PERFORMANCE COMPARISON OF ERROR CONTROL SCHEMES IN HIGH SPEED COMPUTER COMMUNICATION NETWORKS

Amit Bhargava, James F. Kurose,
Don Towsley, Guy VanLeemput

COINS Technical Report 87-92

September, 1987

Performance Comparison of Error Control Schemes in High Speed Computer Communication Networks [1]

Amit Bhargava[2], James F. Kurose[3], Don Towsley[3], Guy VanLeemput[2]

University of Massachusetts
Amherst, Mass. 01003

Abstract

Advances in both high speed switching and fiber optics technology are changing some of the fundamental premises upon which traditional wide-area network packet-switching technology was first developed. It is thus appropriate to carefully reconsider design issues for protocols built upon these raw high-speed transmission and switching facilities.

We examine the performance of two different approaches for handling the loss and/or corruption of messages as they are transmitted between two end users *in a high-speed network*. In the link-by-link approach, two adjacent nodes in an end-to-end path locally detect and recover from message loss or corruption along their joining link. In the end-to-end approach, recovery is done solely on the basis of a single end-to-end protocol. We develop analytic performance models for comparing the performance of these two approaches; these analytic models are validated by simulation. A result of this paper is that for the range of network parameters of practical interest, an end-to-end approach towards error control is superior to a link-by-link approach, even under assumptions that would overly favor the link-by-link approach, while at the same time requiring fewer network resources (e.g., buffers, computation time) than the link-by-link approach. The models developed in this paper also permit quantitative predictions to be made for the delay, throughput, and buffer overflow probabilities and occupancy statistics of these two error control schemes in a high speed network. The models explicitly consider the effects of channel errors, propagation delays of messages and their acknowledgments, both separate and shared VC buffer pools, finite buffer capacities, buffering of unacknowledged messages, and timeout mechanisms.

---

[1]This work supported in part by the Office of Naval Research grant N00014-87-K-0304

[2]Department of Electrical and Computer Engineering

[3]Department of Computer and Information Science

1. Introduction

Recent advances in both high speed switching [5,15,17] and fiber optics technology [6] have now made it physically possible to provide gigabit communication channels between two users in a packet-switched computer communication network. An often-cited goal of this technology is the transmission of diverse and often high bandwidth traffic, including packetized voice, video, and data, over a single integrated communication network. From a networking standpoint, the increasing communication and switching speeds are changing some of the fundamental premises upon which traditional wide-area network packet-switching technology was first developed. First, the ratio of transmission speed to computation speed has increased dramatically, prompting the need to shift computation-intensive protocols that once resided within the subnetwork to the "edges" of the network [16]. Secondly, the ratio of message propagation delay to message transmission time is also increasing, particularly in the case of wide-area networks. Finally, the use of fiber optics as a communication media is resulting in significantly lower channel error rates. Given these fundamental changes, it is appropriate to now carefully reconsider the design of the protocols which lie between the raw high-speed transmission and switching facilities and the higher-level application protocols in a high-speed network.

In this paper, we examine the performance of two different approaches for handling the loss and/or corruption of messages as they are transmitted between two end users *in a high-speed network (HSN)*. Traditionally, this problem has been handled primarily by the data link protocol [2], which permits two *adjacent* nodes in an end-to-end path to *locally* detect and recover from message loss or corruption along their joining link. We refer to this approach as a *link-by-link* approach towards error control. Note that in this approach, a higher-level end-to-end protocol is still required, for example, to recover from message loss due to node crashes. A second approach is to recover from both message loss and corruption solely on the basis of a single end-to-end protocol. In such an *end-to-end* approach, the traditional functionality of the data link protocol is mostly eliminated (although error detection, but not recovery, might still be maintained) and the burden of maintaining an error-free data path falls on the end-to-end protocol.

In this paper, we develop analytic performance models for comparing the performance of the link-by-link versus end-to-end approaches towards error control; these models are validated by simulation. A result of this paper is that for the range of network parameters of practical interest, *an end-to-end approach towards error control is superior to a link-by-link approach, even when the (higher) node processing demands of the link-by-link approach are ignored.* That is, even under assumptions that would overly favor the link-by-link approach, an end-to-end approach is *still* found to provide equivalent or better performance while at the same time requiring fewer network resources (e.g., buffers, computation time) than the link-by-link approach. The models developed in this paper also permit *quantitative predictions* to be made for the delay, throughput, and buffering requirements of these two error control schemes in a high speed environment.

Several previous researchers have examined the problem of end-to-end versus link-by-link error control, although *not* in the setting of a high speed network. In [7], link-by-link and end-to-end protocols were shown to have similar performance when the number of hops between end-users was small. In all other cases, link-by-link was found to be clearly superior (exactly the opposite of our conclusions for the HSN case). We note that while the analytic models in [7] do consider the effects of finite buffer sizes, they do not consider transmission errors, propagation delays (which, as discussed above, are significant in HSN's), acknowledgment delays (which also may be relatively large due to propagation delays), or the fact that unacknowledged packets occupy buffer space at the sending station (which is also important in a HSN, again due to the large propagation delays). Nonetheless, we do follow the overall approach of [7] in developing our model of end-to-end error control. The comparison of link-by-link versus end-to-end in [9] indicated that the end-to-end approach was superior. However, we conjecture that this resulted primarily from the fact that in [9] each message was segmented in the end-to-end case, whereas a message was transmitted as a unit in the link-by-link case. Also, the effects of propagation delays, finite buffer capacities, and non-instantaneous acknowledgments (which we have found to be important in a HSN) were not considered. Finally, a model of link-by-link (only) error control was developed first in [13] and extended in [10]. This model considers the effect of blocking, propagation delays and non-instantaneous acknowledgments. Our model of each node in a link-by-link approach (only) thus closely resembles this model, although our solution technique is considerably simpler due to the structure of our virtual circuit model.

In the following section we describe our general approach towards modeling message flow between the two end-users of a virtual circuit in which each VC maintains separate buffer pools at each node. In section 3, we then describe the end-to-end approach in more detail and develop a queueing-theoretic model for evaluating the performance of end-to-end error control in such a VC; the results of the analytic model are corroborated through simulation. Various modeling assumptions used in both this section and the next are also discussed. Section 4 presents the corresponding model for link-by-link error control and Section 5 compares the performance of the two approaches for various network parameters of interest. In section 6, we study the case in which virtual circuits routed over the same outgoing link at a node share a common buffer pool at the node; our overall conclusions for this case are identical to those drawn from the separate buffer pool model. Finally, section 6 summarizes this paper.

2. Network Model

The network model used for comparing link-by-link versus end-to-end error control is shown in figure 1a. As in [7,9], we focus on the traffic, λ_{VC}, entering a virtual circuit (VC) of M links which begins at some source node (labeled 1 in figure 1) and terminates at some destination node (labeled $M+1$ in figure 1). In sections 3, 4, and 5 we study the case in which the VC is allocated

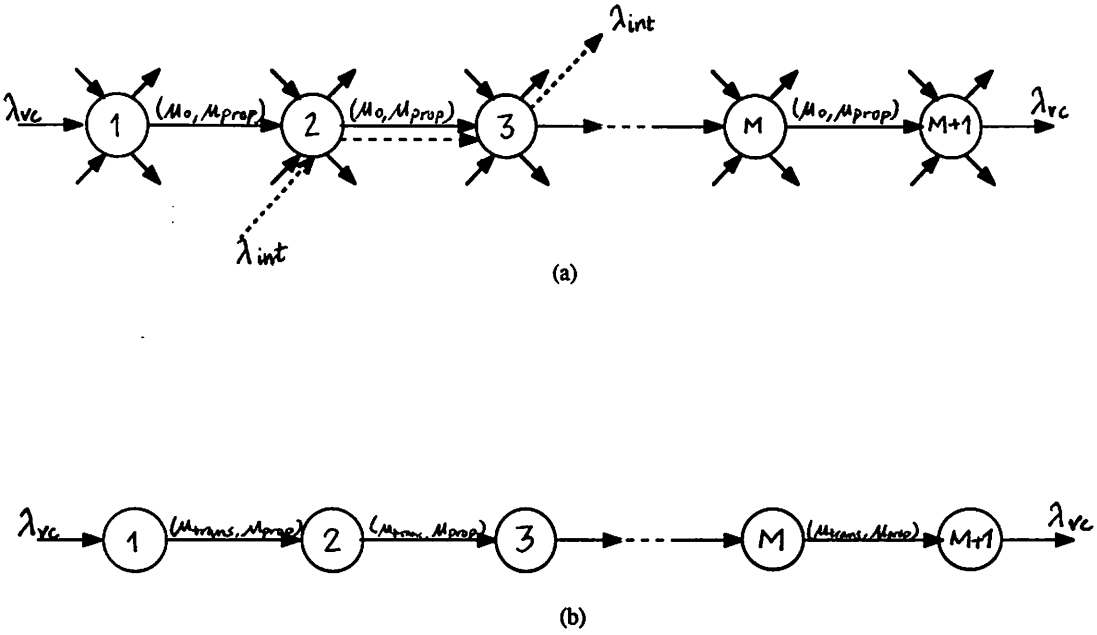


Figure 1: Virtual circuit model

a finite number of buffers, K_i, at all but the first link, $i = 2, \cdots, M$ and that there is enough buffer capacity at the first link to store all new packets. In section 6, we consider the case in which *all* VC's routed over the same outgoing link share the same buffer pool at this link.

We denote the message transmission rate (capacity) of each link, i, along the VC by μ_0^i and denote the mean propagation delay between nodes i and $i+1$ as $1/\mu_{prop}^i$. Note that several VC's may share the same physical link (although maintain separate buffer pools, as discussed above). For example, in figure 1a a VC of rate λ_{int} shares the link between nodes 2 and 3 with λ_{VC}. In sections 3,4 and 5, we adopt the approach of [11,14] (also implicitly used in [7,9]) for modeling message traffic along a virtual circuit with separate buffer pools. We thus model the effects of this interfering traffic by reducing the communication capacity seen by the λ_{VC} traffic along each outgoing link of the VC by some appropriate amount. Our model of the M-hop virtual circuit is thus shown in figure 1b, where μ_{trans}^i now represents the effective (reduced) service capacity of link i on the λ_{VC} virtual circuit. In section 6, our shared buffer pool models explicitly consider the effects of interfering traffic.

3. End-to-End Error Control

In this section we develop an analytic model for studying the performance of the end-to-end error control scheme. Our model explicitly considers the effects of message propagation delays, finite buffer capacities (blocking), channel errors, and the use of a timeout mechanism in the error control scheme. We begin by describing the operation of the end-to-end error control protocol in detail. We then develop a two-level performance model of this scheme. The lower level model, developed in section 3.2, models the interaction between two adjacent nodes along the VC. The higher level model, discussed in section 3.3, incorporates the M lower level models into a single VC-

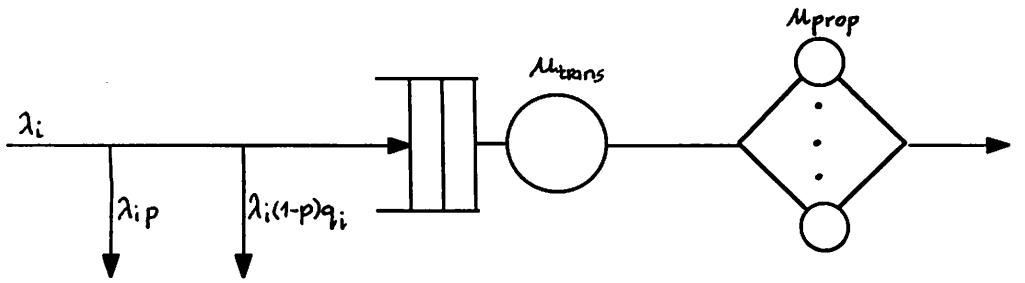


Figure 2: Lower level queueing model for end-to-end error control

level model. The analytic results are compared with simulation results in section 3.4 and various assumptions underlying the model are discussed.

3.1 Description of End-to-End Error Control

In an end-to-end approach towards error control, when the VC source node first transmits a message, it buffers a copy of the message until an ACK message is received from the *final* destination node on the VC. The destination station generates an ACK message only when a message from the source is received correctly. If an ACK is not received within a specified *timeout period*, the message is assumed lost or corrupted and the source node retransmits a copy of the message along the VC. In our study, we will assume (as in all previous studies [10,13,7,9]) that a timeout occurs only when a message has, in fact, been lost or corrupted. We note that in a HSN, where the known, fixed length, propagation delays may be significantly greater than the variable message queueing and transmission times, this assumption is even more justified than in the non-HSN case.

The function of an intermediate node in the end-to-end approach is simple; it need only check arriving messages for errors, discard any incorrectly received messages or any messages that overflow the buffer, and queue all correctly received messages for transmission on the outgoing link of the VC.

3.2 Link-Level Model for End-to-End Error Control

Our queueing model for studying the performance of an *individual link* along a VC under the end-to-end error control scheme is shown in figure 2. For simplicity, we assume that all nodes along the VC are homogeneous, i.e., have the same message transmission rate, propagation delay and

fixed number of buffers; this assumption can be easily relaxed at the expense only of some slightly more complicated notation.

The transmission facility for the i-th outgoing link and its buffers are modeled by an M/M/1/K queueing system, where K represents the fixed number of buffers available for this VC at all nodes other than the source node. As mentioned earlier, the source node has an infinite number of buffers. In section 3.4, we discuss the assumptions of exponential service and inter-arrival times implicit in the M/M/1/K model. The link propagation delay is modeled by an infinite server queue in which a message has a fixed delay of $1/\mu_{prop}$. Note that a message releases its buffer immediately upon completion of its transmission. As we will see, in the link-by-link case, we must explicitly model the fact that a message continues to hold a buffer at each transmitter, i, while it (and its ACK message, if any) propagates across link i.

When a message arrives at node i, one of three events can occur. First, with probability p the message may be received in error, in which case it is discarded (the eventual retransmission of this message at the source node will be incorporated into the higher-level model described in the following subsection). Second, a message may be received correctly but there may be no buffers available. In this case, the message is also discarded and will again be eventually requeued at the source node. In our analytic model we assume that each arriving message *independently* finds a full buffer with probability q_i, where q_i is the steady state probability that each of the K buffers at node i are in use. (Note that q_i will be different for each node along the VC, and hence we must explicitly retain the subscript.) The use of this independence assumption, also adopted in the previously cited earlier studies of error control schemes, will also be validated by our simulation results. The third possibility is that the message is received correctly and is allocated a buffer.

As discussed above, the arrival of messages to node i is modeled as a Poisson process with mean λ_i. Given the above assumptions and given a value of λ_i, the individual link-level model of end-to-end error control can now be easily solved. The average number of messages in an intermediate node i, $E[L_{1,i}]$, is simply the average queue length of the M/M/1/K queue (recall that there are an infinite number of buffers at the source node). Defining:

$$\rho_{1,i} = \lambda_i(1-p)/\mu_{trans}$$

we have [1]:

$$E[L_{1,i}] = \begin{cases} \rho_{1,i}/(1-\rho_{1,i}), & i = 1, \\ \rho_{1,i}/(1-\rho_{1,i}) - (K+1)\rho_{1,i}^{K+1}/(1-\rho_{1,i}^{K+1}) & i = 2, \cdots, M. \end{cases}$$

The average time a packet spends queueing for transmission plus its actual transmission time is denoted $E[W_{1,i}]$. Using Little's formula we have:

$$E[W_{1,i}] = E[L_{1,i}]/\lambda_i(1-q_i-p+q_ip) \tag{1}$$

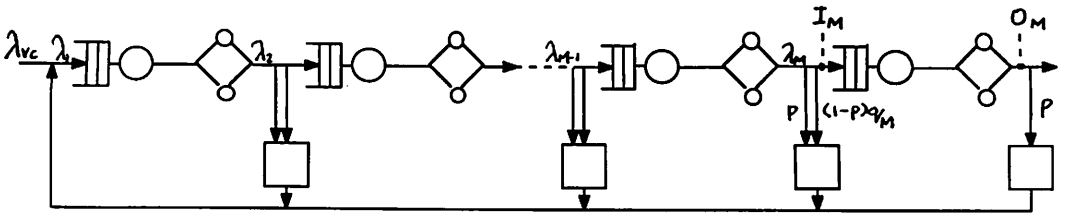


Figure 3: VC-level model of End-to-end error control

and

$$E[W_i] = E[W_{1,i}] + 1/\mu_{prop} \tag{2}$$

where $E[W_i]$ is the average delay (queueing, transmission, and propagation) a message experiences passing through link i.

Finally, the probability, q_i, that all buffers at node i are full (and hence node $i-1$ is blocked) is given by [1]:

$$q_i = Pr[L_{1,i} = K] = \frac{(1-\rho_{1,i})\rho_{1,i}^K}{1-\rho_{1,i}^{K+1}}, \quad i = 2, \cdots, M. \tag{3}$$

Equations 2 and 3 give the average delay and blocking probability at node i. Note that they are each functions of the still undetermined values, $\{\lambda_i\}$; these values will be obtained in the following section, in which the individual link-level models are combined into a single VC-level model.

3.3 VC-Level Model for End-to-end Error Control

Figure 3 combines the M individual link-level models into a single model for the VC. Note that a rejected packet is removed from the VC as soon as an error occurs. As we will see, we need not explicitly consider here the amount of time that elapses between the removal of a message from the VC and its eventual retransmission at the source node. Instead, we need only consider the overall flow rate of messages back to the source node.

In order to actually solve for the blocking probabilities and average delays along the links, we must still determine $\{\lambda_i\}$. The difficulty here is that λ_i and q_i are functionally dependent on each other. Moreover, $\lambda_i \neq \lambda_j \neq \lambda_{VC}$ due to the different node blocking probabilities and the fact that a message is discarded at the point where an error is first detected. Our approach will be to first

solve for the arrival rate and blocking probabilities for the last hop (M) on the VC, use these values to solve for corresponding values for link $M-1$, and then continue to solve the VC-level model working from the end of the VC back towards the front.

Clearly, the message output rate for link M at the point O_M in figure 3 is given by $\lambda_{VC}/(1-p)$. By conservation of flow, the input rate to link M at the point I_M must also equal this value and thus

$$\lambda_M = \frac{\lambda_{VC}}{(1-p)^2(1-q_M)} \tag{4}$$

Substituting this expression for λ_M into equation 3 yields a single equation in a single unknown (q_M) and we may thus solve for q_M immediately. Given a value for q_M, equation 4 gives a value for λ_M. Finally, given a value for λ_M, equation 2 can be solved for the average delay through link M. Once λ_M is known, q_{M-1}, λ_{M-1} and $E[W_{M-1}]$ can be obtained using similar arguments. Working backwards along the VC, the set of unknown values, $\{q_i\}, \{\lambda_i\}$ and $\{E[W_i]\}$ are thus obtained.

Finally, in order to obtain the average delay along the VC, several additional quantities must be defined:

f_i - the probability that a message is removed from the VC at node i. From the link-level model, we have $f_i = q_i + p - q_i p$.

p_{fail} - the probability a particular transmission of a packet from the source node will *not* successfully reach the destination node:

$$p_{fail} = \sum_{i=1}^{M} f_i \prod_{j=1}^{i-1}(1-f_j).$$

N_t - the number of times a packet must be *retransmitted* before it is successfully received at the destination station. Assuming the outcome of each packet retransmission is independent:

$$P[N_t = k] = p_{fail}^k(1-p_{fail})$$

and

$$E[N_t] = \sum_{i=1}^{\infty} k\, P[N_t = k] = \frac{p_{fail}}{1-p_{fail}}. \tag{5}$$

T_{ee} - the end-to-end timeout interval for the VC. The timeout interval begins when a message's *transmission* is completed at the VC source node. If the timeout interval expires and an ACK message has not been received, the message has been lost (see above) and a copy is thus retransmitted.

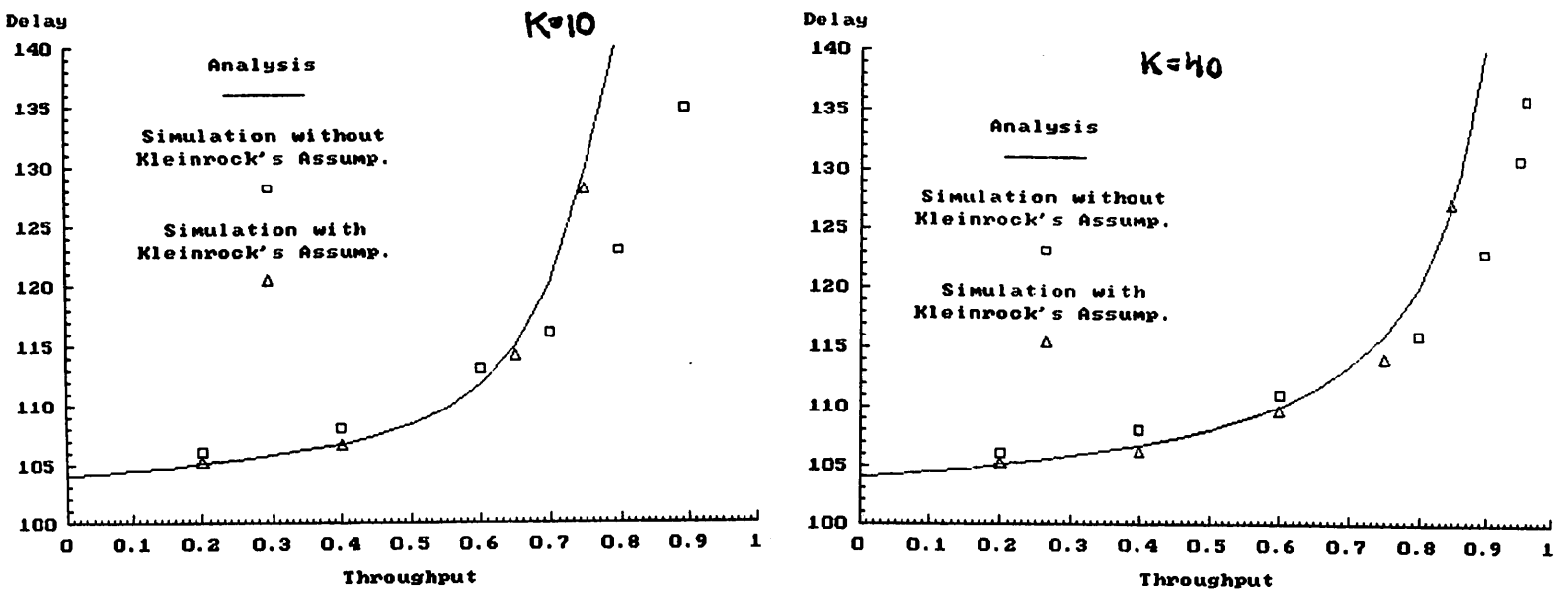


Figure 4: Comparison of simulation and analysis, the end-to-end case

$E[W_{ee}]$ - the average delay experienced by a packet between its initial arrival at the VC source node and its eventual successful receipt at the VC destination node, with an end-to-end error control protocol.

A packet's delay between its initial arrival and eventual successful reception at the destination results from the N_t timeout intervals during which it is unsuccessfully transmitted to the destination node plus the delays it experiences along the VC during its successful trip to the destination station. Hence:

$$E[W_{ee}] = E[N_t](E[W_{1,1}] + T_{ee}) + \sum_{k=1}^{M} E[W_k]. \tag{6}$$

3.4 Discussion

In order to validate the assumptions underlying our analytic model we compare analytic results with performance results obtained via simulation. All time-related values are given in units of the average message transmission time (i.e., in units of $1/\mu_{trans}$). We simulated a 4 hop virtual circuit, with a link propagation delay 25 times the message transmission time, $p = 10^{-5}$, and $T_{ee} = 224$. Given a transmission rate of 100 Mbits/sec, these values would correspond to a packet size of 1000 bits, a bit error rate $= 10^{-8}$ and an inter-node distance = 50 km.

Fig 4 shows both simulation and analytic results plotting $E[W_{ee}]$ versus the virtual circuit arrival rate, λ_{VC}, for the cases of $K = 10$ and $K = 40$ buffers at each intermediate node. Note that the y-axis values do not begin at 0. We note that as a general trend, as the number of buffers increases,

the maximum throughput the protocol can support also increases. Also, for a fixed throughput, the time delay decreases (particularly at higher throughput values) as the number of buffers increases. Both these results are as expected, since a VC with fewer buffers blocks more often than one with more buffers.

In figure 4, simulation results are indicated as point values while analytic results are represented by continuous curves. In these results, exponentially-distributed message lengths were used in both the simulation and the analysis (the appropriateness of this assumption will be examined later when the link-by-link versus end-to-end approaches are compared). Two sets of simulation results are shown in each graph. In the set of simulation results plotted using triangles, message lengths were independently generated at each node along the VC (Kleinrock's independence assumption); note that these results track the analytic results quite closely. Thus the assumption (used in the analysis only) that each message independently finds all buffers full with probability q_i is a good one.

In the set of simulation results plotted with squares, Kleinrock's independence assumption was not used, i.e., a packet's length was preserved (after being chosen from an exponential distribution) as it passed through the VC. These simulation results show noticeably lower delay than the analytic results at high throughput values. This would indicate that the use of Kleinrock's assumption in the analysis tends to give *pessimistic* performance predictions; an observation also noted in [7]. However, as we will see, for the purposes of comparing link-by-link versus end-to-end error recovery this difference will not play a significant role, for *even using the more pessimistic analytic results derived above,* the end-to-end approach is *still* found to be superior.

4. Link-by-Link Error Control

In this section we develop an analytic model for studying the performance of the *link-by-link* error control scheme. The model again explicitly considers the effects of message propagation delays (of both messages and their ACK), finite buffer capacities (blocking), channel errors, and timeouts. Also, we explicitly model the fact that *a message continues to hold a buffer at each transmitter, i, while it (or its ACK message, if any) propagate across link i*. We begin by describing the operation of the link-by-link error control protocol. As in the end-to-end case, we then again develop a two-level performance model and compare the analytic results with those obtained via simulation.

4.1 Description of Link-by-Link Error Control

In the link-by-link approach, *each node* along the VC buffers a copy of each transmitted message until an ACK message is received for that message from the *next node* along the VC. A node generates an ACK message for each message it receives correctly and sends this ACK back to the preceding node on the VC. If an ACK is not received within a specified *timeout period*, the message

is assumed lost or corrupted and a node retransmits a copy of the message to the next node along the VC. We again assume (as before) that a timeout occurs only when a message has, in fact, been lost or corrupted.

4.2 Link-Level Model for Link-by-Link Error Control

Our model for studying the performance of an *individual link* along a VC is slightly more complicated than in the end-to-end case because we must explicitly model the fact that a message continues to hold buffer space in an intermediate node while it and its ACK (if any) propagate through the channel. This behavior may be modeled using the queueing network shown in figure 5(a). For simplicity, we again assume that all nodes along the VC are homogeneous. The FCFS M/M/1 queue with rate $1/\mu_{trans}$ again models the transmitter at node i (although as discussed below, the number of customers waiting in this queue does *not* represent the number of messages buffered at node i). Customers in the infinite server queue represent messages for which a timeout interval has been started; each customer experiences a *fixed* delay of $1/\mu_2 = 2/\mu_{prop}$ in this queue. We may think of this as modeling the propagation delay of a message from node i to $i+1$ plus the propagation of the ACK from $i+1$ back to i, although the analogy is not exact since a lost or corrupted message generates no ACK. Thus, the total number of messages actually buffered at node i is given by the sum of the number of customers in the FCFS and IS queues. The fixed buffer capacity (K) and the fact that a node buffers a message while waiting for an ACK is modeled by not permitting packets to enter the node/link queueing model (i.e., enclosed by dashed lines in figure 5) when this sum is equal to K.

The queueing system shown in figure 5(a) can be modeled as the *product form closed queueing network* illustrated in figure 5(b), where the number of customers in the network corresponds to the buffer capacity K of the VC link. An additional FCFS queue has been introduced to model the arrival of packets to the link. This queue is assigned a Poisson service rate of λ_i to correspond to the packet arrival rate to link i. Note that when there are K customers in the node/link queueing system, the departure rate from the new queue (and hence the arrival rate into the node/link model) is zero, exactly the same behavior shown in figure 5(a).

We are interested in the stationary distribution of the number of packets in the transmission queue, $L_{1,i}$ and the number of packets for which the timeout interval has started, $L_{2,i}$. Let $\pi_{i,m,n} = P[L_{1,i} = m, L_{2,i} = n]$, $0 \leq m,n \leq K$, $m+n \leq K$. Because this queueing network satisfies the assumptions required for product form, i.e., exponential service times for FCFS queues and general service times for delay servers, these probabilities can be written as [3,12]

$$\pi_{i,m,n} = G^{-1}\rho_{1,i}^{m}\frac{\rho_{2,i}^{n}}{n!},\ 0 \leq m,n \leq K,\ m+n \leq K \tag{7}$$

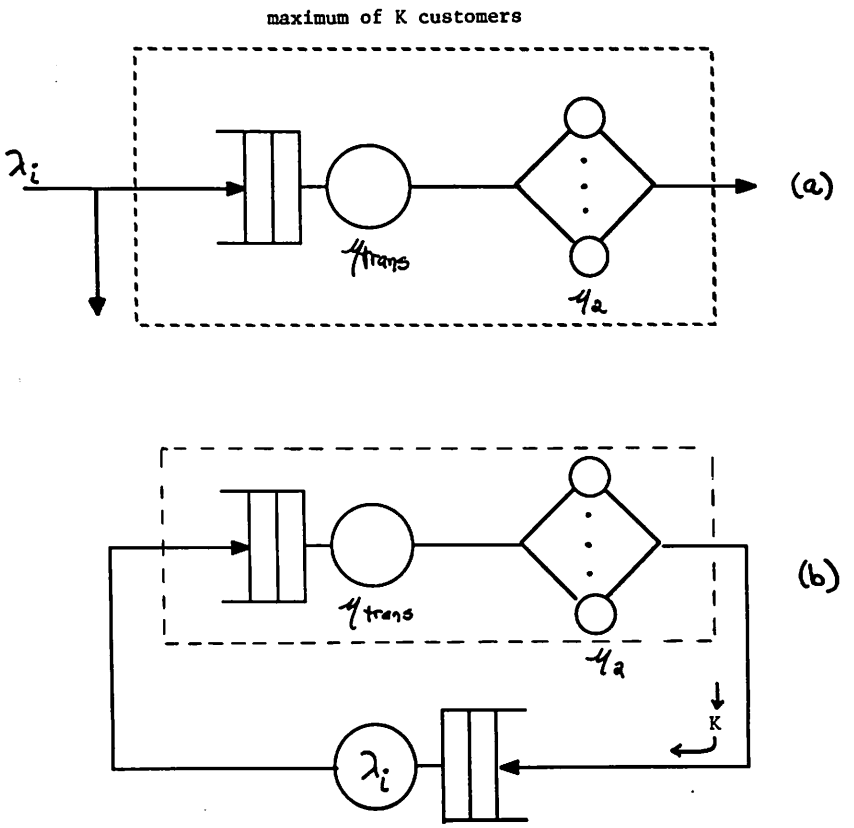


Figure 5: (a) Lower level queueing model for link-by-link protocol. (b) Analogous closed queueing network model.

where

$$\begin{aligned}\rho_{1,i} &= \lambda_i(1-p)/\mu_{trans}, \\ \rho_{2,i} &= \lambda_i(1-p)/\mu_2.\end{aligned}$$

Here G^{-1} is a normalization constant given as

$$\begin{aligned}G &= \sum_{n=0}^{K}\sum_{m=0}^{K-n}\rho_{1,i}^{m}\frac{\rho_{2,i}^{n}}{n!}, \\ &= \frac{1-\rho_1}{\left[S_K(\rho_2)-\rho_1^{K+1}S_K(\rho_2/\rho_1)\right]} \qquad (8)\end{aligned}$$

where

$$S_K(x)=\sum_{j=0}^{K}\frac{x^j}{j!}.$$

The probability, q_i, that all buffers in node i are full is given by:

$$q_i=\sum_{m=0}^{K}\pi_{i,m,K-m} \qquad (9)$$

and the average number of messages waiting to be transmitted along the VC is given by:

$$E[L_{1,i}]=\sum_{m=0}^{K}m\sum_{n=0}^{K-m}\pi_{i,m,n} \qquad (10)$$

From Little's law, the amount of time a message spends actively queued waiting for transmission (i.e., as opposed to waiting for a timeout to expire) is given by:

$$E[W_{1,i}] = E[L_{1,i}]/\lambda_i(1-q_i-p+q_ip) \qquad (11)$$

and the time spent propagating in the channel is again $1/\mu_{prop}$.

As in the case of end-to-end error control, we define $E[W_i]$ to be the average delay (queueing, transmission, and propagation) a message experiences passing through link i. Note that in general $E[W_i] \neq E[W_{1,i}] + 1/\mu_{prop}$ since in the link-by-link case, a message may be retransmitted several times before it is successfully received at node $i+1$. However, we may proceed in a manner analogous to that used in Section 3.3.3. and compute $E[W_i]$ as follows. The following definitions will be useful.

N_t^i - the number of times a packet must be *retransmitted* at node i before it is successfully received at the station $i+1$. Assuming the outcome of each packet retransmission is independent:

$$P[N_t^i = k] = f_{i+1}^k(1 - f_{i+1})$$

where again $f_{i+1} = q_{i+1} + p - q_{i+1}p$ is the probability that a message is either received in error or finds a full buffer at node $i+1$. We thus have

$$E[N_t^i] = \sum_{i=1}^{\infty} k\, P[N_t = k] = \frac{f_{i+1}}{1 - f_{i+1}} \tag{12}$$

T_{ll} - the link timeout interval. The timeout interval again begins when a message's *transmission* is completed. In our examples, it is taken to be equal to twice the propagation delay, i.e., $T_{ll} = 2/\mu_{prop}$.

A packet's delay between its initial arrival at node i and its eventual successful reception at node $i+1$ results from the N_t^i timeout intervals (during which it is not successfully transmitted) plus the delays it experiences during its successful transmission. Hence:

$$E[W_i] = E[N_t^i](E[W_{1,i}] + T_{ll}) + E[W_{1,i}] + 1/\mu_{prop}. \tag{13}$$

Equations 13 and 9 give the average delay and blocking probability at node i. As in the end-to-end case, they are each functions of the still undetermined values, $\{\lambda_i\}$.

4.3 VC-Level Model for Link-by-Link Error Control

The link-by-link VC-level model is *identical* to the end-to-end model; the only difference is that the blocking probabilities are computed using equation 9 rather than equation 3.

The model thus again consists of M link-level models in tandem. In order to sustain an output rate λ_{VC} of *correctly* received messages from the last link, the overall throughput rate through link M must be $\lambda_{VC}/(1-p)$. Since a fraction $(1-p)(1-q)$ of the messages arriving to link M will either be blocked or in error, the overall arrival rate of messages (first time arrivals plus retransmitted messages) to link M must also be given by equation 4. Once again we can now solve for the blocking probability, arrival rate, and average delay for the last hop (M) on the VC, use these values to solve for corresponding values for link $M-1$, and again continue to solve the VC-level model working from the end of the VC back towards the front.

A packet's delay between its initial arrival at the VC source node and its eventual successful reception at the VC destination node is simply the sum of the individual delays it experiences in each link. Thus, for the expected value of this delay, $E[W_{ll}]$, we have:

$$E[W_{ll}] = \sum_{i=1}^{M} E[W_i] \tag{14}$$

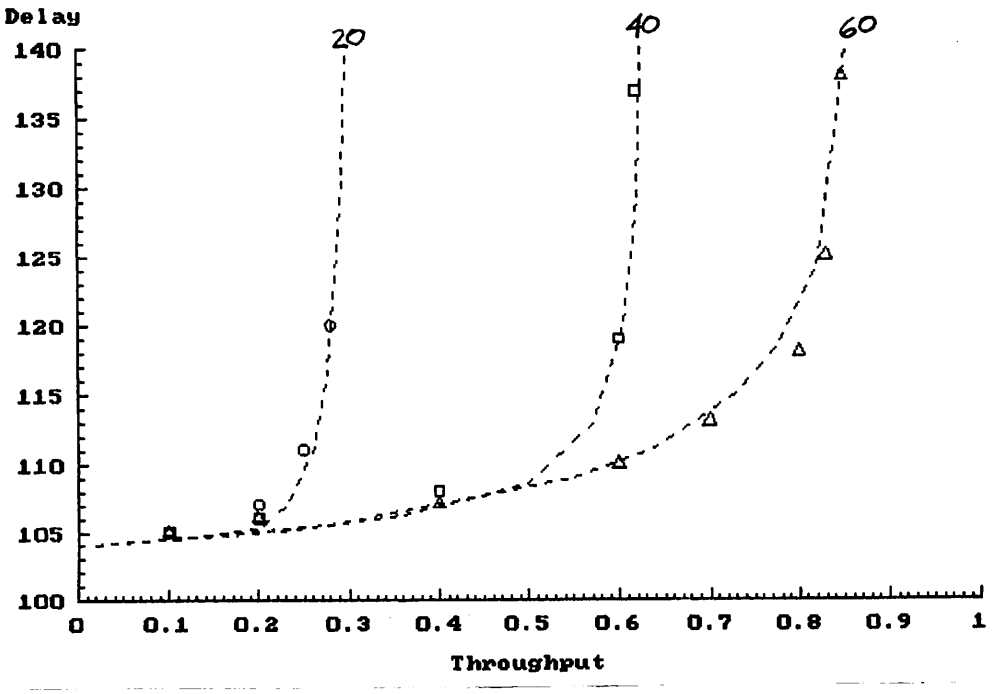


Figure 6: Comparison of simulation and analysis, the link-by-link case

4.4 Discussion

In order to again validate the assumptions underlying our analytic model we compare analytic results with performance results obtained via simulation. Figure 6 shows both simulation and analytic results plotting $E[W_{ll}]$ versus the virtual circuit arrival rate, λ_{VC}, for the cases of $K = 10, 20$ and 40 buffers at each intermediate node. All network parameters are identical to those used in figure 4, and the link-level timeout interval, T_{ll}, was taken to be 56. We again note that as a general trend, as the number of buffers increases, the maximum throughput the protocol can support also increases. We also note that the simulation results (indicated as point values) were obtained *without* Kleinrock's independence assumption. The obviously close correspondence between the analytic and the simulation results again indicates that reasonable assumptions were indeed introduced into the analytic model.

5. Comparison of Link-by-Link versus End-to-End

Having developed and validated our analytic models of the link-by-link and end-to-end error control protocols, we now use these models to compare the performance of these protocols for various network parameters of interest.

In figure 7 we compare the average end-to-end delay along the VC (i.e., the average amount of time between a message's initial arrival at the VC source node and its successful reception at the VC destination node) for the link-by-link and end-to-end protocols for the indicated intermediate-node buffer sizes. The results shown are for the network parameters given in sections 3.4 and 4.4.

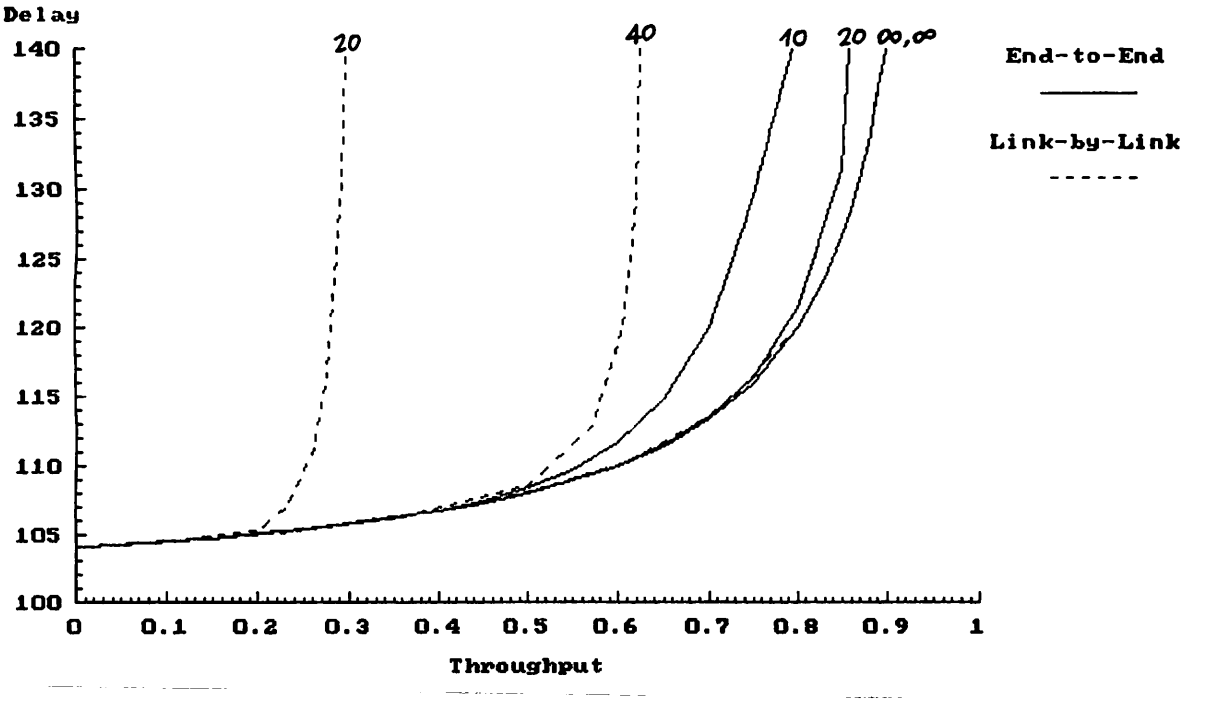


Figure 7: Link-by-link versus end-to-end, baseline case

The results indicate that for low throughput (λ_{VC}) values, the performance of the end-to-end and link-by-link schemes is essentially identical (our numerical results indicate that link-by-link performs slightly better, but by an amount small enough so that the plotted results are indistinguishable). As the VC throughput increases, however, for the same number of buffers, the link-by-link scheme reaches saturation throughput sooner than the end-to-end scheme; consequently, higher delays are also experienced as the link-by-link throughput approaches its saturation value. This results from the fact that the link-by-link scheme continues to buffer a message as the message (and its ACK) propagate across a link; the end-to-end protocol releases the buffer immediately upon completing message transmission, effectively letting the *media* buffer a copy of the message. Thus at higher throughputs, nodes are blocked less frequently in the end-to-end scheme, and hence the delays are lower and the maximal throughput higher.

Other interesting results are also illustrated in figure 7. Note that when either of the protocols is not operating near its saturation throughput, increasing the number of buffers at the intermediate nodes has little effect. Also, when the protocol is operating near saturation throughput, there is little to be gained by increasing the number of buffers beyond a certain value; in the end-to-end scheme, for example, the performance with 20 buffers at each node is quite close to the performance with an infinite number of buffers at each node.

In figure 8, the message error probability has been increased by a factor of 100; this would correspond to a bit error rate of 10^{-6}, high for a fiber optic link but reasonable for other transmission media. In this case, for a fixed number of buffers, the link-by-link scheme performs marginally better than the end-to-end scheme at lower throughput values. This results from the fact that the link-by-link protocol can recover quickly when an error occurs, i.e., after the link-by-link timeout interval has expired, while the end-to-end approach must wait for the longer end-to-end timeout to occur. As the throughput increases, however, the increased buffering requirements, and hence

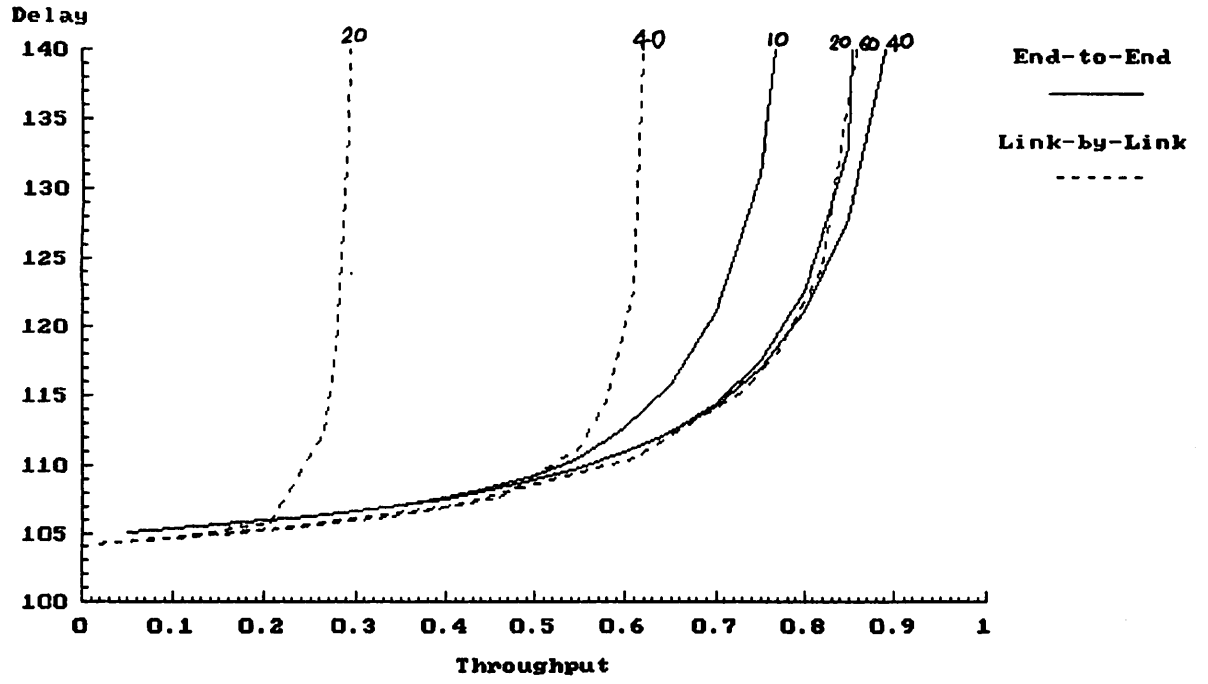


Figure 8: Link-by-link versus end-to-end, $p = 0.001$

blocking probabilities, become significant and the end-to-end scheme again eventually achieves superior performance.

In figure 9, the propagation delay of all links has been decreased to 5 times the message transmission time, and $p = 10^{-5}$ once again. Since the intermediate node buffers are now occupied for a shorter time, the relative performance of link-by-link improves with respect to the baseline case of figure 7. The blocking probabilities for the end-to-end case are unaffected, and the delay versus throughput curves for the end-to-end scheme are simply shifted down, due to the decreased propagation delays.

When the propagation delay of a single link is *increased*, the relative performance of link-by-link worsens. Figure 10 shows the performance of the two schemes when the propagation delay of link 3 is increased by a factor of five. Again, the relative shape of the end-to-end curves remains unchanged, although the absolute value of the average delay is now higher. However, the longer link now becomes a bottleneck for the link-by-link scheme and its performance shows significant deterioration. These effects can be mitigated, of course, but only at the expense of providing additional buffers at the link with the longer propagation delay. For example, the link-by-link curves for 40 and 60 buffers would shift to the curves labeled A and B, respectively, if the number of buffers at this link were doubled.

Figure 11 shows the effect of increasing the number of hops on the VC from 4 to 10. In order to understand the effects of increasing the number of hops, it is worthwhile reviewing the advantages and disadvantages of the end-to-end and link-by-link schemes. End-to-end has the

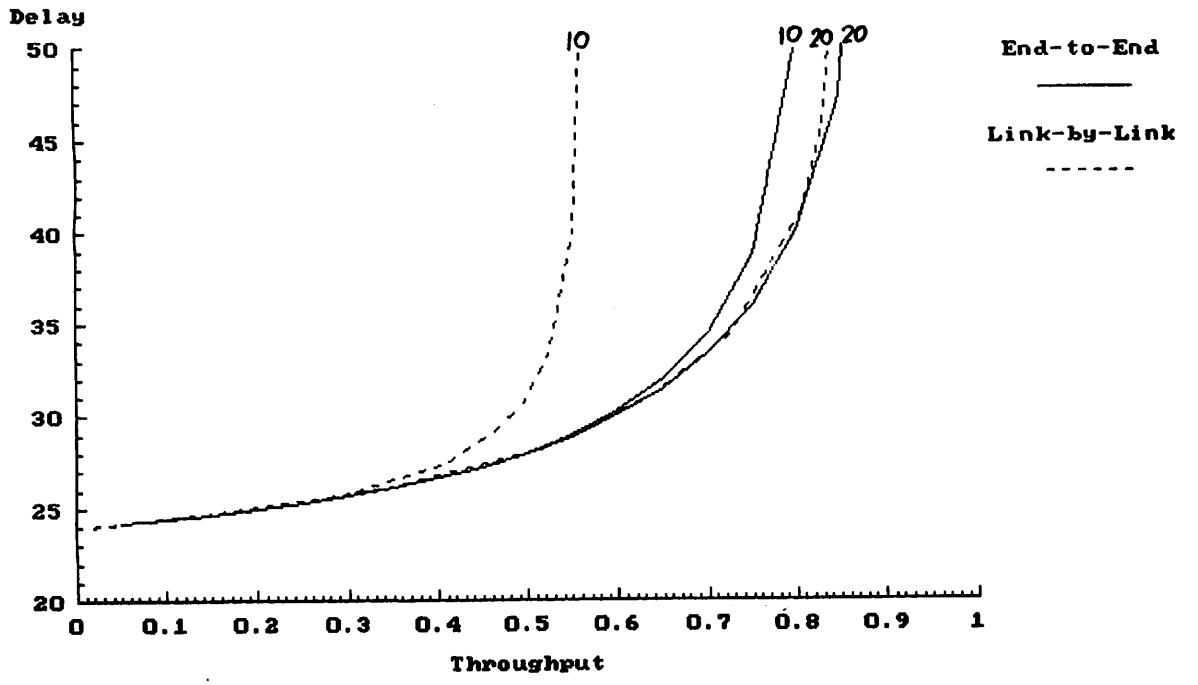


Figure 9: Link-by-link versus end-to-end, effects of a decreased propagation delay

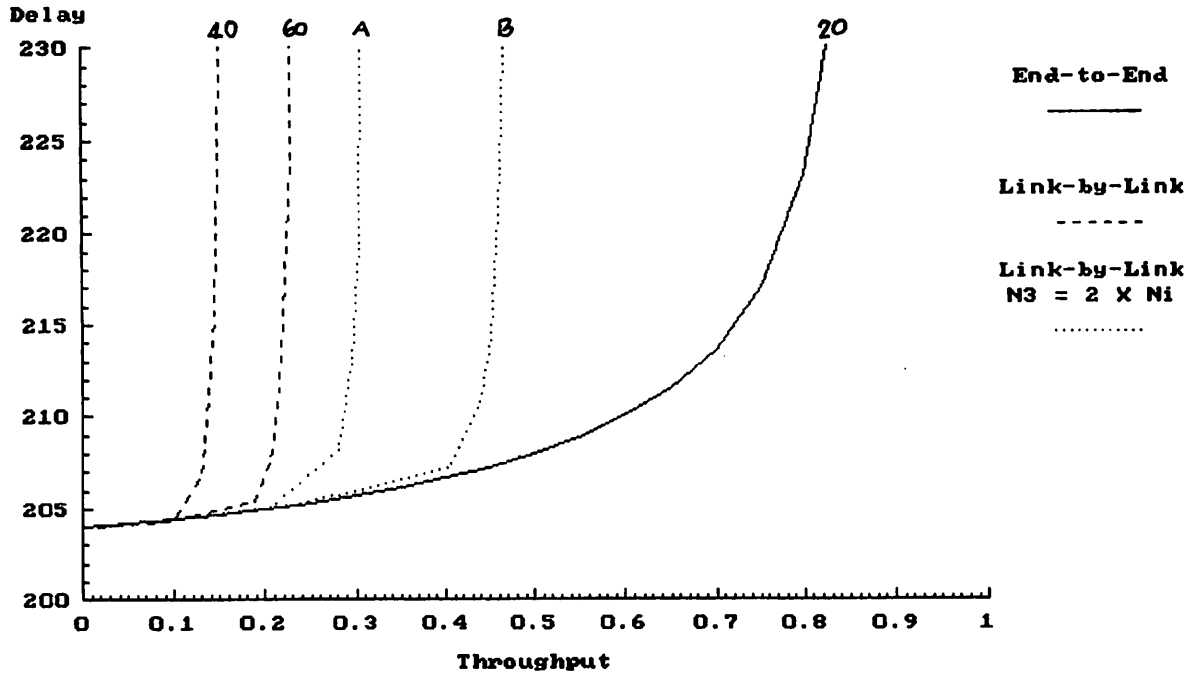


Figure 10: Link-by-link versus end-to-end, with one longer link

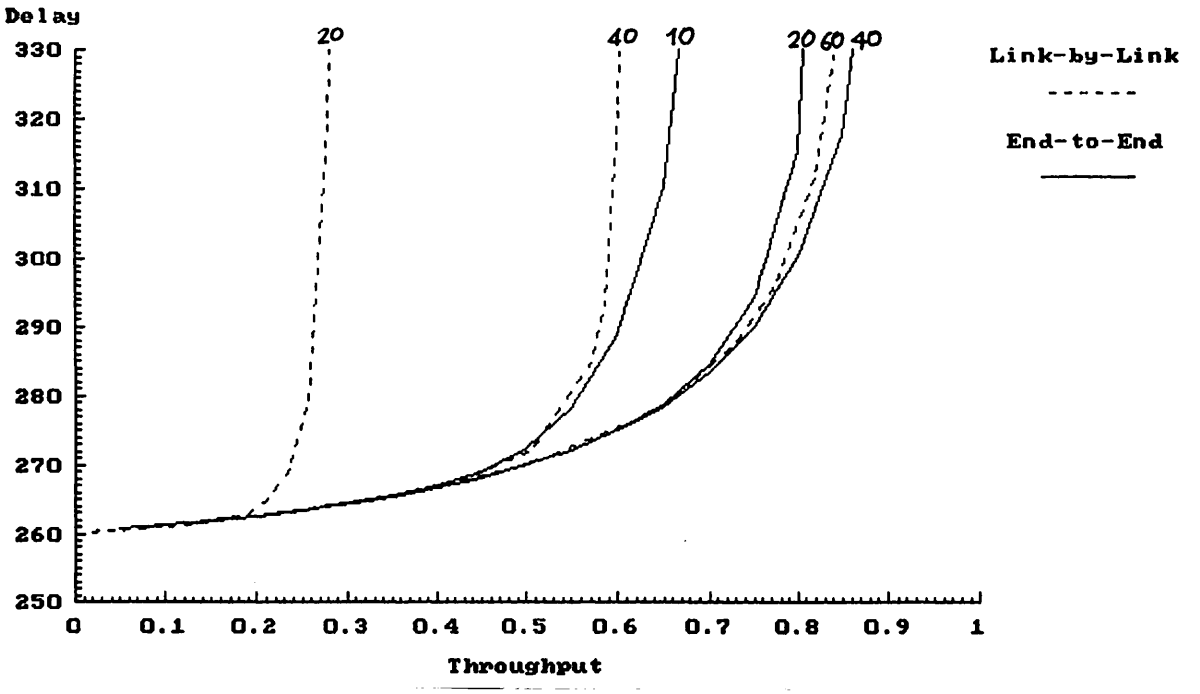


Figure 11: Link-by-link versus end-to-end, M= 10 hops

disadvantage that it recovers from an error or buffer overflow more inefficiently than link-by-link. Specifically, if either of these events occur on a link far from the source, then end-to-end "wastes" the successful transmissions over all earlier links. The problem with link-by-link is that it requires buffer storage for each packet, until it is acknowledged. Figure 11 indicates that the relative difference in the performance of the two protocols decreases as the number of hops increases. We conjecture that, as the number of hops increases even more, the performance of link-by-link will approach and surpass that of end-to-end. This is because the inefficiency in the method by which the end-to-end scheme recovers from errors or buffer overflows worsens to the point that the maximum sustainable throughput approaches zero. On the other hand, the constraints imposed on the sustainable throughput by finite buffer limitations is not sufficient to reduce the throughput to zero even as the number of hops approaches infinity. Although we have not attempted to determine the number of links at which this crossover occurs, we are confident that it occurs for large enough values of M so as to not be of practical interest.

Finally, we note that constant length packets are often assumed in high speed switch architectures [5,15,17], although this is not always the case [4]. In order to examine the effect of constant message sizes, we again simulated a single VC, but in this case we explicitly introduced interfering traffic (at a Poison rate of $3\lambda_{VC}$) at each node along the VC, as shown in Figure 1a; this was done to avoid the artificial "pipelining" of constant length messages as they passed along a VC. Although the interfering traffic did *not* share the same buffer pools, it did utilize the outgoing link. Figure 12 shows that in both the link-by-link and end-to-end schemes, the average delay in the case of constant length messages is uniformly lower than with exponentially distributed message

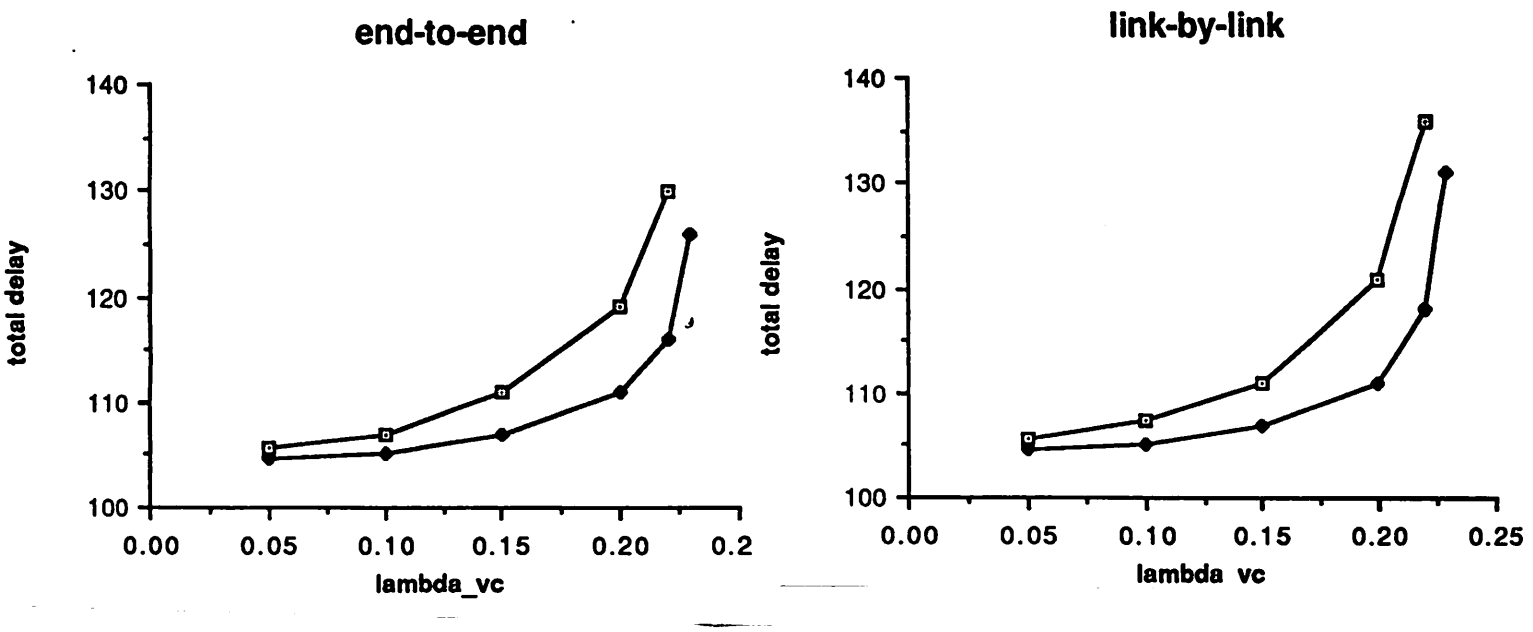


Figure 12: Exponential versus constant message lengths

lengths. Note however, that the *relative* differences in performance are almost identical in both cases. Thus, a central thesis of this paper remains unchanged - end-to-end error protocols still provide equivalent or superior performance over the range of parameters of practical interest.

6. Error Control with Shared VC Buffer Pools

In previous sections we assumed that each VC was allocated a separate pool of K buffers at each node; the effect of other VC's routed over the same outgoing link was modeled by reducing the link service capacity seen by the VC under study. An alternate buffering approach is for all VC's routed over the same outgoing link at a node to share a single common buffer pool. In this section, we examine this latter approach and its impact on the performance of link-by-link and end-to-end error control schemes. Since the models developed in this section are similar to our previous separate-buffer-pool models and since the overall conclusions are identical (end-to-end is again shown to be superior to link-by-link), our discussion of the models will be somewhat abbreviated.

The modeling challenge in studying buffer sharing by multiple VC's is to avoid the complexity of explicitly modeling (in their entirety) each of the VC's sharing a buffer pool at each of the nodes along a given VC. Fortunately, in the case of homogeneous networks (of arbitrary topology) considered below in which all nodes are statistically identical (i.e., have the same traffic characteristics, buffering capacities, and service rates), this complexity can be avoided. In this case, we can study the behavior of interfering VC's along a *single* outgoing link and then extrapolate to obtain the performance of an entire VC.

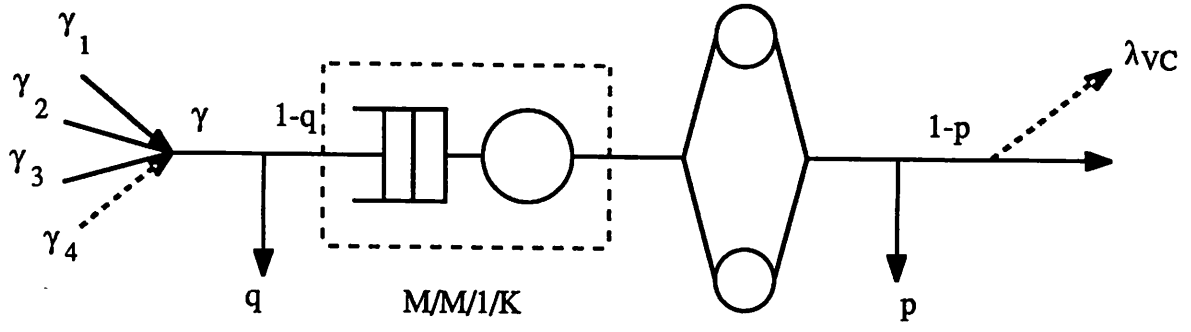


Figure 13: Link-level model for end-to-end control with shared buffer pools

We again consider a network in which each VC is M hops long, μ_0 is the service rate of the link, $1/\mu_{prop}$ is the propagation delay along a link, and K buffers are available (on a shared basis) for all VC's routed along an outgoing link. For simplicity, we assume that each node initiates and terminates a single VC, although this restriction can easily be relaxed. p is again the probability that a message is corrupted during transmission and q is the probability that a message finds all buffers occupied at a node; note that q is the same for all nodes, again due to the homogeneity condition.

An important network parameter once again is the *offered traffic*, γ, at each node. (We use the notation γ, instead of λ to differentiate the shared-buffer models from the separate-buffer models). Here γ represents the rate at which messages arrive for transmission at a node and, as shown in figure 13 and 14 includes both first time message transmissions as well as retransmissions. Note that as a result of the homogeneity condition, γ is now the same for *every* link in the network. In the following, we will find it useful to decompose γ into $\gamma_1, \ldots, \gamma_M$, where γ_i indicates traffic for which the link under study is the ith hop on its VC.

6.1 End-To-End Error Control: the Shared Buffer Approach

Our model of a single link with end-to-end error control is shown in figure 13 (for the case $M = 4$). Note that it is similar to figure 2, except that the offered traffic now contains traffic from multiple VC's. In figure 13, λ_{VC} again represents the rate at which traffic initially enters a VC and thus also the rate at which successfully received traffic exits the network at a destination node. From flow conservation and the homogeneity condition, we have:

$$\begin{aligned} \lambda_{VC} &= \gamma_M(1-f) \\ \gamma_i &= \gamma_{i-1}(1-f) \qquad i = 2, \ldots, M \end{aligned}$$

where $f = p+q-pq$ is the probability that a message is either blocked at, or incorrectly transmitted by a node. Note that f is the same for all nodes, again due to the homogeneity condition. Using the fact that $\gamma = \sum_{i=1}^{M} \gamma_i$, we have after some simple rearrangement:

$$\lambda_{VC} = \gamma \frac{f(1-f)^M}{1-(1-f)^M} \tag{15}$$

Defining $\rho = \gamma/\mu_0$, we again have for the M/M/1/K queue [1]:

$$q = \frac{(1-\gamma)\gamma^K}{1-\gamma^{K+1}} \tag{16}$$

$$E[W_*] = \frac{1}{\gamma(1-q)} \left(\frac{\rho}{1-\rho} - \frac{(K+1)\rho^{K+1}}{1-\rho^{K+1}} \right) + 1/\mu_{prop} \tag{17}$$

where $E[W_*]$ is the average delay (queueing, transmission and propagation) a message experiences passing through a link. For a given offered traffic, we can now solve equation 16 for q and then calculate a corresponding value of λ_{VC} using equation 15.

In order to incorporate the single link model into a VC-level model of end-to-end error control, we again follow the approach of section 3.3, except that p_{fail} must now be computed by

$$p_{fail} = q + (1-q) f \sum_{i=0}^{M-1} (1-f)^i, \tag{18}$$

since a message entering the network can be blocked (and thus require later retransmission) if all the buffers at the first node are full. Using this value of p_{fail}, we can compute the total delay of a message between its initial arrival to the network and its eventual successful reception at the destination station under end-to-end error control as:

$$E[W_{ee}] = E[N_t] T_{ee} + \sum_{k=1}^{M} E[W_*] \tag{19}$$

where $E[N_t]$ is the expected number of times a message is retransmitted (computed using equations 18 and 5) and T_{ee} is the end-to-end timeout interval.

6.2 Link-by-Link Error Control: the Shared Buffer Approach

The lower-level model of link-by-link error control with shared buffer pools is identical to the lower-level model developed in section 4.2, except that the offered message traffic (first time arrivals

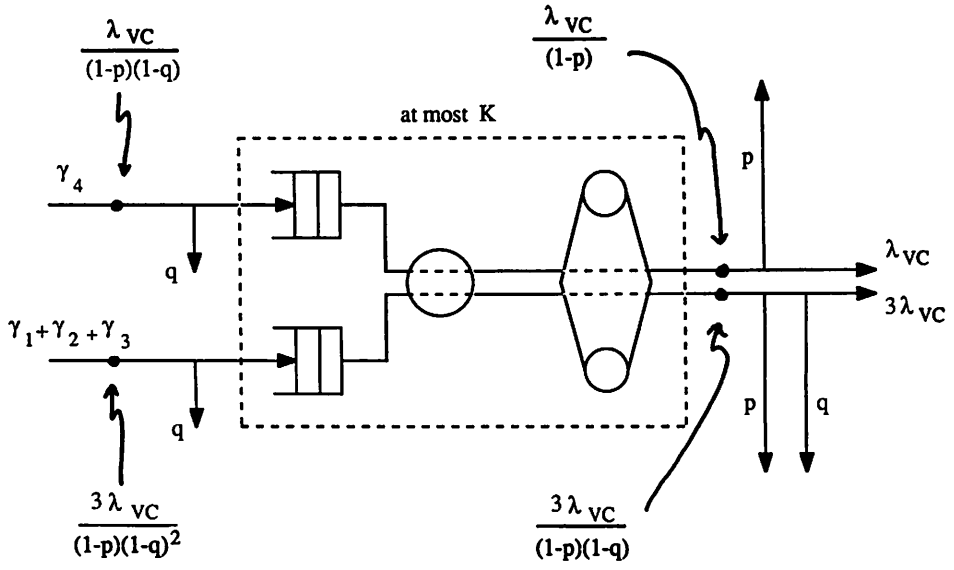


Figure 14: Link-level model for link-by-link control with shared buffer pools

plus retransmissions) is now given by $\gamma = \sum_{i=1}^{M} \gamma_i$ rather than by $\lambda_i(1-p)$ (see figure 5(a) and equation 7). The probability that a message is blocked at a node can thus again be obtained via equation 9.

Equation 3 contains two unknowns, the arrival rate γ and the blocking probability q. We may once again invoke flow conservation arguments to obtain a second equation relating these two unknown quantities. As shown in figure 14 (for the case $M = 4$), in order for the throughput of successfully transmitted and buffered (at the next node) messages through a link to be $M\lambda_{VC}$, the input rate of messages (first time arrivals plus retransmissions) must be:

$$\gamma = \sum_{i=1}^{M} \gamma_i = \frac{\lambda_{VC}}{(1-p)(1-q)} + \frac{(M-1)\lambda_{VC}}{(1-p)(1-q)^2} \tag{20}$$

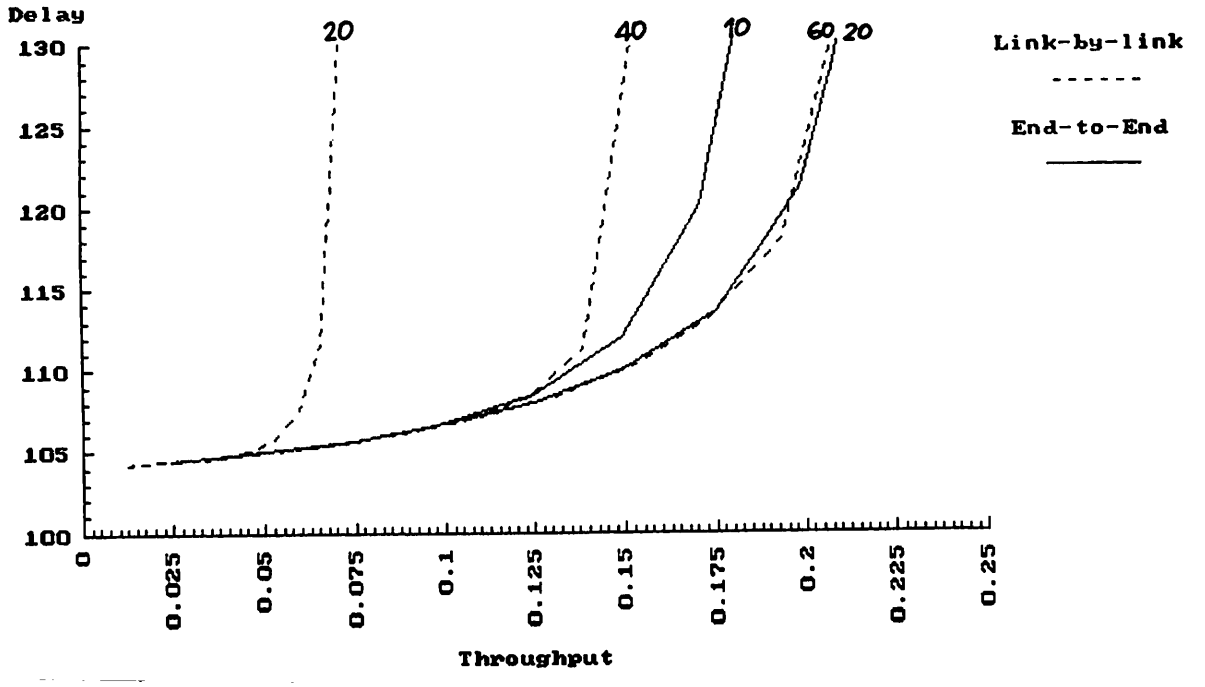
Using this value of γ as the offered message traffic, equations 3 and 20 can be numerically solved for q and thus $E[L_*]$ (equation 10), where $E[L_*]$ is the average number of messages actively queued for transmission at a node. Using Little's Law we have:

$$E[W_*] = \frac{E[L]}{\frac{\lambda_{VC}}{1-p} + \frac{(M-1)\lambda_{VC}}{(1-p)(1-q)}} \tag{21}$$

Finally, the average end-to-end delay under a link-by-link error control scheme in a homogeneous network can now be obtained. The probability that a message must be retransmitted k times on the ith hop of its VC before being successfully received at the next station is given by:

$$P[N_i^i = k] = f_i^k(1 - f_i)$$

where again $f_i = q + p - pq$ for $i = 1, \ldots, M-1$. Also, $f_M = p$ since an error-free message can not blocked at its destination node and $f_0 = q$ since a message can be blocked at the first node on its



VC. Following the same arguments used in section 4.3, the average time between the initial arrival of a message to the VC and its successful reception at the destination station is given by:

$$E[W_{ll}] = E[N_t^0]T_{ll} + \sum_{i=1}^{M} \left(E[N_t^i] \left(E[W_*] + T_{ll} \right) + E[W_*] + 1/\mu_{prop} \right) \tag{22}$$

where T_{ll} is the link-level timeout interval.

6.3 Comparison of Link-by-Link versus End-to-End with Shared Buffers

Figure 15 shows the time delay versus throughput performance comparisons of the two error control schemes using the same network parameters as the baseline case of figure 7. The results indicate that in the case of shared buffers, an end-to-end again provides a performance level equivalent or superior to the link-by-link scheme. We also note that additional studies (not reported here) have shown that in the case of increased error probabilities, decreased/increased propagation delays, and increased VC hop lengths, the shared-buffer models also behave similarly to the separate-buffer models.

7. Conclusion

In this paper, we have examined the performance of a link-by-link versus end-to-end approach for handling the loss and/or corruption of messages as they are transmitted between two end users in a high-speed network. Both analytic and simulation performance models were developed for comparing the performance of these two schemes. Our study has shown that for the range of

network parameters of practical interest, an end-to-end approach towards error control provides equivalent or better performance while at the same time requiring fewer network resources (e.g., buffers, computation time) than the link-by-link approach. This was shown to be the case even when the (higher) node processing demands of the link-by-link approach were ignored and an overly pessimistic analytic model was used for the end-to-end approach.

Research in the area of protocols for high speed networks has just begun. Our present work has shown that traditional wisdom, at least in the case of error control protocols, is no longer valid in a high speed environment. We believe that the same may be true for other high-level protocols such as routing, flow control, and congestion control in a HSN. The exploration of these issues is thus a promising and challenging area for future research.

REFERENCES

[1] A. Allen, *Probability, Statistics and Queueing Theory,*, Academic Press, 1978.

[2] D. Carlson, "Bit-Oriented Data Link Control Procedures", *IEEE Transactions on Communications,* Vol. COM-28, No. 4, (April 1980), pp. 455-467.

[3] K.M. Chandy, J. Howard, D. Towsley, "Product Form and Local Balance in Queueing Networks," *JACM*, Vol. 24, No. 2, (April 1977), pp. 250-262.

[4] K.Y. Eng, M. Hluchyj, Y. Yeh, "A Knockout Switch for Variable-length Messages", *Proc. IEEE ICC 1987,* (Seattle, Wa.), pp. 22.6.1-22.6.5.

[5] A. Huang and S. Knauer, "Starlite: A Wideband Digital Switch", *Proc. IEEE GLOBCOM 1984,*, Atlanta, Ga., pp. 121-125.

[6] Special Issue of Fiber Optic Systems for Terrestrial Applications, *IEEE Journal on Selected Areas in Communications,* Vol. SAC-4, No. 9 (Dec. 1986).

[7] M. Irland and G. Pujolle, "Comparison of Two Packet Retransmission Techniques", *IEEE Transactions on Information Theory,* Vol. IT-26, No. 1, pp. 92-97.

[8] H. Kobayashi, *Modeling and Analysis*, Addison Wesley, Reading, MA (1978).

[9] D. Kuhl, "Error Recovery Protocols: Link-by-Link versus Edge-to-Edge", *Proc. IEEE Infocom 1988,* pp. 319-324.

[10] S. Lam, "Store and Forward Buffer Requirements in a Packet Switching Network", *IEEE Transactions on Communications,* Vol.COM-24, No. 4 (April 1976), pp. 394-403.

[11] M. Pennotti and M. Schwartz, "Congestion Control in Store and Forward Tandem Links", *IEEE Trans. Communications,* Vol. COM-23, No. 12 (Dec., 1975), pp. 1434-1443.

[12] C. Samelson and W. Bulgren, "A Note on Product Form Solution for Queueing Networks with Poisson Arrivals and General Service Time Distributions with Finite Means," *JACM*, Vol. 29, No. 3 (July 1982), pp. 830-840.

[13] P. Schweitzer and S. Lam, "Buffer Overflow in a Store and Forward Network Node", *IBM Journal of Research and Development,* Nov. 1976, pp. 542-550.

[14] M. Schwartz, "Performance Analysis of the SNA Virtual Route Pacing Control", *IEEE Transactions on Communications,* Vol. COM-30, No. 1 (Jan. 1982), pp. 172-184.

[15] J. Turner, "Design of a Broadcast Packet Network", *Proc. IEEE INFOCOM 1986,*, Miami, Fla., pp. 667-675.

[16] J. Turner, "Design of an Integrated Services Packet Network", *IEEE Journal on Selected Areas in Communications,* pp. 1373-1379.

[17] Y. Yeh, M. Hluchyj, A. Acampora, "The Knockout Switch: A Simple Modular Architecture for High Performance Packet Switching", *Proc. International Switching Symposium,* (Phoenix, Az.).