A Family of Window Protocols for Time-Constrained Applications in CSMA Networks*

James F. Kurose
Department of Computer Science

Mischa Schwartz
Department of Electrical Engineering

Columbia University,
NY NY 10027

CUCS-90-83

ABSTRACT

In the design of random access CSMA protocols for time-constrained applications such as packetized voice, the distribution of message waiting times is of critical importance. It is shown that the ordering or scheduling imposed on message transmissions by a particular random access protocol greatly affects the message waiting time distribution. We present a random access protocol which can provide a large class of distributed message scheduling disciplines based on message arrival times. Moreover, this protocol can adaptively vary to provide an optimal service discipline in response to changing system demands.

Analytic models are derived for the waiting time distribution for the cases of FCFS, LCFS and RANDOM scheduling and the analytic results are compared with simulation. Other possible scheduling disciplines are discussed and the impact of the distributed scheduling discipline on the performance of time-constrained applications is examined.

---

*This work was supported in part by National Science Foundation Grant NSF ECS-8110319 and the Defense Advanced Research Projects Agency Project N00039-82-C-0427.

1. Introduction

The use of carrier sense multiple access (CSMA) channels for data communication has been studied now for almost a decade. Much of this research has focused on developing random access strategies which permit users to efficiently share a channel in a distributed fashion. The primary performance metric used to evaluate these random access strategies has been the classical tradeoff between time delay and throughput.

For many data communications applications, average time delay and throughput are adequate to characterize the relevant performance tradeoffs. However, for many time-constrained applications, [e.g. packetized voice, distributed sensor networks [Distributed Sensor Networks 78]] in which data must either be transmitted within a certain time limit or be lost, the additional performance metric of loss must be considered. Thus, rather than the two way tradeoff between time delay and throughput, there is a three way tradeoff among the performance measures of loss, throughput and time delay.

We will show that for time-constrained communication over CSMA channels, the ordering imposed on the message transmissions by a particular channel access strategy is a critical factor in determining system performance. That is, while a random access strategy does provide for distributed sharing of a resource, the channel access mechanism itself also functions as a distributed scheduling mechanism which permits messages distributed among the stations on a channel to be transmitted according to some explicit or implicit scheduling policy. A single general random access protocol based on a generalization of current window mechanisms [Gallager 78] [Towsley 82] can be used to obtain a large class of distributed scheduling disciplines based on message arrival times. Moreover, such a generalized window mechanism can adaptively vary during system operation to provide an optimal service discipline in response to changing system demands.

In the following section we describe this random access protocol and

discuss its use as a distributed scheduling mechanism. In section 3 we then study three particular cases in which the protocol provides FCFS, LCFS and RANDOM scheduling and present approximate analytic models for the waiting time distribution for each of these three disciplines. The analytic results are shown to compare favorably with simulation results.

Using these results, we then compare the time delay versus throughput versus loss performance of the three distributed scheduling disciplines in section 4. Another scheduling discipline which specifically attempts to maximize the percentage of messages with waiting times below a specified time bound is then suggested and discussed. Finally, the performance results are used to illustrate several important features of the scheduling function performed by a random access protocol and the impact of the scheduling function on the time-constrained performance of the protocol.

2. A Protocol for Time-Constrained Communication Over a CSMA Channel

In this section we describe a random access protocol suitable for time-constrained communication over a CSMA channel and demonstrate its use as a distributed scheduling mechanism. A probabilistic model of the protocol's behavior is derived in order to obtain average performance measures for the contention resolution phase of the protocol.

2.1. Description of the Protocol

Let us assume that each station on the multiple access channel possesses a clock which defines the current time, t, and that the clocks at all stations are synchronized. Each station will maintain a value for each message which arrives at the station called the pseudo arrival time of the message. The initial value of a message's pseudo arrival time is the actual arrival time of the message at the station; a message's pseudo arrival time may change as described below. In addition, each station will also maintain a value, t_past, such that all messages currently at any station have a pseudo arrival time greater than t_past; all stations initialize t_past to the initial clock value. Finally, each station has a pseudo random number generator and each station initializes the generator with the same seed and therefore produces the same sequence of pseudo-random numbers.

The operation of the protocol is shown below in figure 1. The pseudo-arrival times of all messages which have not yet been successfully transmitted are shown below the time axes in this figure. The protocol functions as follows. All stations continuously monitor the channel and after each successful message transmission, each station uses its random number generator and a scheduling policy to be described below to select a window of time between t_past and the current time, t; the selection of this window is shown on the second time axis in figure 1. Since all stations have the same value of t_past and generate the same sequence of random numbers, all stations will select the same window of time. After a window has been selected, all stations with messages with pseudo arrival times which fall within this window of time attempt to transmit the message.

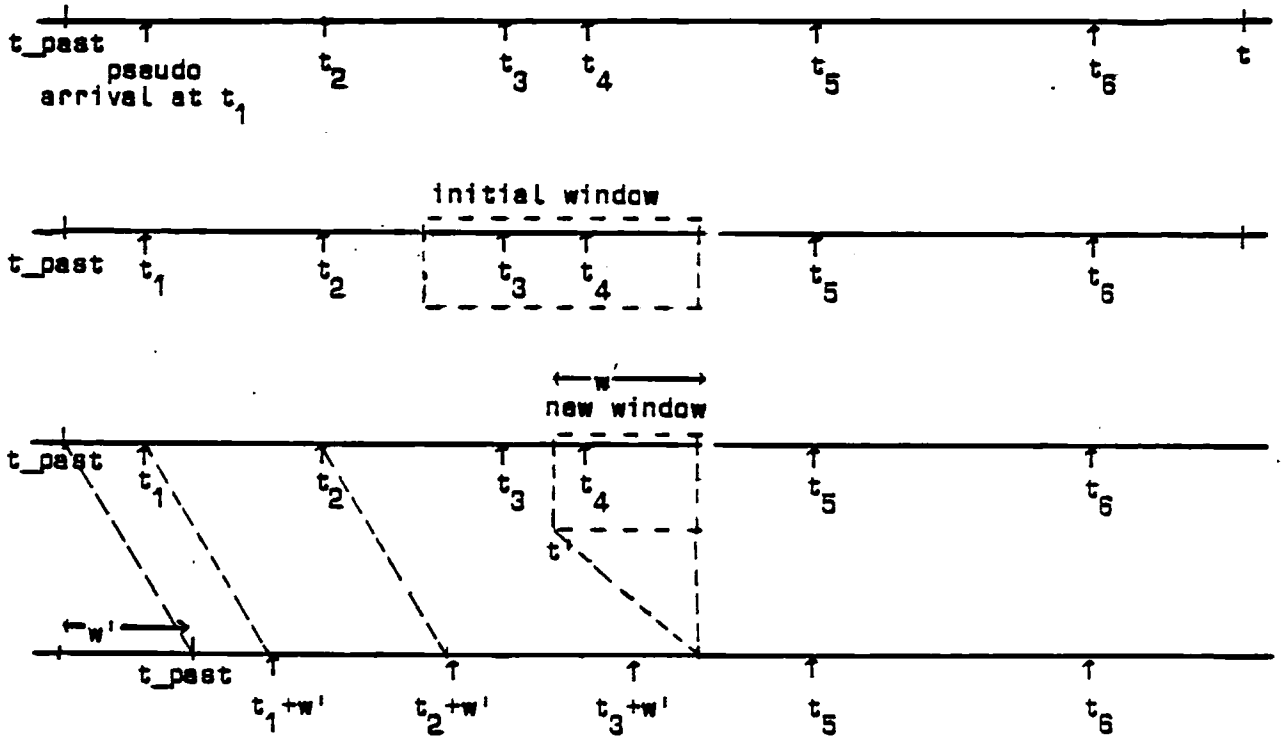


figure 1

If no stations have a message with a pseudo arrival time which falls in this window, then an empty slot occurs on the channel and all stations select a new initial window after updating their values of t_past and the pseudo arrival time of any message as discussed below. If exactly one message falls within this initial window, then its transmission begins immediately.

If more than one station has a message with a pseudo arrival time which falls within the time window, then two or more stations attempt to transmit a message and a collision occurs. All stations continue to monitor the channel and attempt to resolve this collision by splitting the initial time window in half. The stations then use their random number generator and the scheduling policy described below to select one of the two halves of the initial window as shown on the third time axis in figure 1; since all stations use the same sequence of pseudo random numbers, all stations will select the same half of the split window. The random access procedure is then repeated using the selected half of the split window as the new time window. If no message has a pseudo arrival time which falls in the selected half of a split window, an empty slot will occur on the channel and all stations will then select the remaining half of the split window, immediately split this new window (since it is known to contain two or more messages), choose one of the halves of the newly split window, and repeat the above access procedure using that half of the newly split window. This splitting process continues until a single message is finally transmitted.

All stations perform the windowing process and thus each station knows the width, w', and the starting time, t', of every time window which contains either no message arrivals or a single message arrival. Once a window with exactly zero or one message arrivals has been selected and the message (if any) has been transmitted, all stations can effectively remove this window of time from consideration, as if the window of time had never occurred. The effect of removing this window of time is twofold. First, all stations with a message with a pseudo arrival time before t' must update the pseudo arrival time of the message by the width, w', of the window. Secondly, since the pseudo arrival time of each message which arrived before t' has been increased by w', there is a gap of time from t_past to t_past+w' for which it is known that there are no messages present and all stations can update t_past accordingly. The effect of updating the value of t_past and the pseudo arrival times is shown on the fourth time axis in figure 1.

The selection of the position of the initial window and the method by which one of the two halves of a split window is selected determine the scheduling policy implemented by the random access mechanism. Suppose that the time

between t_past and t (the current time) divides into n windows of width w; we note three special cases:

1. The first (oldest) of the n windows is always selected as the initial window and the first half of a split window is always chosen before the second.

2. The last (newest) of the n windows is always selected as the initial window and the second half of a split window is always chosen before the first.

3. Each of the n windows is equally likely to be chosen as the initial window and both halves of a split window are equally likely to be chosen first.

Case (1) above implements FCFS scheduling, case (2) implements LCFS scheduling and case (3) implements RANDOM scheduling. In the most general case:

4. Each of the n windows is selected with some probability according to some discrete probability distribution δ and the first half of a split window is selected with probability q and the second half is selected with probability 1-q.

Depending on the distribution δ chosen in (4) and the value of q, any one of a large class of scheduling disciplines can be selected. As we will see, the optimal selection of δ and q will be dependent on current system demands including throughput and loss tolerances.

2.2. Analysis of the Average Performance of the Window Mechanism

In this section we derive an approximate expression for the average message scheduling time of the protocol. In order to obtain this expression we will first follow Towsley [Towsley 82], Molle [Molle 81] and others and analytically determine the exact value of the message scheduling delay under saturation conditions, which occur when the average arrival rate exceeds the average time needed to schedule and transmit a message. Specifically, we will make use of the property that at saturation, the average waiting time is unbounded and thus the difference between the current time and t_past is always greater than any window width chosen.

Once the average scheduling delay under saturation conditions has been obtained, the actual message arrival rate at which saturation occurs can then be determined. This value for the saturation arrival rate and the average message scheduling time at saturation can then be used to provide an approximate expression for the message scheduling delay for arrival rates less than saturation.

We will assume that time is slotted in units of $2t_0$, where t_0 is the end-to-end propagation delay of the channel. Furthermore, we will assume that all stations can detect message collisions (CSMA-CD) and can abort transmission of a collided message in a negligible amount of time. Finally, we assume that the message arrivals to all stations together constitute a poisson process with rate λ (arrivals/slot).

Since the overall arrival process is poisson, at any point in time, the inter-(pseudo)arrival times of unsent messages waiting at all stations are exponentially distributed with mean $1/\lambda$. This fact can be inductively established since, due to the memoryless property of the exponential distribution, the removal of a time window and the concommitant shifting of pseudo-arrival times as described above preserves this exponential interarrival property. One important consequence of the exponential nature of the inter-(pseudo)arrival times is that any two windows of time between the current time and t_past which are of equal length are statistically identical with respect to the pseudo-arrival times. Thus the order in which windows are selected has no effect on the average message scheduling delay and thus the average message scheduling time is independent of the scheduling discipline implemented by the protocol.

Let us now proceed to determine the average message scheduling time which in general, will be a function of the arrival rate, λ. Define:

$\bar{s}(\lambda)$ - average time (in slots) to schedule a message for traffic arrival rate λ.

As discussed earlier, in order to obtain an approximate expression for $\bar{s}(\lambda)$, we will first determine the exact value of $\bar{s}(\lambda)$, under saturation conditions. Define:

$\bar{s}_{sat}$ - the average number of slots needed to schedule a message at saturation. The last slot (in which only 1 message is transmitted) is <u>not</u> considered part of the scheduling time.

s_k - the number of slots needed to schedule a message given a window is known to contain k messages, with $k \geq 2$.

$q_{k,i}$ - probability that i messages are in one half of a window given there are k messages in the window. Due to the memoryless property of the interarrival process, $q_{k,i}$ is given by:

$$q_{k,i} = \binom{k}{i} 2^{-k}$$

p_i - probability of i arrivals in a window. If the length of a window is given by the parameter ψ (in units of time slots), then:

$$p_i = (\lambda\psi)^i e^{-\lambda\psi} / i!$$

Now, $\bar{s}_{sat}$ can be expressed in terms of s_k by conditioning on the number of arrivals in a window:

$$\bar{s}_{sat} = p_0(1+\bar{s}_{sat}) + p_1 \cdot 0 + \sum_{k=2}^{\infty} p_k(1+s_k)$$

or

$$\bar{s}_{sat} = \frac{p_0 + \sum_{k=2}^{\infty} p_k(1+s_k)}{1 - p_0} \qquad (1)$$

The added 1's in the terms $(1+s_k)$ and $(1+\bar{s}_{sat})$ in (1) are due to the fact that 1 time slot is first required to learn that zero or two or more collisions have occurred. We can now condition s_k on the events of zero, one, or more than one message in the selected half of a split window. Note that if no message occurs in the selected half of a split window (this occurs with probability $q_{k,0}$), no slot is needed to determine that two or more messages occur in the remaining half of the split window and that half can be split immediately. Conditioning s_k thus gives:

$$s_k = q_{k,0}(1+(s_k-1)) + q_{k,1} \cdot 0 + \sum_{i=2}^{k} q_{k,i}(1+s_i)$$

or

$$s_k = \frac{(1 - q_{k,1} - q_{k,0}) + \sum_{i=2}^{k-1} q_{k,i} s_i}{(1 - q_{k,0} - q_{k,k})} \qquad 3 \leq k \qquad (2)$$

with the initial condition $s_2 = 0.5$. Values for s_k can be iteratively obtained from (2) above and then substituted into (1) to determine the value for $\bar{s}_{sat}$. Note, however, that $\bar{s}_{sat}$ is a function of the yet unspecified value of $\lambda\psi$. Clearly, the value of this parameter should be chosen to minimize the

value of $\bar{s}_{sat}$. Numerical methods can be used to show that $\bar{s}_{sat}$ achieves a minimum value of 1.24 slots when $\lambda\Psi$ has a value of 1.2. Thus, whatever the actual arrival rate at which saturation occurs, the initial window size, Ψ, should be chosen such that $\lambda\Psi = 1.2$ and thus $\bar{s}_{sat} = 1.24$ slots, independent of the arrival rate at which saturation occurs.

Equation (1) thus gives the message scheduling delay under saturation conditions to be 1.24 slots; let us now determine the actual arrival rate at which saturation occurs. Following Lam [Lam 80], we note that the overall channel utilization ρ (the arrival rate times the average time for scheduling and transmitting a message) must be bounded above by unity. If we define α as the ratio of the end-to-end propagation delay, t_0, to the fixed message length, then the length of a message in slots is given by $1/(2\alpha)$ and the channel utilization bound is expressed as:

$$\lambda[1/(2\alpha) + \bar{s}(\lambda)] \leq 1 \qquad (3)$$

The saturation value of the arrival rate, λ_{sat}, is that value of λ for which the equality in (3) holds and thus:

$$\lambda_{sat} = 1 / [1/(2\alpha) + \bar{s}_{sat}] \quad \text{or} \quad \lambda_{sat} = 1 / [1/(2\alpha) + 1.24] \qquad (4)$$

If we define the effective channel throughput, ρ', to be the fraction of the channel which is utilized by successfully transmitted messages, (i.e. $\rho' = \lambda/(2\alpha)$) then the effective channel throughput at saturation or the maximum effective channel utilization is given by:

$$\rho'_{sat} = \frac{1/(2\alpha)}{1/(2\alpha) + \bar{s}_{sat}} = \frac{1}{1 + 2\alpha\bar{s}_{sat}}$$

The analysis thus far has provided the average message scheduling delay under saturation conditions as well as the actual message arrival rate (and thus effective throughput) at which saturation occurs. We also know that as λ and thus ρ' approaches zero, the average message scheduling delay also approaches zero since an arriving message would always be sent without contention upon arrival. Given these two endpoint values, we will approximate the intermediate points of the average message scheduling time, $\bar{s}(\rho')$, by fitting a function of the form $\rho' / (a - \rho')$ to these endpoint values, where a is a suitably chosen constant. The results of this approximation are

compared with the average message scheduling times obtained through simulation for various message sizes in figure 2. In the following section, the average message scheduling times will be used to study the waiting time distribution under different scheduling disciplines.

3. Waiting Time Distributions for FCFS, RANDOM and LCFS Scheduling

In this section, we present analytic and simulation results for the distribution of message waiting times for fixed length packets in the cases in which the window random access protocol provides FCFS, RANDOM and LCFS scheduling. Throughout the analysis in this section the messages waiting at stations to be transmitted will be considered as customers in a distributed queue. More importantly, the message scheduling time or contention resolution time immediately preceding a successful transmission will be considered as part of the service time for that message which is successfully transmitted. Thus the service time for a particular message will always have two components : a message scheduling time (i.e. the scheduling delay due to that contention period (if any) which results in the message beginning successful transmission) and the actual transmission time for that particular message.

3.1. Service Time Distribution of Messages

In this section we determine the service time distribution for fixed length messages; note that since the message scheduling time has been shown to be independent of the scheduling discipline imposed by the protocol, this service time distribution will thus also be independent of the scheduling discipline. Let $b(t)$ be the distribution of the message service time, $s(t)$ be the distribution of the scheduling time, and $x(t)$ be the distribution of the message transmission time; throughout this section, time will be in units of slot length. Since the service time is the sum of two independent random variables, the service time distribution time is given by:

$$b(t) = s(t) \circledast x(t) \qquad (5)$$

where $\circledast$ is the convolution operator. For the case of fixed length messages, $x(t)$ is given by:

$$x(t) = \mu_0(t - 1/(2\alpha)) \qquad (6)$$

where Ψ_0 is the unit impulse function and $1/(2\alpha)$ is the fixed message length.

The distribution of the message scheduling time is more difficult to obtain. However, our simulation studies have shown that the geometric distribution is a good approximation for the message scheduling time distribution, where the mean of the geometric distribution is taken to be the mean scheduling time as determined in the previous section. Let $\bar{s}(\lambda)$ be the average message scheduling delay and define c_i to be the probability that the message scheduling delay is i slots. Then:

$$c_i = c(1-c)^i$$

where:

$$c = 1 / (1 + \bar{s}(\lambda))$$

and the probability distribution for the message scheduling time is thus given by:

$$s(t) = \sum_{i=0}^{\infty} c_i \Psi_0(t-i) \qquad (7)$$

Finally, using the value for x(t) and s(t) from (6) and (7) respectively, equation (5) gives the service time distribution for a message as:

$$b(t) = \sum_{i=0}^{\infty} c_i \Psi_0(t - (1/(2\alpha) + i)) \qquad (8)$$

3.2. Distribution for FCFS Scheduling

Since the message scheduling or collision resolution time has been modeled as part of the service time of the messages, our model of the distributed queue reduces to the case of an M/G/1 queue, where the service distribution of the customers is given by (8).

The waiting time distribution for customers in an FCFS M/G/1 queue is given by [Kleinrock 75]:

$$w_{fcfs}(t) = \sum_{k=0}^{\infty} (1-\rho)\rho^k \hat{b}^k(t) \qquad (9)$$

where

$\hat{b}(t)$ = the density function of the residual service time that an arriving customer finds for the customer (if any) in service.

$\hat{b}^k(t)$ = the k-fold convolution of $\hat{b}(t)$

ρ = the server utilization, previously defined as

$$\rho = \lambda(1/(2\alpha) + \bar{s}(\lambda))$$

For values of ρ not especially close to unity, the infinite sum in (9) can be truncated at some finite value of k to produce an excellent approximation to $w_{fcfs}(y)$. The residual service time density for a given arrival rate (λ) and a service time distribution given by (8) can be calculated [Kleinrock 75] to be:

$$\hat{b}(t) = \frac{1 - \sum_{i=0}^{\infty} c_i \psi_{-1}(t - (1/(2\alpha) + i))}{(\ 1/(2\alpha) + \bar{s}(\lambda)\ )} \qquad (10)$$

where ψ_{-1} is the unit step function.

Figure 3 shows the computed waiting time distributions for the case in which the random access protocol provides FCFS service; the sum in equation (9) was terminated at k=9 to obtain these values. Waiting time distributions are shown for 3 different traffic arrival intensities for message lengths 10 and 50 times the end-to-end channel propagation delay (α=0.1 and α=0.02, respectively). The degree to which these analytic results in figure 3 agree with the simulation values indicate that good approximations were introduced to obtain the analytic form of $w_{fcfs}(t)$.

3.3. Distribution for LCFS Scheduling

In order to compute the waiting time distribution under LCFS scheduling, the entire waiting time of a message can be considered as a series of waiting time components as shown below in figure 4. A message has no waiting time with probability γ_1. With probability $1-\gamma_1$ a message has a first waiting time component with a distribution given by $d_1(t)$. A message which finishes the first waiting time component begins service with probability γ_2 and requires a second waiting time component (with a distribution $d_2(t)$) with probability $1-\gamma_2$. In general, γ_i represents the probability, given that a message has completed i-1 waiting time components, that it will begin service after component i-1. Note that in order for a message to have exactly i waiting time components, it must not enter service after completing each of the first i-1 waiting time components and must enter service after the ith component. Thus the probability that a message has exactly i waiting time components is

given by: $(1-\gamma_1)(1-\gamma_2)\cdots(1-\gamma_i)\gamma_{i+1}$. The distribution of the length of the ith component is given by $d_i(t)$.

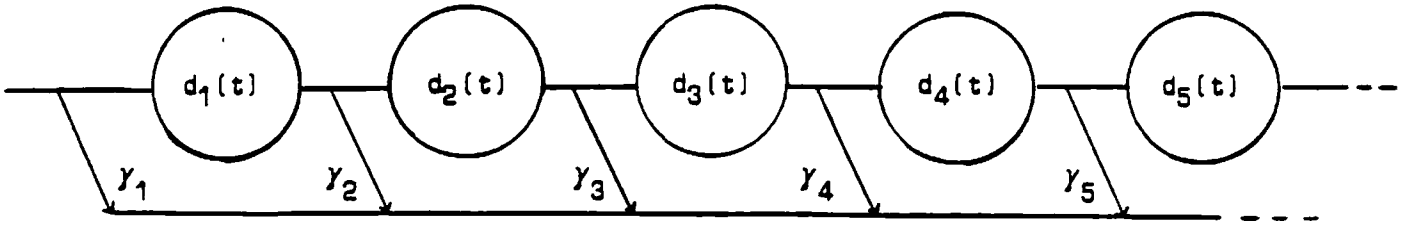


figure 4

If a message experiences any waiting time at all, then the first component of its waiting time, $d_1(t)$, results from the residual service of another message already in service when the message arrives; thus $d_1(t)$ is given by $\hat{b}(t)$. Since the scheduling discipline is LCFS, the remaining components of the message's waiting time result from messages which arrive after the message, but are transmitted before the message; thus $d_i(t)$ is given by $b(t)$ for $i>1$.

Using the above model of the waiting time, the distribution of the waiting time is given by:

$$w_{lcfs}(t) = \gamma_1 u_0(t) + (1-\gamma_1)\gamma_2\hat{b}(t) + \sum_{i=2}^{\infty}(1-\gamma_1)\cdots(1-\gamma_i)\gamma_{i+1}\hat{b}(t)\circledast b^{i-2}(t) \qquad (11)$$

where u_1 and $\hat{b}(t)$ are as previously defined and $b^{i-2}(t)$ is the $(i-2)$fold convolution of the service time distribution. We now compute the unknown values, $\{\gamma_i\}$. To do this, define :

q_0 – probability that no messages are waiting to be sent (queue is empty) at equilibrium.

$\hat{p}_i$ – probability of i arrivals during a residual service time.

p_i' – probability of i arrivals during a message's service time.

p_j^k – probability that j messages have arrived during the first k components of the message waiting time given that the kth component of the message waiting time has just ended.

Q_j^k - probability that j messages have arrived during the first k-1 components of the message waiting time given that the kth component of the message waiting time has just begun.

Clearly, $\gamma_1 = q_0$ and $\gamma_2 = \hat{p}_0$. The value of q_0 can be determined from an analysis for the number in the queue as in [Lam 80]. Since the ith waiting time component is the last component if and only if exactly i-1 messages have arrived during the first i waiting time components, the remaining values for $\{\gamma_i\}$ are given by $\gamma_i = P_{i-1}^i$. The values for Q_j^{k+1} and P_j^{k+1} can be iteratively determined from the initial condition $P_j^1 = \hat{p}_j$. First, Q_j^{k+1} can be computed from P_j^k as follows. The relationship between these two sets of probabilities is shown below in figure 5.

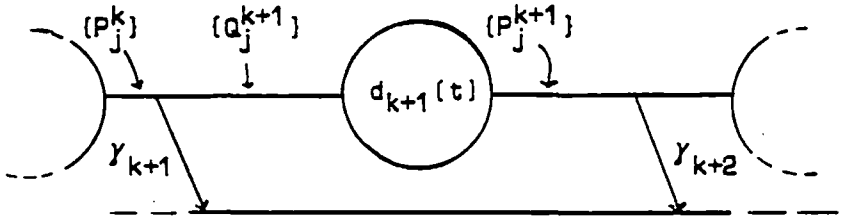


figure 5

Note that for all j less then k-1, P_j^k is known to be zero since in order for a message to complete k waiting time components, k-1 or more messages must have arrived since its own arrival. Now, if a message begins the k+1st waiting time component, there must have been <u>strictly</u> more than k-1 messages after component k was completed (otherwise the message waiting time would have ended after component k); thus although P_{k-1}^k was non zero at the end of the kth waiting time component, at the beginning of component k+1, it is known that $Q_{k-1}^{k+1} = 0$. For values of j greater than k-1, the ratio of Q_j^{k+1} and Q_{j+1}^{k+1} should equal the ratio of P_j^k and P_{j+1}^k. However, the actual values of Q_j^{k+1} and Q_{j+1}^{k+1} must be normalized since at the beginning of component k+1, $Q_{k-1}^{k+1} = 0$. The above considerations indicate that the values of Q_j^{k+1} are given by:

$$Q_j^{k+1} = 0 \qquad 0<j<k \qquad (12)$$

$$= \frac{P_j^k}{1 - P_{k-1}^k} \qquad j >= k$$

Finally, given the values for Q_j^{k+1}, the values of P_j^{k+1} can be easily computed by

conditioning the event of j arrivals immediately following k+1 waiting time components on the number of arrivals during component k+1. Thus:

$$p_j^{k+1} = \sum_{n=0}^{j} a_{j-n}^{k+1} \cdot p_n' \qquad (13)$$

The waiting time distribution for LCFS service can thus be computed using (11), (12) and (13) with the values of $\{Y_i\}$ found above. Figure 6 shows the computed waiting time distributions for 3 different traffic arrival intensities for message lengths given by $\alpha=.1$ and $\alpha=.02$. Note that all the distributions shown in figure 6 rise rapidly for time values less then the message length (corresponding to messages which begin service immediately after a residual service is completed) and then slowly approach the asymptotic value of 1.

3.4. Distribution for RANDOM Scheduling

The waiting time distribution under RANDOM scheduling can also be determined using waiting time components. Specifically, the first component of the waiting time once again results from the residual service time for another message already in service when the message arrives. Also, the remaining components of the waiting time once again result from the service times of other messages; however, since the message scheduling discipline is RANDOM, these other messages may have arrived at any time.

The value for Y_0 will be exactly the same as under LCFS since the probability that an arriving message finds the queue empty is independent of the order in which messages are selected for service [Kleinrock 76]. In order to determine the remaining values for the $\{Y_i\}$, we can use the average number of messages in the distributed queue given there is at least one message in the queue (i.e. the message for which the waiting time distribution is being computed). This conditional value is given by $\bar{q}/(1-q_0)$, where $\bar{q}$ is the unconditional average number of customers in the queue. Since RANDOM scheduling implies that all messages in the queue are equally likely to begin service next, if there are on the average, $\bar{q}/(1-q_0)$ messages in the queue, one way to model the probability that a particular message begins service immediately after waiting time component i is:

$$\gamma_i = \frac{1}{1 - \frac{\bar{q}}{1 - q_0}} \quad \text{or} \quad \gamma_i = \frac{1 - q_0}{\bar{q}} \qquad i>1 \tag{14}$$

The waiting time distribution under RANDOM scheduling can be computed using these values of $\{\gamma_i\}$ and equation (11). Figure 7 shows computed waiting time distributions for RANDOM scheduling for three different traffic arrival intensities and message sizes given by $\alpha=0.1$ and $\alpha=0.02$.

In the following section the results for the waiting time distributions for FCFS, LCFS and RANDOM obtained in this section will be used to discuss the impact of the scheduling discipline imposed by the protocol on the time-constrained performance of the protocol.

4. The Impact of Scheduling Disciplines on the Time-Constrained Performance of Random Access Protocols

As mentioned earlier, many time constrained applications are characterized by two important features: tolerable message loss and the constraint that messages not received at the destination station (or equivalently not beginning transmission at the sending station) within some fixed amount of time after arrival at the sending station are considered lost. Thus, message loss, as well as time delay and throughput, is an important performance measure for time-constrained applications which use random access protocols.

The time delay versus message loss tradeoff for a given random access protocol can be determined from the distribution of the message waiting times. For example, the message loss versus time delay tradeoff for FCFS scheduling and a throughput of $\rho' = .70$ is given in figure 3 - if the waiting time bound is 40 time slots, for instance, then the message loss is approximately 30%; if the time bound is increased to 100 time slots then the message loss is only 6%. Since the waiting time distributions determine the time delay versus loss tradeoff, the comparison of the time-constrained performance of two protocols requires the comparison of their waiting time distributions. In figure 8 we show sample waiting time distributions for FCFS, LCFS and RANDOM scheduling in

order to provide a quantitative example of the impact that a scheduling discipline can have on the time-constrained performance of the protocol.

The results in figure 8 indicate that for the given traffic intensity and message size, none of the three protocols is uniformly the best in the sense of minimizing loss for all possible time bounds. For small time bounds (large loss), LCFS is better than FCFS and RANDOM, while for large time bounds (small loss), FCFS is better than LCFS and RANDOM. Similar results can be found by comparing waiting time distributions from figures 3, 6 and 7. The results in figure 8 also indicate that there can be significant performance differences due to the imposed scheduling discipline. For example, for the same fixed time bound, the message loss for FCFS and LCFS can differ by as much as 20%; for the same message loss, the time bounds required by FCFS and LCFS to realize this loss can differ by as much as 100%. Clearly, the scheduling discipline imposed by a protocol greatly affects the time-constrained performance of the protocol.

Although FCFS and LCFS each perform better than the other (and RANDOM) for certain values of time delay and message loss, the question arises whether there are other scheduling disciplines which perform better than both FCFS and LCFS in such regions. Since we are interested in maximizing the probability that a message has a waiting time below some given bound, a scheduling discipline similar to minimum slack time scheduling in deterministic scheduling [Coffman 76] would seem promising. Under minimum slack time scheduling, that message with a current waiting time closest to, but not exceeding, the waiting time bound is transmitted next. This message can be selected by choosing the beginning of the initial time window to be the current time minus the waiting time bound and resolving collisions within a window on a FCFS basis. Figure 9 shows simulation values for minimum slack time scheduling for waiting time bounds of 50, 70, and 90 time slots. From simulation studies and as evident from figure 9, we have noted that minimum slack time scheduling for a specific time bound performs equally as well or better than both FCFS and LCFS in the region of the waiting time bound. However, as shown in figure 9, the increase in performance is relatively small.

Finally, it should be noted that all the various scheduling disciplines discussed so far can be implemented by the same general window random access mechanism. In practice, system characteristics such as average traffic arrival rate, loss tolerances and acceptable time delays may vary over time. Since the relative time-constrained performance of the different scheduling disciplines depends strongly on these variable system characteristics, a crucial feature of the general window mechanism which makes it particularly suitable for time-constrained applications is the capability of the single window mechanism to impose different scheduling disciplines in response to the changing system characteristics.

5. Conclusion

We have presented an adaptive random access protocol for CSMA networks which is based on a generalization of the time window mechanism. This protocol can be used to provide any of a large class of distributed scheduling disciplines based on message arrival times. We have studied the cases in which the protocol provides FCFS, LCFS and RANDOM scheduling and have presented both analytic and simulation results for the message waiting time distribution. In addition, a protocol which specifically attempts to maximize the probability that message waiting times are below a given bound was also introduced and discussed.

The performance results have demonstrated the critical impact of the scheduling discipline on the time-constrained performance of the protocol, thus indicating that the scheduling discipline imposed by a protocol should be a primary concern in the design of random access protocols for time-constrained applications in CSMA networks.

REFERENCES

[Coffman 78] E.G. Coffman.
Computer and Job Shop Scheduling.
J. Wiley and Sons, N.Y. N.Y., 1976.

[Distributed Sensor Networks 78]
Carnegie-Mellon University.
Proc. Conference on Distributed Sensor Networks, Carnegie-Mellon University, Pittsburgh, Pa., 1978.

[Gallager 78] R. Gallager.
Conflict Resolution in Random Acess Broadcast Networks.
In Proceedings of the AFOSR Workshop in Communication Theory and Applications. AFOSR, Sept, 1978.

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

[Kleinrock 76] L. Kleinrock.
Queueing Systems Volume II: Computer Applications.
J. Wiley and Sons, N.Y., N.Y., 1976.

[Lam 80] S. Lam.
A Carrier Sense Multiple Access Protocol for Local Networks.
Computer Networks 4, January, 1980.

[Molle 81] M. Molle.
Extensions and Unifications of the Multiple Access Communication Problem.
Technical Report CSD-810730, UCLA Computer Science Dept., July, 1981.

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

FIGURE 2: Average Scheduling Delay, $S(\rho')$

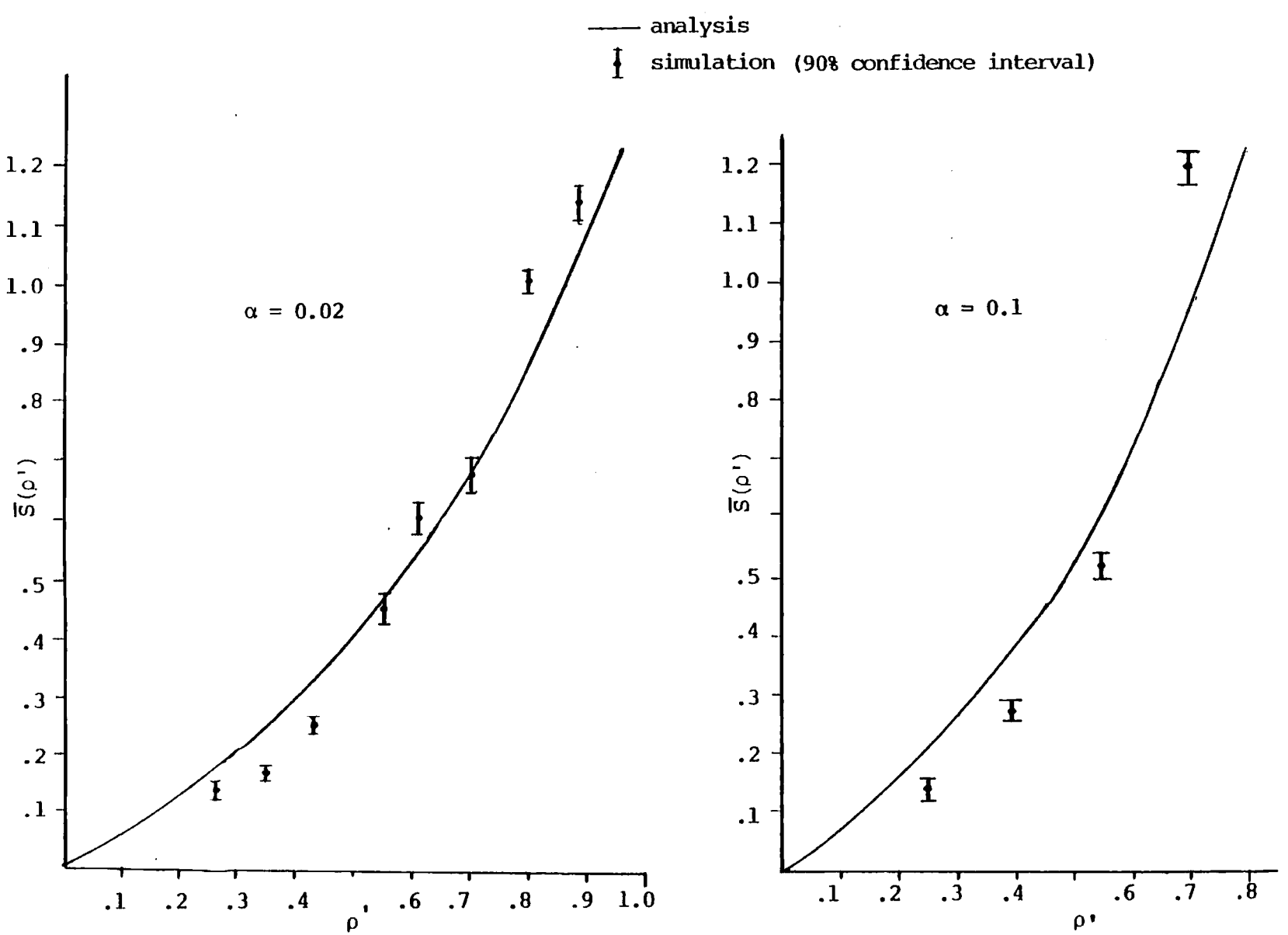
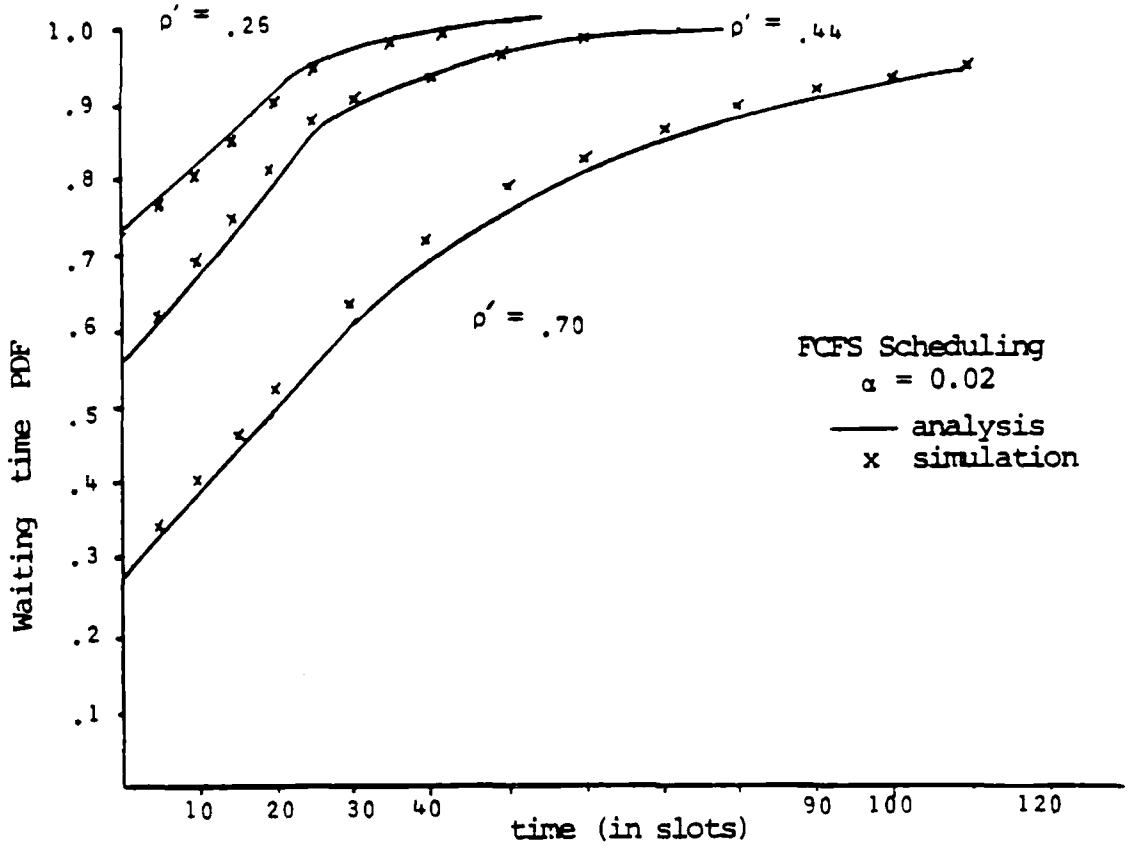


Figure 3


ρ′ = .25
ρ′ = .44
ρ′ = .70
1.0
.9
.8
.7
.6
.5
.4
.3
.2
.1
Waiting time PDF
10
20
30
40
90
100
120
time (in slots)
FCFS Scheduling
α = 0.02
analysis
x simulation

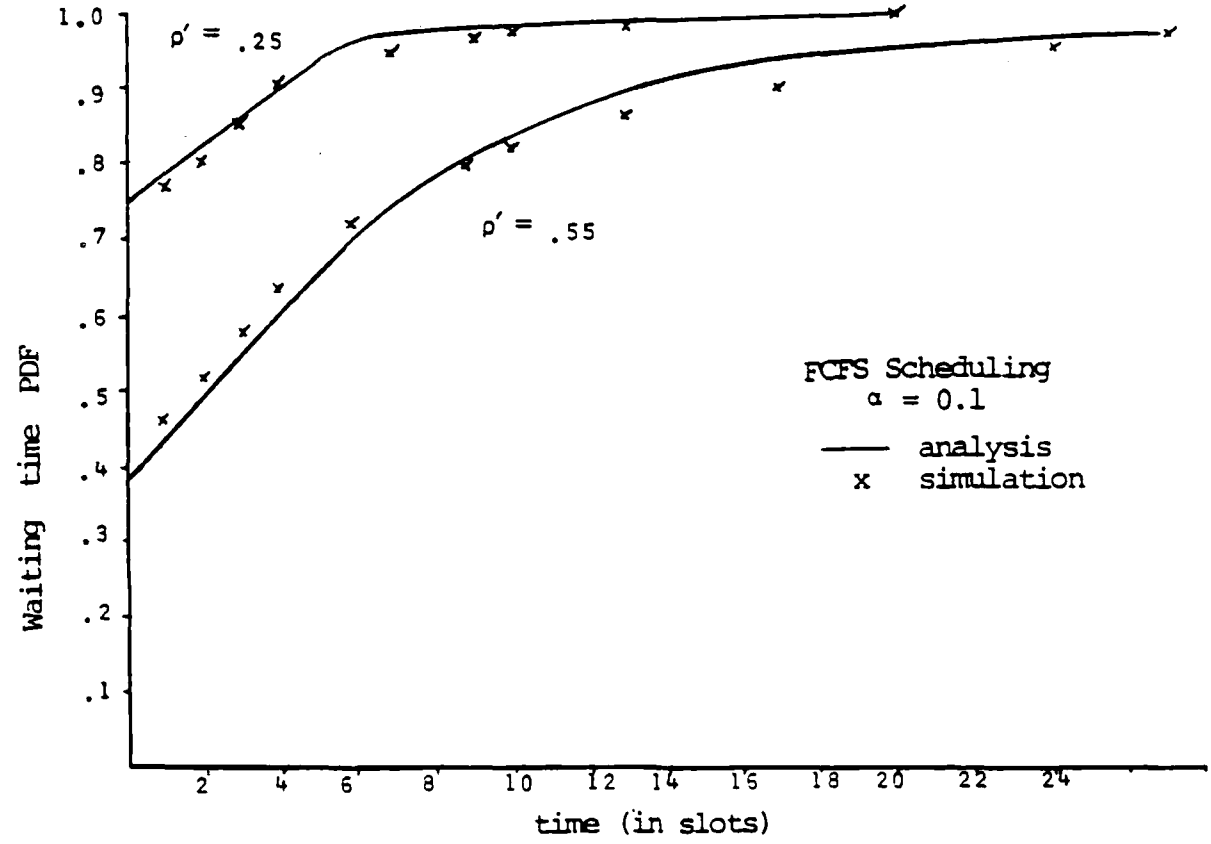
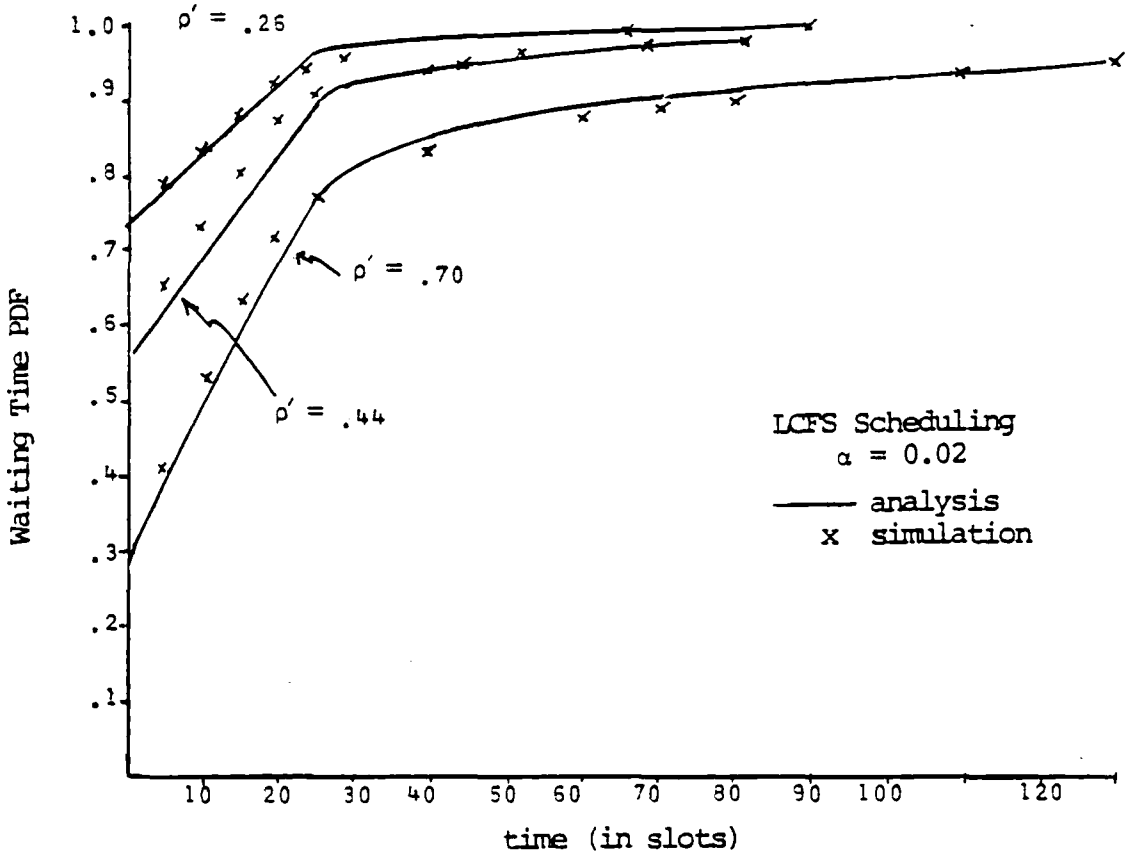

ρ′ = .25
ρ′ = .55
1.0
.9
.8
.7
.6
.5
.4
.3
.2
.1
Waiting time PDF
2
4
6
8
10
12
14
16
18
20
22
24
time (in slots)
FCFS Scheduling
α = 0.1
analysis
x simulation

Figure 6


Waiting Time PDF
1.0
.9
.8
.7
.6
.5
.4
.3
.2
.1
ρ′ = .26
ρ′ = .70
ρ′ = .44
LCFS Scheduling
α = 0.02
analysis
x simulation
10 20 30 40 50 60 70 80 90 100 120
time (in slots)

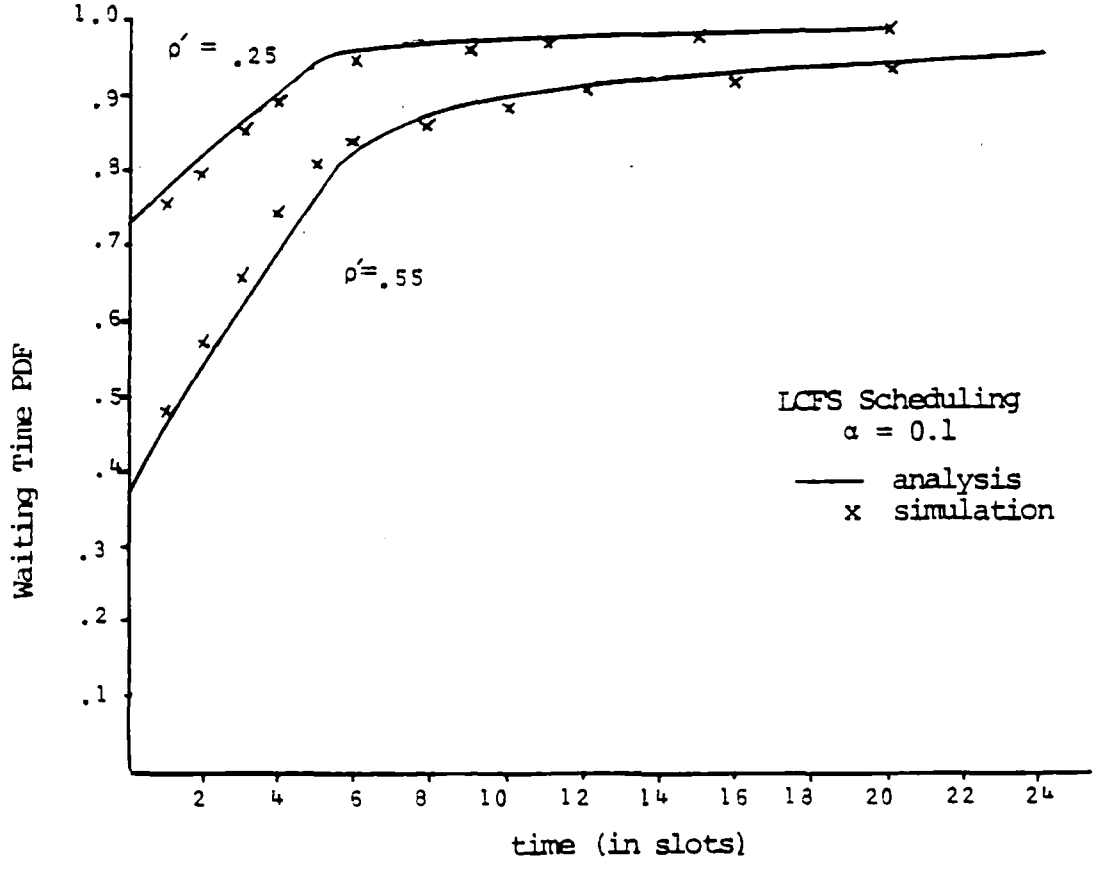
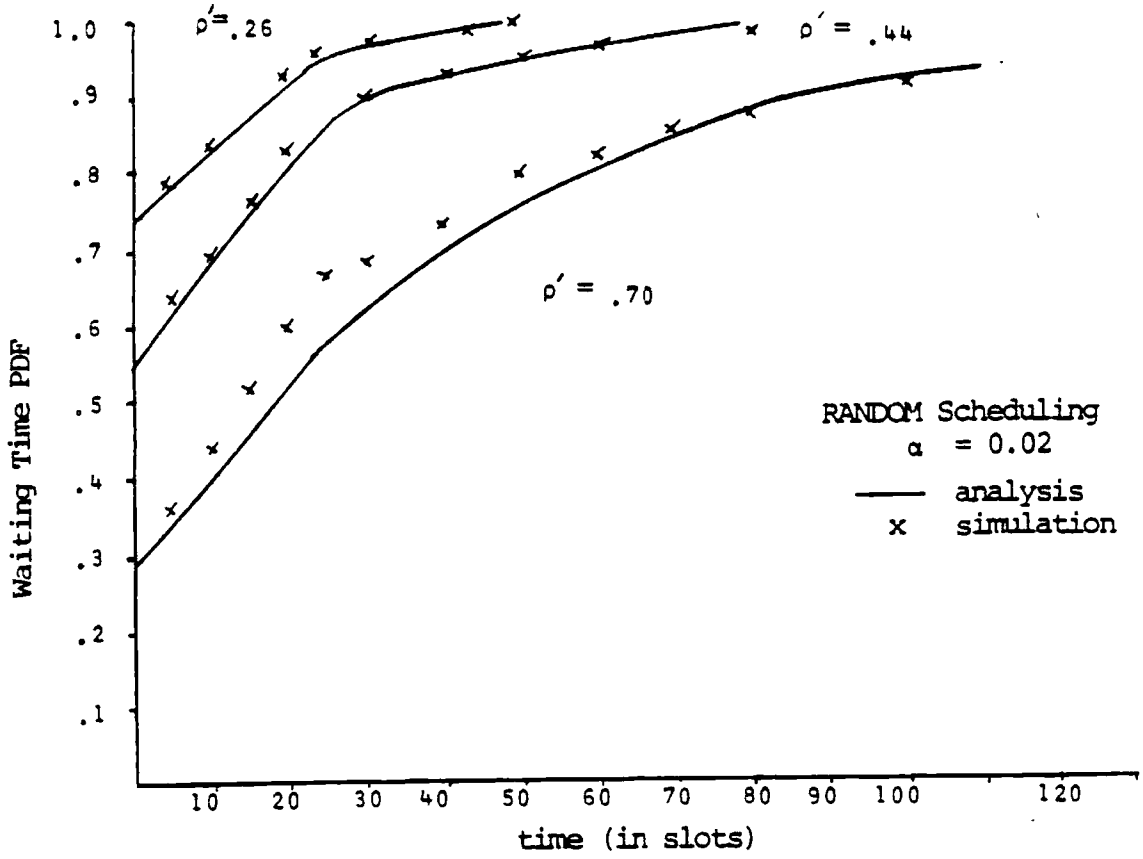

Waiting Time PDF
1.0
.9
.8
.7
.6
.5
.4
.3
.2
.1
ρ′ = .25
ρ′=.55
LCFS Scheduling
α = 0.1
analysis
x simulation
2 4 6 8 10 12 14 16 18 20 22 24
time (in slots)

Figure 7


1.0
.9
.8
.7
.6
.5
.4
.3
.2
.1
Waiting Time PDF
ρ′=.26
ρ′ = .44
ρ′ = .70
RANDOM Scheduling
α = 0.02
analysis
x simulation
10 20 30 40 50 60 70 80 90 100 120
time (in slots)

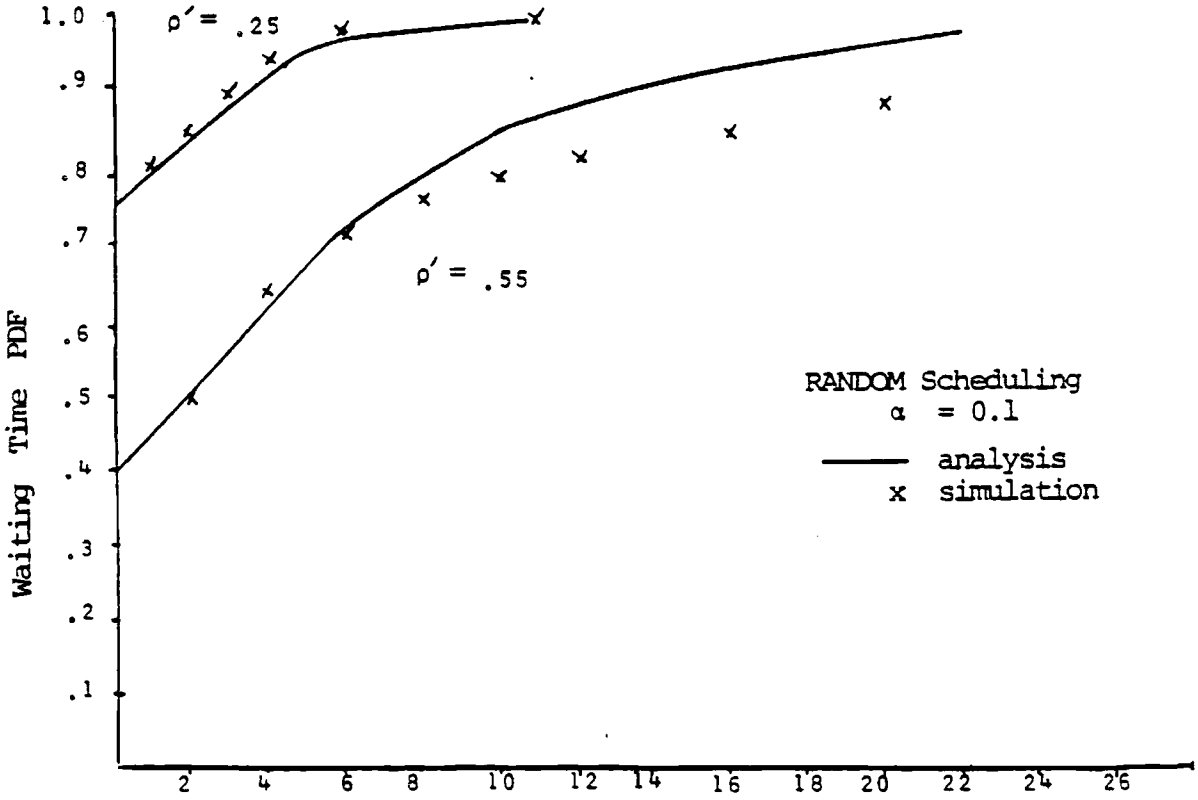

1.0
.9
.8
.7
.6
.5
.4
.3
.2
.1
Waiting Time PDF
ρ′= .25
ρ′ = .55
RANDOM Scheduling
α = 0.1
analysis
x simulation
2 4 6 8 10 12 14 16 18 20 22 24 26

figure 8

Comparison of waiting time distributions

FCFS versus LCFS versus RANDOM

$\alpha = .02$

$\rho = .70$

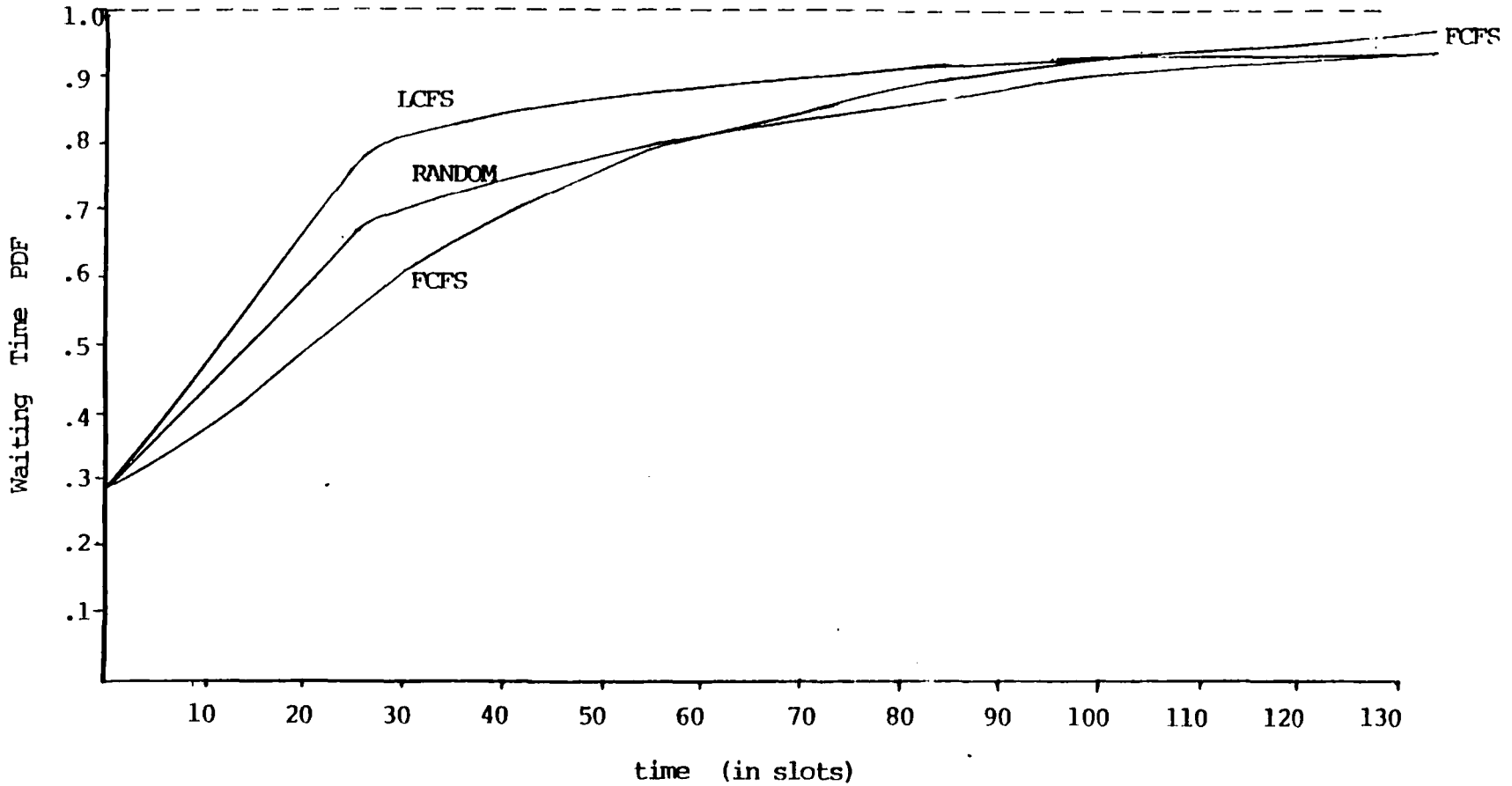
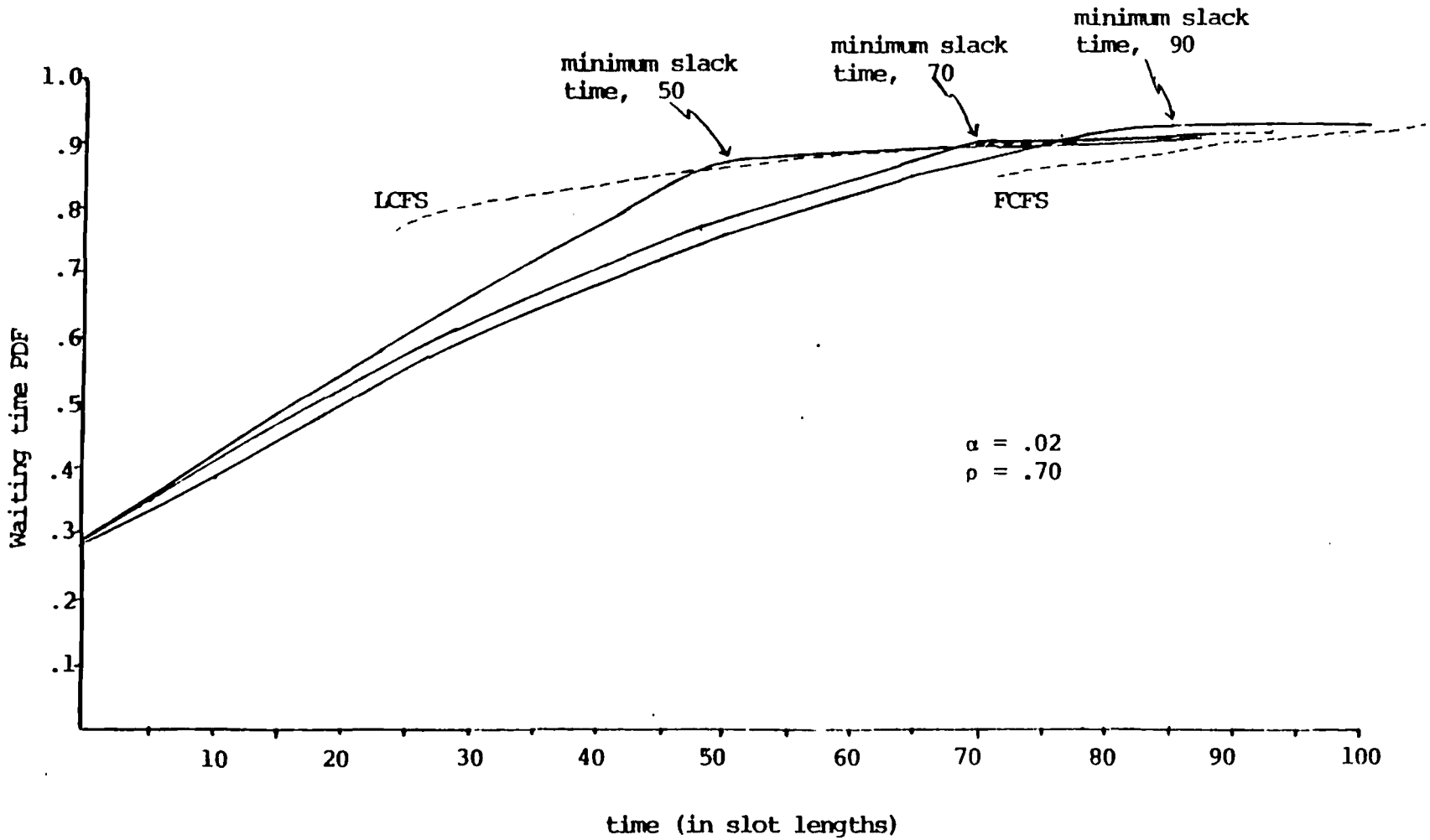


figure 9


minimum slack time, 50
minimum slack time, 70
minimum slack time, 90
LCFS
FCFS
α = .02
ρ = .70
Waiting time PDF
1.0
.9
.8
.7
.6
.5
.4
.3
.2
.1
10
20
30
40
50
60
70
80
90
100
time (in slot lengths)