3.3. Thus P^w_{ms} minimizes the one-step pseudo loss.

Lemma 4 : Given policy element 4 in section 3.1, a policy which minimizes the one-step pseudo loss also minimizes the pseudo loss.

Lemma 4 relates the short term pseudo loss to the long term average pseudo loss and relies on results from decision theory. The proof of lemma 4 can be found in appendix A. Given lemmas 1 through 4 we can now establish theorem 1:

Proof of Theorem 1: Let $\{P^w\}$ be the set of all policies which have the same second policy element and let $P^w_{ms} \in \{P^w\}$ be the policy with elements (1) and (3) as in theorem 1. We want to show that no policy in $\{P^w\}$ has an actual loss less than P^w_{ms} . Suppose there exists some policy, P^w_α which has an actual loss less than that of policy P^w_{ms} :

$$\text{actual loss}(P^w_\alpha) < \text{actual loss}(P^w_{ms})$$

By lemma 2, we then have

$$\text{actual loss}(P^w_\alpha) < \text{actual loss}(P^w_{ms}) = \text{pseudo loss}(P^w_{ms})$$

and then by lemmas 3 and 4:

$$\text{actual loss}(P^w_\alpha) < \text{pseudo loss}(P^w_{ms}) = \min_{P' \in \{P^w\}} \text{pseudo loss}(P')$$

and thus specifically since $P^w_\alpha \in \{P^w\}$:

$$\text{actual loss}(P^w_\alpha) < \text{pseudo loss}(P^w_\alpha)$$

which contradicts lemma 1. Thus P^w_α cannot exist and thus P^w_{ms} is the optimal policy.

An important consequence of theorem 1 (stated in lemma 2) is that there is no difference

between pseudo time and actual time. Thus, there are no "gaps" in time (the shaded regions in figure 2) between those intervals of actual time still possibly containing untransmitted messages. Thus the state space need not be a large and complicated encoding of all the intervals of time known to contain no untransmitted message arrivals. Rather, only a single piece of information need be maintained by the protocol - that point in time closest to, but not exceeding K units of time in the past which may contain untransmitted messages. This value will be known as *t_past*. Under the optimal protocol, it is further known that *all* time between t_past and the current time may contain message arrivals. Figure 4 provides an example of the operation of the optimal protocol and the manner in which t_past is maintained.

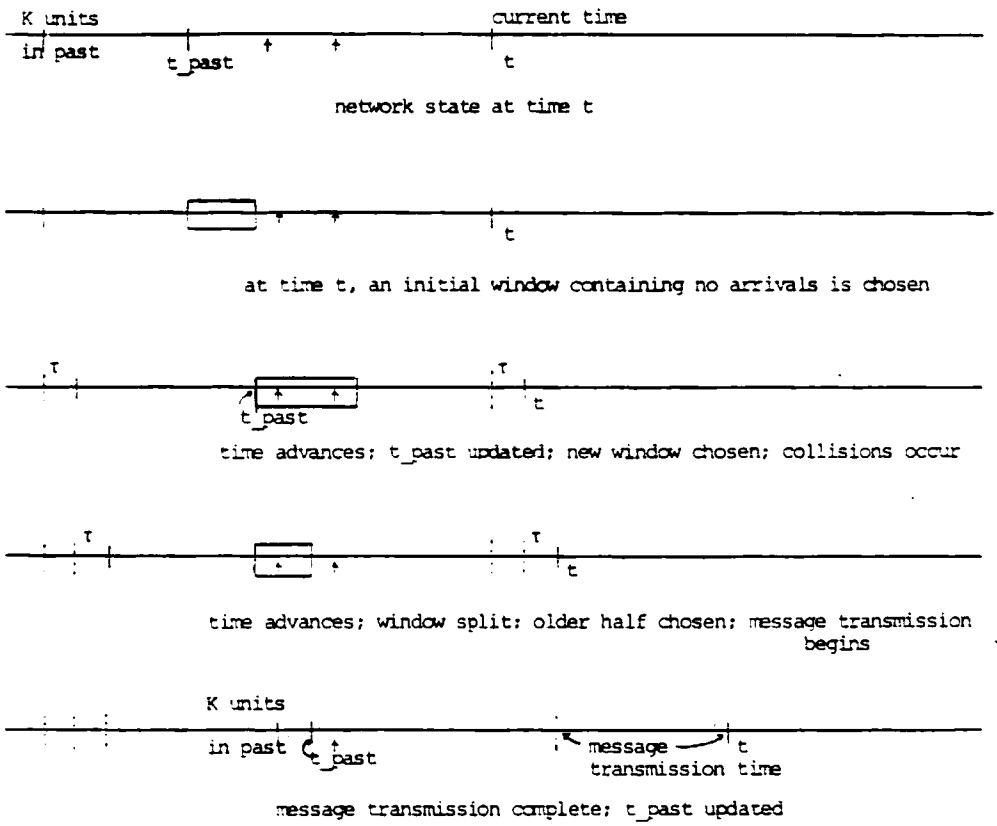


Figure 4: The Controlled Window Protocol

4. A Queueing Model of Protocol Performance

The analysis developed in this section is based on viewing the messages distributed among the stations throughout the network as customers in a *distributed* queue. If we view channel activity as a function of time, the window protocol results in alternate use of the channel for the windowing process (to determine the message to be transmitted) and for the successful

transmission of that message. Suppose we consider the time required by the windowing processes immediately preceding, and resulting in, a message's transmission (i.e., its scheduling time) to be part of its *service* time. The scheduling time component of a message's service time is thus the time between either the end of the most recent message transmission or its own arrival time (whichever is more recent) and the start of its own successful transmission. In this case the distributed queue is equivalent to a centralized queue in which a message's service time consists of two components: the scheduling time component and the actual transmission time component.

4.1 An M/G/1 Queue With Customer Loss

A consequence of policy elements (1), (3) and (4) is that all successfully transmitted messages are sent on a FCFS basis. Furthermore, as a result of policy element (4), messages are lost at the sender (i.e., never sent) only when their waiting time (as defined in section 2) exceeds the given time constraint. Thus, the operation of the optimal policy can be modeled as a FCFS queue in which messages at the front of the queue are lost (denied service) if their waiting time in the queue has exceeded the time constraint; this model is shown below in figure 5a

Fig. 5-1a: Customers are denied service if wait in queue > K.

Fig. 5-1b: Customers determine waiting time and do not join the queue if wait time > K.

Figure 5: Two Models of a Queue With Customer Loss

In terms of server utilization, it makes no difference whether customers are lost upon arrival at the front of the queue or whether they somehow determine their waiting time and join the queue if and only if their waiting time is less than the specified time constraint. Thus, the probability that the server is busy is the same in the model in figure 5a as in the model in figure 5b. We will model the distributed queue using the M/G/1 queueing system shown in figure 5b. It should be noted that this model is only approximate since the "service" times are not truly independent. This second queueing model was recently studied in [Baccelli 81] for the M/G/1 case for the waiting time distribution of customers entering service. If we are only interested in the probability of message loss, an alternative approach can considerably simplify the analysis for the M/G/1 case.

Let us define F(w,t) as the PDF of the unfinished work in the queue at time t:

$$F(w,t) \triangleq P(\text{ unfinished work} < w)$$

Note that the unfinished work in the queue corresponds to the waiting time an arriving customer

would experience under FCFS scheduling. By elementary continuity arguments, the PDF of the unfinished work at time $t+\Delta t$ can be characterized in terms of its value at time t as follows:

$$F(w,t+\Delta t) = (1-\lambda\Delta t)F(w+\Delta t,t) + \lambda\Delta t(1-\mu_1(w-K))\int_0^w B(w-x)\frac{\partial F(x,t)}{\partial x}dx + \lambda\Delta t(\mu_1(w-K))\{ c_1\int_0^w B(w-x)\frac{\partial F(x,t)}{\partial x}dx + (1-c_1)F(w+\Delta t,t)\} \quad (4.1)$$

where K is the time constraint, B(x) is the service time PDF of a customer, $\mu_1(w-K)$ is the unit step function at K and $c_1 = F(K,t)$.

The first term on the right hand side of equation 4.1 relates the value of $F(w,t+\Delta t)$ to the value of the PDF at time t in the case of no message arrivals in Δt. The second term is for the case of one message arrival and unfinished work at time $t+\Delta t$ less than or equal to K. The final term is for the case of one message arrival and unfinished work at time $t+\Delta t$ greater than K; the values of c_1 and $1-c_1$ represent, respectively, the probability that an arriving customer found its waiting time to be less than or greater than K. Rearranging the terms in equation 4.1, taking the limit as Δt approaches 0 and then (assuming equilibrium exists) taking the limit as t approaches infinity, results in the following pair of integro-differential equations for the distribution of unfinished work in the queue:

$$0 = \frac{dF(w)}{dw} - \lambda F(w) + \lambda\int_0^w B(w-x)d_xF(x) \qquad (0<w\leq K)\ (4.2)a$$

$$0 = \frac{dF(w)}{dw} - \lambda c_1F(w) + \lambda c_1\int_0^w B(w-x)d_xF(x) \qquad (K < w)\ (4.2)b$$

Let $f_{w<K}(w)$ be the derivative of the solution to equation 4.2 (i.e., the waiting time density (pdf))in the region $0<w\leq K$ and let $f_{K<w}(w)$ be the waiting time density function in the region $K<w$. Then by conservation of probability, we have:

$$\int_0^K f_{w<K}(w)dw + \int_K^\infty f_{K<w}(w)dw = 1 \quad (4.3)$$

At this point, we could solve (4.2) for $f_{w<K}(w)$ and $f_{K<w}(w)$ using equation 4.3 to determine the unknown constants. The solution to equation 4.2a can be shown [Kleinrock 75] to be:

$$f_{w<K}(w) = P(0)\sum_{i=0}^{\infty}\rho^i\beta^i(w) \quad (4.4)$$

where:

- P(0) is the probability that the server is idle
- $\rho = \lambda\bar{x}$, where λ and $\bar{x}$ are, respectively, the arrival rate and average service times of customers.
- $\beta(w)$ has the form of the residual service time distribution of an M/G/1 queue with no loss, i.e., the distribution of the remaining work that an arriving customer would find for a customer in service.

- $\beta^i(w)$ is the i-fold convolution of $\beta(w)$.

The solution of equation 4.2b is considerably more complex; fortunately, it need not be solved. Note that the second term on the left of equation 4.3 is simply the probability that an arriving customer finds a waiting time greater than K and thus does not join the queue and is lost:

$$\int_K^\infty f_{K<w}(w)dw \quad = \quad p(\text{loss}) \quad = \quad 1 - p(\text{accepted}) \tag{4.5}$$

The loss probability can be related to P(0), the probability that the server is idle, using the simple conservation of flow argument shown in figure 6. The average rate of customers joining the queue is given by $\lambda p(\text{accept})$ and the average rate of customers leaving the queue after service is given by the probability that the server is busy times the rate at which the server services these customers. By conservation of flow, the average arrival rate and departure rate must be the same and thus:

$$p(\text{accept})\rho \quad = \quad 1 - P(0) \tag{4.6}$$

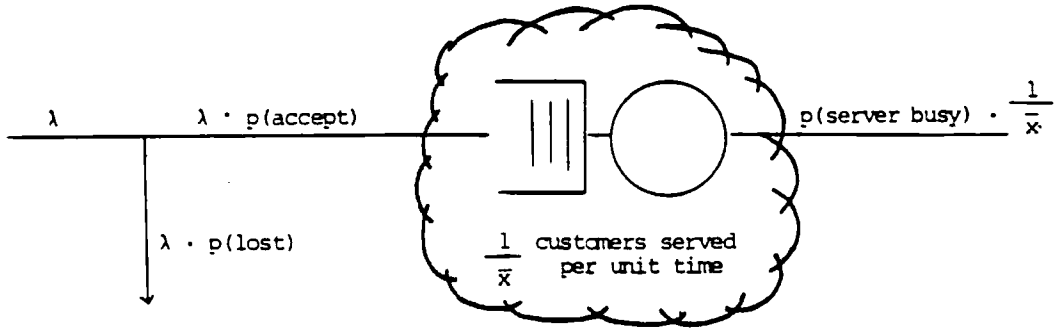


Figure 6: Flow Conservation

Using equations 4.3 - 4.6 we can now determine the probability of message loss:

$$p(\text{loss}) \quad = \quad 1 \quad - \quad \rho^{-1} \quad + \quad \frac{1}{\rho + \rho^2 z(K,\rho)} \tag{4.7}$$

$$\text{where} \quad z(K,\rho) \quad = \quad \sum_{i=0}^{\infty} \rho^i \int_0^K \beta^i(w)dw$$

As a check of equation 4.7, note that in the limit that K approaches infinity, p(loss) approaches 0. As K approaches 0, p(loss) approaches 1-P(0). That is, the probability that a customer is lost approaches the probability that the server is busy, as would be expected since as K approaches 0, a customer would only enter service if the queue was empty.

Equation 4.7 thus gives the probability of message loss under optimal policy elements (1), (3) and (4). Note that we have not yet specified policy element (2) which determines the initial window length. The values chosen for (2) will affect both $\bar{x}$ (and thus ρ) and $\beta(w)$ in equation

4.7. Unfortunately, computing the optimal value of policy element (2) would require solving the set of equation (A1) in the appendix and thus is computationally too expensive to be of practical use; similar problems have been encountered in other protocol models based on semi-Markov decision analysis [Lam 75]. Thus, rather than compute the optimal values for (2), let us examine the performance of the protocol using a heuristic rule for (2).

Specifically, let us assume that (2) is chosen to minimize the *average* amount of time required by the windowing process to schedule a message. Even if the window sizes are selected in this manner, determining the first moment and distribution of the message scheduling time is not an easy task. In [Kurose 83] these values were approximated by exactly determining the average scheduling time for two arrival rates and fitting a function to these endpoints to approximate the average scheduling time for intermediate arrival rates. This average scheduling time was then used as the mean of a geometrically distributed service time. The performance results obtained using these analytic approximations were shown to coincide closely with simulation results.

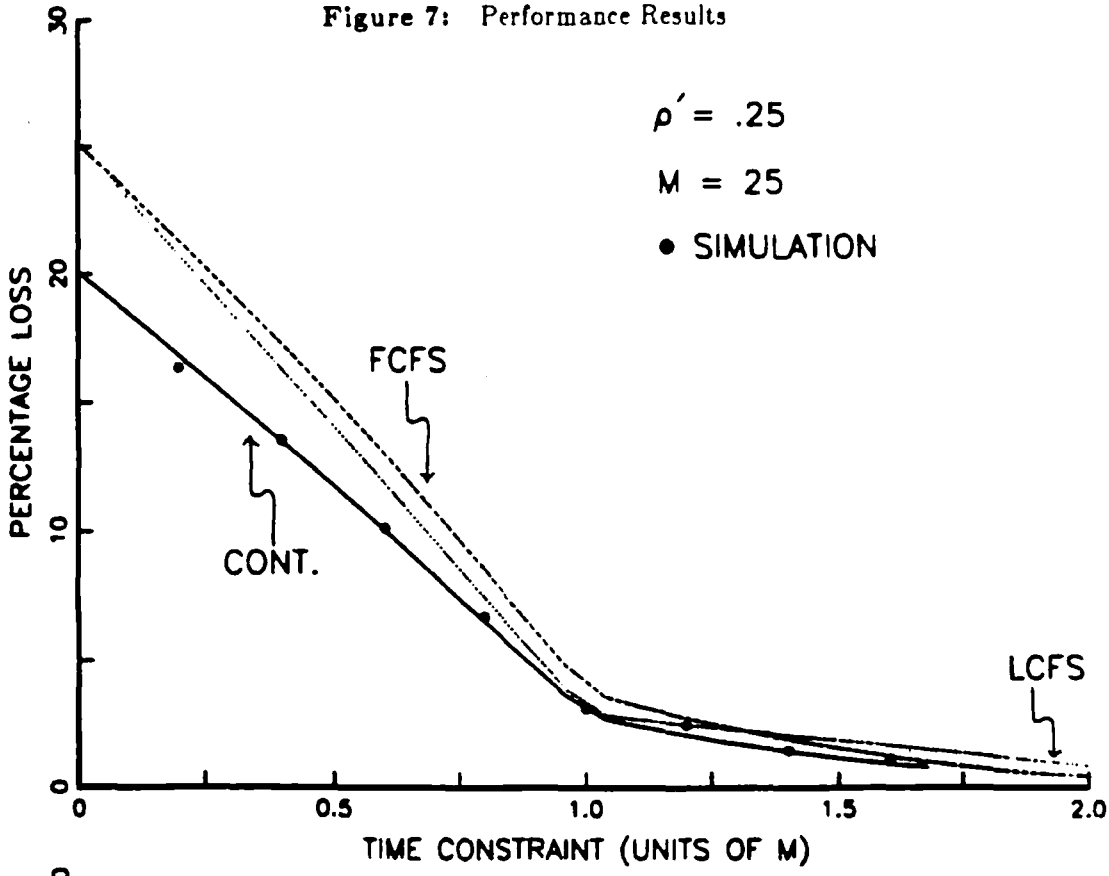
A final complication in evaluating equation 4.7 is that the average scheduling time (and thus message service time) depends on the fraction of messages actually entering the queue and eventually receiving service, i.e., the scheduling time components of $\bar{x}$ and $\beta(w)$ are dependent on p(loss). However, since the scheduling delay is known to be exactly 0 for the case that K=0, $\bar{x}$, $\beta(w)$ and p(loss) can be computed exactly at K=0. The values of $\bar{x}$ and $\beta(w)$ at K=ϵ (ϵ close to 0) can then be closely approximated using the exact fraction of messages entering service for K=0; these values can then be used to compute the loss at K=ϵ. In this fashion the loss at the nth value of K can be iteratively computed using the loss at the (n-1)st value of K to compute $\bar{x}$ and $\beta(w)$.

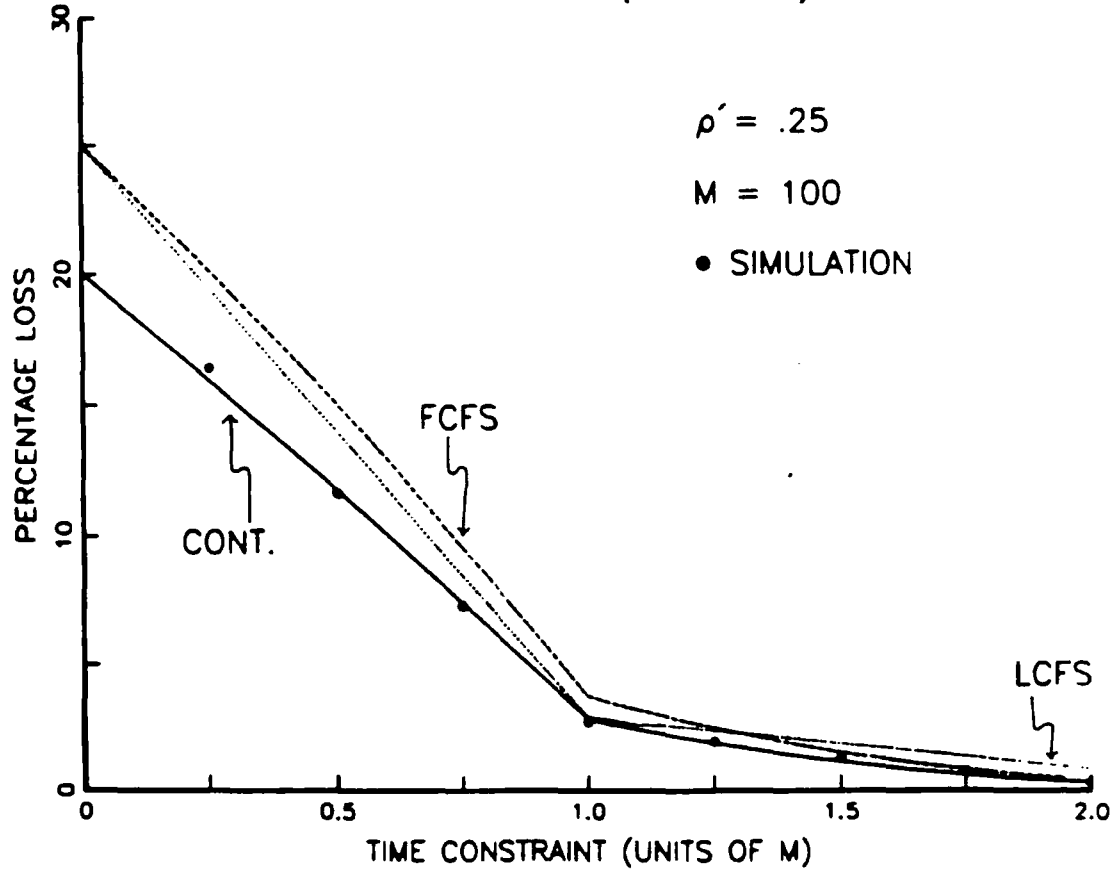
4.2 Some Numerical Results

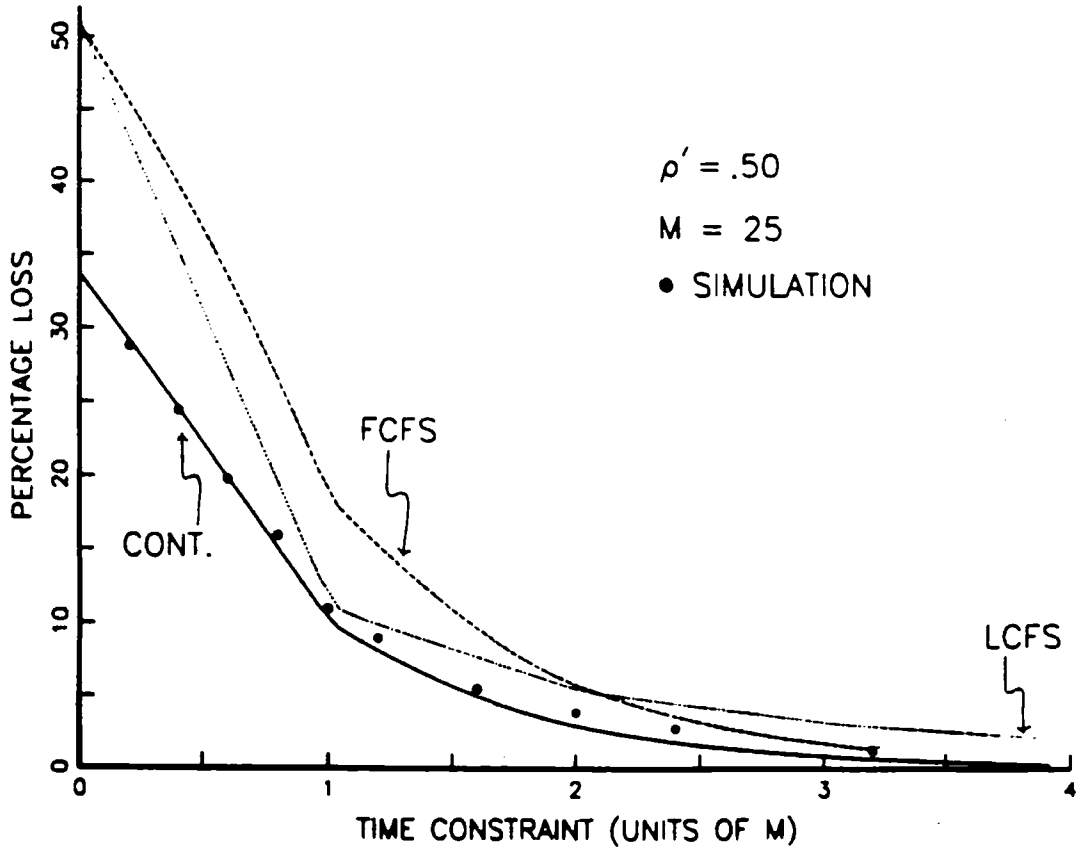
In figures 7 we present some numerical results for the performance of the controlled protocol and compare these results with those from [Kurose 83] in which the protocol provided FCFS and LCFS service and messages were lost only at the receiving stations. Performance results are given for various values of M and ρ', where M is the fixed message length in units of the end-to-end propagation delay of the channel, r, and ρ' is the arrival rate of all messages (lost and transmitted) times

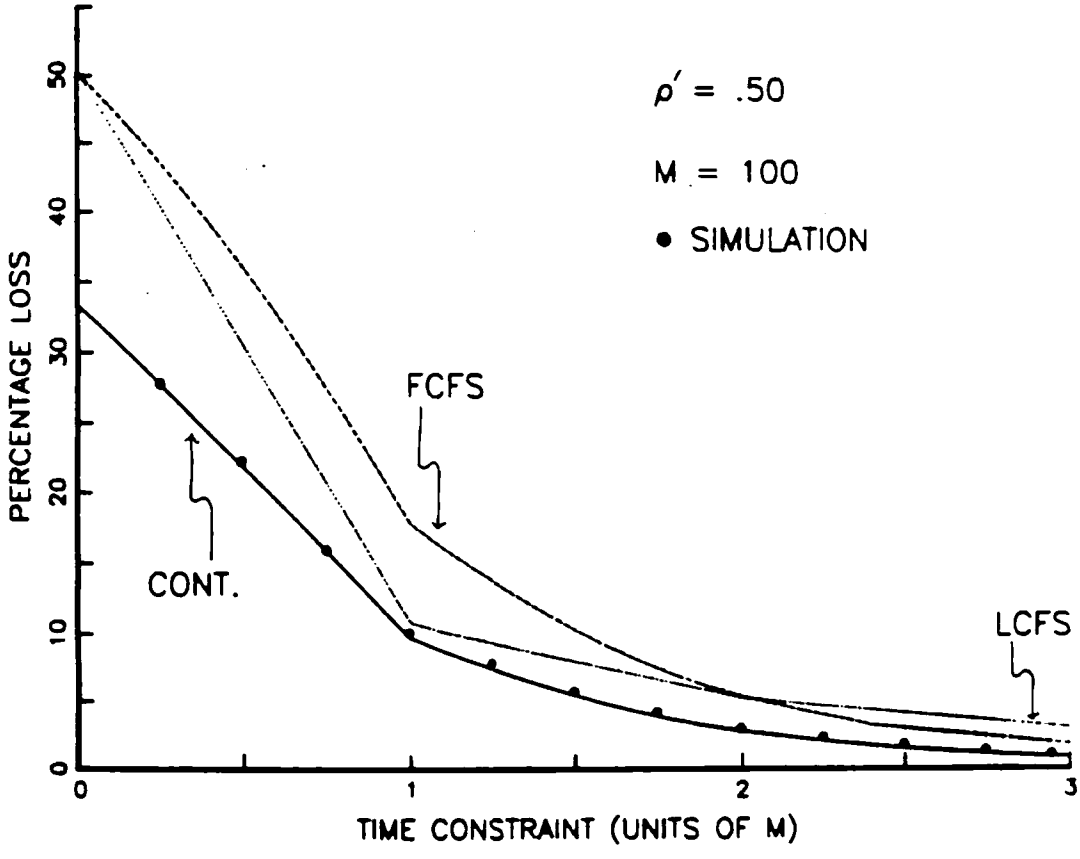
As expected, these results show significant performance improvements over the FCFS and LCFS results. Two factors contribute to this increase in performance. First, the optimal policy elements (1) and (3) have been used; the critical role of these two policy elements was demonstrated in [Kurose 83]. However, the increase in performance probably primarily results from the inclusion of policy element (4). Note that as a result of this policy element, the channel is used only for "useful" work. That is, if the protocol sends a message then, due to element (4) and our definition of waiting time, that message will be accepted at the receiving station with probability 1. Thus, unlike the protocols in [Kurose 83], the channel is *never* used for the transmission of messages which are lost at the receiving station.

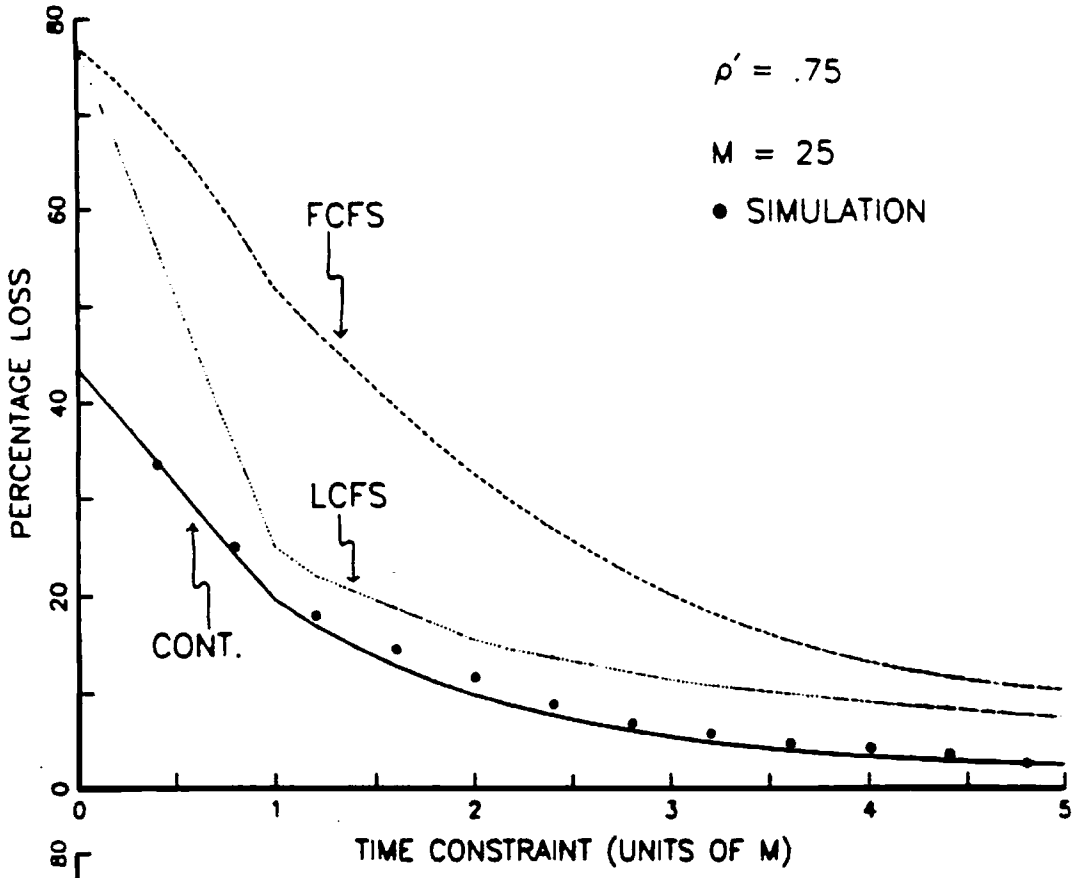
Figure 7: Performance Results

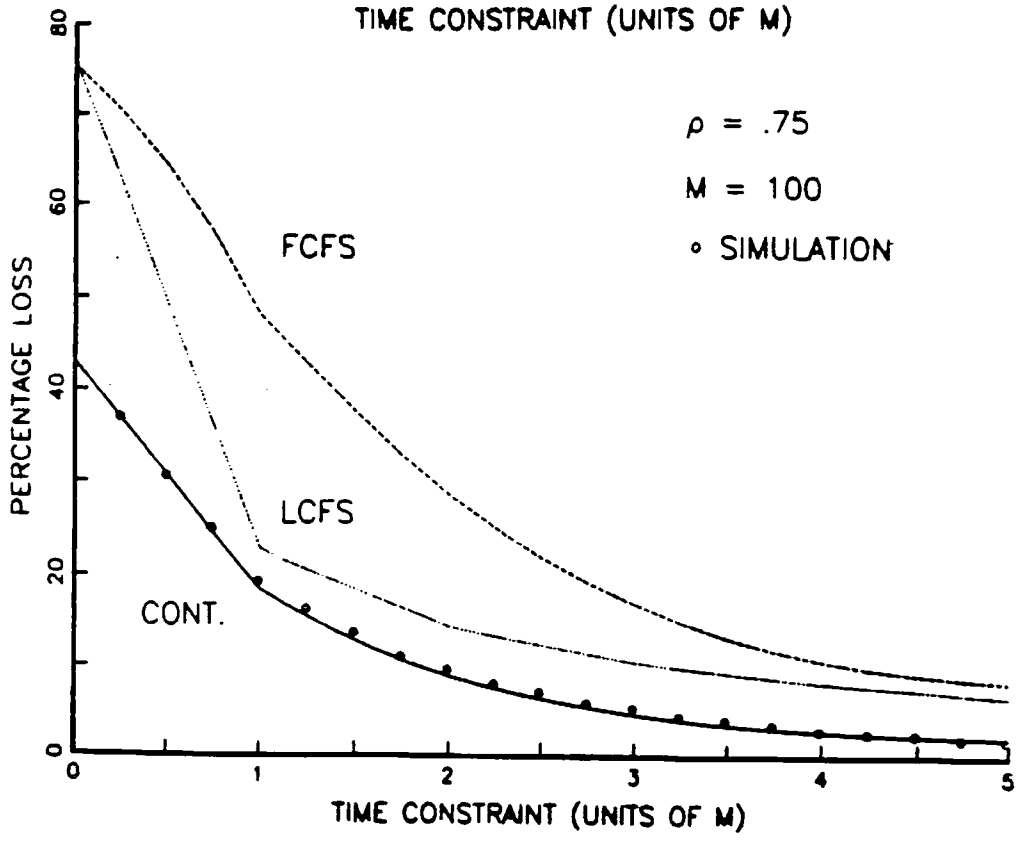





ρ′ = .50
M = 25
• SIMULATION
FCFS
CONT.
LCFS
PERCENTAGE LOSS
TIME CONSTRAINT (UNITS OF M)
0
10
20
30
40
50
0
1
2
3
4


ρ′ = .50
M = 100
• SIMULATION
FCFS
CONT.
LCFS
PERCENTAGE LOSS
TIME CONSTRAINT (UNITS OF M)
0
10
20
30
40
50
0
1
2
3


ρ′ = .75
M = 25
• SIMULATION
FCFS
LCFS
CONT.
PERCENTAGE LOSS
TIME CONSTRAINT (UNITS OF M)
0
20
40
60
80
0
1
2
3
4
5


ρ = .75
M = 100
∘ SIMULATION
FCFS
LCFS
CONT.
PERCENTAGE LOSS
TIME CONSTRAINT (UNITS OF M)
0
20
40
60
80
0
1
2
3
4
5

Recall that our definition of waiting time in section 2 does not include a message's *own* scheduling time as part of its waiting time. This definition only approximates the truer (and more traditional) definition of waiting time: the time between a message's arrival at a sending station and the start of its successful transmission. The waiting time approximation was introduced to avoid considerable complication to the analyses of sections 3 and 4. In the simulation results shown in figure 7, messages were considered lost when their *true* waiting time exceeded the time constraint. (Note that in this case it is possible for messages to be lost at both the sending and receiving stations.) The close agreement between the analytic results and the simulation results indicate that the waiting time approximation as well as the other approximations and assumptions underlying the analysis are indeed reasonable.

5. Conclusion

In this paper we have examined the problem of controlling a group random access protocol based on message arrival times in order to maximize the percentage of messages with delays below a given bound. First, we identified four policy elements which comprised a policy for controlling protocol operation. A semi-Markov decision model was then developed and three of the four optimal policy elements were determined within this model; these elements were found to be both intuitive and simple. A simple closed form characterization of the final optimal policy element was not found.

Although the decision model was shown to be sufficient to provide performance results, it was found to be too computationally expensive to be of practical use. Thus an alternative performance model, based on a queueing system with impatient customers, was developed. A heuristic was then adopted for the final policy element. Protocol performance was then examined and found to be superior to cases in which it was not controlled using optimal policy elements (1), (2) and (4).

Finally, several extensions of this work seem promising to pursue. First, note that our *definition* of a policy (i.e., elements (1) through (4)) is only one policy by which the protocol can be controlled. Introducing additional policy elements (e.g., not necessarily splitting a window in half), may result in further performance improvements. Secondly, the protocol discussed in this paper operates in a synchronous manner. Since the synchronization of distributed stations is often a difficult task, it would be desirable for the protocol to operate in an asynchronous manner. It would be interesting to explore different approaches for achieving this asynchronous operation and the effect of these approaches on the time-constrained performance of the protocol. Molle [Molle 83] has recently investigated several aspects of this problem. Finally, the present protocol requires that all stations select initial windows of the same size. This may not be possible in practice and in some cases may not even be desirable! For example, if different stations have different priorities, then one form of priority be can achieved by permitting stations to choose different initial window sizes. Determining a policy for choosing these window sizes based on station priority and analyzing the performance of such a policy is an interesting, but potentially difficult, problem for future work.

[Baccelli 81] Francois Baccelli and Gerard Hebuterne.
On Queues With Impatient Customers.
In *Performance '81*, pages 159-179. North-Holland Publishing Company, 1981.

[Capetanakis 79] J. I. Capetanakis.
Generalized TDMA: The Multi-Accessing Tree Protocol.
IEEE Transactions on Communications COM-27:1476-1484, October, 1979.

[Coffman 76] E.G. Coffman.
Computer and Job Shop Scheduling.
J. Wiley and Sons, N.Y. N.Y., 1976.

[Cohen 77] D. Cohen.
Issues in Transnet Packetized Voice Communication.
In *Conference Report.* Proceedings of the Fifth Data Communications Symposium, Snowbird, Utah, 1977.

(DSN 82) MIT Lincoln Laboratories.
Workshop on Distributed Sensor Networks, MIT Lincoln Laboratories, Lexington, Mass., 1982.

[Gallager 78] R. G. Gallager.
Conflict Resolution in Random Access Broadcast Networks.
In *Proc. of the AFOSR Workshop in Communication Theory and Applications.* AFOSR, 1978.

[Gitman 81] I. Gitman.
Analysis and Design of Hybrid Switching Networks.
IEEE Transactions on Communications COM-29(9):1290-1300, September, 1981.

[Hayes 78] J. F. Hayes.
An Adaptive Technique for Local Distribution.
IEEE Transactions on Communications COM-26:1178-1186, August, 1978.

[Howard 71] Ronald A. Howard.
Dynamic Probabilistic Systems, Volume 2: Semi-Markov and Decision Processes.
J. Wiley and Sons, N.Y. N.Y., 1971.

[Kleinrock 75] L. Kleinrock.
Queueing Systems Volume I: Theory.
J. Wiley and Sons, N.Y., N.Y., 1975.

[Kurose 83] J.F. Kurose and M. Schwartz.
A Family of Window Protocols for Time-Constrained Applications in CSMA Networks.
In *Proceedings*, pages 405-413. IEEE INFOCOMM '83, 1983.

[Lam 75] S.S. Lam and L. Kleinrock.
Packet Switching in a Multiaccess Broadcast Channel: Dynamic Control Procedures.
IEEE Transactions on Communications COM-23:891-904, September, 1975.

[Molle 83] M. Molle.
Asynchronous Multiple Access Tree Algorithms.
In *ACM SigComm 1983 Proceedings*. ACM, March, 1983.

[Schwartz 77] M. Schwartz.
Computer Communication Network Design and Analysis.
Prentice Hall, 1977.

[Tanenbaum 81] A. S. Tanenbaum.
Computer Networks.
Prentice Hall, Englewood Cliffs, N.J., 1981.

[Tobagi 80] F. Tobagi.
Multiaccess Protocols in Packet Communication Systems.
IEEE Transactions on Computers COM-28:468-488, April, 1980.

[Towsley 82] D. Towsley and G. Venkatesh.
Window Random Access Protocols for Local Computer Networks.
IEEE Transactions on Computers COM-31(8), August, 1982.

Appendix A: Proof of Lemma 4

Let $P' \epsilon \{P^w\}$ be a policy with a decision, k', which for some state $s_i \epsilon S$ does *not* minimize the one step pseudo loss. Also define:

- p_{ij}^k to be the probability that state s_j immediately succeeds state s_i given decision k is made upon entry to state s_i.
- τ_i^k to be the average time from when a decision is made upon entry to state s_i to the time when the next decision is made, given decision k is made upon entry to state s_i.
- r_i^k to be the one step pseudo loss in state s_i, given decision k is made upon entry to state s_i.

Recall that $\{P^w\}$ is a set of policies which choose the same action for policy element (2), i.e., choose the same size initial window when in the same state. Since any two equal length intervals of time are statistically identical, then if policies P and P' are in $\{P^w\}$, and choose actions k and k' respectively when in state s_i then

$$p_{ij}^k = p_{ij}^{k'} \text{ and } \tau_i^k = \tau_i^{k'}$$

We now want to determine if P' minimizes the average pseudo loss even though it does not (by hypothesis) minimize the one step pseudo loss for state s_i. If P' does not minimize the average pseudo loss then there must exist a policy iteration [Howard 71] starting from P' which yields a policy with a smaller pseudo loss. To determine if such a policy iteration exists, we would normally have to solve the set of simultaneous equations:

$$v_n + g\tau_n^{k'} = -r_n^{k'} + \sum_{j=1}^{K} p_{nj}^{k'} v_j \qquad n=1,2 \ldots K \quad (A1)$$

where $v_K=0$ and $\tau_n^{k'}$, $r_n^{k'}$ and $P_{nj}^{k'}$ are computed using the decisions of policy P′ to determine the values of g and $\{v_j\}$. g is known as the *gain* under policy P′ and can be related to the average pseudo loss; $\{v_j\}$ is known as the set of relative values. To determine if a policy iteration exists, we must examine the value of γ_i^k, defined below, for all possible decisions, k, in state s_i:

$$\gamma_i^k = -(r_i^k/\tau_i^k) + (1/\tau_i^k) \sum_{j=1}^{K} p_{ij}^k v_j \qquad (A2)$$

Fortunately, for the purposes of proving that P′ does not minimize the average pseudo loss, we need not actually solve A1 (note, however, that a numerical value for the loss can be obtained from solving A1). Now, if we can show that there is some alternate decision, *k*, such that

$$\gamma_i^{k'} < \gamma_i^k$$

then P′ does not maximize the gain and hence does not minimize the average pseudo loss [Howard 71]. By our above arguments, p_{ij}^k, and τ_i^k are independent of k for all $P \epsilon \{P^w\}$. Thus, A2 can be expressed:

$$\gamma_i^k = -c_1 r_i^k + c_2$$

where c_1 and c_2 are constant with respect to k and c_1 is positive. Since $r_i^k \geq 0$, the maximum value of γ_i^k occurs when r_i^k is minimized. By hypothesis, P′ chooses k′ such that r_i^k is *not* minimized and thus there exists some *k* such that r_i^k is less than $r_i^{k'}$. Thus there exists a policy iteration on P′ and hence P′ does not minimize the average pseudo loss. Thus, any policy which does not minimize the one step pseudo loss in every state does not minimize the average pseudo loss.