The Helical Window Token Ring

FRANK R. KSCHISCHANG, STUDENT MEMBER, IEEE, AND MART L. MOLLE, MEMBER, IEEE

Abstract —A new access rule for token ring local area networks called the "helical window" token ring protocol is introduced. This protocol features the use of a window that limits the allowable messages a token-holding station may send. With the window, the operation of the protocol approaches that of a central single-server queueing system in the sense that messages are delivered in "near" first-come–first-served order on a networkwide basis. The introduction of the window also makes analysis of the network tractable. Exact analytical formulas for the capacity, and the mean, variance, and moment-generating function of the message waiting time are derived for both the continuous (infinite population) and the discrete (finite population) case. Numerical simulation is used to verify the results. Comparisons with continuous polling systems show that the imposition of the windowed access rule can lead to significant reductions in the delay variance (but at the cost of increasing the mean system time) when the traffic is heavy and/or the message transmission time is large with respect to the walk time of the ring.

I. Introduction

In a token ring network, access to the channel is controlled by a form of distributed hub polling in which permission to transmit messages rests with the current holder of a reserved symbol, called the *idle token* (or simply "token"). To transmit a message, a station must 1) wait for the arrival of the idle token from its inbound channel; 2) take control of the channel by substituting another symbol called the *busy token* (or "connector") on the outbound channel; 3) transmit its message(s), separated by busy tokens; and finally, 4) release control of the channel by regenerating another idle token on the outbound channel.

The network protocol governs the number of messages a station holding the token may send. The usual protocols may be grouped into three categories. *Ordinary service* limits each station to transmitting just one message before giving up the token. *Gated service* allows the station to transmit all messages that were ready for sending before acquisition of the token but none that became ready while the token is held. *Exhaustive service* allows each station to empty its buffer of messages before passing on the token.

Manuscript received August 10, 1987. This work was supported in part by the University of Toronto through an Open Graduate Fellowship, and in part by the Natural Sciences and Engineering Research Council of Canada under Operating Grant A5517. The material in this paper was partially presented at the Thirteenth Biennial Symposium on Communications, Kingston, ON, Canada, June 1986.

F. R. Kschischang is with the Department of Electrical Engineering, University of Toronto, Toronto, ON, Canada, M5S 1A4.

M. L. Molle is with the Computer Systems Research Institute, University of Toronto, Toronto, ON, Canada, M5S 1A4.

IEEE Log Number 8928189.

Although the usual protocols are fair in the sense that each station is treated statistically identically, usually no attempt is made to deliver messages in the order in which they arrive to the system. A station located unfavorably with respect to the token may be forced to wait while intermediate stations deliver messages out of turn.

In this paper a new access rule for local area token ring networks—which, for reasons explained later, is called *helical window* service [1]—is introduced. This protocol controls possession of the token, and hence access to the channel, with a *window*. The protocol is fair in the sense that it will deliver messages on a networkwide first-come–first-served basis, at least up to the resolution of the window size. The helical window token ring is interesting from a theoretical standpoint because the introduction of the window uncouples the message queues at the individual stations. This greatly simplifies the analysis of the protocol.

The operation of the helical window protocol is described in Section II. In Section III, the protocol is analyzed for both the continuous (infinite population) and the discrete (finite population) cases, yielding exact expressions for the message waiting time statistics. Section IV derives parameter settings for maximizing capacity or for minimizing mean delay. Section V gives the results of a computer simulation verifying the theoretical analysis, and in Section VI the helical window system is compared with the continuous polling systems analyzed in [2], [3] and finite population token rings with ordinary, gated, and exhaustive service, as analyzed in [4]. Conclusions are given in Section VII. We have also included an Appendix in which we analyze a particular class of "moving server" queueing systems; the results are used in the main portion of the paper but may also be of independent interest.

II. Description of the Helical Window Token Ring

The evolution of a ring network through time may be represented on the surface of an infinitely long hollow cylinder in space–time, with the boundary of a circular cross section of the cylinder representing the spatial extent of the ring and the longitudinal axis representing time. Events such as message arrivals and deliveries occur at discrete points on the surface of the cylinder. In a token ring network, the motion of the idle token in space and time describes a continuous curve on the cylinder surface.

An object moving around the ring at constant angular velocity would describe a helix of constant pitch in

0018-9448/89/0500-0626$01.00

space–time. (See Fig. 1.) Such a helix would partition the time axis, as viewed from any particular station, into a sequence of equal-length segments that we will call *windows*. The helical window service allows each station seeing the idle token for the kth time to transmit only those messages that arrived during the kth window. Thus a station located at position c_0 on a ring of circumference c and holding the idle token after its kth pass may transmit all messages whose arrival time t satisfies

$$\left(\frac{c_0}{c}+k-2\right)w < t \le \left(\frac{c_0}{c}+k-1\right)w \tag{1}$$

where w is the length of a window.

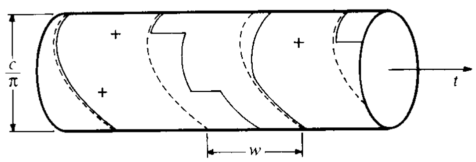


Fig. 1. Helical window token ring represented on surface of cylinder in space–time.

The movement of the idle token in the helical window token ring differs from that of the usual token ring systems. The idle token moves around the ring at one of two different speeds, depending upon its position with respect to the leading edge of the window. When the idle token is caught up with the leading window edge, it moves around the ring (with speed c/w) at the same rate at which the window advances. When it stops in space to allow a particular station to transmit, the idle token falls behind the window. When the transmitting station regenerates the idle token, it will again circulate around the ring, only at a greater speed ($\eta c/w, \eta > 1$) to catch up to the leading window edge. Once it has caught up with the window, possibly after having stopped to allow the transmission of some additional messages, the idle token reduces its speed from $\eta c/w$ to c/w. When the idle token has stopped to service a customer or is traveling around the ring at rate $\eta c/w$, we shall say that the system is backlogged.

The motion of the idle token can also be described in analogy with the classic cyclic repairman problem. In the latter problem, a repairman is confronted with a population of machines along some closed tour [2]. Here the repairman is analogous to the idle token. The window can be described by the motion of a "dispatcher," who moves around the tour at a constant rate. Whenever the dispatcher encounters a newly broken machine, he marks it with his tour iteration number. This repairman initially moves in step with the dispatcher. However, after stopping to repair the first broken machine (requiring some service time) the repairman falls behind. In an effort to catch up, he speeds up by a factor of η but still continues to stop and service broken machines, repairing, however, only those machines tagged with his own tour iteration number. If he encounters a machine tagged with a number greater than his tour iteration number, he defers repairing it until he has completed enough tours. Eventually, if the machines fail at a low enough rate, the repairman will catch up with the dispatcher again. When this happens, the repairman once again slows down to move in step with the dispatcher.

In Fig. 2, the cylinder has been "unrolled" into an infinite two-dimensional strip. Station positions are represented by the symbol l, plotted along the horizontal axis. l is periodic in the ring circumference so that, for n an integer, l and $l + nc$ correspond to the same station with message arrivals in different windows. Time t is plotted along the vertical axis. The area bounded from above (and containing) the line $t = wl/c$ and bounded from below by the line $t = w(l/c - 1)$ represents the surface of the cylinder and hence contains all message arrival points. The two boundary lines also represent the helical window edges. The motion of the idle token is described by the dotted curve. The channel is said to be busy whenever the idle token has stopped, i.e., when the idle token trajectory is vertical. When the idle token is moving, the channel is idle.

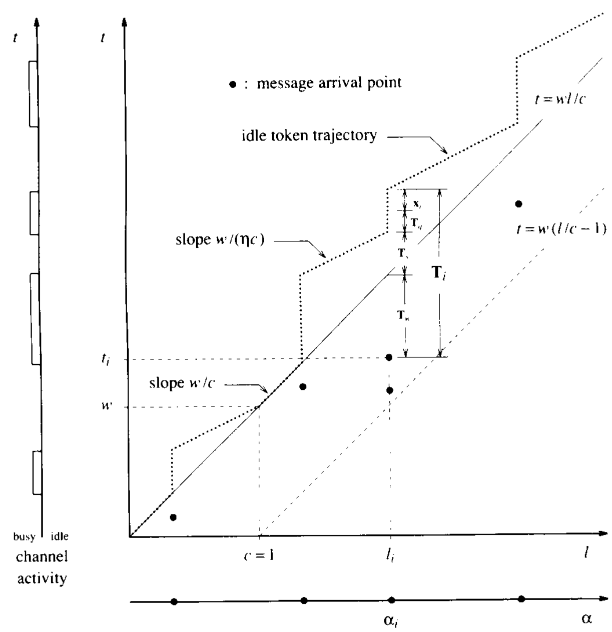


Fig. 2. Two-dimensional representation of helical window token ring.

Observe that the definition of the helical window access rule—namely, that a station holding the idle token for the jth time should transmit exactly those messages (if any) whose arrival times fall within the jth window—is independent of any particular message arrival process, message service time distribution, or the particular distribution of stations in the network. However, the *analysis* of the performance of the protocol depends critically on these factors.

Practical Considerations

In spite of our seemingly unrealistic assumptions, the implementation of the helical window access rule in a real token ring network is straightforward. As discussed in [5] with reference to the operation of virtual time carrier sense multiple access (CSMA), this type of window-based access rule is robust in the face of inaccurate time keeping at various stations.

It is not hard to modify a token ring system to allow the idle token to circulate at one of two speeds. Inactive stations (i.e., those that have crashed or been taken off line) can simply allow the data on the ring, including idle tokens, to pass by unhindered, in the usual way. Similarly, the active stations follow the usual token ring protocol, with the following exception. Following the notation of (1), assume that at time $t_{0,k}$ the idle token arrives for the kth time to a station at position c_0 on the ring. In this case, upon detecting the arrival of an idle token on the incoming channel, the station starts a "rest period" on the outgoing channel of duration

$$\max\left\{\left(\frac{c_0}{c}+k-1\right)w-t_{0,k},0\right\},$$

thus delaying the apparent arrival of the idle token until the completion of the corresponding window. Thereafter, the station continues in the usual way, either by releasing the idle token, since it now knows there will not be any messages to send in the current window, or by transmitting a busy token, the message(s) scheduled for transmission in the current window separated by busy tokens, and finally an idle token.

The implementation of "rest periods" on the outgoing channel can be done as follows. If tokens are represented as reserved bit strings [6], then the rest periods can be created using variable length tokens. Thus instead of using 1^8 and $1^7 0$ to represent idle and busy tokens, respectively, we could use $1^{7+j}01$ and $1^{7+j}00$, where j is any nonnegative integer. Notice that the minimum length token is 1 bit longer than before (or that bit stuffing, to prevent the appearance of a token in a message, must be done 1 bit earlier) and that the delay at each ring interface must be increased to two bit times. Alternatively, it is possible to create tokens that are of the same duration as a single bit time using violations of the standard Manchester encoding for data bits [7]. In this case, since the complete token fits within the 1-bit delay in the station's ring interface, it is a simple matter for the station to "remove" the idle token and output ordinary data bits for the duration of the rest period.

There are several obvious performance improvements to the above access rule, most of which are trivial to implement, but which we will not consider to simplify the analysis. These include 1) moving the rest period to the end of the transmission period (if any) and 2) avoiding the rest period altogether by taking the window size to be the minimum of the constant w and the current backlog of the algorithm at this station. Both improvements reduce the delay for those messages that are waiting to be sent when the idle token arrives "ahead of schedule" because they avoid the rest period. In addition, the delay for subsequent messages is also reduced, since the amount by which the idle token falls further behind the leading edge of the window is the difference between the (sum of the) message transmission time(s) and the length of the rest period. Unfortunately, either improvement adds significant complexity to the analysis, since each introduces a dependence between the arrival process (i.e., the number of messages at each station waiting for the next arrival of the idle token) and the state of the system (i.e., the lag of the algorithm).

III. Analysis of System Time

In this section we analyze the performance for homogeneous symmetric systems in which the number of stations may be either finite or infinite (i.e., 'continuous' polling). In our analysis we assume, without loss of generality, that the message service time $\boldsymbol{x}$ has mean $\bar{x}=1$, i.e., time is measured in units of mean message service time. Furthermore, we will assume that the ring circumference c is unity, i.e., lengths are measured in units of ring circumference.

In the discrete (or finite population) case, N stations are regularly spaced around the ring, so that the ith station is located at position $l_i=i/N$, $i\in\{0,\cdots,N-1\}$. Messages arrive independently at each station at rate G/N. The number of messages that arrive during an interval of duration t is Poisson distributed with parameter Gt/N. This implies that messages arrive independently at the system (considered as a whole) at rate G according to a Poisson process.

In the continuous case, we let $N\to\infty$ to obtain an infinite population system where stations are uniformly distributed along the ring circumference. System message arrivals form a two-dimensional Poisson process with a mean number G messages arriving per unit surface "area"[1] of the cylinder in Fig. 1. Letting $\boldsymbol{n}(A)$ represent the random number of messages that arrive in an area A on the cylinder, the probability density function of $\boldsymbol{n}$ is

$$P[\boldsymbol{n}(A)=k]=\frac{e^{-AG}(AG)^k}{k!}.$$

Moreover, messages arrive independently in disjoint regions of the cylinder.

With any collision-free protocol in equilibrium, the rate at which messages are delivered must equal the rate at which they arrive. Thus the throughput S, defined as the mean number of messages delivered per unit time, is equal to the arrival rate Gc, where c is the ring circumference. Since c is equated to unity, we have the simple result

$$S=G. \tag{2}$$

[1]Here "area" is the product of a spatial length and a temporal length.

In the following analysis we assume that the protocol operates in equilibrium so that (2) holds.

A. Decomposition of System Time

Consider the ith message arrival point (l_i, t_i) in Fig. 2. We shall define the system time $\boldsymbol{T}_i$, for this message as the total time it spends in the system. The system time consists of four components (see Fig. 2):

- $\boldsymbol{T}_{w_i}$ window delay, or time elapsed $wl_i - t_i$ between the arrival of the message and the end of the window in which the message arrived;
- $\boldsymbol{T}_{s_i}$ server delay, or time elapsed between the end of the message arrival window and the arrival of the idle token on the correct tour iteration;
- $\boldsymbol{T}_{q_i}$ queueing delay, caused by the service of messages that arrived at the same station during the same window and before the tagged message. (Notice that, with probability one, $\boldsymbol{T}_q$ is zero in the continuous system.)
- $\boldsymbol{x}_i$ message service (transmission) time, is characterized by its probability density $f_{\boldsymbol{x}}(x)$ and the corresponding moment generating function $\Phi_{\boldsymbol{x}}(s)$, defined as the Laplace transform of $f_{\boldsymbol{x}}(x)$:

$$\Phi_{\boldsymbol{x}}(s) \triangleq \int_0^{\infty} f_{\boldsymbol{x}}(x) e^{-sx}\,dx.$$

$f_{\boldsymbol{x}}(x)$ may be any positive probability density with mean $\bar{x} = 1$. To ensure that the higher moments of $\boldsymbol{T}$ exist, all derivatives of $\Phi_{\boldsymbol{x}}(s)$ at the origin must also exist. Thus

$$\boldsymbol{T}_i = \boldsymbol{T}_{w_i} + \boldsymbol{T}_{s_i} + \boldsymbol{T}_{q_i} + \boldsymbol{x}_i.$$

We assume that the service of different messages are statistically independent and independent of the operation of the network. $\boldsymbol{T}_{w_i}$ and $\boldsymbol{T}_{q_i}$ depend on the message arrival process at a single station within a single window and are independent of the operation of the ring. $\boldsymbol{T}_{s_i}$ depends only on the service times of messages encountered by the idle token before reaching the position of the ith message arrival point on the correct tour iteration. Thus since the arrivals from disjoint intervals are independent in a Poisson process, $\boldsymbol{T}_{w_i} + \boldsymbol{T}_{q_i}$, $\boldsymbol{T}_{s_i}$, and $\boldsymbol{x}_i$ are mutually independent random variables so that knowledge of their marginal density functions is sufficient to obtain a complete description of $\boldsymbol{T}_i$. And since all stations are identical, the equilibrium statistics of $\boldsymbol{T}_i$ are the same for all stations. For this reason, we shall henceforth suppress the subscripts identifying the ith message arrival, except when necessary.

B. Window Delay Plus Queueing Delay

Since the arrival process is Poisson, $\boldsymbol{T}_w$ is uniform in $[0, w)$, hence

$$f_{\boldsymbol{T}_w}(t_w) = \begin{cases} 1/w, & t_w \in [0, w) \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

is the probability density function of $\boldsymbol{T}_w$. We will temporarily define the random variable $\boldsymbol{m}$ as the number of messages that arrived before a randomly tagged message within an arrival window. Since messages arrive at rate G/N with Poisson density, the distribution of $\boldsymbol{m}$ conditional upon the window delay $\boldsymbol{T}_w$ of the tagged message is

$$\begin{aligned} P_{\boldsymbol{m}|\boldsymbol{T}_w}(i|t_w) &\triangleq P[\boldsymbol{m} = i | \boldsymbol{T}_w = t_w] \\ &= e^{-G(w-t_w)/N}\left[G(w-t_w)/N\right]^i / i!. \end{aligned} \tag{4}$$

Given that $\boldsymbol{m} = i$, we may write

$$\boldsymbol{T}_q = \sum_{j=1}^{i} \boldsymbol{x}_j \tag{5}$$

where $\boldsymbol{x}_j$ are the service times of the i messages ahead of the tagged message in the station queue. Let $f_{\boldsymbol{T}_q|\boldsymbol{m}}(t_q|i)$ be the probability density function of $\boldsymbol{T}_q$ given that $\boldsymbol{m} = i$. From Bayes' theorem, we may obtain the joint density function

$$f_{\boldsymbol{T}_q,\boldsymbol{m},\boldsymbol{T}_w}(t_q, i, t_w) = f_{\boldsymbol{T}_q|\boldsymbol{m},\boldsymbol{T}_w} \cdot P_{\boldsymbol{m}|\boldsymbol{T}_w} \cdot f_{\boldsymbol{T}_w}.$$

Observe that $f_{\boldsymbol{T}_q|\boldsymbol{m},\boldsymbol{T}_w} = f_{\boldsymbol{T}_q|\boldsymbol{m}}$ since, given $\boldsymbol{m}$, *a priori* knowledge of $\boldsymbol{T}_w$ is unnecessary to determine the distribution of $\boldsymbol{T}_q$. Thus

$$f_{\boldsymbol{T}_q,\boldsymbol{T}_w}(t_q, t_w) = \sum_i f_{\boldsymbol{T}_q|\boldsymbol{m}=i} \cdot P_{\boldsymbol{m}|\boldsymbol{T}_w} \cdot f_{\boldsymbol{T}_w}. \tag{6}$$

From the joint density of $\boldsymbol{T}_q$ and $\boldsymbol{T}_w$, we obtain the density of their sum,

$$f_{\boldsymbol{T}_w+\boldsymbol{T}_q}(\psi) = \int_{-\infty}^{\infty} f_{\boldsymbol{T}_q,\boldsymbol{T}_w}(\psi - \gamma, \gamma)\,d\gamma,$$

and the moment generating function of their sum,

$$\Phi_{\boldsymbol{T}_w+\boldsymbol{T}_q}(s) \triangleq \int_{-\infty}^{\infty} f_{\boldsymbol{T}_w+\boldsymbol{T}_q}(\psi) e^{(-s\psi)}\,d\psi.$$

Recognizing from (5) that $\Phi_{\boldsymbol{T}_q|\boldsymbol{m}=i} = (\Phi_{\boldsymbol{x}})^i$, since the message service times are independent, we obtain, after substitution of (3) and (4) into (6) and some manipulation,

$$\Phi_{\boldsymbol{T}_w+\boldsymbol{T}_q}(s) = \frac{e^{Gw(\Phi_{\boldsymbol{x}}(s)-1)/N} - e^{-sw}}{Gw(\Phi_{\boldsymbol{x}}(s)-1)/N + sw}.$$

From $\Phi_{\boldsymbol{T}_w+\boldsymbol{T}_q}(s)$, we may obtain the moments of $\boldsymbol{T}_w + \boldsymbol{T}_q$. In particular, the first moment

$$\begin{aligned} E[\boldsymbol{T}_w + \boldsymbol{T}_q] &= -\lim_{s \to 0} \frac{d}{ds} \Phi_{\boldsymbol{T}_w+\boldsymbol{T}_q}(s) \\ &= \frac{w}{2}\left(\frac{G}{N}\bar{x} + 1\right) \end{aligned}$$

where $\bar{x}$ is the mean message service time. The second moment may similarly be obtained:

$$\begin{aligned} E\left[(\boldsymbol{T}_w + \boldsymbol{T}_q)^2\right] &= \lim_{s \to 0} \frac{d^2}{ds^2} \Phi_{\boldsymbol{T}_w+\boldsymbol{T}_q}(s) \\ &= \frac{1}{3} w^2 \left[1 + \frac{G}{N}\bar{x}\left(1 + \frac{G}{N}\bar{x}\right)\right] + \frac{Gw\overline{x^2}}{2N}. \end{aligned}$$

The variance of this component of the system time is then

$$\sigma^2_{T_w+T_q}=\frac{w^2}{12}\left(1-\frac{G}{N}\bar{x}\right)^2+\frac{Gw\overline{x^2}}{2N}.$$

In the continuous case, where $T_{q_i}\equiv 0$, these results may be simplified to obtain $\Phi_{T_w}(s)=(1-e^{-sw})/sw$, $E[T_w]=w/2$ and $\sigma^2_{T_w}=w^2/12$, respectively.

C. Server Delay

In a moving server queueing system [5], [8], a server traverses an infinite tour along which he encounters customers requiring service. As shown in the Appendix, it is trivial to obtain the delay characteristics for a moving server system, given the corresponding results for a related "synthetic" queueing problem, provided the latter problem can be solved. To obtain the moment generating function of T_s, we will show that T_s may be viewed as the waiting time in a moving server queueing system.

The moving server queueing system is obtained from the helical window token ring by projecting the message arrival points, idle token trajectory, and window edge helix from Fig. 2 onto an "arrival time" (α-)axis, obtained by scaling the l-axis by a factor of w/c (changing the units from distance to time). A point (l,t) in Fig. 2 is mapped to the point wl on the α-axis. (Recall that $c\equiv 1$.) Thus a message arrival at (l_i,t_i) in the helical window token ring is mapped to a customer arrival at $\alpha_i\triangleq wl_i$ in the moving server queueing system. The result of this projection is a moving server system with Poisson arrivals at intensity G in which the server (the projection of the idle token) "moves" along the arrival time line at unit speed (the rate of advance of the projection of the window edge helix), if he is not backlogged; otherwise, he is either stopped (serving a customer) or moving at the accelerated rate η.[2] The waiting time in the moving server system is identical to the server delay T_s, in the helical window system.

Analyses of the mean delay in moving server queueing systems based on our transformational approach have appeared in [5], [8]. However, the presentation in the Appendix extends the method to show that our approach holds for the distribution of delay and not just its mean. The basic idea is that any moving server queueing system can be transformed into a corresponding central single-server synthetic queueing system by inflating each service time by a factor of β to account for the server walking time, where

$$\beta\triangleq\frac{\eta}{\eta-1}. \tag{7}$$

If we can solve for the waiting time $\tilde{W}$, in the synthetic queueing system then we will also have solved for the waiting time in the moving server queueing system, since

$$T_s=\frac{\tilde{W}}{\beta}. \tag{8}$$

Continuous Case: In the continuous case, the synthetic queueing system is an $M/G/1$ queue with arrival rate G, service time $\tilde{x}=\beta x$ and server utilization $\tilde{\rho}=\beta G\bar{x}$. Hence the moment generating function of $\tilde{W}$, obtained from the Pollaczek–Khinchin transform equation [9], is

$$\Phi_{\tilde{W}}(s)=\frac{s(1-\beta G\bar{x})}{s-G+G\Phi_x(\beta s)}.$$

The moments of $\tilde{W}$ may be obtained either by differentiating $\Phi_{\tilde{W}}(s)$ at the origin, i.e.,

$$E[\tilde{W}^j]=\lim_{s\to 0}(-1)^j\frac{d^j}{ds^j}\Phi_{\tilde{W}}(s)$$

or from the Takács recurrence formula [9, p. 201] and the first moment of $\tilde{W}$,

$$E[\tilde{W}]=\beta\frac{\overline{x^2}}{2}\frac{\beta G}{1-\beta G}.$$

Using (8), we may obtain all the statistics of T_s from the statistics of $\tilde{W}$. Specifically,

$$\Phi_{T_s}(s)=\frac{s(1-\beta G\bar{x})}{s-\beta G+\beta G\Phi_x(s)}.$$

is the moment generating function of T_s, and

$$E[T_s]=\frac{\beta G\overline{x^2}}{2(1-\beta G\bar{x})}$$

$$\sigma^2_{T_s}=E^2[T_s]+\frac{\beta G\overline{x^3}}{3(1-\beta G\bar{x})}$$

are the mean and variance of T_s, respectively.

Discrete Case: In the analysis of the discrete case, we will follow the same approach as in the continuous case and transform the system into a synthetic single server queueing system. However, now customer arrivals can only occur when the window boundary (or, in the terminology of Section II, the "dispatcher") is at a station position, which happens periodically once every ω time units, where

$$\omega\triangleq w/N. \tag{9}$$

Because of our assumption of homogeneous stations generating Poisson traffic, customer interarrival times in the moving server system have a geometric distribution, with probability

$$p=1-e^{-Gw/N} \tag{10}$$

of encountering a customer at each station position. Furthermore, each "customer" in the synthetic queueing system no longer corresponds to an individual message, but to a nonempty bulk arrival of messages that were generated at a single "busy" station during a single window. Letting a_i be the number of messages contained in the ith such

[2]It is interesting to observe that this moving server system is identical to the one that is used in [5], [8] to estimate the initial delay (until the first transmission attempt) for virtual time CSMA. Indeed, if Fig. 2 is modified by mapping message arrival points to the end of the window (i.e., $(l_i,t_i)\to(l_i,\alpha_i)$), we obtain the identical mapping from arrival times to transmission times as shown in [5, fig. 2].

bulk, we have

$$P[\boldsymbol{a}_i = j] \triangleq P_a(j) = \frac{e^{-Gw/N}(Gw/N)^j}{(1-e^{-Gw/N})j!}, \qquad j \ge 1.$$

Since we assume that the message lengths are independent, the conditional service time distribution for the ith customer given that $\boldsymbol{a}_i = j$, $f_{\boldsymbol{b}_i|\boldsymbol{a}_i}(b_i|j)$, may be found as the j-fold convolution of the message length distribution.

Solution of the synthetic queueing system in the general case is difficult because it combines a geometric arrival process in discrete time with a general service time distribution in continuous time. However, we shall restrict the service time for each message so that its expanded service time βx is a multiple of ω, the discrete time unit for the arrival process, so that the synthetic system reduces to a geometric/G/1 queue in discrete time. This restriction is equivalent to requiring that the backlog of the token always be cleared at a station position.

To solve for the waiting time $\tilde{\boldsymbol{W}}$ in the synthetic system, it is convenient to use ω as the elementary time unit, so that

$$\boldsymbol{T}_s = \omega \tilde{\boldsymbol{W}}/\beta. \tag{11}$$

In this case, the transformed service times satisfy

$$\tilde{\boldsymbol{x}} = \frac{\beta \boldsymbol{x}}{\omega} = k, \qquad k \in \{1,2,\cdots\}$$

and are characterized by a nonnegative discrete probability distribution with discrete moment generating function $\tilde{X}(z)$, which satisfies

$$\tilde{X}(e^{-\omega s/\beta}) = \Phi_x(s).$$

Let $\tilde{\boldsymbol{b}}$, with discrete moment generating function $\tilde{B}(z)$, be the service time for a customer in the synthetic system (corresponding to the total service time for all packets in the bulk arrival at a busy station). Then

$$\begin{aligned}\tilde{B}(z) &\triangleq \sum_{i=0}^{\infty} P[\tilde{\boldsymbol{b}} = i] z^i \\ &= \sum_{j=1}^{\infty} P_a(j) \sum_{i=0}^{\infty} P\left[\tilde{\boldsymbol{b}} = i | \boldsymbol{a} = j\right] z^i \\ &= \sum_{j=1}^{\infty} P_a(j) \tilde{X}^j(z) \\ &= \frac{e^{Gw(\tilde{X}(z)-1)/N} - e^{-Gw/N}}{1-e^{-Gw/N}}.\end{aligned} \tag{12}$$

The mean transformed service time of such a bulk is

$$E[\tilde{\boldsymbol{b}}] = \frac{GwE[\tilde{\boldsymbol{x}}]/N}{1-e^{-Gw/N}}. \tag{13}$$

Since the synthetic system is a geometric/G/1 queueing system with probability p of a customer arrival at a (discrete) arrival point, service time $\tilde{\boldsymbol{b}}$, and server utilization $\tilde{\rho} = pE[\tilde{\boldsymbol{b}}]$, the moment generating function for the waiting time is given by [10, p. 230]

$$\tilde{W}(z) = \frac{(1-\tilde{\rho})(1-z)}{p\tilde{B}(z) - z + 1 - p}.$$

Substituting from (10), (12), and (13), we obtain

$$\tilde{W}(z) = \frac{(1-GwE[\tilde{\boldsymbol{x}}]/N)(1-z)}{e^{Gw(\tilde{X}(z)-1)/N} - z}.$$

Now, using the fact that $\Phi_{T_s}(s) = \tilde{W}(e^{-\omega s/\beta})$, $\tilde{\boldsymbol{x}} = \beta \boldsymbol{x}/\omega$, and $\omega = w/N$, we obtain

$$\Phi_{T_s}(s) = \frac{(1-\beta G\bar{x})(1-e^{-sw/N\beta})}{e^{(\Phi_x(s)-1)Gw/N} - e^{-sw/N\beta}}.$$

From $\Phi_{T_s}(s)$, we may obtain all the moments of $\boldsymbol{T}_s$. In particular, the mean is

$$E[\boldsymbol{T}_s] = \frac{\beta G\overline{x^2}}{2(1-\beta G\bar{x})} - \frac{Gw\bar{x}}{2N},$$

and the second moment is

$$\begin{aligned}E\left[(\boldsymbol{T}_s)^2\right] = 2E^2[\boldsymbol{T}_s] + \frac{\beta G\overline{x^3}}{3(1-\beta G\bar{x})} \\ - \frac{w}{N\beta}(1-2\beta G\bar{x})\left[E[\boldsymbol{T}_s] + \frac{Gw\bar{x}}{3N}\right].\end{aligned}$$

Thus the variance of $\boldsymbol{T}_s$ is

$$\begin{aligned}\sigma_{T_s}^2 = E^2[\boldsymbol{T}_s] + \frac{\beta G\overline{x^3}}{3(1-\beta G\bar{x})} \\ - \frac{w}{N\beta}(1-2\beta G\bar{x})\left[E[\boldsymbol{T}_s] + \frac{Gw\bar{x}}{3N}\right].\end{aligned}$$

Notice that the expression for the mean and variance of $\boldsymbol{T}_s$ are increasing functions of N and that they reduce to our previous results for the continuous case as $N \to \infty$.

D. System Time

Let us now obtain the statistics for the system time $\boldsymbol{T}$. Since $\boldsymbol{x}$, $\boldsymbol{T}_s$, and $\boldsymbol{T}_w + \boldsymbol{T}_q$ are independent random variables, we have

$$\Phi_T(s) = \begin{cases} \Phi_x(s)\dfrac{(1-e^{-sw})(1-\beta G\bar{x})}{sw[s-\beta G+\beta G\Phi_x(s)]}, & N = \infty \\ \Phi_x(s)\dfrac{(e^{Gw(\Phi_x(s)-1)/N} - e^{-sw})(1-\beta G\bar{x})(1-e^{-sw/N\beta})}{(Gw(\Phi_x(s)-1)/N + sw)(e^{(\Phi_x(s)-1)Gw/N} - e^{-sw/N\beta})}, & N < \infty \end{cases} \tag{14}$$

$$E[\boldsymbol{T}] = \bar{x} + \frac{w}{2} + \frac{\beta G\overline{x^2}}{2(1-\beta G\bar{x})} \tag{15}$$

and

$$\sigma_T^2 = \sigma_x^2 + \frac{w^2}{12}\left[1 - \frac{2G\bar{x}}{N}\left(1 - \frac{1}{N\beta}\right)\right] + \left[\frac{\left(\overline{x^2}\right)^2}{4}\frac{\beta G}{1-\beta G\bar{x}} + \frac{\overline{x^3}}{3}\right]\frac{\beta G}{1-\beta G\bar{x}}. \quad (16)$$

E. Population Invariance of the Mean Delay

Observe that (15), the expression for the mean system time, is independent of N, the number of stations in the system. This implies that the system capacity is also unaffected by N, and we obtain the same capacity in both the discrete and continuous cases.[3] That this invariance should hold can readily be seen from the following constructive argument.

Clearly, the window delay $\boldsymbol{T}_w$ and service time $\boldsymbol{x}$ are unaffected by N; therefore, to show the invariance it suffices to show that $\zeta \triangleq E[\boldsymbol{T}_s + \boldsymbol{T}_q]$ is the same in both cases. Notice that, providing a steady state exists, ζ can be found from any sample execution trace as

$$\zeta = \lim_{K\to\infty}\frac{1}{K}\sum_{i=1}^{K}(\delta_i - \alpha_i)$$

where $\alpha_i \triangleq wl_i$ and δ_i represent the times at which the ith message observes the completion of the window and enters into service, respectively. From each sample execution trace for the continuous case we can construct a corresponding execution trace for the discrete case with N stations by mapping $(l_i, \alpha_i) \to (l_i', \alpha_i')$ in the following way. Each arrival position $l_i \in [j/N, (j+1)/N)$ is mapped to the discrete arrival position $l_i' = j/N$, corresponding to a station location. The corresponding window completion time α_i is mapped to $\alpha_i' = wl_i'$. The service order in the discrete system is preserved by the mapping. Due to the uniformity of the Poisson arrivals, the mapping reduces α_i by $w/(2N)$ on average. The key step is to observe that this mapping also reduces δ_i by the same amount on average, thus leaving ζ unaffected. To see this, we proceed in two stages. First, we note that the average starting time of each "busy period" (during which the token is continuously backlogged) is also reduced by $w/(2N)$ on average, due to the mapping of the arrival time of its first message to a discrete arrival time. Provided the message service times satisfy the restriction required for the discrete analysis—that each inflated service time be an integer multiple of the discrete time unit $\omega \triangleq w/N$—the ending time of the busy period will also be shifted by the same amount on a average. Thus distinct busy periods in the continuous case will remain nonoverlapping in the discrete case. Next, we see that the average difference between the start of the busy period in which the ith message arrives and the start of its own service is unaffected by the mapping. This result follows because, in both cases, this difference represents the sum of the service times for all messages that are served before the ith message within the same busy period (which is preserved) and of the time for the token to advance (at rate η/w) from the position at which the busy period started to the location of the ith message (which is the same on average). This completes the proof of the invariance.

[3]Note that this result is a mathematical idealization. In an actual implementation, increasing the system population increases the ring circumference c due to the necessary bit delay at each station.

IV. Optimizing System Performance

Two parameters may be tuned in the helical window token ring, namely w and η. From Section III-A, clearly only the server delay $\boldsymbol{T}_s$ depends on η, and we see by induction from (A.2) in the Appendix that $\boldsymbol{T}_{s_k}$ is a monotonically nonincreasing function of η for every k. Thus, for fixed w, we should always choose η as large as possible, subject to the physical constraint that the maximum speed of the idle token is limited to v revolutions of the ring per unit time, i.e.,

$$\eta = \frac{w}{1/v}. \quad (17)$$

Thus it remains to find the "optimal" value of w.

The system capacity C is defined to be the supremum over all attainable values of throughput for which the expected message system time is finite. From (15), we see that the mean system time has a singularity at $G = 1/\beta\bar{x}$; hence, from (2) and (7) and using the fact that $\bar{x}$ is normalized to unity, the capacity of the helical window token ring is

$$C = 1 - \frac{1}{\eta}. \quad (18)$$

Thus, for the mean system time to be bounded, we require from (18) that

$$\eta > \frac{1}{1-G}, \quad (19)$$

where it is assumed that $0 \le G < 1$.

If the optimization criterion is to maximize capacity, then we should let η, and hence w, grow arbitrarily large. However, recall that the fairness of the access rule (i.e., that the order of message transmissions is approximately global first-come–first-served, up to the resolution of the window size) depends on choosing a small value for w; thus we see that fairness and high capacity are conflicting goals.

Finally, if the optimization criterion is to minimize the mean system time without regard to fairness, we must find values of η and w satisfying (17) and (19) such that (15) is minimized. A global minimum is attained at $(\eta, w) =$

(η_0, w_0), where

$$\eta_0 = \frac{1 + \left(vG\overline{x^2}\right)^{1/2}}{1 - G\bar{x}}$$

$$w_0 = \frac{1 + \left(vG\overline{x^2}\right)^{1/2}}{v(1 - G\bar{x})}. \tag{20}$$

V. Simulation Results

Fig. 3 is a plot of mean system time versus arrival rate for the helical window system with Poisson arrivals as analyzed in Section III. The service time was fixed at unity and the maximum token speed v, was fixed at $\eta/w = 3$.

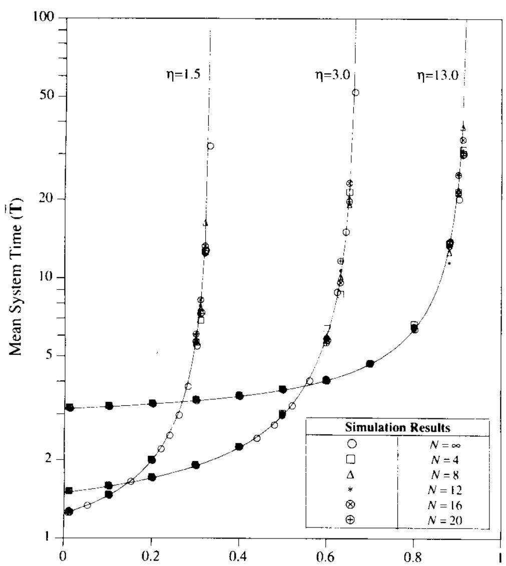
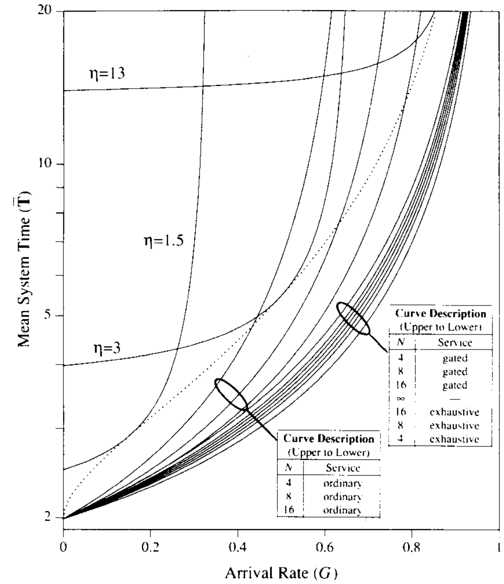


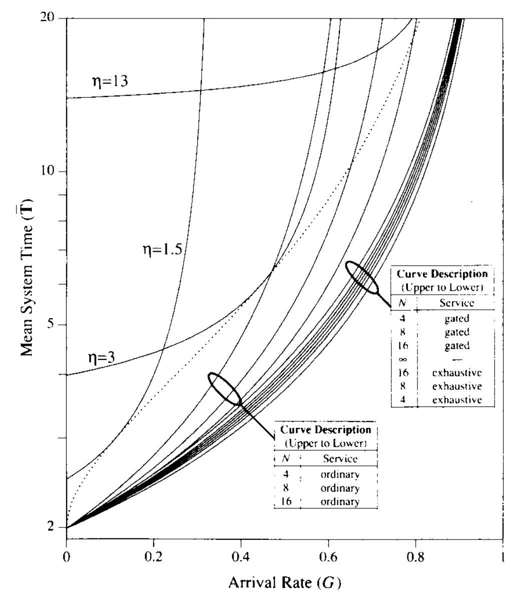
Fig. 3. Comparison of analysis and simulation of mean system time versus arrival rate for $v = \eta/w = 3$.

Curves for $\eta = 1.5$, $\eta = 3$, and $\eta = 13$ are plotted. Analogous curves are included in Fig. 4, except that the maximum token speed v has been reduced to 0.5. Observe that at low arrival rates, the mean system time is dominated by the window delay since, for large values of η, the system time is almost proportional to $1/v$. At arrival rate values approaching the system capacity (which is independent of v for a given value of w), the mean system time is dominated by the server delay, caused by the token backlog. The mean delay at low arrival rates may be decreased by decreasing w, but at the expense of a lower system capacity.

A computer simulation of the helical window token ring was performed, and simulation results for both the infinite population and various finite populations are plotted in Fig. 3. The simulation values agree well with the analytically obtained values. This is not surprising since no approximation was made in the analysis, i.e., the results are exact.



(a)



(b)

Fig. 4. Comparison of mean system system time performance of helical window token ring with token rings under ordinary, gated, and exhaustive service. Dotted curve displays performance of helical window system using optimal parameter settings. (a) Constant service time. (b) Exponentially distributed service time.

VI. Comparisons

In this section, the system time characteristics of the helical window token ring are compared with symmetric finite population token rings as presented in [4] and continuous (infinite population) polling systems as presented in [2], [3].

The mean system time (waiting time + service time) for a symmetric token ring network is given by [4]

$$E[T_{N,o}] = \frac{G\overline{x^2} + \frac{1}{v}\left(1 + \frac{G\bar{x}}{N}\right)}{2\left(1 - G\left(\bar{x} + \frac{1}{vN}\right)\right)} + \bar{x}$$

for ordinary service, and

$$E\left[T_{N,g,e}\right] = \frac{G\overline{x^2} + \frac{1}{v}\left(1 \pm \frac{G\bar{x}}{N}\right)}{2(1 - G\bar{x})} + \bar{x}$$

for gated (+) and exhaustive (−) service, respectively.[4] As $N \to \infty$, $E[T_N]$ converges to

$$E[T_\infty] = \frac{G\overline{x^2} + \frac{1}{v}}{2(1 - G\bar{x})} + \bar{x}, \tag{21}$$

a result also obtained by Ferguson [3].

Fig. 4(a) is a plot of mean system time versus, arrival rate for constant service times and $v = 0.5$. The corresponding plot for exponentially distributed service times is shown in Fig. 4(b). The curves labeled $\eta = 1.5$, 3, and 13 display the performance of the helical window system. The dotted curve represents the performance of the helical window system using the optimal values of η given by (20). Curves representing the performance of finite population symmetric token rings with ordinary, gated, and exhaustive service for $N = 4$, 8, and 16 are also plotted. The performance of the infinite population (continuous polling) system is also shown.

Fig. 4 shows that the mean system time of the helical window system is greater than the continuous polling system and finite population token rings with gated or exhaustive service. This result is not too surprising since the token does not always travel at maximum speed but moves slower when it is not backlogged. This increases the average walk time and hence the mean system time. Observe, however, that the mean system time of the helical window system may be less than the finite population token ring with ordinary service. This result is not surprising, since the helical window system admits the possibility of serving more than one arrival per station in a window. When the message arrival rate is high, so that the probability of encountering more than one message per station in a window is high, a smaller mean system time results.

Limiting Behavior

In Fig. 5 the ratio of the mean system times for the helical window system and the continuous polling system

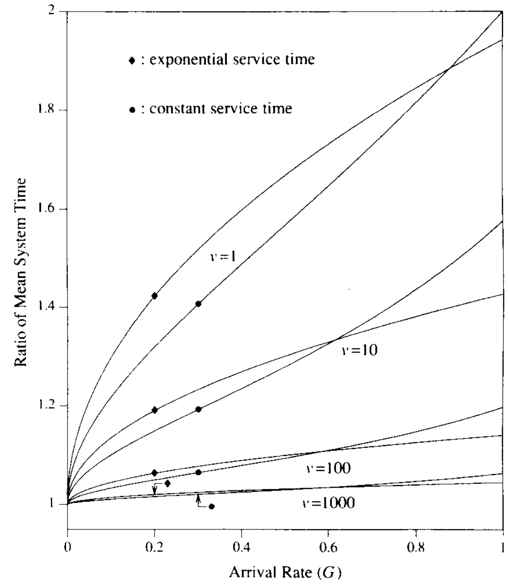


Fig. 5. Ratio of mean system time in helical window system and mean system time in continuous polling system with both constant and exponentially distributed service time.

are plotted versus arrival rate for various values of v. Both constant service times and exponential service times are shown. The mean system time of the helical window system is obtained by substituting the optimal values of η and w given by (20).

We observe that the mean delay of the helical window system approaches that of the continuous polling system as v becomes large. Indeed, in the limit as $v \to \infty$, we obtain (21) from (15). This behavior is to be expected since, in the limit, both systems become $M/G/1$ queues. The difference between the systems, however, is that the helical window system will provide service on a first-come–first-served basis, while the continuous polling system, in general, will not [2], [3]. Since the variance of delay is minimized with the first-come–first-served queueing discipline [11], we would expect the variance of the helical window system to be lower than that of the continuous polling system, for large values of v.

Fig. 6 is a plot of the ratio of the variance of system time in the helical window system and the variance of system time in the continuous polling system against message arrival rate, for various values of v. The variance of system time for the helical window system is obtained by using the values of η and w given in (20), which optimize mean system time, and not the variance. The variance of system time for the continuous polling system is given by [3]

$$\sigma^2_{T_\infty} = \sigma_x^2 + \frac{G\overline{x^3}}{3} + \frac{(3 - G\bar{x})\left\{\left(G\overline{x^2}\right)^2 + \left(G\overline{x^3}\right)(G\bar{x})(1 - G\bar{x})\right\}}{3(2 - G\bar{x})(1 - G\bar{x})^2} + \frac{(1/v)^2 + 6(1/v)G\overline{x^2} - 3\left(G\overline{x^2}\right)^2}{12(1 - G\bar{x})^2}.$$

[4] We have omitted a term of $1/2v$ in these expressions that explicitly accounts for the mean propagation delay from source to destination, assuming that the destinations are uniformly distributed around the ring. Instead, following [2], we have implicitly included this component in the service time.

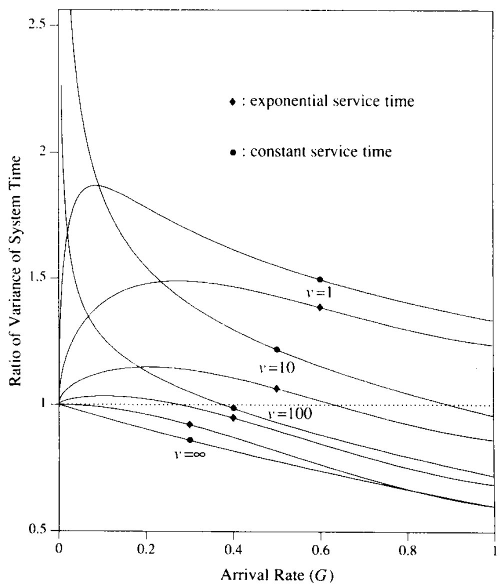


Fig. 6. Ratio of variance of system time in helical window system and variance of system time in continuous polling system with both constant and exponentially distributed service time.

We see that as v increases (and the walk time of the ring decreases), the variance of system time in the helical window system indeed becomes smaller than that of the continuous polling system, especially for large message arrival rates.

VII. Conclusion

A new access rule for token ring local area networks has been described and analyzed. The access rule employs a window to determine which messages the token-holding station may send. The window has two major effects.

1) The introduction of the window improves the fairness of the system over the usual gated and exhaustive service disciplines. This is because the helical window access rule makes an explicit attempt to deliver messages on a systemwide first-come–first-served basis, at least to the extent of the resolution of the window.

2) The introduction of the window uncouples the queues at the various stations. This decoupling leads to an exact analytical characterization of the message system time statistics for two important special cases, using only the tools of elementary queueing theory.

Exact formulas for the mean, variance, and moment generating function of the message system time were obtained for both the continuous and the discrete case. Higher moments are readily available from the moment generating function. Computer simulation of the helical window token ring showed close agreement between the simulation statistics and the values predicted by the analysis.

The helical window token ring network was compared with the usual token rings with ordinary, gated, and exhaustive service [4], and (infinite population) continuous polling systems [2], [3]. The results of the comparison are as follows.

1) The mean message system time is greater in the helical window system than in the token ring networks with gated and exhaustive service or the continuous polling system. However, for some combinations of system parameters, and especially at high message arrival rates, the mean message system time in the helical window system may be smaller than the corresponding delay in the token ring network with ordinary service.

2) For small ring walk times and/or large message arrival rates, the variance of message system time in the helical window system may be smaller than the corresponding variance in the continuous polling system.

Acknowledgment

The authors would like to thank Mr. Benoit Champagne for his help during the early stages of the writing of this paper, and the anonymous referees for suggesting various improvements. The first author would also like to thank Professor S. Pasupathy for his support and encouragement, and for providing an environment in which this work was possible.

Appendix

In a moving server queueing system, a server traverses an infinite tour along which he encounters customers requiring service. A particular class of moving server queueing systems is of interest in the analysis of several systems [5], [12], [13] including the helical window token ring. In this class, the server moves along the tour at one of two different rates depending upon his position with respect to some constantly advancing reference point. When the server is caught up with the reference point, he advances in step with it; when he is behind (because he had to stop to serve a customer) he speeds up by a factor of $\eta, \eta > 1$, to catch up. In what follows, we shall normalize rates so that the speed of the reference point is unity.

The random variable of particular interest in these systems is the difference between the time the advancing reference point encounters a customer requiring service (which we will call the arrival time) and the time the server encounters that customer. We shall refer to this random variable as the moving server delay $\tilde{W}$. In this Appendix we will show that, by a suitable transformation of the moving server system, an ordinary single server queueing system is obtained in which the waiting time is proportional to the moving server delay.

Consider the sequence of customer waiting times $W^{(1)}, W^{(2)}, \cdots$, in an ordinary first-come–first-served single server queueing system. Assume that $x^{(k)}$ and $A^{(k)}$ are the service time for the kth customer and the interarrival time between the $(k-1)$st and kth customers, respectively, the distributions of which are unimportant for our present discussion. Clearly,

$$W^{(k)} = \max\{W^{(k-1)} + x^{(k-1)} - A^{(k)}, 0\}, \qquad (A.1)$$

since the kth customer enters service at either the $(k-1)$st customer's departure time or at his own arrival time, whichever occurs last.

Now consider a corresponding sequence of moving server delays $\tilde{W}^{(1)}, \tilde{W}^{(2)}, \cdots$, in a moving server queueing system. Clearly, $\tilde{W}^{(k)} = 0$ if and only if the moving server has caught up to the reference point when he encounters the kth customer. Otherwise, the kth customer must wait until the $(k-1)$st customer's departure time (so the server can start moving again), and then for enough additional time for the server, walking at rate η, to reach the kth customer's arrival time, namely, $A^{(k)}/\eta$. Thus

$$\tilde{W}^{(k)} = \max\{\tilde{W}^{(k-1)} + x^{(k-1)} - A^{(k)} + A^{(k)}/\eta, 0\}. \quad \text{(A.2)}$$

Comparing (A.1) and (A.2), we see that waiting times in a moving server system contain an extra term corresponding to "scan time" overhead, where a customer's entry into service may be delayed even though the server is not offering service to any customer. Unlike most models of queueing systems with vacations, this overhead is proportional to (rather than independent of) the corresponding customer's interarrival time.

Define

$$\beta \triangleq \frac{\eta}{\eta - 1} \quad \text{(A.3)}$$

to be a constant that depends on the moving server's speedup factor η. Multiplying both sides of (A.2) by β and recognizing that $-A^{(k)} + A^{(k)}/\eta = -A^{(k)}/\beta$, we have that

$$\beta\tilde{W}^{(k)} = \max\{\beta\tilde{W}^{(k-1)} + \beta x^{(k-1)} - A^{(k)}, 0\}. \quad \text{(A.4)}$$

Notice that after a change of variable from $\beta\tilde{W}^{(\cdot)}$ to $W^{(\cdot)}$, (A.4) has exactly the same form as (A.1) and thus represents the relation between successive customer waiting times in a *synthetic queueing system*. The synthetic queueing system is an ordinary queueing system that has exactly the same sequence of interarrival times $A^{(1)}, A^{(2)}, \cdots$ as the moving server system, but each customer's service time has been proportionally increased from $x^{(k)}$ to $\beta x^{(k)}$, respectively, compared to the moving server system, and there is no scan time overhead. Since $W^{(1)} = \tilde{W}^{(1)} \triangleq 0$, the above observation implies that

$$\tilde{W}^{(k)} = W^{(k)}/\beta \quad \text{(A.5)}$$

must hold for all k: and since this correspondence holds for each customer taken individually, it must hold for the mean and distribution of the waiting time in the two systems as well. Hence (A.5) allows us to compute the statistics of waiting time in the moving server queueing system from those of the synthetic queueing system.

The increase in service times that results from transforming a moving server system into a corresponding synthetic queueing system has an interesting physical interpretation. Recall that while the server stops to offer service to some customer, he is being left further behind by the reference point at rate unity; whenever the server is moving at rate η, he is gaining on the reference point at the rate of $\eta - 1$. Thus, to maintain his relative position with respect to the reference point over the long run, the system must satisfy a "global balance" condition, which states that the server must spend $1/(\eta - 1)$ time units moving at the accelerated rate for every time unit spent serving a customer. One way to accomplish this would be to impose a "local balance" condition, where we define the kth customer's service time to be

$$\tilde{x}^{(k)} = x^{(k)} + \frac{x^{(k)}}{\eta - 1} = \beta x^{(k)},$$

the sum of his actual service time together with enough scan time overhead for the server to regain his former position with respect to the reference point.[5] In the synthetic system, we assume that a customer's entire service time is served without interruption, while in the moving server system, the component corresponding to scan time overhead is preempted if the server encounters subsequent customers in his walk.

Note also that since no assumptions about service or interarrival time distributions were made, the above method—where the analysis of a moving server system is transformed into the corresponding synthetic queueing problem—applies equally well to discrete and continuous time systems.

References

[1] M. L. Molle, "An old problem with a new twist—The Helical Window token ring," in *Proc. 13th Biennial Symp. Commun.*, Queen's Univ., Kingston, ON, Canada, June 1986, pp. B.5.1–B.5.3.

[2] E. G. Coffman, Jr., and E. N. Gilbert, "A continuous polling system with constant service times," *IEEE Trans. Inform. Theory*, vol. IT-32, pp. 584–591, July 1986.

[3] M. J. Ferguson, "Computation of the variance of the waiting time for token rings," *IEEE J. Sel. Areas Commun.* vol. SAC-4, pp. 775–782, Sept. 1986.

[4] H. Takagi, *Analysis of Polling Systems*. Cambridge MA: MIT Press, 1986.

[5] M. L. Molle and L. Kleinrock, "Virtual time CSMA: Why two clocks are better than one," *IEEE Trans. Commun.*, vol. COM-33, pp. 919–933, Sept. 1985.

[6] A. S. Tanenbaum, *Computer Networks*. Englewood Cliffs, NJ: Prentice-Hall, 1981.

[7] J. L. Hammond and P. J. O'Reilly, *Performance Analysis of Local Computer Networks*. Reading, MA: Addison-Wesley, 1986.

[8] M. L. Molle, "Modelling the delay in 'moving server' random access protocols," in *Current Advances in Distributed Computing and Communications*, Y. Yemini, Ed. Computer Science Press, 1987, pp. 17–32.

[9] L. Kleinrock, *Queueing Systems, vol. I: Theory*. New York: Wiley-Interscience, 1975.

[10] J. J. Hunter, *Mathematical Techniques of Applied Probability, vol. 2—Discrete Time Models: Techniques and Applications*. New York: Academic, 1983.

[11] J. F. C. Kingman, "The effect of queue discipline on waiting time variance," *Proc. Cambridge Phil. Soc.*, vol. 58, pp. 163–164, 1962.

[12] G. C. Polyzos and M. L. Molle, "Delay analysis of a window tree conflict resolution algorithm in a local area network environment," presented at the ACM SIGMETRICS' 87 Conf., Banff, AB, Canada, May 1987.

[13] M. L. Molle and M. W. Wu, "Prioritized Virtual time CSMA: Head-of-the-line priorities without channel overhead," in *Proc. IEEE Int. Conf. Commun.*, Chicago, IL, June 1986, pp. 35.4.1–35.4.6.

[5] We interpret scan time overhead as a component of service time because the server appears to be busy (even though he is not engaged in service to any customer) while he is moving at rate η, since waiting times are strictly positive for all customers who arrive to the system while the server is moving at rate η and are identically zero for all customers who arrive while the server is moving at rate unity.