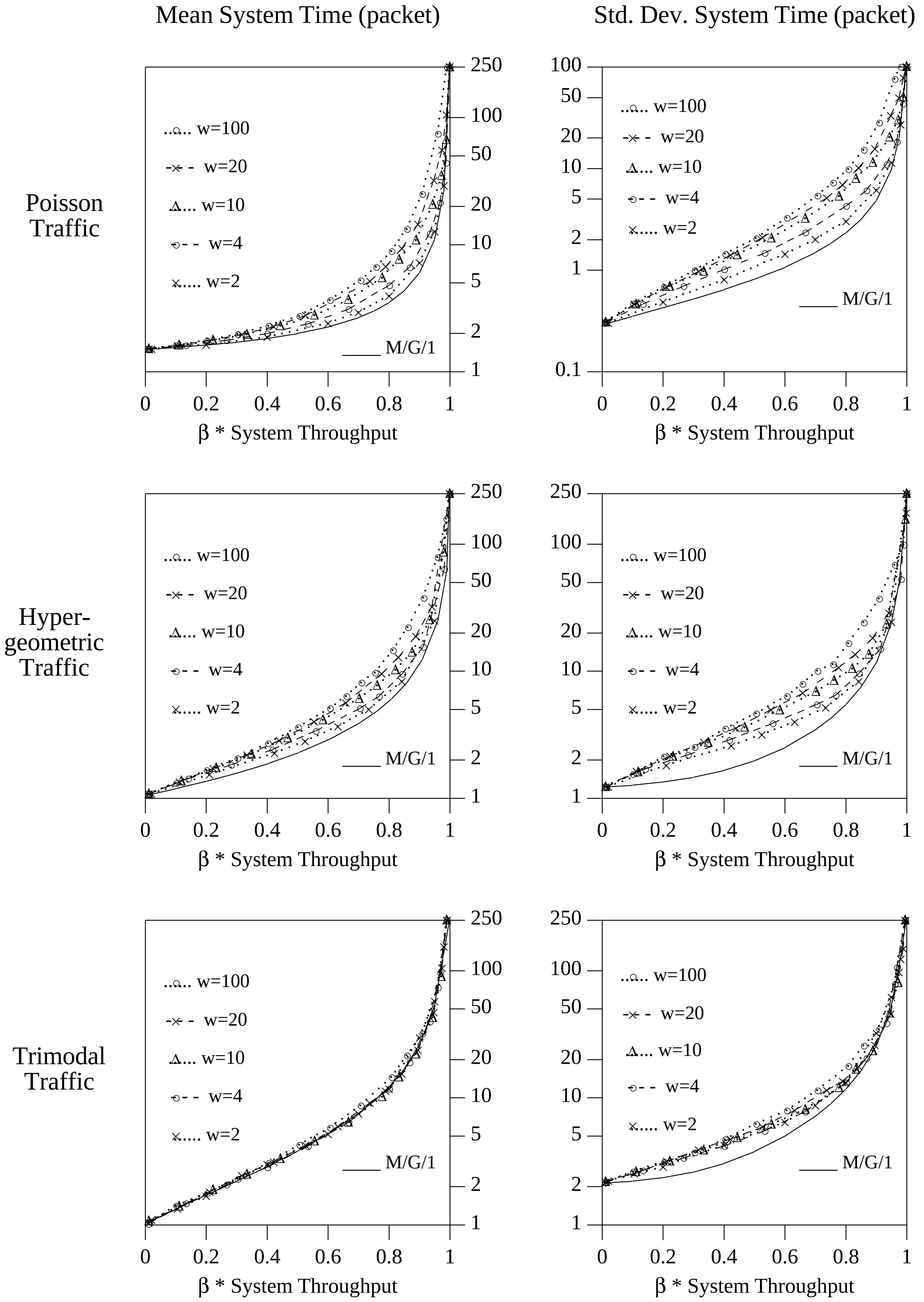

Figure 5.3: Packet Delay vs (β * Throughput) for Varying Window Size.

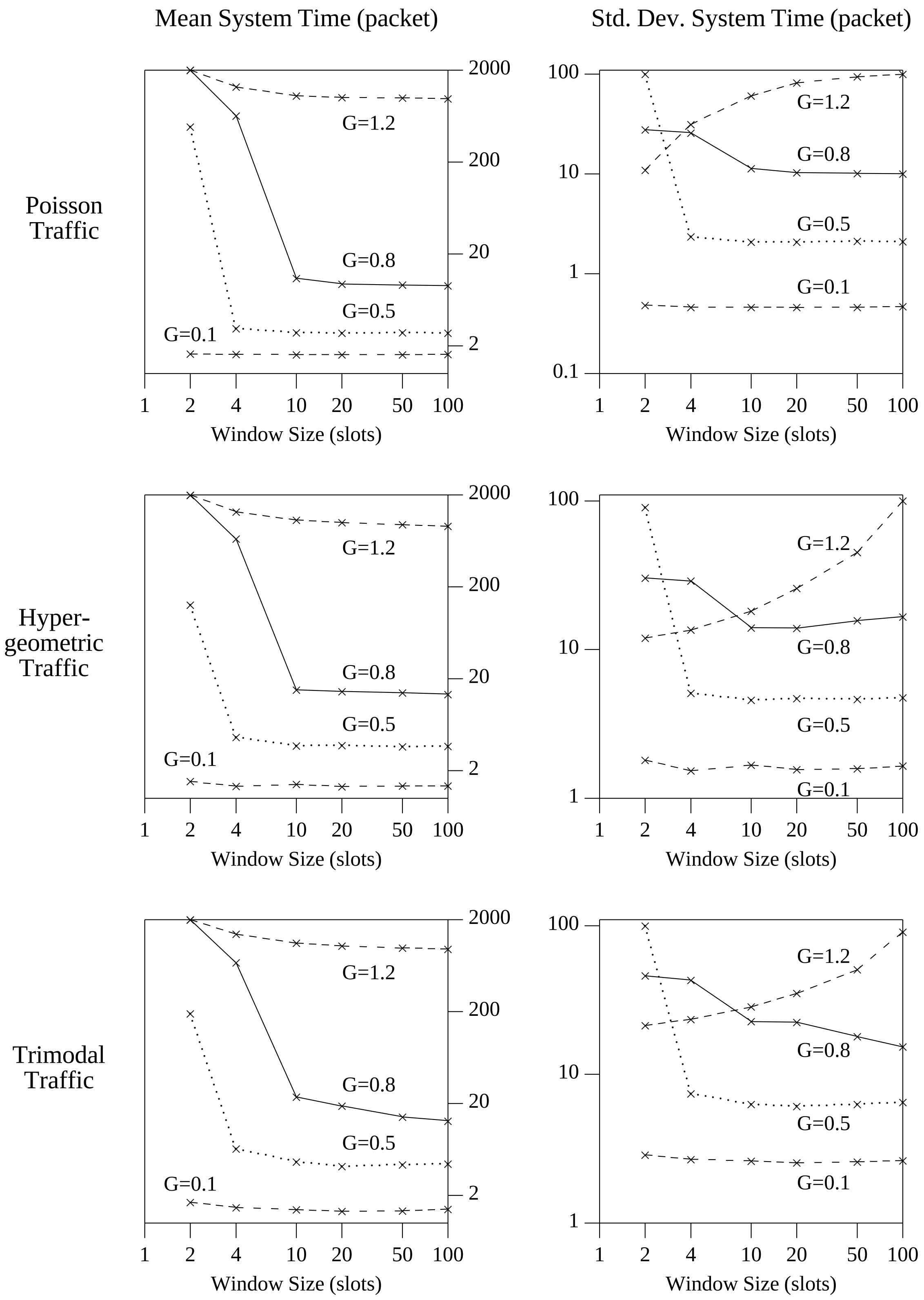

Figure 5.4: Packet Delay vs Window Size for Constant System Loads.

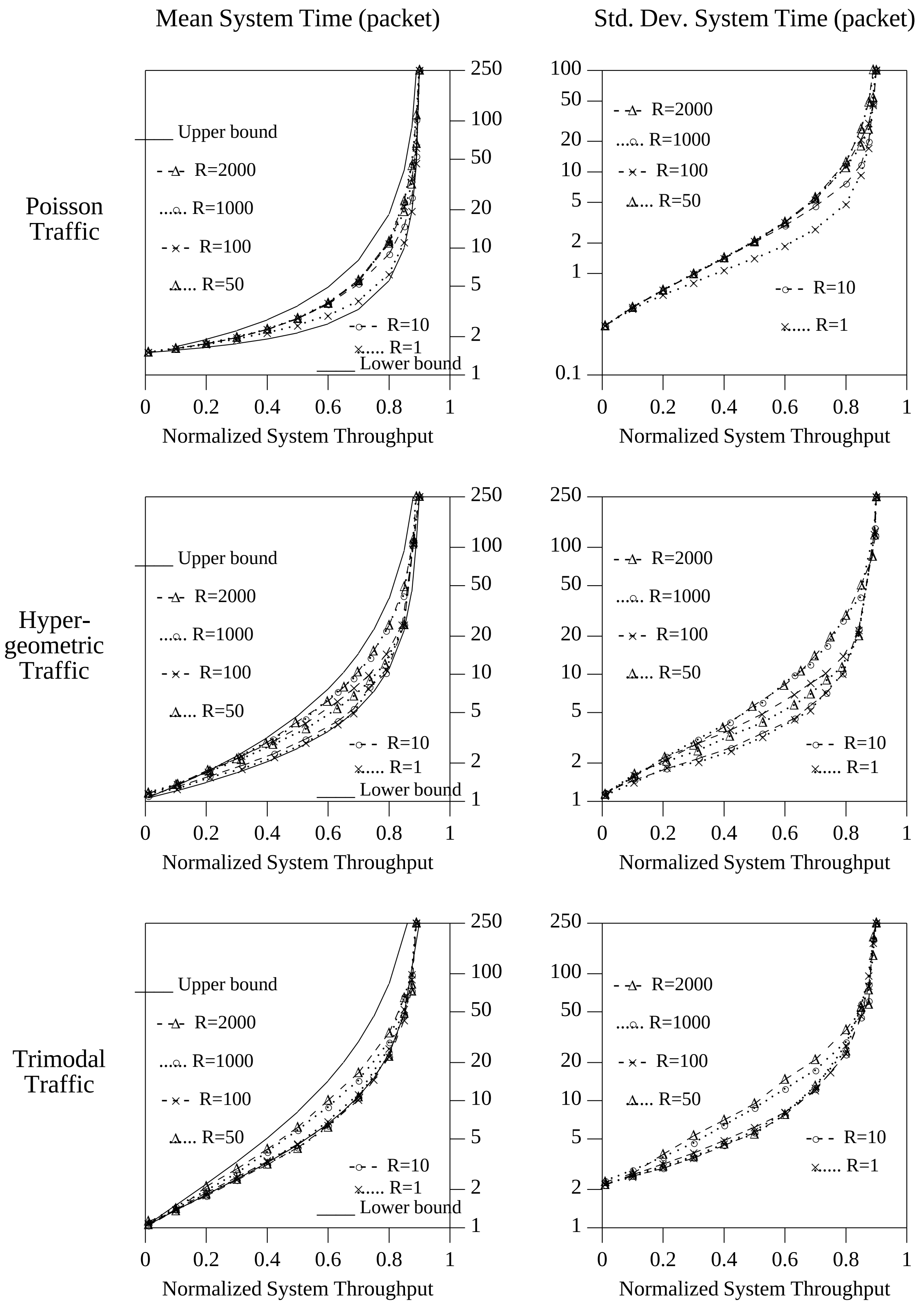

Figure 5.5: Packet Delay vs Throughput for Varying Normalized Ring Walk-Time.

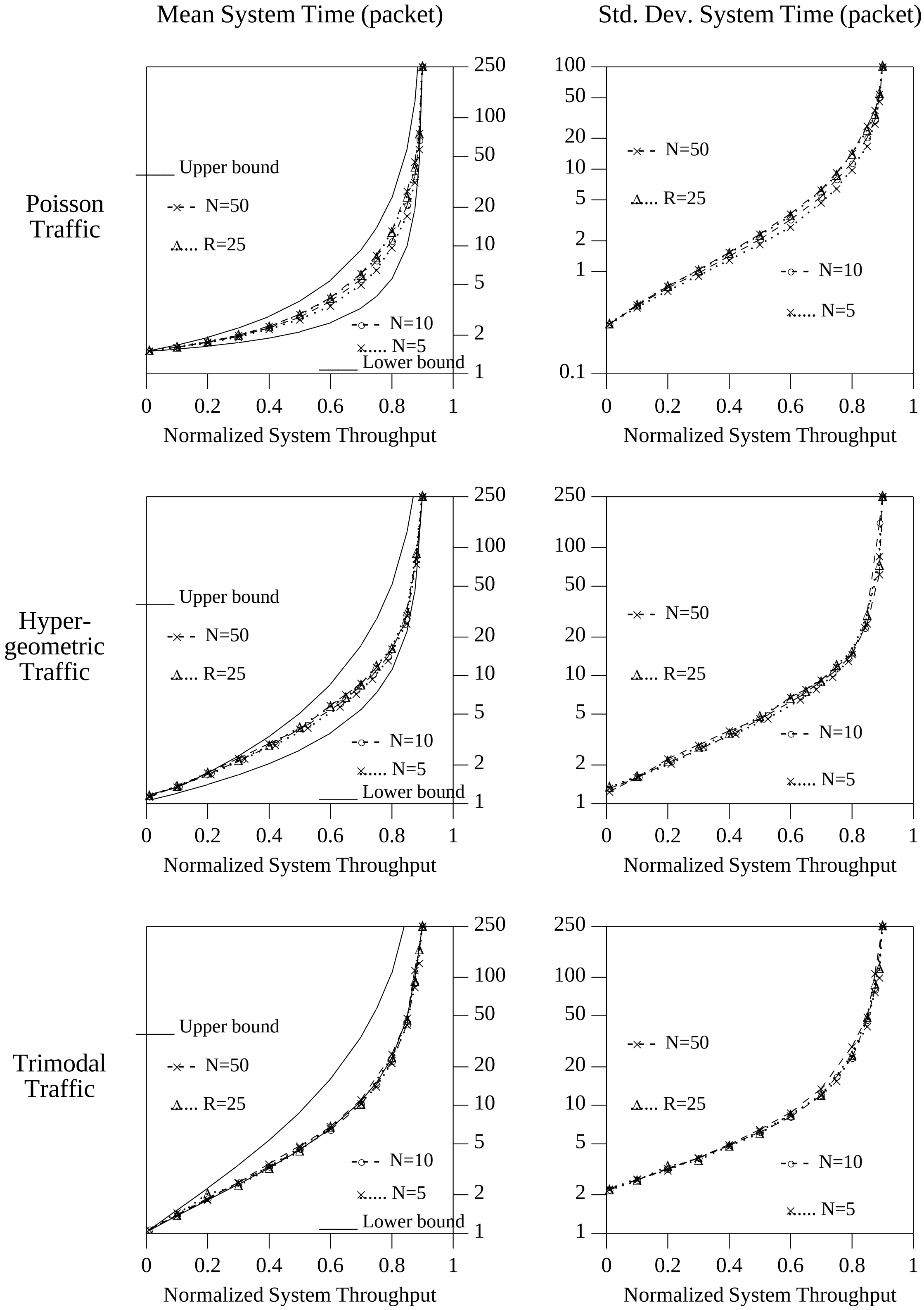

Figure 5.6: Packet Delay vs Throughput for Varying Number of Stations.

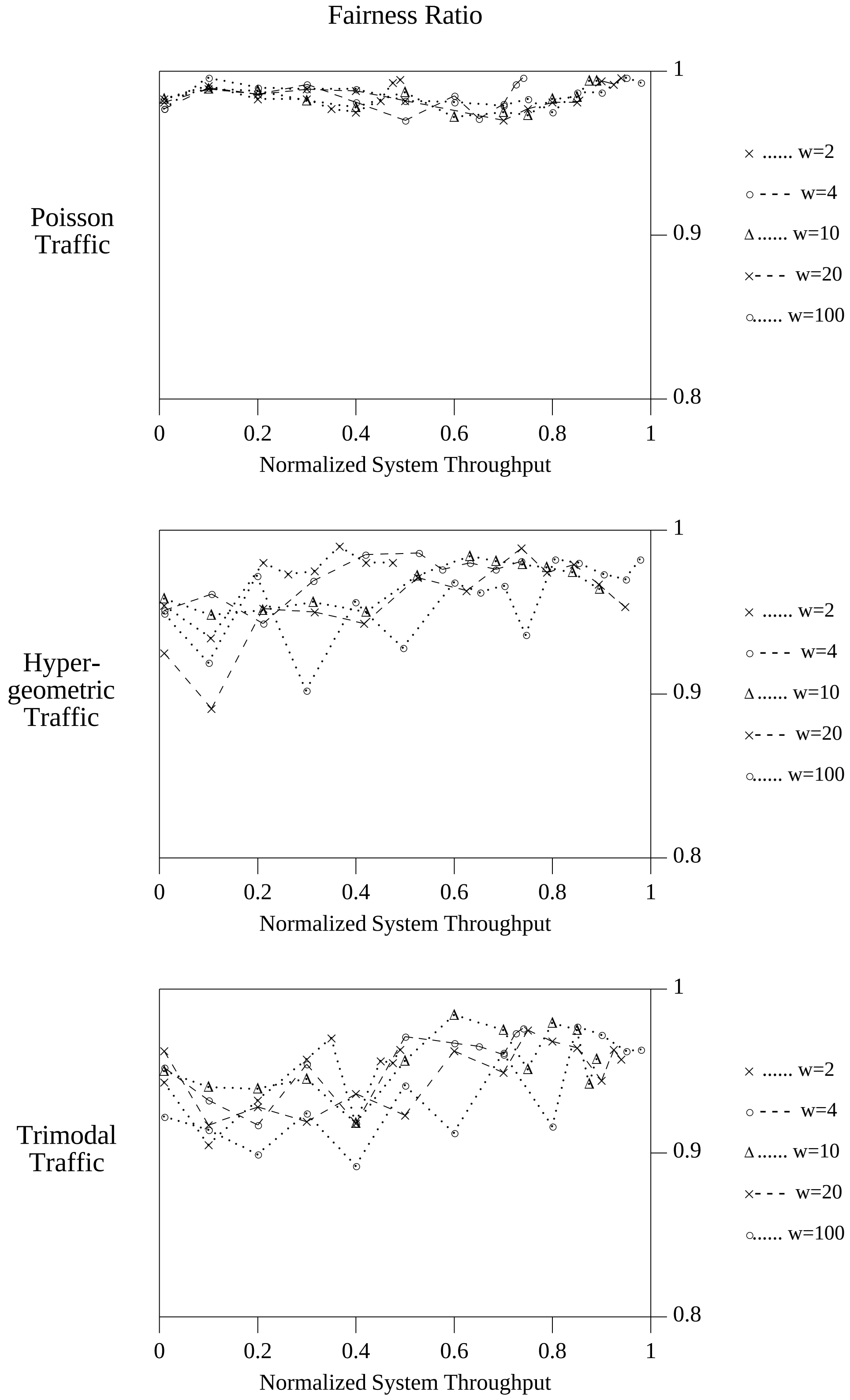

Figure 5.7: Fairness Ratio for Varying Window Size.

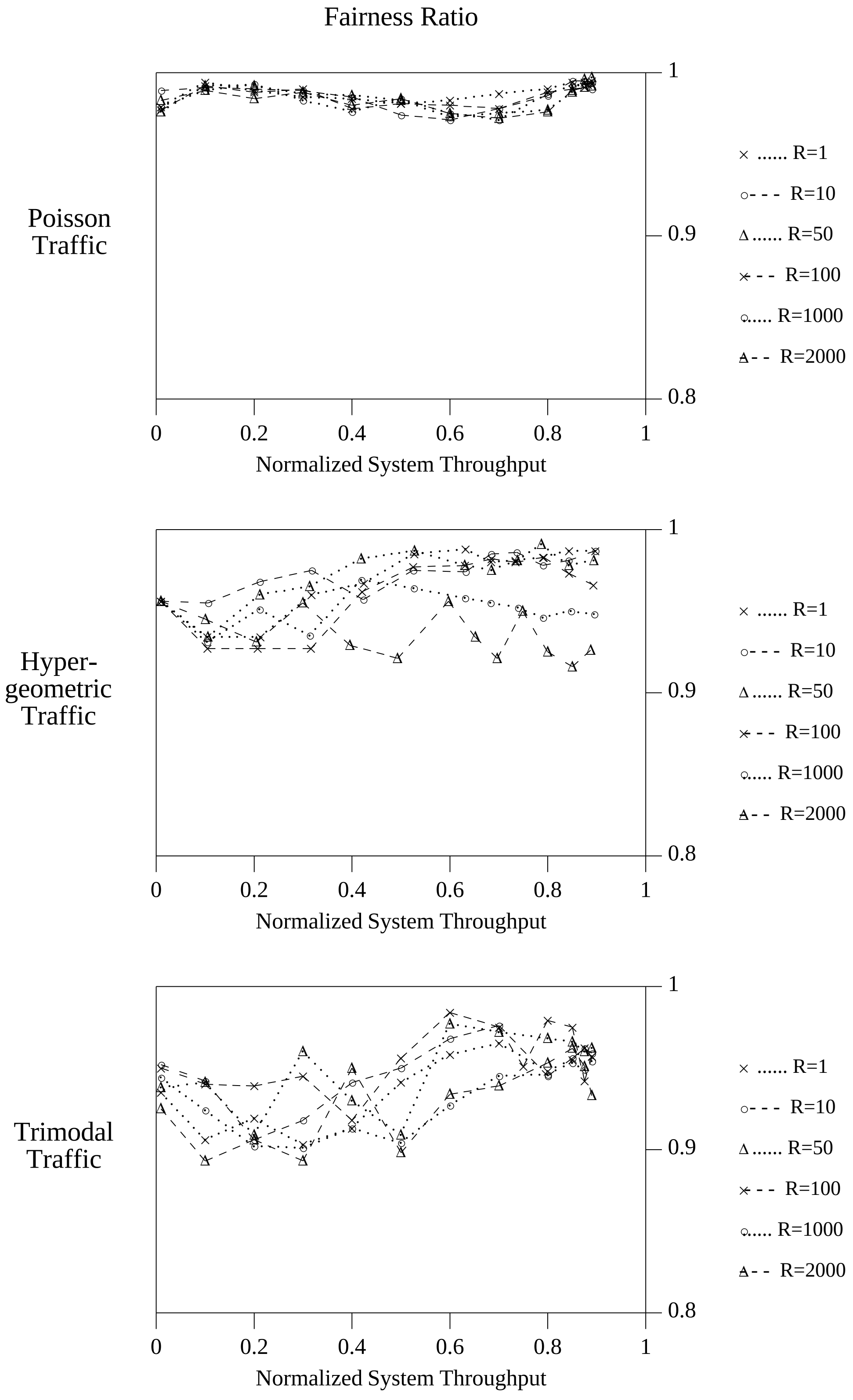

Figure 5.8: Fairness Ratio for Varying Normalized Ring Walk-Time.

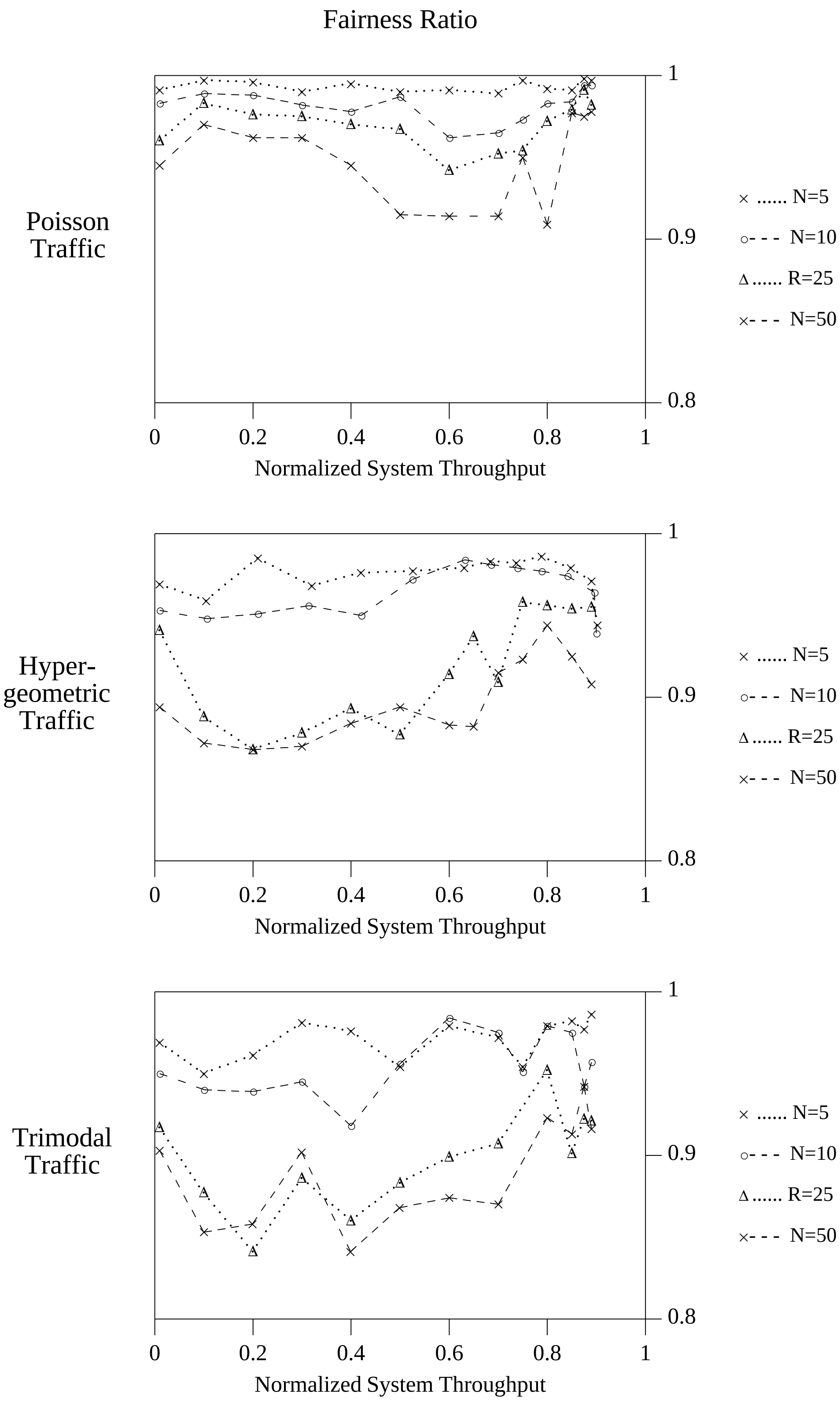

Figure 5.9: Fairness Ratio for Varying Number of Stations.

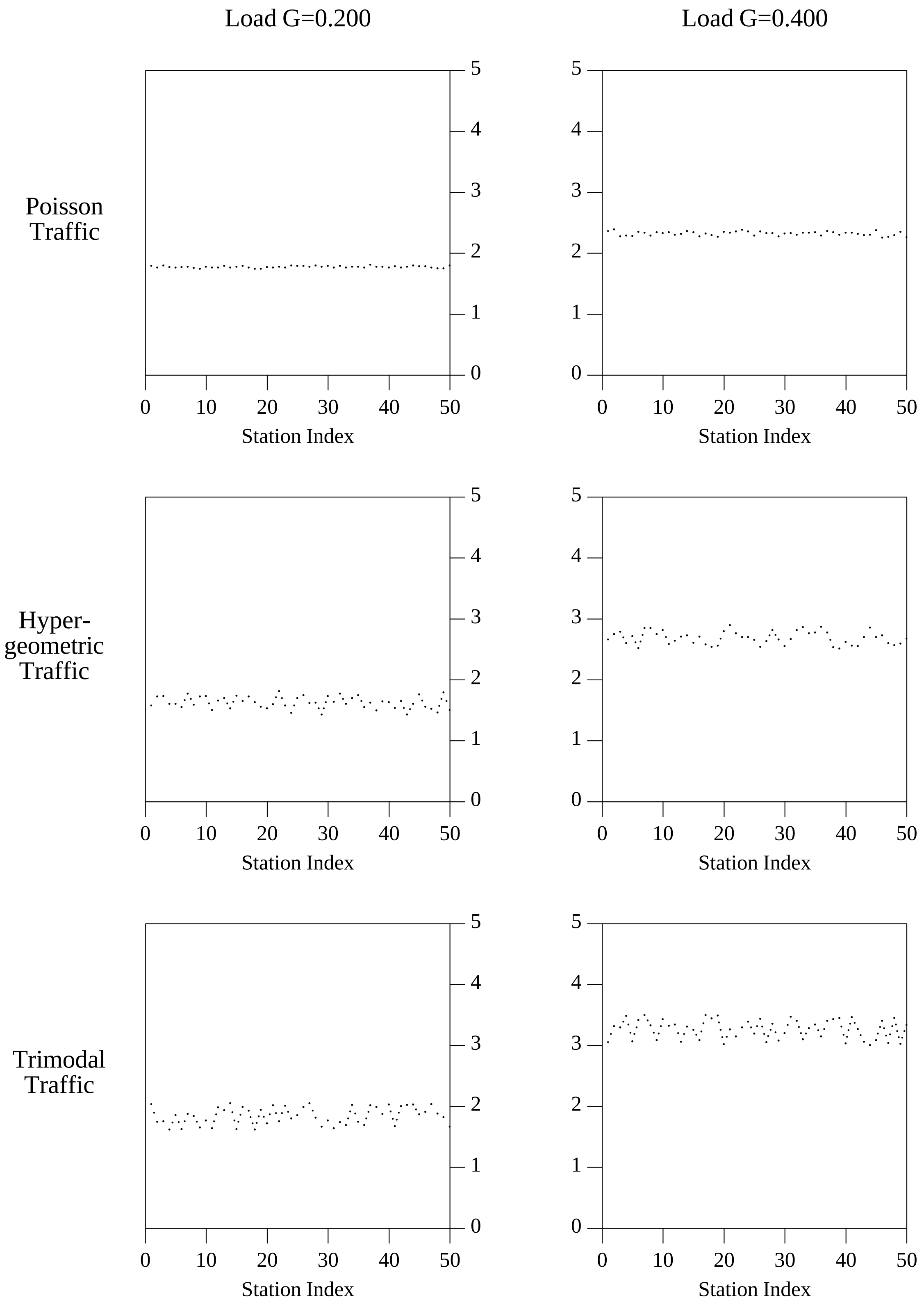

Figure 5.10: Frame Access Delay for N=50 stations, G = 0.2 and 0.4.

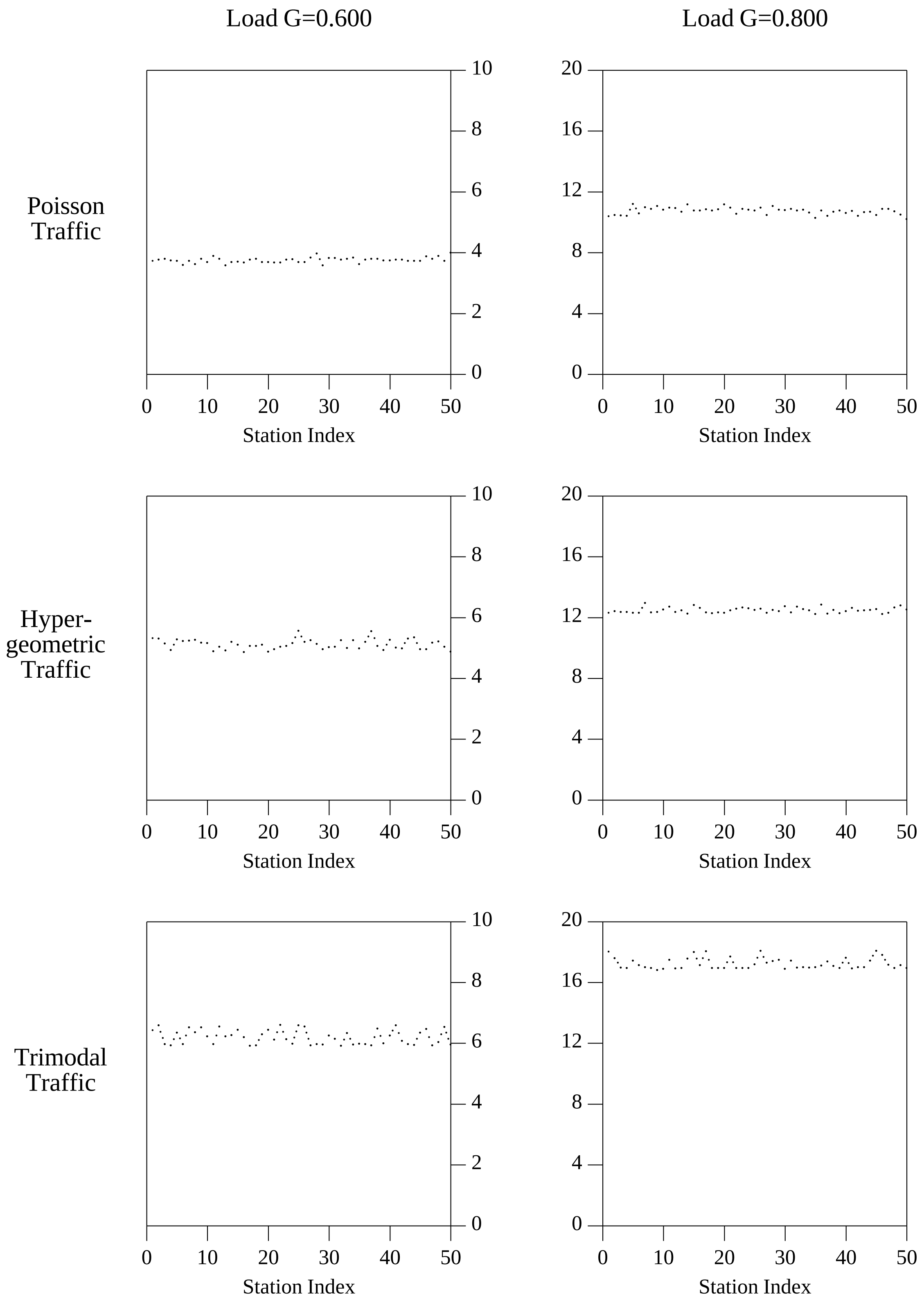

Figure 5.11: Frame Access Delay for N=50 stations, G = 0.6 and 0.8.

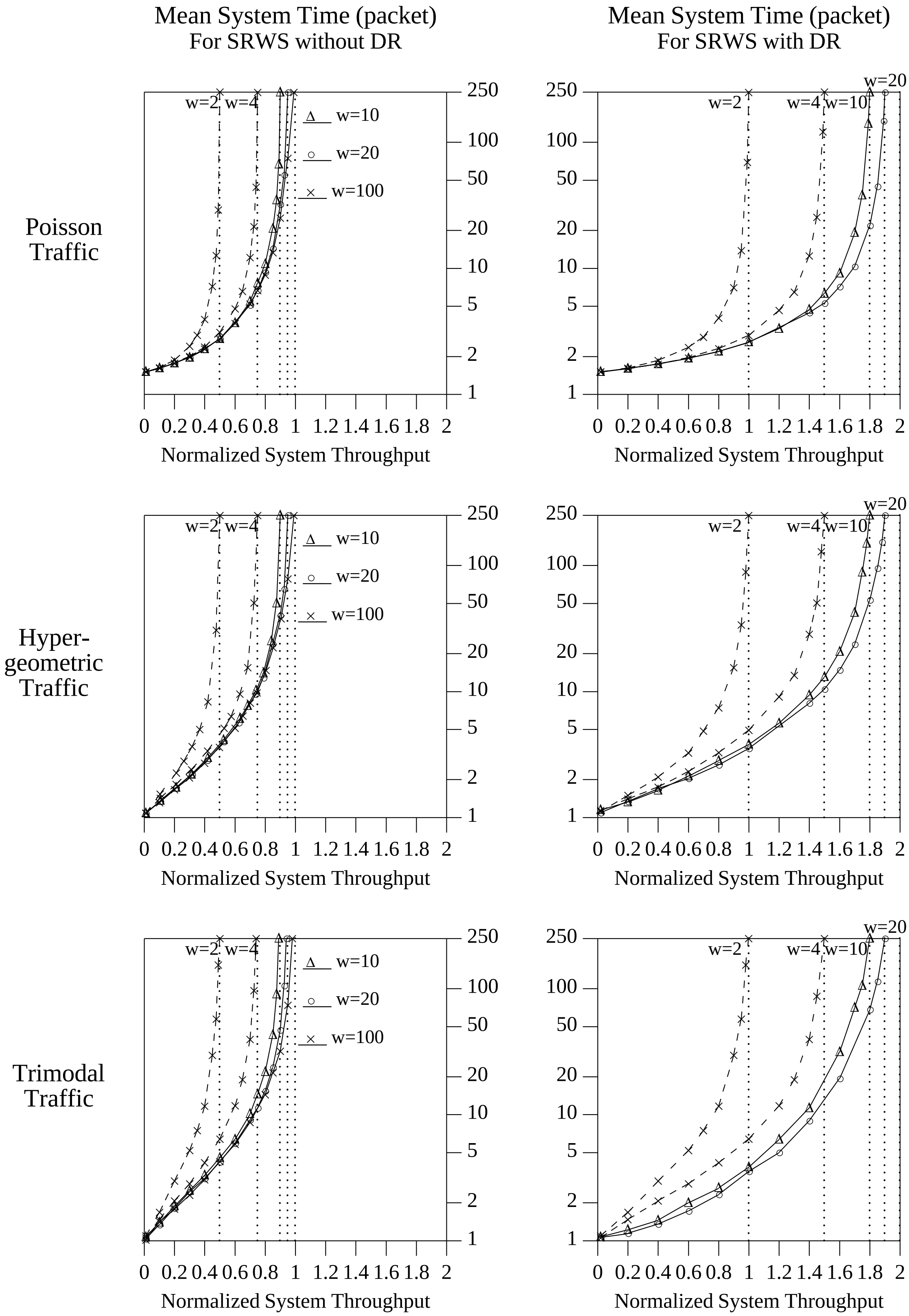

Figure 5.12: Mean Packet Delay Comparison of SRWS and SRWS/DR.

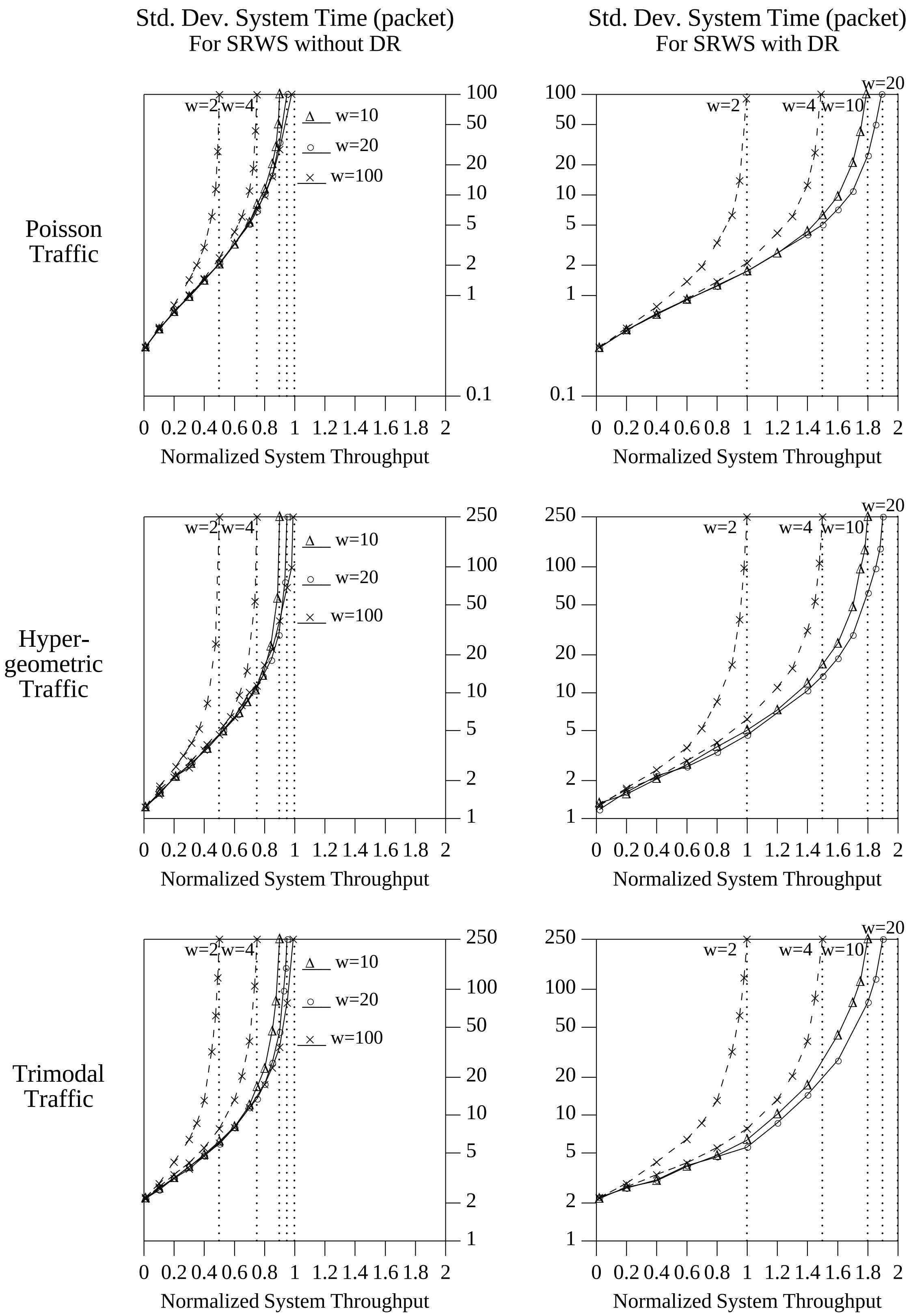

Figure 5.13: Variance Packet Delay Comparison of SRWS and SRWS/DR.

Chapter 6

Comparison to Other Protocols

In this chapter, we compare the performance of the SRWS protocol to several other alternatives, notably an ideal M/G/1 queueing system, a token ring with exhaustive service, Cyclic Reservation Multiple Access (CRMA), and Distributed Queue Dual Bus (DQDB). M/G/1 with First-Come-First-Served (FCFS) scheduling represents an unattainable lower bound to the mean packet system time and has ideal fairness properties. The token ring with exhaustive service is well known to have the lowest mean system time among all token ring protocols. DQDB and CRMA are designed specifically for high-speed LANs and MANs; both impose a similar fixed-size slot structure onto the channel, and feature capacities that are independent of the network size and distance. Except for DQDB, we were able to find analytical formulae for the mean delay for each of the other protocols that we used in our comparisons. The results for DQDB were obtained from a simulation model, which was constructed using the same methodology we described for SRWS in section 5.1.

6.1 Comparison to M/G/1, Token Ring, and CRMA

To compare the performance of SRWS to the other protocols, we consider the following set of network parameters to represent a typical high-speed MAN: $N = 50$ stations; network speed $B = 150$ Mbit/s; network length $L = 100$ km; hence a normalized walk-time $R = 150$ slots. The traffic model used is the hypergeometric packet length traffic model described in section 5.1.2. As before, we assume that the N stations equally spaced around the ring and all N stations have equal load, with identical and independent Poisson packet arrivals at normalized rate of G/N packets per mean packet transmission time.

The performance of the SRWS protocol is evaluated using the simulation model described in section 5.1.1 with the window size $W = 20$ slots. The performance of M/G/1, token ring and CRMA were evaluated using analytical models. Based on the assumptions above, the relevant parameter values are: a packet transmission time distribution with mean $\overline{x} = 10$ slots, squared coefficient variation $C_b^2 \equiv var(x)/(\overline{x})^2$ $= 1.5$, and second moment $\overline{x^2} = 250$ slots; normalized round-trip propagation delay $T_{walk} = 150$ slots; time period between successive reserve commands in CRMA $W_{CRMA} = \overline{x} = 10$ slots; and arrival rate G packets per mean packet transmission time.

The mean system time formula for the ideal slotted M/G/1 queueing system is the sum of the well-known Pollaczek-Khinchin (P-K) mean value formula [Kle75] plus the mean residual life of a slot. The additional term $T_{slot}/2$ is the result of forcing service to be be slot aligned, and is a special case of the vacation models discussed in

[LK86]. Thus, we obtain:

$$\overline{T}_{M/G/1} = \overline{x} + \frac{T_{slot}}{2} + G\overline{x}\frac{\overline{x}(1+C_b^2)}{2(1-G\overline{x})} = \overline{x} + \frac{T_{slot}}{2} + \frac{G\overline{x^2}}{2(1-G\overline{x})}\,. \tag{6.1}$$

Similarly, the variance of the system time for the slotted M/G/1 queueing system is the sum of three variance terms, because the three components are independent. These terms are: i) the variance of service time, $Var(x)$, since the service time is independent of everything in M/G/1; ii) variance of the slot time, $Var(T_{slot})$, i.e., variance of the residual life of a slot, since we consider a "slotted" M/G/1 system, not a "pure" M/G/1 system, the fact that this term can be added in this way is based on the same results from the vacation models by Levy and Kleinrock in [LK86], since they proved that the vacation time was independent of the rest of the system time; and iii) the variance of the waiting time for a "pure" M/G/1 system, $Var(W)$, which can be found in [Kle75]. Thus, we obtain:

$$Var(T_{M/G/1}) = Var(x) + \frac{(T_{slot})^2}{12} + \frac{G\overline{x^2}}{2(1-G\overline{x})} + \frac{G\overline{x^3}}{3(1-G\overline{x})}\,. \tag{6.2}$$

The mean system time formula for a symmetric token ring with exhaustive service is given in [Tak86]:

$$\overline{T}_{TR} = \overline{x} + \frac{T_{walk}}{2(1-G\overline{x})}\left[1 - \frac{G\overline{x}}{N}\right] + \frac{G\overline{x^2}}{2(1-G\overline{x})}\,. \tag{6.3}$$

An exact expression for the mean system time of a folded-bus CRMA system (but without the back-pressure mechanism, so the protocol reduces to a window service system) assuming Poisson traffic and arbitrary station locations was found in [LM90]. The "back-pressure" mechanism should have little effect under our symmetric traffic assumption, and in any case triggering it can only serve to increase the mean system time. From [LM90], we obtain:

$$\overline{T}_{CRMA-1} = \overline{x} + \frac{W_{CRMA}}{2} + T_{walk} + \frac{G\overline{x^2}}{2(1-G\overline{x})}\,. \tag{6.4}$$

By studying the operation of CRMA, it is clear that the only changes between the folded bus (CRMA-1) and dual-bus (CRMA-2) are a reduction in the load on each bus from G to G' ($G' = G/2$ assuming symmetric traffic), and a reduction in the set-up time for a reservation from T_{walk} to T_{set-up}. This is because the set-up time for a reservation on bus-A now consists of a trip to the head-end of bus-A and back (which depends on the position of the requesting station) instead of the complete trip around the network (which is independent of the position of the requesting station) in CRMA-1. Assuming symmetric traffic and equally spaced stations, the average value for T_{set-up} in CRMA-2 can be found as follows:

$$T_{set-up} = 2 * \sum_{i=1}^{N} \left[\frac{1}{N} \frac{T_{walk}}{N-1} \frac{N-i}{N-1} (i-1) \right] = \frac{N-2}{3(N-1)} T_{walk}\,. \tag{6.5}$$

Thus:

$$\overline{T}_{CRMA-2} = \overline{x} + \frac{W_{CRMA}}{2} + \frac{N-2}{3(N-1)} T_{walk} + \frac{G'\overline{x^2}}{2(1-G'\overline{x})}\,. \tag{6.6}$$

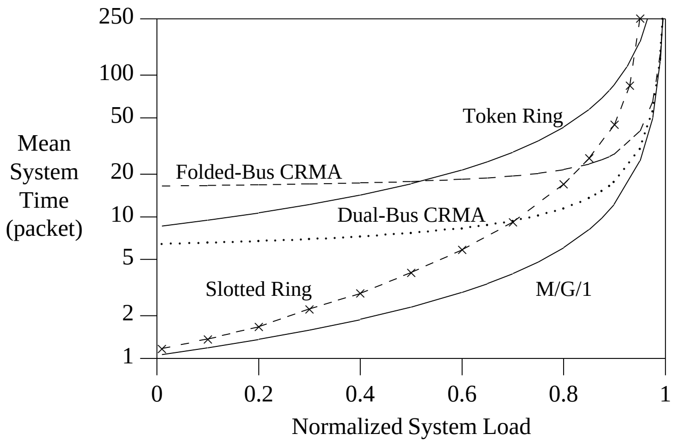

Figure 6.1: Protocol Comparison for a High-Speed MAN (B=150Mbit/s, L=100km).

Figure 6.1 plots mean packet system time versus normalized system throughput for the considered network. From figure 6.1, we see that even at high speeds and long distances, the SRWS protocol has very low mean packet system time, comparable to that of the M/G/1 queueing system. However, the mean packet system time for both token ring and CRMA are much higher, except under very heavy load. The excellent performance of the SRWS protocol at low load is because most of the slots are empty in this case, and hence packets experience negligible queueing and access delays.

In figure 6.2, we fix the system load (G) and vary the network size, R, between 1 and 1000 slots to study the effects of network speed and distance on the performance of these protocols at low load (G = 0.1), medium load (G = 0.5), and high load (G = 0.8). We see that, the SRWS protocol is quite similar to an M/G/1 queueing system, exhibiting very low packet system time, independent of the network speed and distance for all loads. However, the packet system time of token ring and CRMA increases linearly with R.

This linear behavior among the other protocols is evident from examing equations (6.3), (6.4) and (6.6), or by studying the operation of the protocols. In a token ring system, a station with a packet to transmit has to wait for next arrival of the "token" which requires the "token" to travel half way around the ring on average. In CRMA, a station with a packet to transmit has to wait for the "reserve command" to come by, add its reservation, and then wait for the corresponding "free" slots to arrive from the "head-end" at least a "round-trip" time later. Therefore, for high speed and/or large distance networks, both token rings and CRMA have large mean packet delays, even when the load on the network is very low.

6.2 Comparison to DQDB

Having shown that the SRWS protocol offers significant advantages over both token rings and CRMA, it now remains to compare it with DQDB. Although DQDB has been widely studied in the literature, we were unable to find a suitable analytical model. Thus, we constructed a simulation model for DQDB using the same methodology we described for SRWS in section 5.1. Since fairness is generally believed to be the key issue with DQDB, our comparison focus on this issue.

Using the two simulation models, we compared the performance of SRWS and DQDB under the following two networks: the first one with $N = 50$ stations and normalized walk-time $R = 150$ slots (network speed $B = 150$ Mbit/s and network length $L = 100$ km, i.e., the same network as described above); the second one with $N = 50$ stations but normalized walk-time $R = 15$ slots (network speed $B = 150$

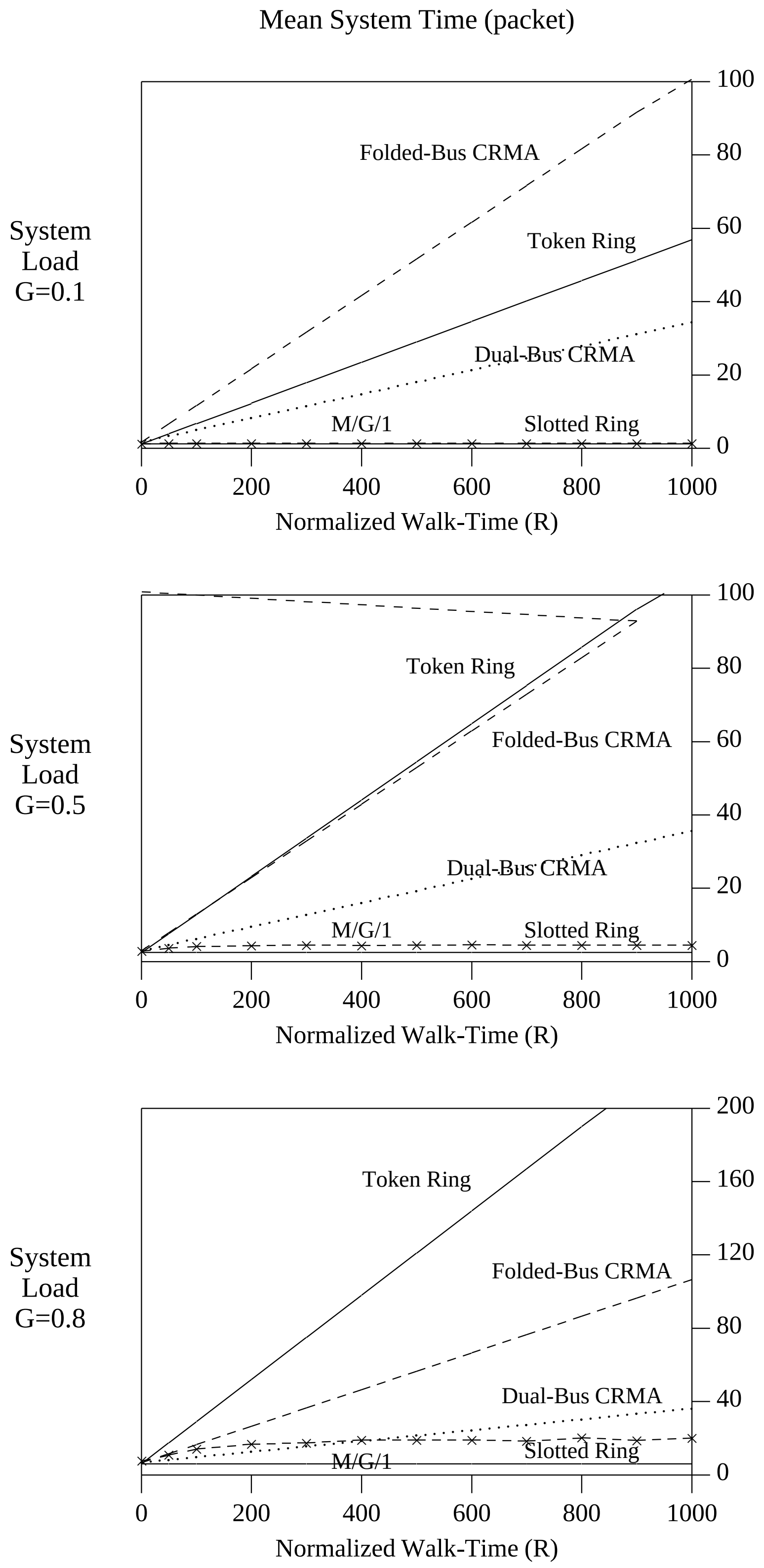

Figure 6.2: Protocol Comparison for Varying Normalized Ring Walk-Time.

Mbit/s and network length $L = 10$ km) instead of 150. Figure 6.3 shows the mean and variance of packet system time and fairness ratio among the stations for the two networks as a function of system load for SRWS with $W = 20$ slots and DQDB with and without bandwidth balancing (i.e, BWB_MOD = 8 and BWB_MOD = 0, respectively).

From figure 6.3, we see that for a network with the number of slots (R) small, SRWS has mean packet system time comparable to that of DQDB, but it has lower packet system time variance than DQDB whether or not we apply bandwidth balancing. However, when the number of slots (R) is large, the mean and variance of packet system time of SRWS is slightly higher than that of DQDB. This is because when the number of slots is large compared to the number of stations (i.e., $R \gg N$), the "slot reservation" mechanism in DQDB is largely unused. We found that the two stations next to the slot generators, which have most traffic on each bus under our symmetric traffic assumption, sent most of their data frames using "unreserved slots". We do not consider this result very significant, because the symmetric traffic assumption seems inappropriate for large networks, where studies have shown that the network traffic can be much more bursty and diverse. Also, if some slots are reserved for *synchronous traffic* (as might be expected in a MAN environment) then the number of slots available for normal data traffic would be reduced. In both cases, we found that the fairness ratio among the stations in SRWS remains around 90% for all system loads, whereas the fairness ratio of DQDB with BWB_MOD = 8 is only around 75%, and DQDB BWB_MOD = 0 has the fairness ratio degrade dramatically when the system load is high.

Figures 6.4 and 6.5 show the mean frame access delay at each station on SRWS and bus-A of DQDB for $R = 15$ and $R = 150$ slots, respectively. Similar to the graphs for fairness ratio in figure 6.3, these two figures show that the frame access delay at each station on SRWS is independent of the system load and the position of the station on the network. However, this is not true with DQDB. Under low to medium loads ($G = 0.1$ and 0.5, respectively), the stations on DQDB have approximately the same frame access delay. But when the system load is high ($G = 1.2$), the system is extremely is unfair.

Moreover, the (un)fairness of DQDB depends not only on the system load, but also the size of the network (R) and distances between the stations. From figures 6.4 and 6.5, we see that for the case $R = 15$ slots the station next to the slot generator (station 1) has the highest frame access delay, but in the case of $R = 150$ slots the station with longest distance from the slot generator (station 50) has the highest frame access delay. Our simulation results show that for the case of $R = 15$ slots the first few stations next to the slot generator dropped more than 90% (almost all) of their packets, where as in the case of $R = 150$ the portions of packets dropped at these stations are much smaller. Similar results on the (un)fairness of DQDB are also found in [Won89, Mar89, KC90, CGL91].

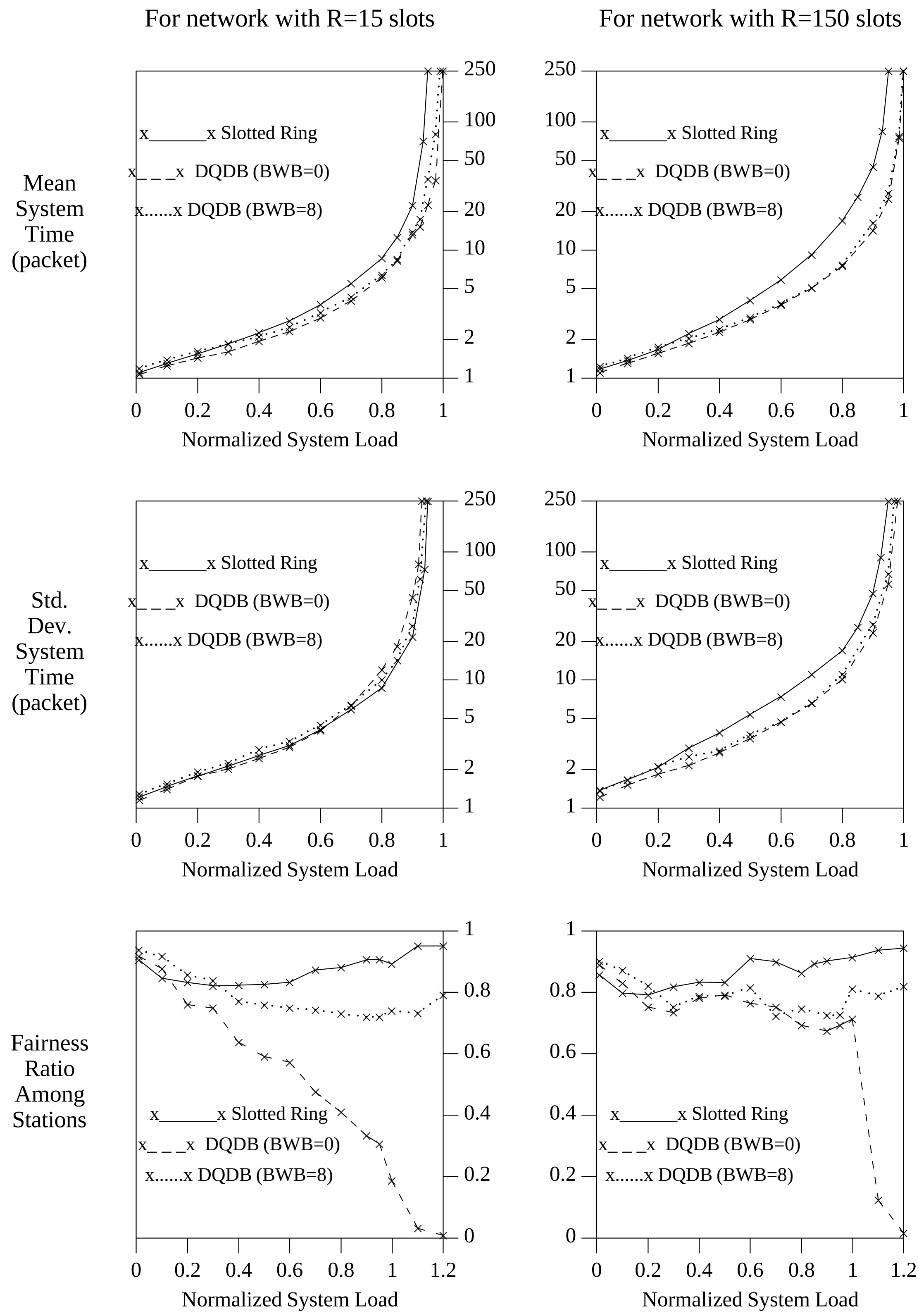

Figure 6.3: Comparison of SRWS and DQDB for $R = 15$ and 150 slots.

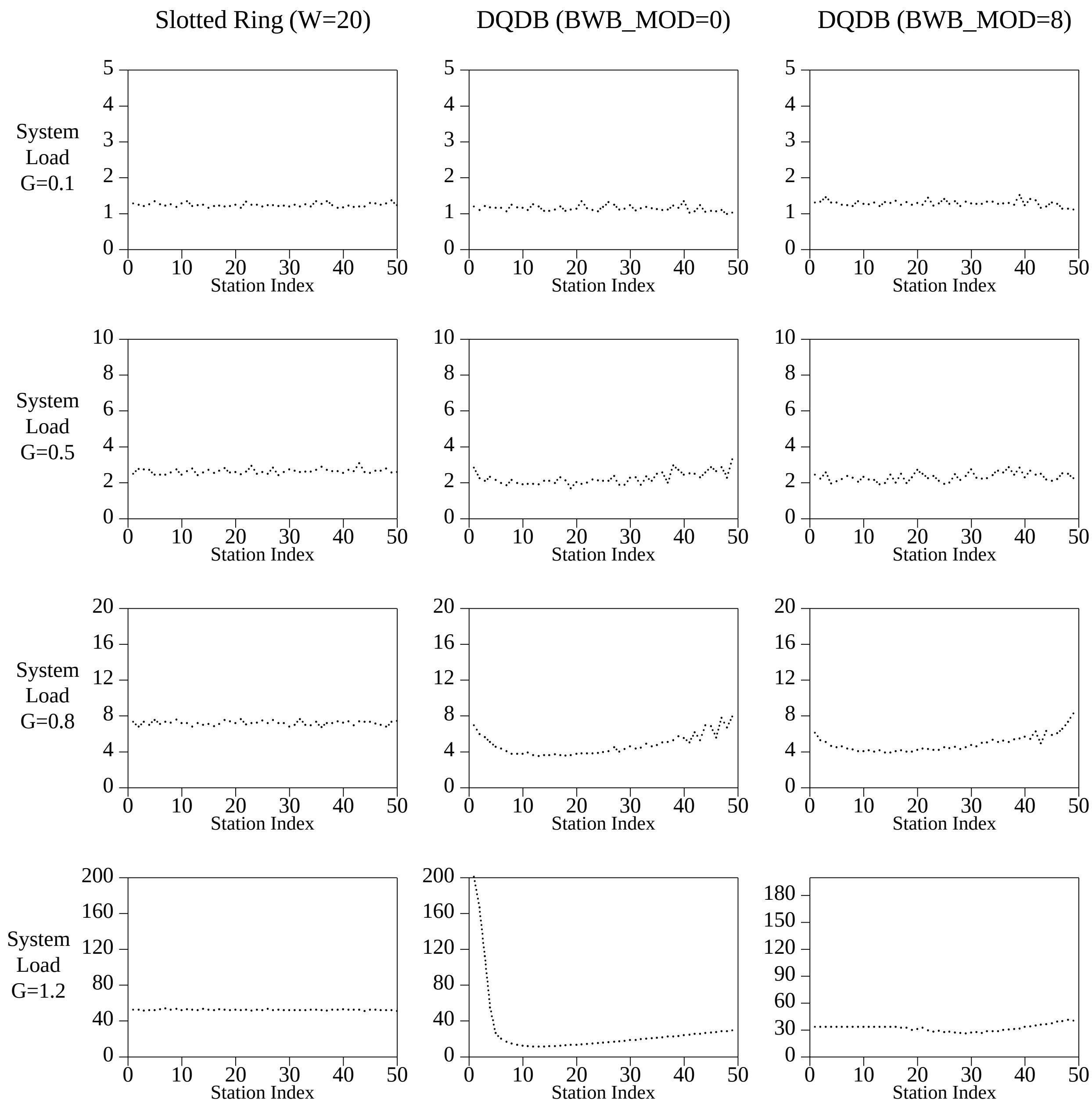

Figure 6.4: Station Frame Access Delay of SRWS and DQDB for $R = 15$ slots.

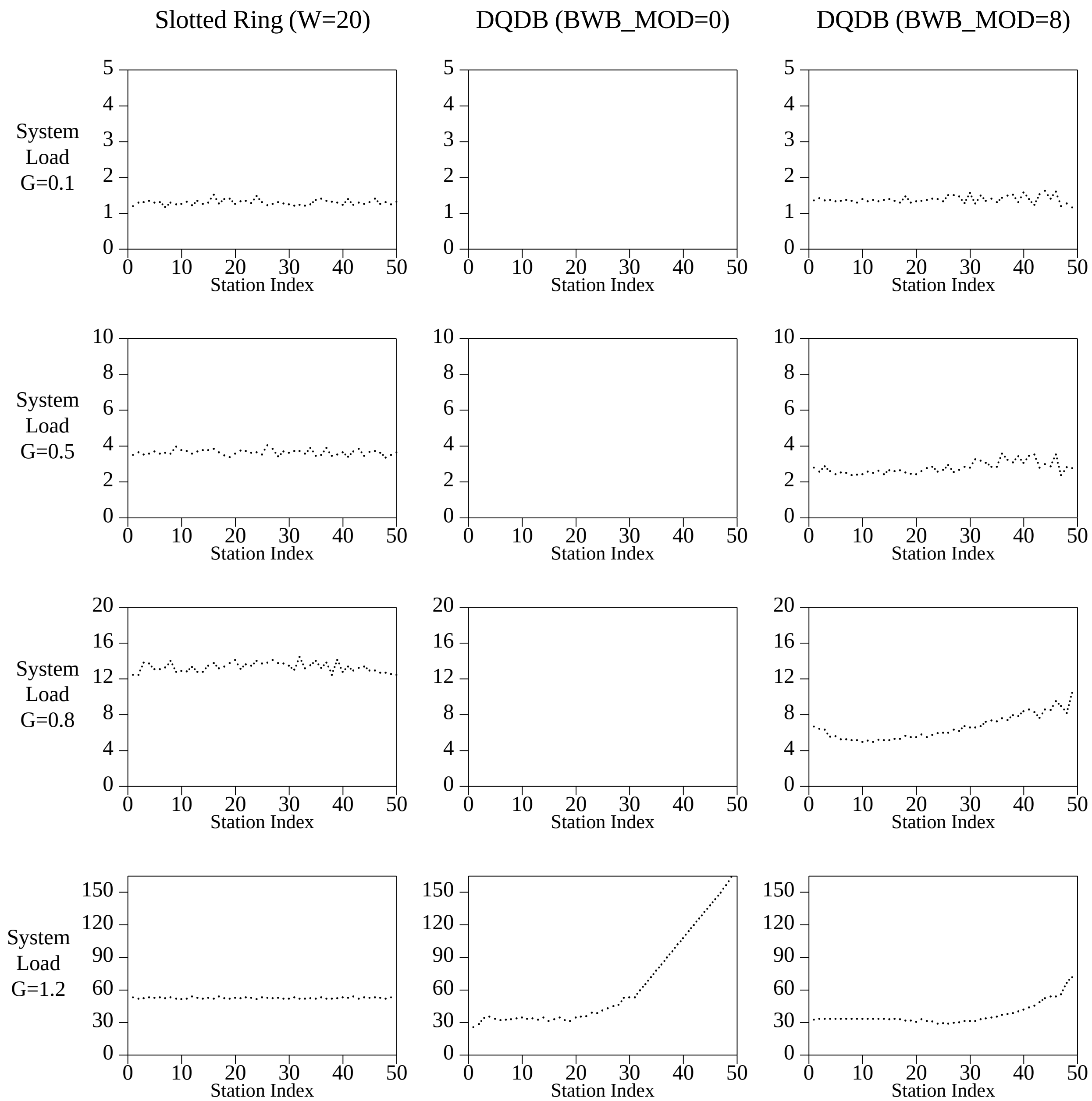

Figure 6.5: Station Frame Access Delay of SRWS and DQDB for $R = 150$ slots.

Chapter 7

Summary

7.1 Conclusions

In this thesis, we have shown that Slotted Ring with window service (SRWS) can be a very efficient protocol for high-speed LANs and MANs. The protocol offers high capacity, low mean and variance of the packet system time, and fairness among the stations that is almost independent of the network speed, distance, and the number of stations. The protocol has very low mean packet system time, comparable to that of an M/G/1 system. When the system load is low, the system time of packets is just the packet transmission time, because most of the slots are available in this case, and packets can be transmitted without any queueing or access delays.

In contrast to most previous performance comparisons, where token rings generally were found to out-perform slotted rings, we showed that, for high-speed (Gbits/s) and long distance (hundreds of kilometers) networks, SRWS has much lower packet system time than both token rings and CRMA. Furthermore, although the mean and variance of packet system time for SRWS is comparable to that of DQDB, the fairness among the stations does not degrade the way DQDB does when the system load is very high.

Using the analytical results from Token Ring with Window-Gated service (TRWS) and moving server queueing systems with Head-of-the-Line (HOL) priority order of service, we found analytical formulae for the system capacity as well as lower and upper bounds on the mean packet system time of SRWS. All of our simulation results agreed closely with the formula for the system capacity and were consistent with the lower and upper bounds for the mean packet system time.

We also proposed the SRWS/DR protocol, which can be used to double the capacity of a slotted ring by having the busy slots released (or re-used) by the destination stations. Our simulation results for symmetric systems showed that this modification allows us to double the system throughput without affecting the mean and variance of packet system time significantly.

7.2 Future Research

In this thesis, we considered only one class of service. However, because of their high transmission rate and long distance coverage, we expect that most of high-speed LANs and MANs will support multiple levels of priority and service classes. However, one can adapt a priority reservation scheme from a ring network, such as that used in a token ring [IEE85c], to SRWS, so that the protocol can provide multiple classes of priority and service.

In our simulations, even though we considered three different traffic models (three different packet length distributions with different squared coefficients of variation) to model different degrees of burstiness of computer traffic, we always used the assumptions of symmetric traffic and balanced load among all the stations. However, in high speed and/or large distance networks, real network traffic is likely to be much more bursty and diverse. Thus, more sophisticated simulation experiments with more realistic workload characteristics and traffic distributions (ideally, real measurement traffic distributions taken from empirical studies) should be constructed to study SRWS under asymmetric workloads. We expect that the performance of SRWS in such studies will be better than the other protocols because the window service rule offers implicit bandwidth balancing in proportion to a station's traffic rate, which will be more important in an unbalanced system.

Although the performance of SRWS is insensitive to the network speed and distance, our simulation results did show that the mean packet system time of SRWS increases slightly as the number of slots (normalized ring walk-time, R) increases, and the rate of increase decreases rapidly as the number of slots increases. Furthermore, our lower and upper bounds for the mean system time are based on the cases $R = 1$ and $R \to \infty$, respectively. We, therefore, conjecture that the mean packet system time of SRWS is a bounded, monotonically increasing function of R. However, a formal proof for this is yet to be found. Even more challenging would be to find the exact analytical formulae for the mean and variance of packet system time for SRWS for arbitrary value of R.

In appendix A of this thesis, we also described a recursive batch mean analysis procedure, which is used to test for batch independence. The procedure was successfully applied in all of our simulation experiments, and can be very useful in many other simulation studies and experiments. However, finding a suitable test for convergence for a general case is still a subject for further study.

Appendix A

Recursive Batch Mean Analysis

In this appendix, section A.1 first describes the principle of batch mean analysis. Section A.2 shows that if the batches are independent, then combining batches pairwise does not change the variance estimator or the confidence interval for the mean. Lastly, section A.3 uses the result from section A.2 and proposes a recursive batch mean analysis procedure which can be used to control the run time of a simulation experiment and automatically stop when the batch size is large enough for the batch means to be approximately independent and the desired confidence interval is achieved.

A.1 Batch Mean Analysis Method

The method of **batch mean analysis (BMA)** [Con63, LK91] is a commonly used method for calculating the confidence interval on the mean of a sequence of correlated observations collected from a simulation experiment. With this method, the n correlated observations from a simulation experiment, x_1, x_2, ..., x_n, are divided into k batches of size m ($n = k.m$), which (we hope) have uncorrelated and normally distributed batch means X_1, X_2, ..., X_k, where

$$X_i = \frac{1}{m}\left[\sum_{j=1}^{m} x_{(i-1)m+j}\right] \quad for \;\; i = 1, 2, ..., k\,. \tag{A.1}$$

The point estimator for the mean μ of the X_i's is the grand sample mean

$$\overline{X} = \frac{1}{k}\sum_{i=1}^{k} X_i = \frac{1}{n}\sum_{j=1}^{n} x_j\,, \tag{A.2}$$

which has a $(1-\alpha)100\%$ confidence interval of

$$\overline{X} \pm t_{k-1,1-\alpha/2}\cdot\sqrt{\frac{S_k^2}{k}}\,, \tag{A.3}$$

where $t_{k-1,1-\alpha/2}$ is the $(1-\alpha/2)$ quantile of the t-distribution with $(k-1)$ degrees of freedom,

$$S_{\overline{X}}^2 = \frac{S_k^2}{k} = \frac{1}{k}\left[\frac{1}{(k-1)}\sum_{i=1}^{k}(X_i - \overline{X})^2\right] \tag{A.4}$$

is the estimator for the variance of the batch means (X_i's), and

$$H_{\alpha,k} = t_{k-1,1-\alpha/2}\cdot\sqrt{\frac{S_k^2}{k}} \tag{A.5}$$

is called the *half-width* of the $(1-\alpha)100\%$ confidence interval.

The method of batch mean analysis is based on the following two assumptions:

i) The effects of the initial *transient period* have been removed, and

ii) The *dependency* and *nonnormality* of the batch means X_i's are negligible.

It has been shown that if the batch size, m, is chosen large enough, then the X_i's will be approximately uncorrelated and normally distributed, and thus satisfy the assumptions of batch mean analysis [LC79, LK91]. The major source of error in batch mean analysis is choosing the batch size too small, which results in the X_i's possibly being highly correlated and $S^2_{\overline{X}}$ being a severely biased estimator for the variance of the batch means. In particular, if the X_i's are positively correlated, then the equations (A.4) and (A.3) will give a variance estimator that is biased low and a confidence interval that is too small. Thus, the confidence interval will cover the mean μ with a probability that is lower than the desired $(1-\alpha)$ [LK91].

Schmeiser [Sch82] pointed out that the effects of less than 10 batches are large and the effects of more than 30 batches small, i.e., $10 \leq k \leq 30$ is reasonable for most of simulation experiments. Thus, the major problem in batch mean analysis is choosing the batch size, not the number of batches. Choosing the batch size is not an easy task, because what is sufficiently large for one experiment may not be adequate for another. Furthermore, using batch sizes larger than necessary to satisfy the independence and normality assumptions of batch mean analysis will also increase the cost of running the experiment. In the following sections, we describe a procedure that uses a fixed number of batches and recursively doubles the batch size until the batch means X_i's satisfy a test for approximate independence and the desired confidence interval is obtained.

A.2 Effects of Combining Batches on Confidence Interval

In this section, we will show that, for the batch mean analysis method, if the batches are independent (i.e., the batch size is large enough), then combining batches pairwise (which doubles the batch size and halves the total number of batches) does not change the variance estimator, and hence the confidence interval for the mean. On the other hand, if there are strong correlations between the batches (i.e., the batch size is too small), then combining batches will significantly change the variance estimator and the width of the confidence interval for the mean.

Suppose that we divide the n observations from a simulation experiment into $(k+1)$ batches of size m, B_0, B_1, B_2, ..., B_k, with respective batch means of X_0, X_1, X_2, ..., X_k. Assuming that batch B_0 is large enough to cover the *transient period,* then the approximate mean and variance estimators for the X_i's are:

$$\overline{X} = \frac{1}{k}\sum_{i=1}^{k} X_i \ ,$$

and

$$S^2_{\overline{X}} = \frac{S^2_X}{k} \ ,$$

where

$$S^2_X = \frac{1}{k-1}\sum_{i=1}^{k}(X_i - \overline{X})^2 \ .$$

Assume that k is even. Let $C_1 = B_1 \cup B_2$, $C_2 = B_3 \cup B_4$, ..., $C_{k/2} = B_{k-1} \cup B_k$, i.e., we merge each of the $k/2$ pairs of adjacent batches into a single batch of size $2m$, with respective batch means of Y_1, Y_2, ..., $Y_{k/2}$, where

$$Y_j = \frac{X_{2j-1} + X_{2j}}{2} \quad for\ j = 1, 2, ..., k/2\ .$$

Then the approximate mean and variance estimators for the Y_j's are:

$$\overline{Y} = \frac{1}{(k/2)} \sum_{j=1}^{k/2} Y_j = \frac{2}{k} \sum_{j=1}^{k/2} \frac{X_{2j-1} + X_{2j}}{2} = \frac{1}{k} \sum_{i=1}^{k} X_i = \overline{X}\ ,$$

and

$$S^2_{\overline{Y}} = \frac{S^2_Y}{(k/2)}\ ,$$

where

$$S^2_Y = \frac{1}{(k/2-1)} \sum_{j=1}^{k/2} (Y_j - \overline{Y})^2$$

$$= \frac{2}{k-2} \sum_{j=1}^{k/2} \left(\frac{X_{2j-1} + X_{2j}}{2} - \overline{X} \right)^2$$

$$= \frac{1}{2k-4} \sum_{j=1}^{k/2} \left(X_{2j-1} + X_{2j} - 2\overline{X} \right)^2$$

$$= \frac{1}{2k-4} \left[\sum_{j=1}^{k/2} (X^2_{2j-1} + X^2_{2j}) + \left(\frac{k}{2}\right) \left(4\overline{X}^2\right) - \left(4\overline{X}\right) \sum_{j=1}^{k/2} (X_{2j-1} + X_{2j}) + \sum_{j=1}^{k/2} (2.X_{2j-1}.X_{2j}) \right]$$

$$= \frac{k}{k-2} \left[\frac{\overline{X^2}}{2} + \overline{X}^2 - 2\overline{X}^2 + \frac{1}{k} \sum_{j=1}^{k/2} (X_{2j-1}.X_{2j}) \right]$$

$$= \frac{k}{k-2} \left[\frac{\overline{X^2}}{2} - \overline{X}^2 + \frac{1}{k} \sum_{j=1}^{k/2} (X_{2j-1}.X_{2j}) \right]\ .$$

Case 1: If the k batches X_i's are 100% independent, then:

$$\frac{1}{k}\sum_{j=1}^{k/2}(X_{2j-1}.X_{2j}) = \frac{\overline{X}^2}{2}\,.$$

Thus,

$$S_Y^2 = \frac{k}{k-2}\left[\frac{\overline{X^2}}{2} - \overline{X}^2 + \frac{\overline{X}^2}{2}\right] = \frac{\overline{X^2} - \overline{X}^2}{2 - 4/k} \approx \frac{S_X^2}{2}\,,$$

and hence

$$S_{\overline{Y}}^2 = \frac{S_Y^2}{(k/2)} \approx \frac{S_X^2}{k} = S_{\overline{X}}^2\,.$$

Therefore, if the batches are independent (i.e., the batch size is large enough), then $S_{\overline{Y}}^2 \approx S_{\overline{X}}^2$. That is, combining batches pairwise will not change the variance estimator or the confidence interval for the mean.

Case 2: If the k batches X_i's are 100% correlated, then:

$$\frac{1}{k}\sum_{j=1}^{k/2}(X_{2j-1}.X_{2j}) = \frac{\overline{X^2}}{2}\,.$$

Thus,

$$S_Y^2 = \frac{k}{k-2}\left[\frac{\overline{X^2}}{2} - \overline{X}^2 + \frac{\overline{X^2}}{2}\right] = \frac{\overline{X^2} - \overline{X}^2}{1 - 2/k} \approx S_X^2\,,$$

and hence

$$S_{\overline{Y}}^2 = \frac{S_Y^2}{(k/2)} \approx \frac{2 * S_X}{k} = 2 * S_{\overline{X}}^2\,.$$

Therefore, if there are strong correlations between the batches (i.e., the batch size is too small), then $S_{\overline{Y}}^2 \gg S_{\overline{X}}^2$. That is, combining batches pairwise will significantly increase the variance estimator and the width of the confidence interval for the mean.

A.3 Recursive Batch Mean Analysis Procedure

It is shown in section A.2 above that if the batch means (X_i's) are approximately independent (i.e., the batch size is large enough), then combining batches pairwise does not change the variance estimator or the confidence interval for the mean. We now use this result to design a procedure to control the run length of a simulation experiment that will automatically detect when the batch size is large enough for the batch means to be approximately independent. This procedure, called **recursive batch mean analysis,** uses bounded storage to hold only a fixed even number of batches. It begins the simulation with a reasonable estimate for the minimum batch size and thereafter recursively increases it by factors of 2 until the test for batch independence is satisfied.

First, k batches of minimum batch size ($m = m_{min}$) are run to obtain k batch means X_1, X_2, ..., X_k. Next, we merge each pair of adjacent batches to form $k/2$ new batches, i.e.,

$$
\begin{aligned}
Y_1 &\leftarrow X_1 + X_2 \ , \\
Y_2 &\leftarrow X_3 + X_4 \ , \\
&\ \ \vdots \\
Y_{k/2} &\leftarrow X_{k-1} + X_k \ .
\end{aligned}
$$

Third, we compute the variance estimators $S^2_{\overline{X}}$ and $S^2_{\overline{Y}}$ using equation (A.4). In all of the calculations, we use only the last $(k-2)$ batches of the X_i's and the last $(k/2-1)$

batches of the Y_j's, because we want to avoid the initial transient period, where the simulation is not yet in *steady-state* condition. Four, we compute the variance estimator difference, $\Delta S^2 \leftarrow \left| S_{\overline{X}}^2 - S_{\overline{Y}}^2 \right| / \overline{X}^2$. Note that we normalize ΔS^2 by $\overline{X}^2$, i.e., we compute the difference of the squared coefficients of variation, because we want to take into consideration both the magnitudes of the mean and the variance.

If $\Delta S^2 \approx 0$, the batches are independent (i.e., the batch size is large enough), stop and compute the mean and confidence interval using equations (A.2) and (A.3). Otherwise, the batches are still positively correlated (i.e., the batch size is not large enough), so we initialize the first $k/2$ "new" batches via $X_j \leftarrow Y_j$, $1 \leq j \leq k/2$, double the batch size, and filling in the batches $X_{k/2+1}$, ..., X_k by running another $k/2$ replications with the new batch size. At this point, we are ready to repeat the test for independence by recomputing the variance estimators $S_{\overline{X}}^2$ and $S_{\overline{Y}}^2$ and the variance estimator difference ΔS^2, and so on.

The procedure above can be repeated until the variance estimator difference ΔS^2 converges. Unfortunately, for some random variables, it may take a long time (a few days (!), even on powerful workstations such as Silicon Graphics SGI 4D/240 and Digital Equipment DECstation3100) to converge. Therefore, there should be an upper limit on the maximum batch size so that the procedure always stops within a reasonable time limit. Furthermore, we also want the confidence interval to be within a certain level of precision. Thus, we also compute and check the half-width of the confidence interval, $H_{\alpha,k}$, and force the algorithm to continue if either the difference of the variance estimators (ΔS^2) or the half-width ($H_{\alpha,k}$) is too large. The description

Step 0 (Initialization step):
    Initialize all batches
    Run k batches of minimum batch size m_{min}

Step 1 (Calculation step):
    /* Use only the last $(k-2)$ batches in the following calculations */
    Compute mean $\overline{X}$ using equation A.2
    Compute variance $S^2_{\overline{X}}$ using equation A.4
    Compute half_width $H_{\alpha,k}$ using equation A.5
    Compute relative_error $\leftarrow H_{\alpha,k}/\overline{X}$
    Merge pairs of adjacent batches into single batches
    Compute variance $S^2_{\overline{Y}}$ /* Use only batches from 2 to $k/2$ */
    Compute variance difference $\Delta S^2 \leftarrow \left| S^2_{\overline{X}} - S^2_{\overline{Y}} \right| / \overline{X}^2$

Step 2 (Test for Independence and Confidence Interval):
    If $(\Delta S^2 > \epsilon$ or relative_error $> \gamma)$ and (batch_size $\leq$ max_batch_size) then
        Multiply the batch size by a factor of 2
        Run another $k/2$ batches with the new batch size
        Return to step 1
    Else
        Exit to step 3
    End-if

Step 3 (Final step):
    Output mean $\overline{X}$ and confidence interval $\overline{X} \pm H_{\alpha,k}$

Table A.1: Recursive Batch Mean Analysis Procedure.

of the recursive batch mean analysis procedure and a "C-like" program are given in table A.1 and table A.2, respectively.

```
/* Initialization */
γ ← max_relative_error /* percentage of relative error */
k ← number_of_batches /* an even constant */
For i from 1 to k do
      sum(i) ← 0
End-for
batch_size ← min_batch_size
k0 ← 1 /* start batch number */
ΔS² ← 2 * ε /* ε is a small positive real number */
relative_error ← 2 * γ

While (ΔS² > ε or relative_error > γ) and (batch_size ≤ max_batch_size) do
      For i from k0 to k do
            For j from 1 to batch_size do
                  /* Run the simulation and collect data */
                  sum(i) ← sum(i) + x_j
            End-for
      End-for
      /* Use only the last (k − 2) batches in the following calculations */
      Compute mean X̄ using equation A.2
      Compute variance S²_X̄ using equation A.4
      Compute half_width H_α,k using equation A.5
      Compute relative_error ← H_α,k / X̄
      /* Merge pairs of adjacent batches into single batches */
      k0 ← (k/2) + 1
      For i from 1 to (k0 − 1) do
            sum(i) ← sum(2i-1) + sum(2i)
      End-for
      For i from k0 to k do
            sum(i) ← 0
      End-for
      Compute variance S²_Ȳ /* Use only batches from 2 to k/2 */
      Compute variance difference ΔS² ← |S²_X̄ − S²_Ȳ| / X̄²
      If ΔS² > ε or relative_error > γ then
            /* Double the batch size */
            batch_size ← batch_size * 2
      End-if
End-while
Output mean X̄ and confidence interval X̄ ± H_α,k
```

Table A.2: Recursive Batch Mean Analysis Program.

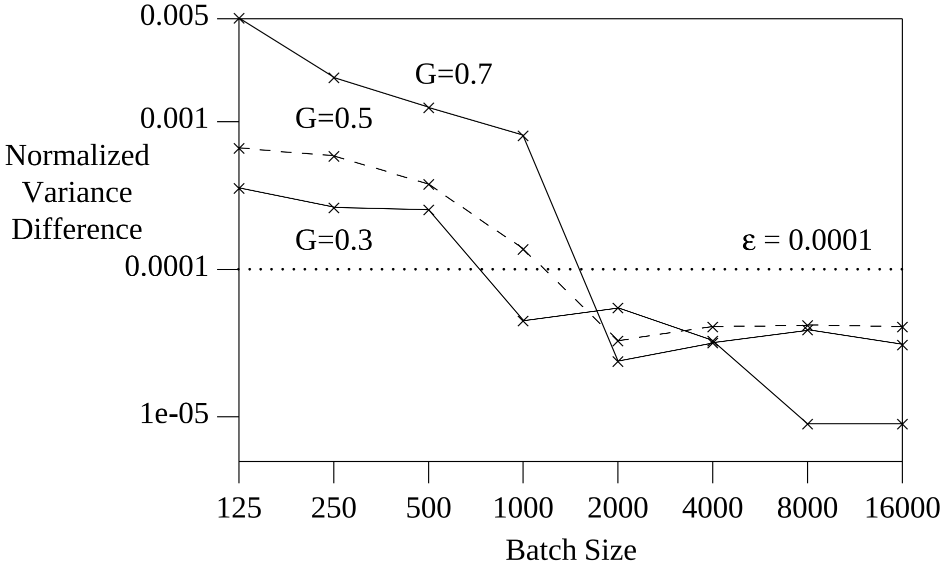

Figure A.1: Normalized Variance Estimator Difference vs Batch Size.

Determining The Value For Epsilon (ϵ)

One of the critical input parameters for the recursive batch mean analysis is the value of ϵ, which is used to specify how small we want to the value of the normalized variance estimator difference, ΔS^2, to converge to. In order to determine a value for ϵ that is good for all (or most) of our simulation experiments, we made several initial runs with different values of system loads ($G = 0.3$, 0.5, and 0.7) to examine the behavior of ΔS^2. The system parameters used were number of stations $N = 10$, normalized ring walk-time $R = 100$ slots, and window size $w = 10$ slots; and the traffic model was the hypergeometric packet length traffic model. The results from these initial runs are tabulated in table A.3. In figure A.1, we plot ΔS^2 (normalized by the square of the mean) as a function of the batch size. From figure A.1, we see that, for all the system loads we ran, ΔS^2 always starts from a large value, however, it decreases as the batch size increases, and converges to a very small value after a few iterations. For our set of initial runs, ΔS^2 always converges to a value that is less than 0.0001 when the batch size reaches the value of 2,000 packets; and thereafter it does not decrease any further. This behavior of ΔS^2 is similar to that of the half-width of the confidence interval ($H_{\alpha,k}$), even though their rates of convergence may be very different. From our initial runs, we see that the value $\epsilon = 0.0001$ is reasonably good for all of our simulation experiments, and thus, we chose to use this value in all our simulation experiments.

Load (G)	Batch (m)	Mean ($\overline{X}$)	Variance ($S^2_{\overline{X}}$)	Variance ($S^2_{\overline{Y}}$)	$S^2_{\overline{X}} - S^2_{\overline{Y}}$	ΔS^2	$H_{\alpha,k}$	$H_{\alpha,k}/\overline{X}$
0.300	125	2.231	0.008528	0.010304	0.001776	0.000357	0.194027	0.086954
	250	2.181	0.008507	0.009632	0.001125	0.000264	0.193778	0.088852
	500	2.192	0.003598	0.004817	0.001219	0.000254	0.126030	0.057496
	1000	2.136	0.001362	0.001568	0.000206	0.000045	0.077546	0.036302
	2000	2.127	0.000879	0.000632	0.000247	0.000055	0.062298	0.029288
	4000	2.125	0.000429	0.000279	0.000150	0.000033	0.043520	0.020478
	8000	2.139	0.000259	0.000220	0.000039	0.000009	0.033835	0.015821
	16000	2.162	0.000172	0.000131	0.000041	0.000009	0.027547	0.012742
0.500	125	3.875	0.119147	0.109166	0.009981	0.000665	0.725216	0.187170
	250	3.589	0.023929	0.016341	0.007588	0.000589	0.325001	0.090561
	500	3.711	0.023568	0.018354	0.005214	0.000379	0.322544	0.086917
	1000	3.750	0.008170	0.010115	0.001945	0.000138	0.189906	0.060638
	2000	3.743	0.008371	0.008827	0.000456	0.000033	0.192227	0.056359
	4000	3.745	0.003460	0.004035	0.000575	0.000041	0.123586	0.032998
	8000	3.820	0.001754	0.001143	0.000611	0.000042	0.088001	0.023034
	16000	3.815	0.001303	0.000709	0.000594	0.000041	0.075850	0.019884
0.700	125	7.042	0.463767	0.738825	0.275058	0.005547	1.430791	0.203192
	250	9.473	0.595723	0.416892	0.178831	0.001993	1.621617	0.171182
	500	8.751	0.612649	0.699045	0.086396	0.001251	1.644493	0.187913
	1000	8.732	0.193864	0.255578	0.061714	0.000809	0.925070	0.105940
	2000	8.422	0.084168	0.085869	0.001701	0.000024	0.609537	0.072372
	4000	8.204	0.051485	0.053637	0.002152	0.000032	0.476724	0.058112
	8000	8.297	0.023792	0.026466	0.002674	0.000039	0.324073	0.039061
	16000	8.268	0.012810	0.010720	0.002090	0.000031	0.237796	0.028762

Table A.3: Intermediate Results from a Simulation using Recursive BMA.

Example: To demonstrate how the recursive batch mean analysis procedure works, we modified our simulation to print out all intermediate results for the set of simulation runs described as "initial runs" above. The simulation started with the minimum batch size of 125 packets and did not stop until the batch size reaches the maximum size of 16,000 packets. The results from these simulation runs are summaried in table A.3.

In figure A.2, we plot the normalized variance difference, ΔS^2, and relative error in the half-width of the confidence interval, $H_{\alpha,k}/\overline{X}$, as a function of the batch size, m. (The graph for ΔS^2 as a function of m is the same as the graph in figure A.1, where it was used to show how the value of ϵ was chosen.) From figure A.2, we can see that both the variance difference and relative error decrease as the batch size increases. For the system load $G = 0.3$, when the batch size $m = 1000$, it was coincident that both the batch independence and confidence interval tests were satisfied at the same time. However, for the system loads $G = 0.5$ and 0.7, when the batch size $m = 2000$, they both satisfied the batch independence test, but they did not satisfy the confidence interval test until batch sizes $m = 4000$ for $G = 0.5$, and $m = 8000$ for $G = 0.7$. We also see that, similar to the half-width of the confidence interval, the variance estimator difference converges at a small batch size for low system load, and at a large batch size for high system load. Observe that in all three cases, the batch independence test was always satisfied before or at the same time as the confidence interval test. This is because we have a very tight requirement for the confidence interval (i.e., $\gamma = 0.05$). If a higher value of γ is used, then it will be quite

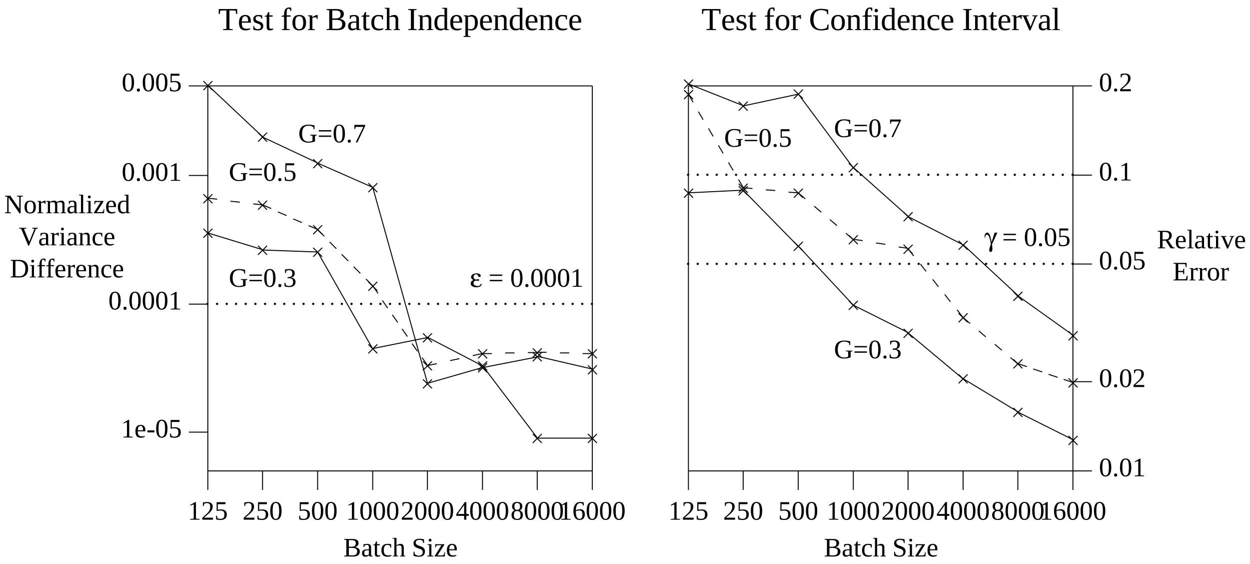

Figure A.2: Batch Independence and Confidence Interval Tests in Recursive BMA.

different, however. From figure A.2, we see that if $\gamma = 0.10$ was used, then, for all three system loads, the confidence interval test would be satisfied much earlier than the batch independence test.

Figure A.3 shows all detailed intermediate batch means from the simulation run for the system load $G = 0.5$. In this figure, we plot both the batch means X_i's (in dashed lines) and the combining batch means Y_j's (in solid lines) for each of the batch sizes (500, 1000, 2000, 4000, 8000, and 16000). From this figure, we see that, when the batch size is large enough ($m \geq 2000$), the curves for X_i's and Y_j's are very close to each other; but when the batch size is too small ($m < 2000$), they are not.

In figure A.4, we apply the *moving average* method described in [LK91] to the (initial) data set with batch size $m = 500$ to find the length of the transient period and the appropriate batch size and compare it to the batch size determined by our recursive batch mean analysis procedure. The moving average $Z_i(w)$ (where w is the window size and is a positive integer such that $w \leq \lfloor k/2 \rfloor$) is defined in [LK91] as follows:

$$Z_i(w) = \begin{cases} \frac{1}{2i-1}\sum_{s=-(i-1)}^{i-1} X_{i+s} & if \quad i = 1, ..., w \\ \frac{1}{2w+1}\sum_{s=-w}^{w} X_{i+s} & if \quad i = w+1, ..., k-w \end{cases}$$

We calculate and plot the resulting moving averages for $w = 1, 2, 4, 6, 8, 10$ in figure A.4. From this figure, we see that clearly the window $w = 4$ is the minimum window size that can smooth out the batch means in the given data set. Thus, the method of moving average indicates that the appropriate batch size to use is $4 \times 500 = 2000$, which is exactly the same as what was found by our recursive batch mean analysis. However, with our recursive batch mean analysis, this is done automatically by the simulation program using our test for independence rather than visually by the user.

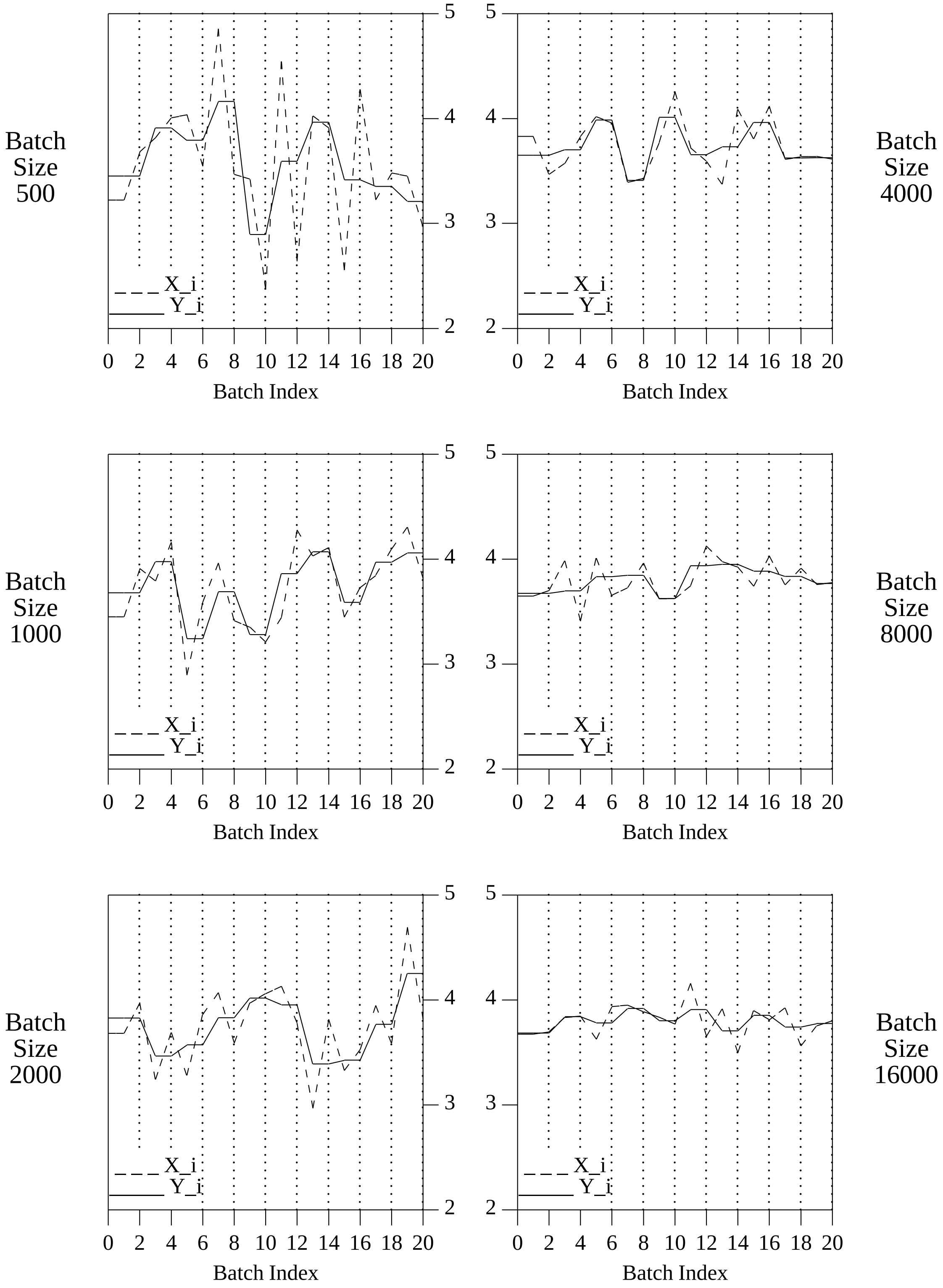

Figure A.3: Batch Means from a Simulation using Recursive BMA.

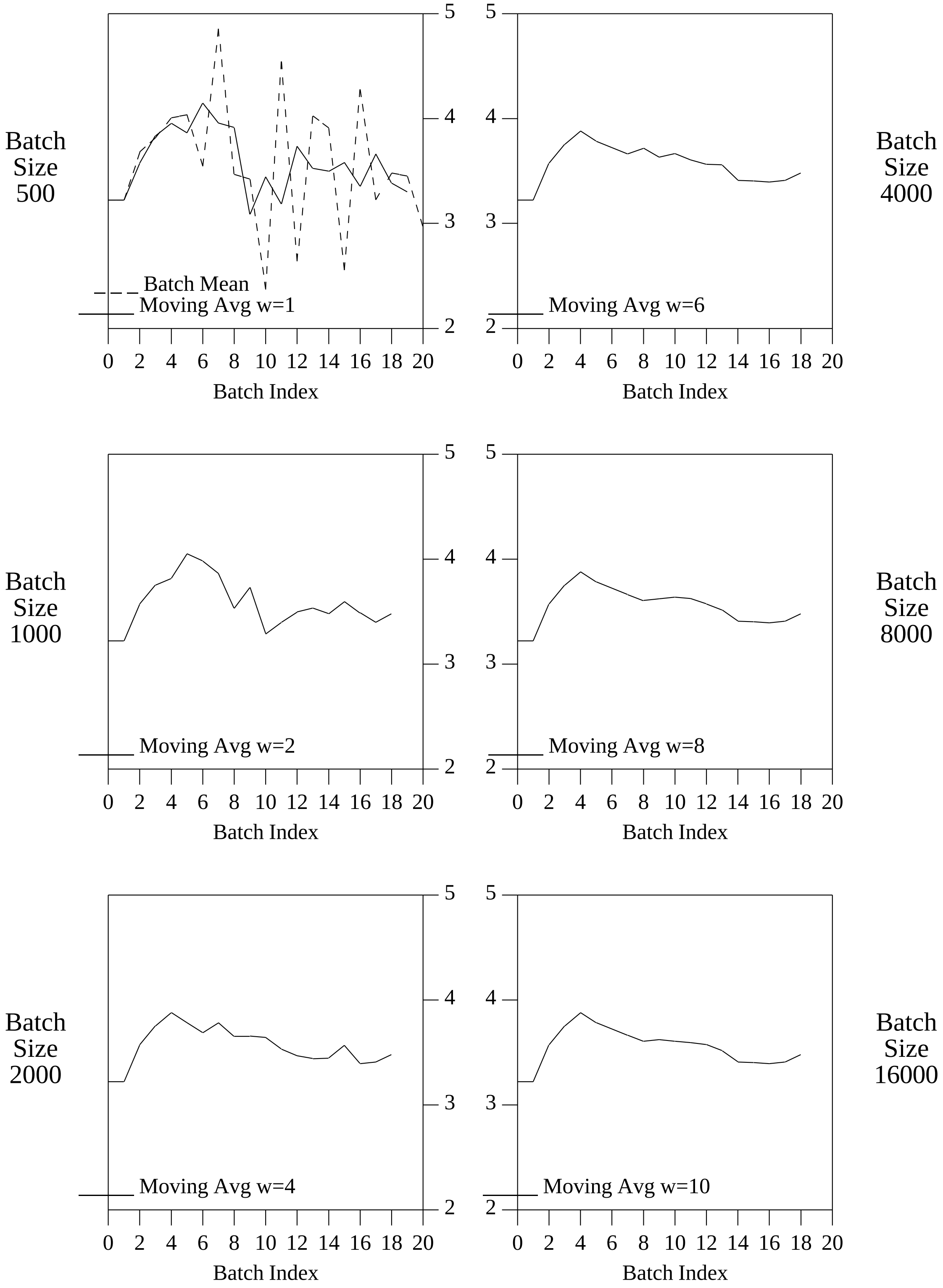

Figure A.4: Moving Average for Data from a Simulation using Recursive BMA.

Bibliography

[AAG78] A. K. Agrawala, J. R. Agre, and K. D. Gordon. The Slotted Ring vs. The Token-Controlled Ring: A Comparative Evaluation. In *IEEE COMPSAC 1987*, pages 674–679, Chicago, IL., 1978.

[Abr70] N. Abramson. The ALOHA System – Another alternative for Computer Communications. In *Proceedings FJCC 1970 AFIPS Conference*, volume 37, pages 281–285, Montevale, New Jersey, 1970.

[ANS86] ANSI. Fiber Distributed Data Interface (FDDI) Token Ring Media Access Control (MAC). American National Std. for Info. Systems ANSI X3T9.5/85-16, February 1986.

[BG87] Dimitri Bertsekas and Robert Gallager. *Data Networks*. Prentice Hall, Englewood Cliffs, New Jersey, 1987.

[BHN+86] Z. L. Budrikis, J. L. Hullet, R. M. Newman, D. Economou, F. M. Fozdar, and R. D. Jeffery. QPSX: A Queued Packet and Synchronous Circuit Exchange. In *Proceedings of the 8th International Conference on Computer Communications*, pages 288–293, Munich, Germany, September 1986.

[Bux81] Werner Bux. Local-Area Subnetworks: A Performance Comparison. *IEEE Transactions on Communications*, COM-29(10):1465–1473, October 1981.

[CGL91] Marco Conti, Enrico Gregori, and Luciano Lenzini. A Methodological Approach to an Extensive Analysis of DQDB Performance and Fairness. *IEEE Journal on Selected Areas in Communications*, 9(1):76–87, January 1991.

[Con63] R. W. Conway. Some Tactical Problems in Digital Simulations. *Management Science*, 10:47–61, 1963.

[CSK88] G. H. Clapp, M. Singh, and S. Karr. Metropolitan Area Network Architecture and Services. In *IEEE GLOBECOM 1988*, pages 38.4.1–38.4.9, 1988.

[DSM83] R. C. Dixon, N. C. Strole, and J. D. Markov. A Token-Ring Network for Local Data Communications. *IBM Systems Journal*, 22(1-2):47–62, 1983.

[FA85] Michael J. Ferguson and Yehuda J. Aminetzah. Exact Results for Nonsymmetric Token Ring Systems. *IEEE Transactions on Communications*, COM-33(3):223–231, March 1985.

[FHT+88] S. Fujita, N. Henmi, I. Takano, M. Yamaguchi, T. Torikai, T. Suzaki, S. Takano, H. Ishihara, and M. Shikada. A 10 Gb/s – 80 km Optical Fiber Transmission

Experiment using a Directly Modulated DFB-LD and a High Speed InGaAs-APD. In *Tech. Dig. Optical Fiber Communications Conference*, pages 25–28, New Orleans, LA, 1988.

[FN69] W. D. Farmer and E. E. Newhall. An Experimental Distributed Switching System to handle Bursty Computer Traffic. In *Proceedings ACM Symposium on Problems in the Optimization of Data Communications*, pages 1–33, October 1969.

[FT84] Micheal Fine and Fouad A. Tobagi. Demand Assignment Multiple Access Schemes in Broadcast Bus Local Area Networks. *IEEE Transactions on Computers*, C-33(12):1130–1159, December 1984.

[Fuh85] S. W. Fuhrmann. Symmetric Queues Served in Cyclic Order. *Operations Research Letters*, 4(3):139–144, October 1985.

[GKDC88] A. H. Gnauck, B. L. Kasper, N. K. Dutta, and T. Cella. An 8 Gbit/s – 76 km Transmission Experiment using a Distributed Feedback Laser at 1.3 μm Wavelength. In *IEEE GLOBECOM 1988*, pages 15.5.1–15.5.4, 1988.

[Gro82] Robert M. Grow. A Timed Token Protocol for Local Area Networks. In *Electro 1982 Conference*, volume 7, pages 17.3.1–17.3.7, Boston, Massachusetts, May 1982.

[Gus88] Riccardo Gusella. The Analysis of Diskless Workstation Traffic on an Ethernet. Technical Report UCB/CSD 87/379, University of California, Berkeley, Computer Science Division, Berkeley, California, December 1988.

[Gus90] Riccardo Gusella. A Measurement Study of Diskless Workstation Traffic on an Ethernet. *IEEE Transactions on Communications*, COM-38(9):1557–1568, September 1990.

[HSW87] R. Heidemann, U. Scholz, and B. Wedding. 5 Gbit/s Transmission System Experiment over 111 km of Optical Fiber. *Electronics Letters*, 23(19):1030–1031, September 1987.

[IEE85a] IEEE. ANSI/IEEE Standard 802.3 – Carrier Sence Multiple Access with Collision Detection (CSMA/CD), 1985.

[IEE85b] IEEE. ANSI/IEEE Standard 802.4 – Token Passing Bus Access Method, 1985.

[IEE85c] IEEE. ANSI/IEEE Standard 802.5 – Token Passing Ring Access Method, 1985.

[IEE90] IEEE. Proposed ANSI/IEEE Standard 802.6/D12 – Distributed Queue Dual Bus (DQDB) Metropolitan Area Network (MAN), February 1990.

[KC90] Harbans Kaur and Graham Campbell. DQDB – An Access Delay Analysis. In *IEEE INFOCOM 1990*, pages 630–635, San Francisco, California, June 1990.

[KGK+87] S. K. Korotky, A. H. Gnauck, B. L. Kasper, J. C. Campbell, J. J. Veselka, J. R. Talman, and A. R. McCormick. An 8 Gbit/s Transmission Experiment over 68 km of Optical Fiber using a Ti:LiNbO3 External Modulator. *IEEE Journal Lightwave Technol.*, LT-5(10):1505–1509, October 1987.

[KL73] L. Kleinrock and S. Lam. Packet Switching in a slotted Satellite Channel. In *National Computer Conference AFIPS Conference*, volume 42, pages 703–710, Montevale, New Jersey, 1973.

[Kle75] Leonard Kleinrock. *Queueing Systems – Volume I: Theory.* John Wiley & Sons, New York, 1975.

[Kle76] Leonard Kleinrock. *Queueing Systems – Volume II: Computer Applications.* John Wiley & Sons, New York, 1976.

[KM89] F. R. Kschischang and M. L. Molle. The Helical Window Token Ring. *IEEE Transactions on Information Theory*, IT-35(2), March 1989.

[KS86] J. Keilson and L. D. Servi. Oscillating Random Walk Models for GI/G/1 Vacation Systems with Bernoulli Schedules. *Journal of Applied Probability*, 23(3):790–802, September 1986.

[KT75] L. Kleinrock and F. A. Tobagi. Packet Switching in Radio Channels: Part I – Carrier Sense Multiple Access Modes and their Throughput-Delay Characteristics. *IEEE Transactions on Communications*, COM-23(12):1400–1416, December 1975.

[Kue79] P. J. Kuehn. Multiqueue Systems with Nonexhaustive Cyclic Service. *Bell System Technical Journal*, 58(3):671–698, March 1979.

[LC79] Averill. M. Law and John S. Carson. A Sequential Procedure for Determining The Length of a Steady-State Simulation. *Operations Research*, 27(5):1011–1025, September 1979.

[LHG82] M. T. Liu, W. Hilal, and B. H. Groomes. Performance Evaluation of Channel Access Protocols for Local Computer Networks. In *Proceedings COMPCON Fall 1982*, pages 417–426, 1982.

[LK86] H. Levy and L. Kleinrock. Queue with Starter and a Queue with Vacations: Delay Analysis by Decomposition. *Operations Research*, 34(3):426–436, May 1986.

[LK91] Averil M. Law and W. David Kelton. *Simulation Modeling and Analysis.* McGraw-Hill Inc., New York, second edition edition, 1991.

[LM90] J. Liu and M. L. Molle. Performance Analysis of Cyclic Reservation Multiple Access Scheme. Computer Systems Research Institute (CSRI), University of Toronto, September 1990.

[Mar89] Peter Martini. The DQDB Protocol – What about Fairness. In *IEEE GLOBECOM 1989*, pages 8.7.1–8.7.5, 1989.

[MB76] Robert M. Metcalfe and David R. Boggs. Ethernet: Distributed Packet Switching for Local Computer Networks. *Communications of The Association for Computing Machinery (ACM)*, 19(7):395–404, July 1976.

[MK85] M. L. Molle and Leonard Kleinrock. Virtual Time CSMA: Why Two Clocks Are Better Than One. *IEEE Transactions on Communications*, COM-33(9):919–933, September 1985.

[MNW+90] Hans R. Müller, M. Mehdi Nassehi, Johnny W. Wong, Erwin Zurfluh, Werner Bux, and Pitro Zafiropulo. DQMA and CRMA: New Access Schemes for Gbit/s LANs and MANs. In *IEEE INFOCOM 1990*, pages 185–191, San Francisco, California, June 1990.

[Mol86] M. L. Molle. An old problem with a new twist – The Helical Window Token Ring. In *Proceedings of the 13th Biennial Symposium on Communications*, pages B.5.1–B.5.3, Queen's Univ., Kingston, Ontario, Canada, June 1986.

[Mol89] M. L. Molle. Analysis of a Class of Distributed Queues with Applications. *Performance Evaluation*, 9:271–286, March 1989.

[Mol90] M. L. Molle. Waiting Times in Token Rings with Helical Window and Window-gated service. In *Architecture and Performance issues of high-capacity Local and Metropolitan Area Networks*, Sophia Antipolis, France, June 1990. NATO Advanced Research Workshop.

[Mol91] M. L. Molle. Prioritized Virtual Time CSMA: Head-of-the-Line Priority Classes with Added Overhead. *IEEE Transactions on Communications*, June 1991.

[MW85] M. L. Molle and Ming-Wen Wu. Prioritized Virtual Time CSMA: Head-of-the-Line Priority Classes without Channel Overhead. In *Proceedings IEEE International Conference on Communications*, pages 35.4.1–35.4.6, Chicago, IL., 1985.

[Nas90a] M. Mehdi Nassehi. CRMA: An Access Scheme for High-Speed LANs and MANs. In *Proceedings IEEE International Conference on Computer Communications*, pages 1697–1702, April 1990.

[Nas90b] M. Mehdi Nassehi. Cyclic-Reservation Multiple-Access Scheme for Gbit/s LANs and MANs based on Dual-Bus Configuration. In *Proceedings EFOC/LAN 1990*, pages 246–251, Munich, Germany, June 1990.

[NBH88] R. M. Newman, Z. L. Budrikis, and J. L. Hullet. The QPSX MAN. *IEEE Communications Magazine*, 26(4):20–28, April 1988.

[Nee79] R. Needham. Systems Aspects of the Cambridge Ring. In *Proceedings of the 7th ACM Symposium on Operating Systems Principles*, pages 82–85, Boston, Massachusetts, 1979.

[NH86] R. M. Newman and J. L. Hullet. Distributed Queueing: A Fast and Efficient Packet Access Protocol for QPSX. In *Proceedings of the 8th International Conference on Computer Communications*, pages 294–299, Munich, Germany, September 1986.

[PB79] B. K. Penney and A. A. Baghdadi. Survey of Computer Communications Loop Networks: Part 1 and 2. *Computer Communications*, 2:165–180 and 224–241, 1979.

[PM87] G. Polyzos and M. L. Molle. Delay Analysis of a Window Tree Conflict Resolution Algorithm in Local Area Network envirnment. In *Proceedings ACM SIGMETRICS 1987 Conference on Measurement and Modeling of Computer Systems*, pages 234–241, Banff, Alberta, Canada, 1987.

[Rod90] Manoel Rodrigues. Evaluating Performance of High-Speed Multiaccess Networks. *IEEE Network Magazine*, pages 36–41, May 1990.

[Sch82] Bruce Schmeiser. Batch Size Effects in The Analysis of Simulation Output. *Operations Research*, 30(3):556–568, May 1982.

[SRC84] J. Saltzer, D. Reed, and D. Clark. End-to-End Arguments in System Design. *ACM Transactions on Computer Systems*, 2(4):277–288, November 1984.

[Sta84] William Stallings. Local Network Performance. *IEEE Network Magazine*, 22(3):27–36, February 1984.

[Sta85] William Stallings. *Data and Computer Communications*. Macmillan, New York, 1985.

[Str87] Norman C. Strole. The IBM Token Ring Network – A Functional Overview. *IEEE Network Magazine*, 1(1):23–30, January 1987.

[Tak86] Hideaki Takagi. *Analysis of Polling Systems*. The MIT Press, Cambridge, Massachusetts, 1986.

[Tan88] Andrew S. Tanenbaum. *Computer Networks*. Prentice Hall, Englewood Cliffs, New Jersey, second edition edition, 1988.

[Tob80] Fouad A. Tobagi. Multiaccess Protocols in Packet Communication Systems. *IEEE Transactions on Communications*, COM-28(4):468–488, April 1980.

[Won89] J. W. Wong. Throughput of DQDB Networks under Heavy Load. In *Proceedings EFOC/LAN 1989*, pages 146–151, Amsterdam, The Netherlands, June 1989.

[WW79] M. V. Wilkes and D. J. Wheeler. The Cambridge Digital Communication Ring. In *Proceedings Local Area Communications Networks Symposium*, pages 288–293, Boston, Massachusetts, May 1979. Mitre Corp. and National Bureau of Standards.