A Simulation Model for Network Routing

Udo W. Pooch, Charles Neblock, and Rahul Chattergy

Texas A&M University, Hardin Simmons University, and University of Hawaii, respectively.

Abstract

The simulation program described in this paper was devised as a vehicle for the study of communication network routing procedures. It was designed to model the behavior of a wide range of network topologies and routing disciplines. An event driven simulation approach was chosen to minimize program development time and complexity.

GASP-IV was selected as the simulation language. The determining factor in choosing the language was the clean interface it presents to FORTRAN. The use of GASP-IV permits utilizing the full power of FORTRAN in describing complex routing algorithms while simoultaneously relieving the programmer of the responsibility for such essential housekeeping functions as enqueuing and dequeuing messages, file manipulation and event sequencing.

As an application of this simulator, a loop-free distributed adaptive routing algorithm is proposed and analyzed. Simulation results are presented and the effect of the algorithm on overall network performance is examined. In addition to a loop-free property, the algorithm is shown to provide a partial solution to the single path routing problem. Furthermore, a modification to the algorithm is given which precludes message loss, and is shown to be superior in performance to current update algorithms (PUA).

INTRODUCTION

Computer-communications networks(1,9) are accepted as applying to the following types of systems:

1. A set of computers providing services to an end user or another computer, known as hosts.
2. A collection of hardware and software components, referred to collectively as the communications subnet, responsible for the transfer of information between hosts. Each subnet switchpoint is termed a node.

Switching modes characterize networks even further according to being circuit switched, message switched, or packet switched.

Three basic functional forms of computer networks are known to exist(9).

1. Random access networks (RAN) support communication between end users and host computers, either in the form of terminal access (dialog) or remote batch. Such networks are common and their properties are well understood. Current systems of this type include TYMNET, CYBERNET, and INFONET (5).
2. Value added networks (VAN) are networks in which computers or other facilities have been interfaced to existing carriers to increase total system value to the end user. A store-and-forward system interconnected by commercial circuit-switched links is a typical example.
3. Mission-oriented networks (MON) are value added networks in which the system components are controlled by a single administrative organization. This arrangement permits some degree of global optimization of resources, and thus achieving a higher degree of network efficiency than the VAN. Perhaps the most ambitious network of this type to date is the World-Wide Military Command and Control System (WWMCS).

NETWORK DESIGN ISSUES

Many interrelated issues are of importance in the design of computer-communications networks. The design problem is frequently viewed as consisting of five major issues (4): Topology, Flow Control, Security, Protocol, and Routing.

Topology is concerned with the placement and interconnection of network nodes. The selection of a particular topology is heavily influenced by available line capacities, link and node reliabilities, and the desired degree of control distribu-

CH1437-3/79/0435-0445$00.75 © 1979 IEEE

1979 Winter Simulation Conference

tion.

Flow control is the mechanism used to prevent network saturation. Two methods are generally employed, "controlled permission to send" and "discard unwanted data". Corollary issues are those dealing with buffer size, allocation, error detection, and correction mechanisms.

The security issue, unlike the other design issues, does not have a direct implication for the communication subnet. The problem primarily involves the design of host computer mechanisms which limit access to processes and files. A secondary problem is to insure the integrity of transmitted information and the prevention of unauthorized interception and use. Transmission security is modt often implemented by any one of a number of well-known link encryption techniques.

Protocols consist of a set of agreements concerning the format of messages exchanged through the network. Network protocols reflect the layered structure of the computer network because they contain control information at an inter-process level, a host to host level, a node to node level, and finally at a hardware level. Layered protocols are designed to make each successive higher protocol invisible to the lower level immediately preceeding it. All system control information is provided by the protocol, and consequently careful protocol design is essential to the effectiveness of the flow control and routing systems.

Routing is concerned with the selection of the transmission path for the message. Routing mechanisms can be grouped into two categories: static or fixed path routing, and dynamic or best estimated path routing. Static routing procedures have been extensively investigated and are reasonably well understood (2 , 10 , 11). Dynamic routing has desirable adaptability characteristics, and theoretically is capable of providing lower message delay than static procedures. Two major problems remain unresolved in dynamic routing schemes. The first is the limitation of peak bandwidth to the speed of a single line, since all data to a given destination is routed via a single neighboring node. The second problem is that of looping, in which a message repeatedly traverse the same set of nodes as a result of subtle deficiencies in control timing.

A NETWORK MODEL BASED ON QUEUING THEORY

INTRODUCTION

Analytic methods have been largely confined to queue theoretic models, with computer-communication systems characterized as networks of interconnected queues (13). Classic queueing theory, however, has been largely confined to the study of single server queues (12). More complex structures are often not amenable to direct analytic solution. Hence, the basic approach to analysis involving queueing systems has been the so-called decomposition technique. In this method, complex nets are reduced to a set of single server problems, each of which has a well-defined solution. After single server analysis has been completed the network must be synthesized by collecting the individual solutions, and reuniting them to form a composite solution which reflects the properties of the original network. Justification for the decomposition-synthesis process has been provided by Jackson (8) and by Burke (3).

The results of Jackson and Burke are not sufficient for the analysis of communication networks. Since messages maintain their length throughout the system, dependencies arise between interarrival and service times, thus violating the constraints of the queueing model. Kleinrock (10), however, proposed and demonstrated the "independence assumption" by noting that for networks having sufficient connectivity the interactions between nodes causes the service and the interarrival time correlation to disappear.

THE ABSTRACT NETWORK MODEL

The abstract computer-communications network consists of a set of data sources and sinks, termed "hosts", a set of message processors called nodes, and a set of communication channels. Each host is connected by a bi-directional link to a network node. The hosts are external to the communication subnetwork and their only interface to the network is via the associated node. The node performs all communication functions including error detection, routing, message buffering, and network protocol handling.

Node pairs are interconnected by, at most, one full-duplex channel having a fixed, finite capacity measured in bits per second. Not all channels in a network need have the same capacity. For analysis purposes full-duplex lines are considered as a pair of distinct simplex lines of opposite sense.

Traffic densities in the network are described by a "traffic matrix," T, whose entries $\gamma_{ij} \in T$ specify the number of messages per second entering the network at node i and departing the network at node j. Alternatively r_{ij}, $r \in \hat{T}$, may be used to specify traffic in bits per second from source i to destination j. The traffic matrix is assumed to describe traffic at the network design point and corresponds to a one hundred percent rated load for the specified throughput and delay. A common performance metric for a network is the effective data rate, R_E , and is used as a linear scaling factor for the traffic matrix entries. Thus, a network is considered to be twenty percent loaded when $R_E = .2$ and the current traffic matrix, T', entries are $R_E * T$.

For analytic purposes, fixed routing policy is assumed in which a single unique path is defined for each message departing source i and destined for sink j. Such a message encounters a series of delays along the path. These delays consist of:

1. Encoding time from host i to node i;
2. A constant nodal processing delay of 10^{-3} seconds at each node traversed.
3. A queuing delay associated with each channel
4. An encoding delay (channel service time) for each channel.

5. A channel propagation delay of eight microseconds delay per mile of line.
6. A fixed modem delay of 70 μsec per channel.
7. An encoding delay from node j to host j.

DELAY ANALYSIS FOR A SINGLE CLASS OF CUSTOMER

A computer-communication network may be represented by a model of a network of queues. Define T_i as the message arrival rate at node i, and $P\{j \mid i\}$ as the probability that a message departing facility i will next join queue j. Assume that arrivals are Poisson and that message lengths are exponentially distributed. With these assumptions, Jackson's result (8) applies and the steady stae behavior of the system is equivalent to that of the set of independent single server queues.

Consider a service time distribution $A(\tau)$, an interarrival time distribution B(t) with an average message arrival rate of λ, and a single server. If $A(\tau)$ is exponentially distributed, the waiting time in the system is:

$$W(t) = \frac{\lambda \cdot \bar{t}^2}{2(1-\rho)} \qquad (2.1)$$

and the average time in the system is:

$$T = W + \bar{t} = \frac{\lambda \bar{t}^2}{2(1-\rho)} + \bar{t} \qquad (2.2)$$

where $\bar{t}$ and $\bar{t}^2$ are the first and second moments of B(t), and ρ is the utilization factor: $\rho = \lambda \bar{t}$. When B(t) is also an exponential distribution the equation reduces to

$$T = \bar{t} \,/\, (1-\rho) \qquad (2.3)$$

Thus, T represents the total time in a single server system, (M/M/1), averaged over all system entities.

The network must next be decomposed into an equivalent set of independent single server queues.

Let Z_{ij} be the average delay for messages having origin i and destination j. Message arrival is a Poisson process with an average arrival time of γ_{ij} messages per second. Message lengths are assumed to be exponentially distributed with an average message length of $1/\mu$ bits per message. With the total traffic entering the network being:

$$\gamma = \sum_{i,j} \gamma_{ij} \qquad (2.4)$$

the average delay is

$$T = \sum_{i,j} \frac{\gamma_{i,j}}{\gamma} * Z_{ij} \qquad (2.5)$$

However, the path followed by the message having delay Z_{ij} may consist of several channels, and the system must yet be reduced to a single channel level. Noting that routing is fixed for a given source - destination pair, the message is constrained to follow a fixed path $\pi(i,j)$ composed of individual channels. The average message rate, λ_k, for the k-th channel becomes:

$$\lambda_k = \sum_{i,j} \gamma_{ij} \qquad \forall_{ij} \ni \lambda_i \in \pi(i,j)$$

Therefore,

$$T = \sum_k \frac{\lambda_k}{\gamma} \cdot T_k \qquad (2.6)$$

Furthermore, service time on the k-th channel is message length divided by channel capacity. Letting C_k denote the capacity of the k-th channel,

$$\rho = \lambda_k / \mu_k \cdot C_k \qquad (2.7)$$

is substituted yielding

$$T = \sum_k \frac{\lambda_k}{\gamma} \left(\frac{1}{\mu_k C_k - \lambda_k} \right) \qquad (2.8)$$

T provides an estimate of the average of the average network delay due to message queuing and encoding on the k-th channel. However, delay in the abstract model was defined to include nodal processing time, propagation time, and modem delay. Propagation time was found to be the product of line length (in miles) and channel delay (per mile), plus a (70 μsec) modem delay. If P_k is propogation time and L_k is the length of line k, then:

$$P_k = L_k \ (8 * 10^{-6}) + 7 * 10^{-5} \qquad (2.9)$$

Nodal processing delay was assumed to be a constant 10^{-3} seconds per message. Adding these two terms, T becomes

$$T = \sum_k \frac{\lambda_k}{\gamma} \left(\frac{1}{\mu_k C_k - \lambda_k} + P_k + 10^{-3} \right) \qquad (2.10)$$

All enroute delays are now accounted for. However, the message passes through one more node than the number of channels. Furthermore, encoding time on the host to node and node to host links must be included in total delay. Assuming that all node to host links have the same capacity, ℓ bits per sec., the total delay can be expressed as:

$$T = \sum_k \frac{\lambda_k}{\gamma} \left(\frac{1}{\mu_k C_k - \lambda_k} \right) + P_k + 10^{-3} + 10^{-3} + 2/\mu\ell \qquad (2.11)$$

DELAY ANALYSIS FOR A PRIORITY SYSTEM HAVING TWO CLASSES OF CUSTOMERS

In a realistic network analysis it is necessary to also model the effect of acknowledgement traffic on network delay. Such an analysis is complicated by two factors. First, acknowledgements are fixed length, thus violating the exponential length distribution assumption. Second, acknowledgement traffic must take priority over message traffic in order to clear nodal buffers as soon as poosible.

The message delay analysis is based on Hawkes' work (7) as modified by Fultz (6). Consider,

1. Acknowledgement length = $1/\mu_a$
2. Message length = $1/\mu$ with coefficients of variance C_v .

The average delay on channel k is

$$T_k = \mu/C_k + W_k + P_k \qquad (2.12)$$

with average channel utilization as

$$P_k = \lambda_k/\mu C_k \quad , \qquad (2.13)$$

where λ_k/μ is the data rate on line k without acknowledgement traffic. Note that on acknowledgement is transmitted on line k for each message received on the reverse line $\bar{k}$, weighted by the ratio between acknowledgement length and average length. Hence

$$pa_k = \lambda_k/\mu C_k \left(\frac{\mu}{\mu_a}\right) \qquad (2.14)$$

Applying Hawkes' results, the appropriate equation for message waiting time is

$$W_k = \frac{pa_k\left(\frac{1}{\mu_a C_k}\right) + p_k \frac{1}{\mu C_k}\left(1+C_v^2\right)}{2\left(1-pa_k\right)\left(1-pa_k-p_k\right)} \qquad (2.15)$$

Concluding, the average message delay with acknowledgement traffic is obtained to be

$$T = \sum_k \frac{\lambda_k}{\gamma}\left(T_k + 10^{-3}\right) + 10^{-3} + 2/\mu l \qquad (2.16)$$

INTRODUCTION

Simulation provides an experimental tool for systems for which real world experiments are not feasible, or for which mathematical analysis is not adequate. In the case of this research a real-world system is not available, and current mathematical analysis techniques are not sufficiently powerful to model the behavior of systems utilizing other than fixed routing schemes.

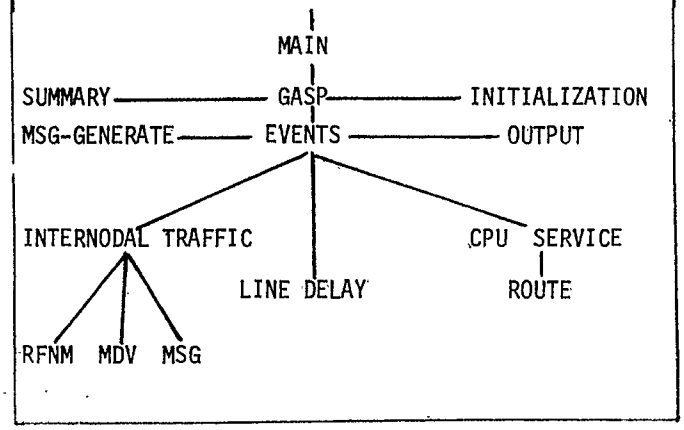
This simulation program was designed to model a wide range of topologies, protocols and routing procedures. The simulation is based on the abstract computer network model introduced previously and embodies the assumptions made in its specification. The analytical expressions for message delay will be used as the primary metric for validating the simulation model. An event driven approach was chosen to minimize development time and programing complexity.

GENERAL PROGRAM FLOW

The general system flow is shown in Figure 3-1. The entities in the simulation are messages. Four distinct message classes are recognized: acknowledgement, request-for-next-message (RFNM), update vectors for adaptive routing algorithms, and regular messages. Each entity consists of full 25 full word attributes and two full word pointers. Of the 25 attributes used only six are required to describe the message. The remaining attributes provide a trace of nodal transits, and are used to detect and record message looping.

The main program reads the system specification. Line numbers are then assigned to each link a_{ij} . An edge-vertex incidence matrix is built and assigned line numbers are mapped to logical port numbers at the node. The user provided traffic matrx, [T], is examined to determine average message interarrival time at each node, and [T] is then transformed to a cumulative probability matrix for use in determining message destination by a Monte Carlo technique.

Figure 3-1
GENERAL SYSTEM FLOW



Event routines are provided for message generation, nodal processing, routing, and line transmission. At the conclusion of the run a number of statistics are provided:

1. Rated system load.
2. Observed system load.
3. Last message generated.
4. Current simulation time.
5. Number of buffers per node.
6. Routing discipline.
7. Delay vector update interval.
8. Delay vector bias.

In addition, the mean, standard deviation, standard deviation of the mean, coefficient of variance, minimum and maximum values and number of observations are provided for each of the following measures:

1. Link busy time.
2. CPU busy time.
3. Line busy time.
4. Message delay.
5. Round-trip time.
6. Message rejected for lack of buffer space.
7. Transmission errors.
8. Number of hops.
9. Message delay for messages not looping.
10. Retransmission required by lack of available path.
11. Queue lengths for four selected nodes.
12. Number of retransmission for all causes.
13. Average system queues.

SYSTEM ENVIRONMENT

As in the abstract model, message arrivals are assumed to be Poisson distributed. Nodal processing delays are fixed at 10**(-3) seconds per message and line propagation time is constant at 8 microseconds per mile, with a 70 microsecond modem delay added. All host to node links have a fixed capacity of 800 Kilobits per second. Nodes are regarded as being 100 percent reliable, and all lines have fixed bit error rate of 10 **(-5).

The host computers form the data sources and sinks, and are regarded as external to the network. Message queues at the host are regarded as FIFO and essentially infinite. Nodal queues are of finite length, determined by the amount of available nodal storage. All node queues are ordered FIFO by priority class. The priority structure in their priority order consists of (i) acknowledgements, (ii) delay table update vector, (iii) request for next message (RFNM), and (iv) regular message. The simulation is structured to permit interconnecting lines to run free at their maximum data rate. That is, no CPU intervention is required once a message is placed in a nodal output buffer.

The network nodes are designed to allocate and deallocate buffer storage, provide acknowledgement for incoming packets, check for transmission errors, and perform message routing. Unlike the channels, host-to-node links are not free-running. The host is permitted to place a message on the node input queue only if adequate free node storage is available to insure the integrity of node to node traffic. If a storage of buffers is noted, the host-to-node link is blocked until time as buffers are available. At this time the CPU unblocks the link and polls the host for any outstanding messages.

The network environment based upon user supplied specifications consists of:

1. The number of nodes in the network.
2. The average message length.
3. The routing algorithm to be used.
4. A stopping parameter, either in messages to be processed or simulated in time. If the run is to be stopped on a time parameter the time must be supplied in the GASP control cards. The maximum number of messages supported is 2000.
5. The network loading factor.
6. The update interval in seconds if the selected routing is adaptive.
7. A number of switch settings to prescribe the portion of the system environment to be read, read and printed, or calculated and printed.
8. The increment by which the loading factor is to be increased for successive runs if multiple runs are desired.
9. The upper limit on network load.
10. The bias term to be applied to any delay in use.
11. The number of buffers per node.
12. The protocol to be used in the simulation.

In addition, the network topology must be specified by:

1. An adjacency matrix $[A]$, whose element $a_{ij} \in \{0,1\}$, a 1 in position a_{ij} indicates that a directed edge exists between source vertex i and destination vertex j. By default if $a_{ij} = a_{ji} = 1$ a duplex line exists between nodes i and j, if $a_{ij} \neq a_{ji}$ the line is simplex.
2. A distance matrx $[D]$, whose elements d_{ij} represent the line distance between nodes i and j, in miles.
3. A capacity matrix $[C]$ whose elements c_{ij} are the line capacity of the directed edge from node i to node j in kilobits per sec.

If no capacity matrix is given, all line capacities default to 50 Kilobits per second. No default exist for adjacency or distance matrices and failure to provide them constitutes a fatal system error. Matrices $[A]$, $[D]$, and $[C]$ are checked for consistency.

ROUTING ALGORITHMS

Four routing algorithms are intrinsic to the program and a stub is provided for an arbitrary user supplied algorithm. The intrinsic algorithms are random routing, fixed routing, a "shortest-queue with bias" periodic update algorithm, and the regional node mapping algorithm. In the random algorithm the next node to be traversed is selected with equal probability from all adjacent nodes except the node last traversed by the message.

A NETWORK SIMULATION MODEL ..... (continued)

In the fixed routing algorithm the next node is picked deterministically from a user supplied routing table.

The shortest-queue with bias algorithm has been extensively investigated and documented by Fultz (6). In essence the node maintains a table of estimated delays to every other node along each of its output lines. The immediate neighbor on each output line periodically supplies its own estimate of delays to every other node, called a minimum delay vector. The receiving node adds its estimated delay for the line on which it received the vector. The minimum delay vector plus internal estimates then replace the existing delay table entry for the line. Periodically the node searches the delay table for the line having the shortest delay to every node in the table. These new shortest delays are then posted to the nodes minimum delay vector. The number of the line having the shortest delay is posted in the routing vector, and the new delay estimates are transmitted to each neighboring node. Routing policy then consists of looking up the line in routing vector corresponding to the minimum delay route to the message destination.

The basic flow control mechanism is a buffer-limiting scheme. The number of buffers per node is selected by the user. The simulator blocks incoming host messages when 50 percent of the buffer pool is filled.

Protocols available include positive acknowledgement, negative acknowledgement, and end-to-end acknowledgement by use of a RFNM. It should be noted that the model uses the RFNM only to determine round-trip message time, and it is in no way associated with input quenching. RFNM should be used sparingly since its presence doubles and the effective data load in the network and can, although such an effect has never been observed, possibly lead to exceeding the maximum number of queue entries in a heavily loaded network.

THE EXPERIMENTAL MODEL

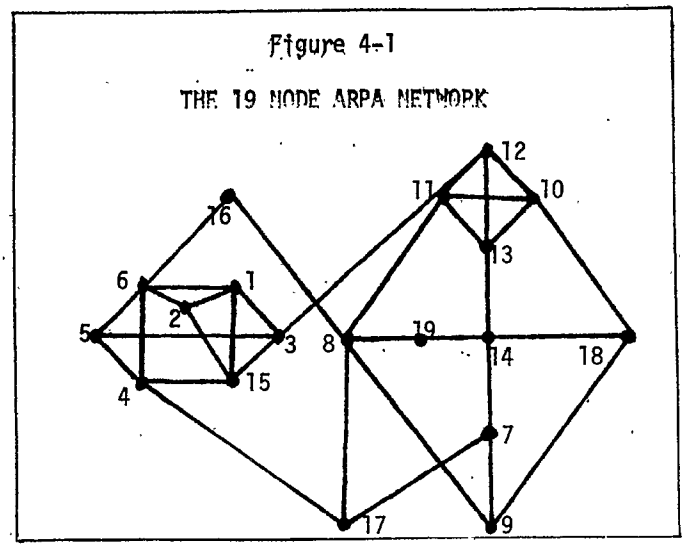
INTRODUCTION

The experimental model chosen for the simulation is the 19-node ARPA network (Figure 4-1). The network was selected both because it is representative of a large class of distributed networks and because of the availability of analytic, simulation and observed data for the network. Since the performance of the network is well documented, validation of the model is unusually easy.

Several peculiarities exist in the model and are worth mentioning. First, RFNM's are considered to be elements of host-to-host prorocol, and are therefore eliminated from consideration in the model. Second, the distribution of message lengths is taken from Fultz (6) and reflects several pertinent features of the ARPA network:

Figure 4-1

THE 19 NODE ARPA NETWORK



1. Some minimum length is required to account for message headers and framing characters.
2. Short messages are preponderant over long messages as a result of the large number of interactive users in the system.
3. The maximum message length is set at 1024 bits. The spike occurring at this point is due to some users attempting to maximize their data transfer rate.

This distribution has been used throughout the simulation runs in order to allow direct comparison between the results obtained and those reported in the literature. The simulator has shown remarkably good agreement with analytic results.

PERFORMANCE CHARACTERISTICS OF THE PERIODIC UPDATE ALGORITHM

In the process of validating the network simulator using the 19-node ARPA network (see Figure 4-1), loop avoidance schemes based on locally detectable systems states were investigated. No strong correlations were found between states known to the nodes, and the formation of loops. However, a great many observations were gathered concerning the performance of the network under a variety of load conditions, update intervals and delay table biases. Analysis of the data gathered produced considerable insight into the performance of the simulated network and led directly to the formulation of the final network model. A brief comparison of network responses to various parameters is given in Figures 4-2 and 4-3.

As s result of the initial network simulations an update interval of 640 msec. was chosen for all subsequent simulation. Variations in performance of biased and unbiased algorithms were deemed to be great enough to warrant analysis, especially the effect of routing policy with both unbiased estimate and estimates biased by a 60 msec. delay. Average message delays are given in Figure 4-4.

Figure 4-2

AVERAGE MESSAGE DELAY WITH RFNM (REQUEST-FOR-NEXT-MESSAGE) WITH AND WITHOUT BIAS

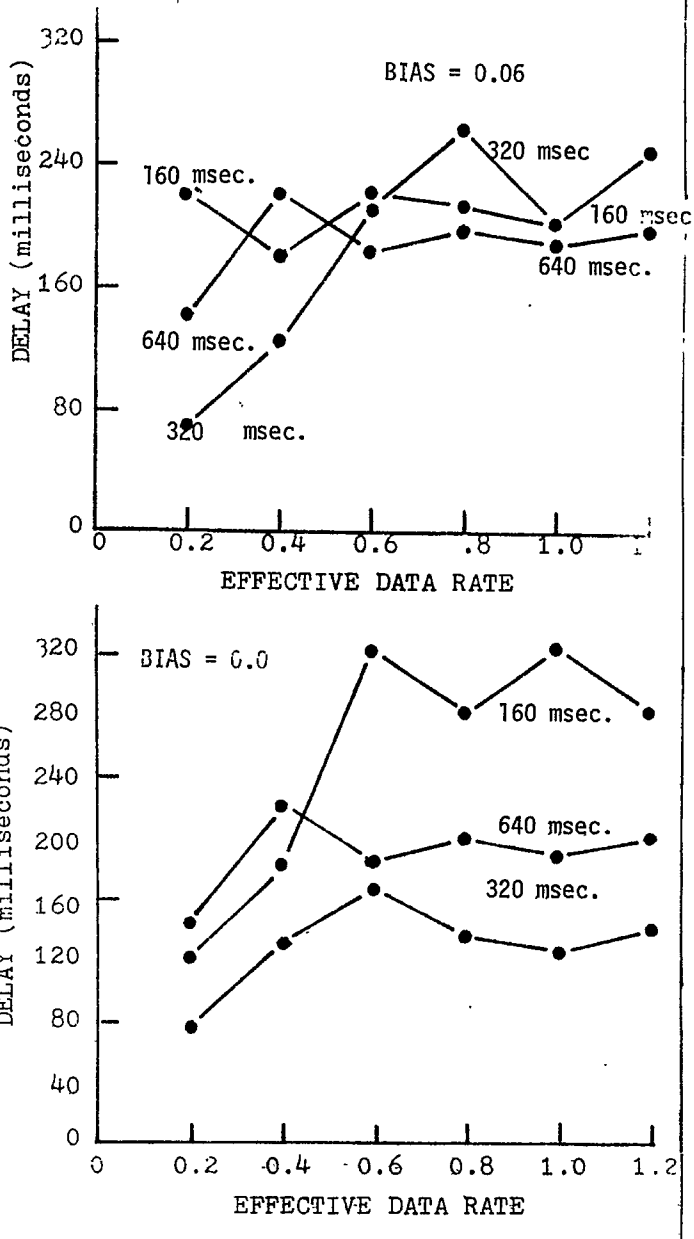
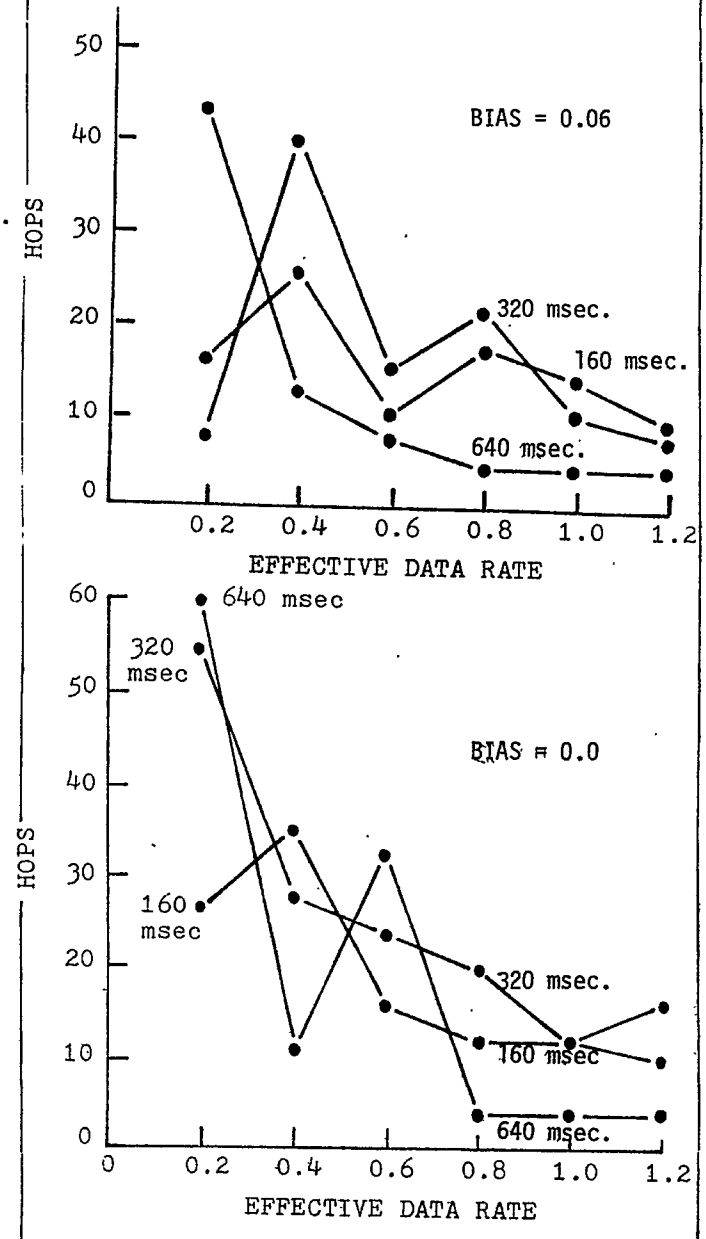


Figure 4-3

MAXIMUM NODE TRAVERSALS WITH RFNM (WITH AND WITHOUT BIAS)



AN APPLICATION TO LOOP-FREE DISTRIBUTED ADAPTIVE ROUTING

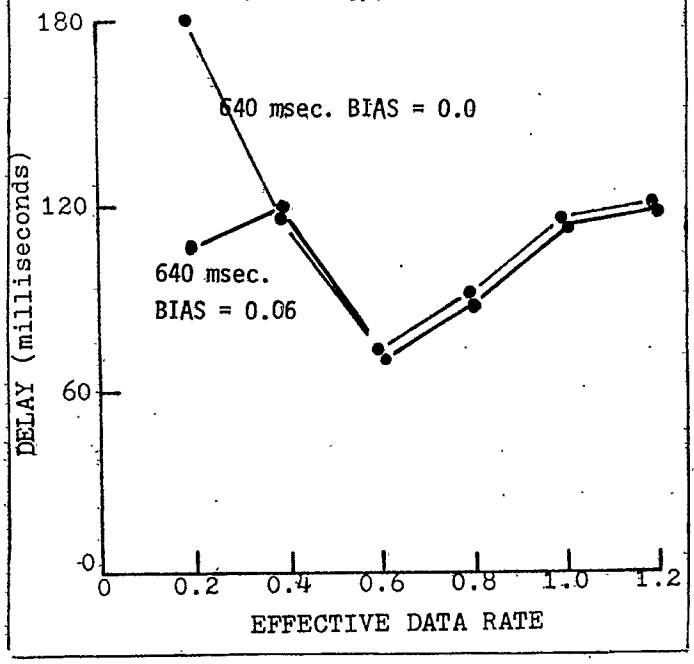
INTRODUCTION

Looping occurs when a message is returned to a node it has previously transited. Loops form because a given node has no knowledge of the path followed by the message up to the time of receipt. If the previous path were known, loop suppression would become the trivial process of refusing to route the message to a node it has previously visited, assuming some other route is always available. Such knowledge can be easily provided by appending a trace of the route to the packet. Providing such a trace requires that each node have a unique site identification, and that it append this identification to the packet as it passes through the node. The trace information is global to the network rather than restricted to neighboring nodes, and thus long loops may be detected and suppressed as well as ping-pong loops. Three objections may be raised to this procedure:

1. Networks typically require the node to have no knowledge of network topology or the identity of their neighbors, so that the network can be re-configured to accomodate failures without having to alter nodal programs. This objection, however, is not insurmountable. Each node could append its site identification to each minimum delay vector it transmits. The identification could be maintained with the delay table, freeing the receiving node from any require-

Figure 4-4

AVERAGE MESSAGE DELAYS WITHOUT RFNM (WITH AND WITHOUT BIAS)



ment for a prior knowledge of the network.

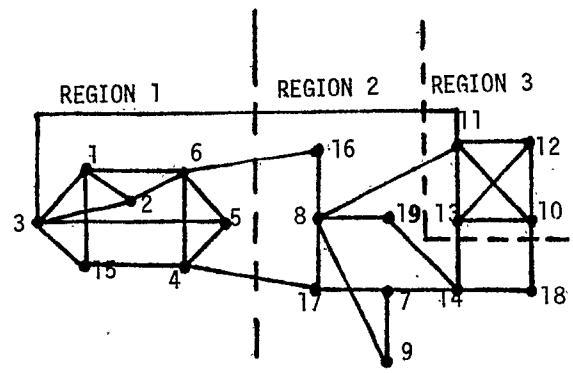
2. The trace can conceivably require a large bandwidth. If messages are in the form of packets having a fixed maximum length, text length must be reduced each time a packet visits a node in order to accomodate the trace for the node. Since portions of the text connot be discarded to provide additional space for the trace a field large enough to store the maximum possible trace length must be reserved prior to text insertion. This is wasteful of bandwidth because the average message will not transit a maximum number of nodes and much of the trace field will be unused. Furthermore, it is clear that in even a moderate sized network such a trace could conceivably exceed nominal packet size. This defect can be minimized by using a bit mapping scheme rather than an address trace.
3. The third objection,"trapping" is fatal to global procedures.

REGIONAL LOOP-FREE ROUTING

The network of Figure 4-1 forms a graph whose vertices are, at a minimum, two-connected. The objective of global algorithms is to find the minimum delay simple path from the source to the destination vertices. A region of the graph will be defined as a vertex disjoint subgraph such that each vertex is reachable from every other vertex within the subgraph. Figure 5-1 shows such partitioning of the network of Figure 4-1.

Figure 5-1

19 NODE NETWORK WITH REGIONAL NODE PARTITIONING



A boundary node is defined to be one vertex of an edge whose other vertex lies in a different region. An interior node is a node having no links to another region. A message passing through a network partitions the graph into two disjoint node sets. Namely, those nodes through which the message has not previously passed, called traversable nodes and designated π, and those previously visited nodes, called nontraversable nodes and designated by $\bar{\pi}$. The simple path property is preserved by prohibiting the message from traversing any path containing a node which is an element of $\bar{\pi}$.

Let the node addresses be of the form (α, ω), where $\alpha(x)$ is the area containing node x, and $\omega(x)$ is the node number within the area. Let S be the message source node and D be the message destination node. Let k be an arbirary region, and $M = \min\{\alpha(S), \alpha(D)\}$, and $N = \max\{\alpha(S), \alpha(D)\}$. Then let $k \in \pi$ iff $M \leq k \leq N$, $k \in \bar{\pi}$ otherwise.

With these definitions, a regional node mapping algorithm can be described as in Table I.

REGIONAL MAPPING ALGORITHM

TABLE I

1a.	If this is the source vertex and is an interior node, set the region bits to the current area, turn the source map bit on and route the message to the next node on the minimum delay path to the destination, else
1b.	If this is the source node, as well as a boundary node bordering a region $k \in \pi$, route the message to the next region; else route the message to the node on the minimum delay path within the source region.
2a.	If this is the destination node remove the message, else

2b. If this is a boundary node to the last region traversed, set the region bits to the current area and turn the node map bit on. Note that this maps all nodes in the region just exited into π, hence, the messages cannot "turn around," else

2c. If this is a boundary node to the next region to be traversed, route the message to the next region, else

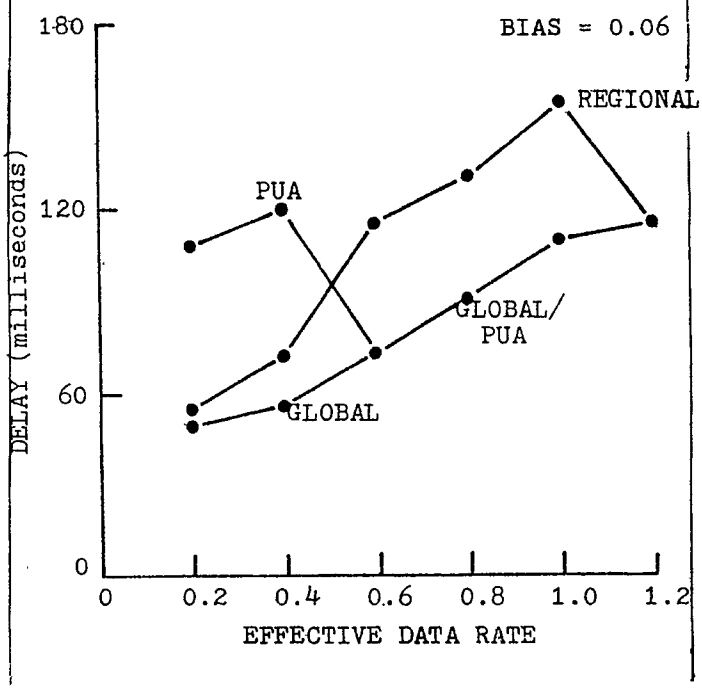
2d. Select the minimum delay line from the routing table, AND the trace map with the next node map, if the next node π route the message, else

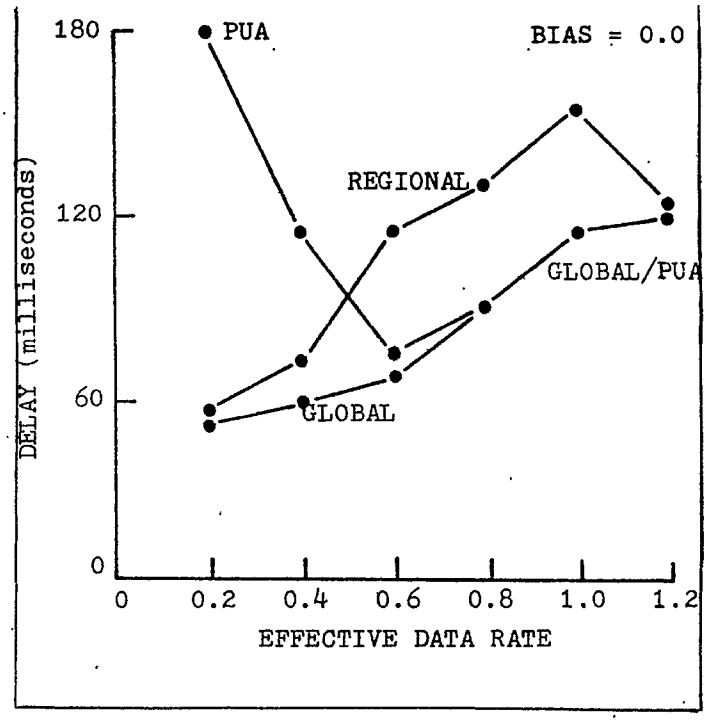
2e. Select the next best output line from the delay table and return to step 2d.

Figure 5-2 compares the performance of the global, regional, and periodic update (PUA) algorithms for both biased and unbiased delay estimates. Both global and regional algorithms provide better average delays during periods in which loops form using periodic updates without mapping. The global technique performs at least as well as the unmodified periodic update scheme throughout the range of the simulation. Performance of the regional scheme declines rapidly once network loading has reached densities at which loops no longer form. The performance of the regional algorithm at large loads can be explained by noting that the procedure forces messages across regional boundaries as they are encountered and frequently results in choosing radically suboptimal paths. Only when the network approaches saturetion loads does the regional routing again become competitive.

Figure 5-2

AVERAGE DELAY COMPARISON
(WITH AND WITHOUT BIAS)





IMPROVED LOOP-FREE ALGORITHMS

A brief description of nodal message processing is required prior to introducing the algorithm. Assume the existence of two nodes in tandem connected by a communication channel. Designate one node as the "sending node" and the other as the "receiving node." After accepting a message from some external source the sending node determines the next node to be visited and queues the message for the line to that node. A copy of the queued message is simultaneously placed on an internal "sent" queue. The line then transmits the queued message without further intervention from the sending node CPU. The receiving node checks the message, and if no errors are detected returns an acknowledgement to the sending node. If the sending node receives the acknowledgement within a specified time interval, the copy of the message on the send queue is destroyed, otherwise it is retained and a copy of it is requeued for retransmission. Also, assume that CPU utilization is very low, and that delays awaiting checking, acknowledgement, and routing are negligible at the receiving node.

Now consider the process of trap identification. The receiving node recognizes the existence of a trap when it interrogates its routing tables and finds that all output lines terminate in nodes which are elements of $\overline{\pi}$. At this time the node discards the message, thus , losing it to the system.

Furthermore, consider the effect of the receiving node withholding acknowledgement until after the routing tables have been consulted. Because CPU time is negligible, essentially no additional delay will be incurred. If a route is found out of the receiving node an acknowledgement is sent and the sending node may discard its copy of the message. If a trap is encountered the receiving node may transmit a negative acknowledgement (NAK)

and discard its copy of the message.Receipt of a NAK then causes the sending node to immediately remove its copy of the message from the sent queue, mark the message node map as having transited the node rejecting the message, and reapply the routing algorithm.Since a copy of the message has been retained at the sending node and the receiving node's copy has been destroyed, no message loop has formed nor has the message trapped, and multiple copies of the message are not in the system. Hence, a simple change in protocol permits trap avoidance without the necessity of returning messages to the sender.

Elimination of "boundary node forcing" is less straightforward but still reasonably cheap and simple to implement. The original boundary node constraint was imposed to insure that messages transit regions in a monotonic fashion, thus, preventing return of the messages to a region previously traversed. This requirement was necessary because a message once exiting a region "forgets" the trace of the nodes within that region. Thus, if permitted to reenter a previous region the message may unknowingly revisit nodes previously transited, thereby permitting loops to form. Hence, a mechanism is neede tp prevent the message from "running backward" into previously visited regions, without the inherent suboptimality of forced entry into the next region. Such a mechanism is described next. Node 16 and the links (6,16) and (16,6) joining it to node 6 are selected to illustrate the method. Note that node 16 and 6 are adjacent boundary nodes lying in different regions.

Each node maintains two routing tables, one for messages having $\alpha(S) > \alpha(D)$ and one for messages having $\alpha(S) \leq \alpha(D)$ where $\alpha(S)$ is the source region and $\alpha(D)$ the destination region. Consider a message in which $\alpha(S) \leq \alpha(D)$. The message should pass easily from node 6 to node 16, but should not pass from node 16 to node 6. To accomplish this, the entry in the $\alpha(S) \leq \alpha(D)$ table to node 6 is set to the normal delay estimate generated by the PUA algorithm. The delay in the table at node 16 is set to an arbirary delay value sufficiently high to ensure it is never reached by PUA estimates. This high delay estimate serves to ensure that the line from node 16 to node 6 is never picked by the minimum delay routing algorithm. Consequently, the duplex channel between nodes 6 and 16 appears to the message to be a simplex channel from node 6 to node 16.

For messages in which $\alpha(S) \geq \alpha(D)$ the reverse situation appertains. The $\alpha(S) \geq \alpha(D)$ delay table at node 16 is set to the normal delay estimates and the line entry in the node 6 delay table is set to the unattainable delay value. Hence, to messages in which $\alpha(S) \geq \alpha(D)$ the link appears to be a simplex line from node 16 to node 6.

The routing algorithm then reduces to testing the message header to determine the relationship between source and destination areas, and selecting the appropriate delay table. Once the delay table is selected, routing becomes identical with the familiar PUA algorithm.

Delay table updates are handled in the usual manner, except that minimum delay vectors are propagated for each delay table.

Simulation average delays are shown for both the 34 and the 35 line networks in Figures 5-3 and 5-4. As previously described, looping occurs at effective data rates less than 60 percent. Both the global and regional algorithms are superior to PUA in the looping region. Regional routing is slightly inferior to global routing at low data rates due to the prohibition agains re-entering previously visited regions. Hence, it is unable to take advantage of minimum delay paths which may appear in the previous region after the message has crossed the regional boundary.

Figure 5-3

AVERAGE DELAY COMPARISON - MODIFIED PROTOCOL (WITH AND WITHOUT BIAS)

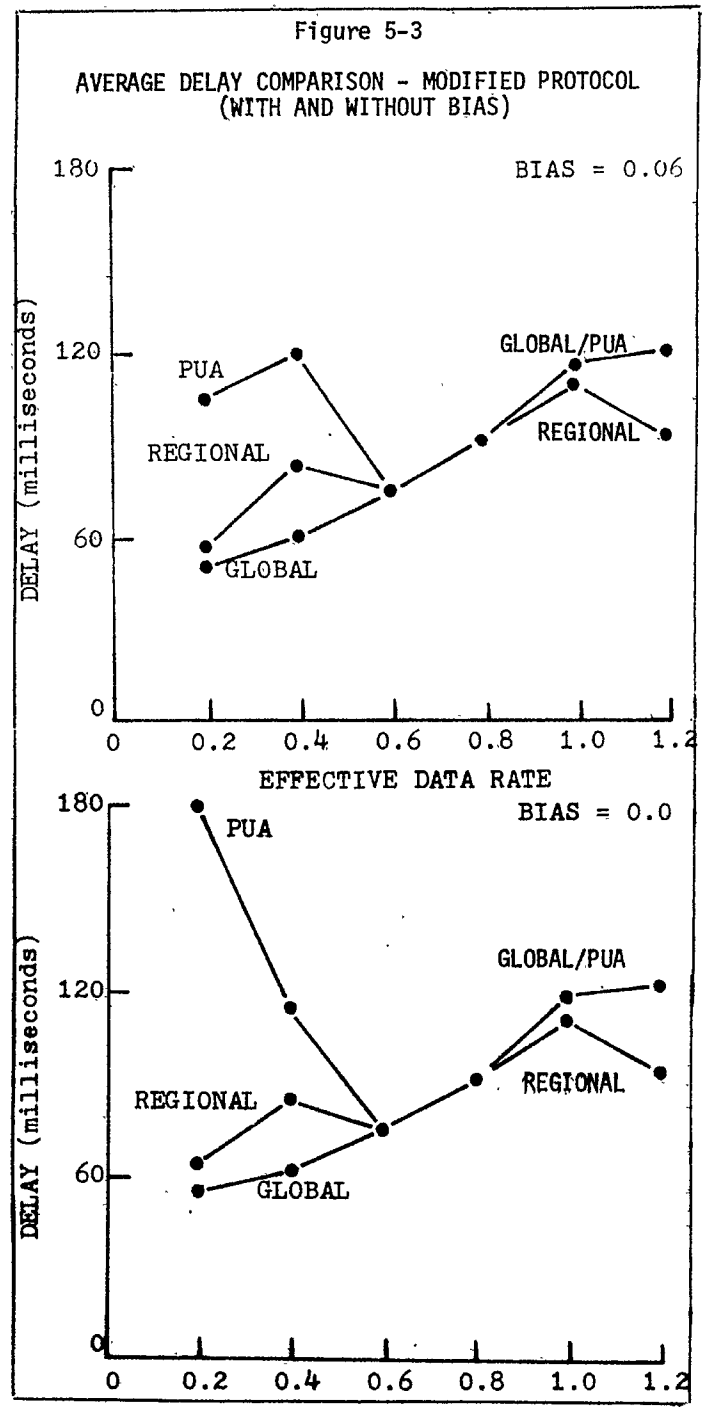
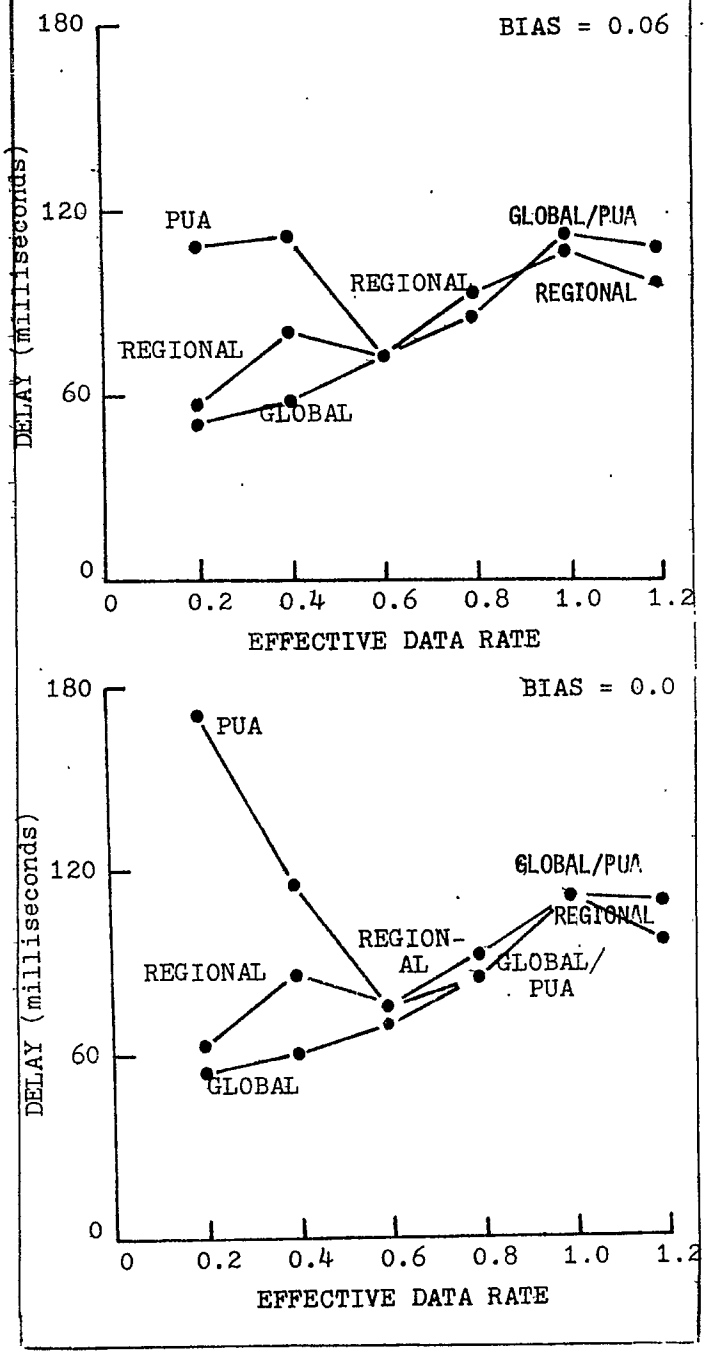


Figure 5-4

AVERAGE DELAY COMPARISON - MODIFIED PROTOCOL - LINE 16 - 19 ADDED (WITH AND WITHOUT BIAS)



In the middle range loads, 60 percent to 90 percent, the three algorithms perform equally well. At or near system saturation the regional routing approach establishes clear superiority over either of the other algorithms. This effect is observed as a result of the impulse in the regional algorithm to keep the message moving in the direction of the destination by prohibiting passage to previously transited regions. On the other hand, PUA and global algorithms show a marked tendency to oscillate at higher load levels. This oscillation, caused by large queue variations and relatively short update intervals in relation to queueing times, permits messages to circulate through a small set of neighboring nodes for long periods. Thus, long delays are produced by these algorithms at large effective data rates.

A word of caution should be added. The effectiveness of the trap suppression mechanism is heavily influenced by the partitioning employed. Partitions containing tandem, two-connected interior nodes afford no guarantee of preserving the trap-free property.

BIBLIOGRAPHY

1. Belford, G.G., Bunch, S. R., Day, J.D., et al. A state of the art report on network data management and related technology. Nat'l Tech. Info. Service, AD-A014233 (April 1975).

2. Boehm, B.W. and Mobley, R.L. Adaptive routing techniques for distributed communications systems IEEE Trans. on Communication Technology COM-17, 3 (June 1969), 340-349.

3. Burke, P.J. The output of a queuing system. Operations Research 4 (1956), 699-704.

4. Crocker, S., Heafner, J., Metcalfe, R. and Postel, J. Function oriented protocols for the ARPA computer network. Proc. AFIPS 1972 SJCC, Vol.40, AFIPS Press, Montvale, N.J., 1972, 271-279.

5. Crowther, W.R., Heart, F.E., McKenzie, A.A., McQuillan, J.M. and Walden, D.C. Issues in packet switching network design. Proc. AFIPS Nat. Comput. Conf. (May 1975), 182-196.

6. Fultz, G.L. Adaptive routing techniques for message switching computer-communications networks. Rep. No. UCLA-ENG-7252 (ARPA) (July 1972), UCLA Computer SCI. Dep. Available from NTIS as document AD-749678.

7. Hawkws, A.G. Time dependent solution of a queue with bulk arrivals. Operations Research 13, 4 (July 1965), 586-595.

8. Jackson, J.R. Networks of waiting lines. Operations Research 5, (1957), 518-521.

9. Kimbleton, S.R. and Schneider, G.M. Computer communications networks, approaches, objectives and performance considerations. Comput. Surv. 7, 3 (Sept. 1975), 129-173.

10. Kleinrock, L. Communication Nets: Stochastic Message Flow and Delay. McGraw-Hill, N.Y., 1964.

11. Kleinrock, L. Models for computer networks. Proc. of the International Communication Conference (June 1969), 21-9 to 21-16.

12. Kleinrock, L. Queueing Systems, Vol. I: THeory. Wiley-Interscience, N.Y., 1975.

13. Kleinrock, L. Queueing Systems, Vol. II: Computer Applications. Wiley-Interscience, N.Y., 1976.