ADAPTIVE ROUTING TECHNIQUES FOR STORE-AND-FORWARD COMPUTER-COMMUNICATION NETWORKS*

by Gary L. Fultz and Leonard Kleinrock

Computer Science Department
University of California, Los Angeles, California

ABSTRACT

A study is made of routing techniques applicable to store-and-forward computer networks (e.g., the ARPA Network) in order to show their importance in relation to the theoretical design of these networks and to the performance of existing networks. The major attempt has been to classify routing techniques and to specify their parameters as well as a means for evaluating their performance. Using average message delay as a measure of network performance, a number of routing techniques are compared via theoretical and computer simulation results.

I. INTRODUCTION

This paper considers message flow in a specific class of networks denoted as store-and-forward computer-communication nets. Such nets accept message traffic from external sources (computers) and transmit this traffic over some route within the network to the destination; this transmission takes place over one link at a time, with possible storage of the message at each intermediate switching node due to congestion. One of the fundamental problems in these nets is the routing of messages in an orderly manner to insure their rapid delivery. The requirements for such a system differ considerably from those of the telephone system employing circuit or line switching and from those of military communication networks required to operate in extremely hostile environments.

The study of routing techniques is important because of the central role they play in the design and operation of low cost computer-communication nets. The abstract design of a low cost computer-communication network was first stated by Kleinrock[11] as follows:

minimize T (the average message delay)

$$\text{over the design variables} \begin{cases} \text{link capacity assignment} \\ \text{message priority discipline} \\ \text{routing doctrine} \\ \text{topology} \end{cases} \quad (1)$$

subject to

a suitable cost criterion and external traffic requirement

All of the design variables are interdependent and a general solution technique is unknown, although significant progress has been made for some interesting special cases.[7,11,12,13]

Before the general solution of Eq. (1) can be undertaken, it is important to determine how the variation of the design parameters in this equation influences the average message delay T. Here we address the routing doctrine question. Key areas which require study are: what should a routing technique achieve; how can routing techniques be classified; how are routing algorithms specified; what are the appropriate performance measures and; how are routing algorithms evaluated? Below, we attempt to answer these questions in relation to the selected computer-communication network model.

II. THE COMPUTER-COMMUNICATION NET

In order to properly characterize what an adaptive routing technique (algorithm) should achieve, the universe in which it operates must first be specified. This requires a characterization for computer-communication networks.

The class of networks considered in this paper can be depicted as shown in Figures 1† and 2 and are modeled after the Defense Department's Advanced Research Projects Agency (ARPA) experimental computer network.[7,9,13,16]

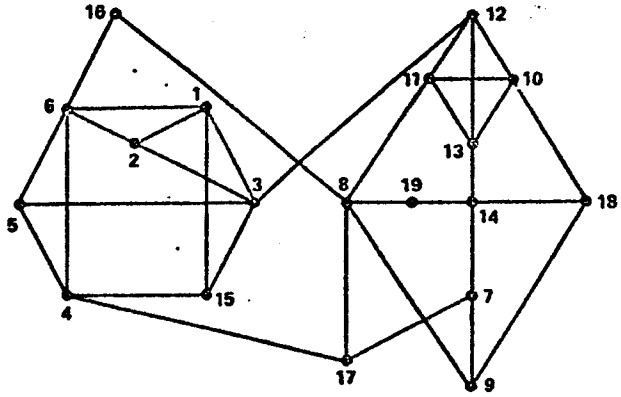


Figure 1. Network Topology

The characteristics of the network model are:

1. Each pair of nodes (N_i, N_j) can be connected by at most one dedicated high-quality (low error rate) full duplex digital communication line.
2. Each communication link has fixed capacity.
3. Each node has finite storage and operates in a store-and-forward fashion.
4. Satellites are not utilized as nodes.[6]

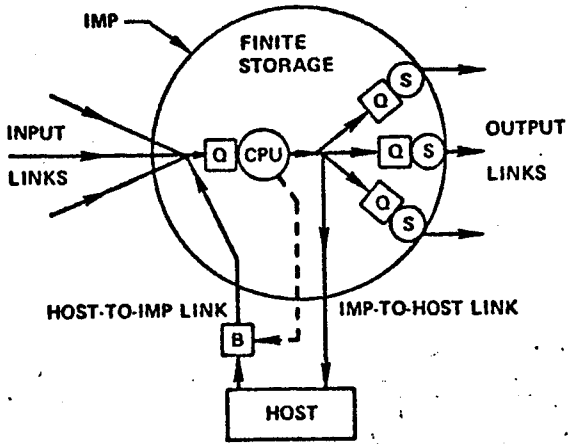
The basic unit of information passed between any pair of nodes is called a "packet" with maximum size of approximately 1000 bits. When a packet is received at a node, it is stored and checked for errors via an error detecting code. If correct and if this node is willing to "accept" the packet, then a positive acknowledgment is sent back to the preceding node indicating this fact; otherwise, a negative acknowledgment is sent back (negative acknowledgments, however, are not used in the ARPA network). When a node receives a positive acknowledgment, it destroys its copy of the packet; otherwise the packet is retransmitted. If a packet is not destined for the node at which it was received, it is relayed (routed) further along its path to a neighboring node.

*This work was supported by the Advanced Research Projects Agency of the Department of Defense (DAHC-15-69-C-0285).

†The ARPA network topology has since changed significantly. However, we continue to utilize it in order to compare our current simulation and theoretical results with those contained in Refs. 12 and 13.

The routing procedure determines the path a packet traverses from a source node N_S to a destination node N_D. For example, the paths π_1 = (5,6,16,8) and π_2 = (5,4,17, 8) are two of the many possible paths from N_S = 5 to N_D = 8 as shown in Figure 1.

The assumed internal structure of a node, shown in Figure 2, consists of a store-and-forward switch referred to as an IMP (Interface Message Processor) and a HOST (external computer system). The function of the IMP is to allocate storage for incoming packets, perform



B = SWITCH WHICH CAN BLOCK TRAFFIC FLOW TO THE IMP
CPU = CENTRAL PROCESSING UNIT ROUTINE
Q = QUEUE
S = SERVER (REPRESENTS THE FINITE RATE OF TRANSMISSION ON THE OUTPUT LINKS)

Figure 2. Basic Node Structure

routing for packets which must be relayed, acknowledge accepted packets and perform other routine functions (i.e., packet error checking, circuit fault detection, traffic measurement, etc.). In addition, the CPU routine can block incoming messages from its HOST when sufficient storage is unavailable.

Messages, which originate at a HOST, have a maximum length of approximately 8000 bits. The IMP segments a HOST's message into packets (i.e., as many maximum sized packets as necessary, plus a "remainder" packet). These packets are then handled by the network as independent entities until they reach their destination node. There the packets of a message are collected and the message is reassembled before it is transferred to the destination HOST. Messages which consist of only a single packet are given higher priority than multi-packet messages so that the network can support interactive users.

Using this network model, the message routing requirements for the computer-communication network can be simply stated:

1. Message routing should insure rapid and error-free delivery of messages.
2. The routing technique should adapt to changes in the network topology resulting from node and communication link failures.
3. The routing technique should adapt to varying source-destination traffic loads.
4. Packets should be routed around nodes that are congested or temporarily blocked due to a full storage.

III. CLASSIFICATION OF ROUTING TECHNIQUES

It is desirable to classify network routing techniques in order to gain insight into their structure, complexity and performance; from this, one may then compare them as candidates for operational network algorithms. The two major classifications selected are (1) deterministic, and (2) stochastic techniques. Deterministic routing techniques compute routes based upon a given deterministic decision rule and produce a loop-free routing procedure (i.e., packets cannot become trapped in closed paths). Stochastic techniques, on the other hand, operate as probabilistic decision rules, utilizing topology and either no information about the state of the network (random routing) or estimates of the present state of the network. With these techniques, packets may be trapped in loops for short time periods. Figure 3 shows a more complete classification of the applicable routing techniques.

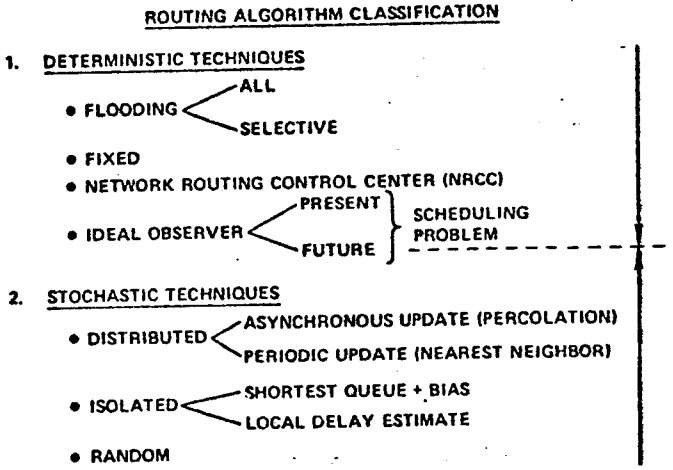


Figure 3. Routing Algorithm Classification

Deterministic Techniques

The four basic deterministic techniques are:

1. Flooding. Each node receiving or originating a packet transmits a copy of it over "all" outgoing links or over a set of "selective" outgoing links; this transmission occurs only after the node has checked to see that it has not previously transmitted the packet, or that it is not the destination of the packet. This technique has been discussed by Boehm and Mobley.[3] Their conclusion is that the inefficiency of this technique is tolerable if one has only a few messages to deliver. However, a large volume of communications traffic necessitates more efficient routing techniques. Another drawback to this technique is that each node requires a mechanism to recognize previously transmitted messages.

2. Fixed Routing. Fixed routing algorithms specify a unique path $\pi(N_S, \ldots, N_D)$ (route) followed by a packet which depends only upon the source-destination node pair (N_S, N_D). To accomplish this, each node has a routing

table similar to that shown in Figure 4. If a packet must be relayed, its destination is used to enter the routing table. The entry contained in the routing table specifies the next unique node in the packet's path. Kleinrock[11,13] and Prosser[15] have examined several of these techniques. Fixed routing techniques require completely reliable nodes and links, except for the occasional retransmission of a packet due to channel bit errors. However, they do allow for highly efficient high volume traffic flow and are very stable.

DESTINATION NODE NUMBER	NEXT NODE NUMBER
1	16
2	11
3	17
4	17
5	17
6	16
7	9
9	9
10	11
11	11
12	11
13	11
14	19
15	17
16	16
17	17
18	9
19	19

Figure 4. Node 9 Routing Table

3. Network Routing Control Center (NRCC). With this technique, one of the network nodes is designated as the NRCC. This center collects performance information about the network operation, computes routing tables and then transmits the appropriate routing table to each node in the network. Computation of the routes by the NCRR is done on a global basis and this insures loop-free paths between all source-destination node pairs. Thus, a fixed routing procedure is maintained between NRCC updates.

There are a number of drawbacks to this technique. By the time the nodes begin using the new routing tables, the performance information that was used in the computation of the routing tables may be out of date in relation to the current state of the network. In addition, transmission costs and vulnerability become significant considerations.

4. Ideal Observer Routing. This technique is essentially a scheduling problem. Each time a new packet enters a node from the HOST, its route is computed to minimize its travel time to its destination node, based upon the complete present information about the packets already in the network and their known routes. If the ideal observer has information about the occurrence of future events, then this information could also be utilized in the computation of the route. This technique is obviously impractical for an operational network, but from a theoretical viewpoint, provides the minimum average message delay to which all other routing techniques may be compared.

Stochastic Techniques

The three basic stochastic techniques are:

1. Random Routing. Random routing procedures are those decision rules in which the choice as to the next node to visit is made according to some probability distribution over the set of neighbor nodes. The set of neighbor nodes utilized in the decision rule can be "all" of the connected nodes or can be based "selectively" over that set of nodes which are in the general direction of the packet's destination.

Kleinrock[11] and Prosser[14] have investigated numerous random routing techniques and have shown that they are highly inefficient in terms of message delay, but are extremely stable (i.e., they are relatively unaffected by small changes in the network structure).

2. Isolated and 3. Distributed Techniques. All of the isolated and distributed routing algorithms operate in basically the same manner. A delay table is formed at each node as shown in Figure 5. The entries $\hat{T}_J(D,L_N)$ are the estimated delays to go from the node under consideration (say node J) to some destination node D using line L_N as the next step in the path to D. A routing table is then formed by choosing, for each row (say the i^{th} row), that output line number $OL_N(i)$ whose value in the delay table is minimum as follows:

$$OL_N(i) = \min_{\{L_N\}} \hat{T}_J(i,L_N) \qquad (2)$$

where $\{L_N\}$ is the set of output line numbers for node J. Figure 5 shows an example.

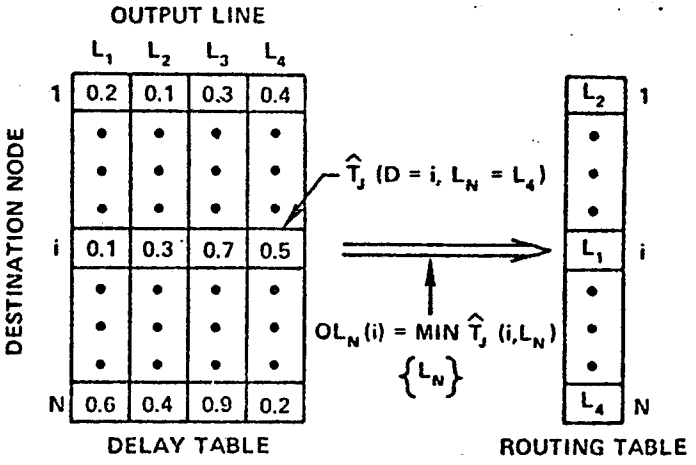


Figure 5. Node J Delay and Routing Tables

The manner in which the estimates $\hat{T}_J(.,.)$ are formed and updated and how often the delay tables are interrogated depends upon the specific structure of the routing algorithm.

In the shortest queue + zero bias algorithm, a packet's route is selected by placing it in the shortest output channel queue. This is essentially Baran's Hot Potato routing concept[1,2]. Since the route selected is independent of the packet's destination, the delay table would require only one row, where the row entries would reflect the output channel queue lengths. The non-zero bias case will be discussed later as a limiting case of a distributed routing technique.

In the local delay estimate algorithm, a packet's route is selected via Eq. (2). The delay table is updated after a packet is received (say at node J) by the following scheme

$$\hat{T}_J(D{=}S,L_N^{(R)})_{new} = K_1 \cdot \hat{T}_J(D{=}S,L_N^{(R)})_{old} + K_2 \cdot TIN(S,J) \qquad (3)$$

where

$TIN(S,J)$ = the Time the packet has spent In the Network traveling from its source node S to the current node J,

$L_N^{(R)}$ = the reverse (outgoing) line corresponding to the forward (incoming) line L_N of the full-duplex pair upon which the packet entered node J,

and

K_1 and K_2 are constants.

This technique, called backwards learning, has been extensively investigated by Baran[1]. Boehm and Mobley[3]

offer modifications to the basic technique (Eq. (3)) to improve its performance.

In the distributed routing techniques classification, all routing algorithms utilize the same basic techniques to compute and update the delay table estimates, but the instants at which these tables are updated and the route selection procedure differs depending upon the particular structure of the algorithm.

There are basically two different mechanisms which cause entries in the delay tables to change: (1) as packets are placed on (or taken off) an output channel queue, all delay table entries in a column corresponding to that output line must be increased (or decreased) to reflect the change in expected delay for the channel; and (2) when delay information from neighboring nodes is utilized to update the delay table estimates. In the latter case, the following procedure is used.

Suppose a decision at node J has been made to inform its neighbors (say N_1, N_2 and N_3, as in Figure 6) of its current minimum estimated delays to reach all nodes within the network. Node J forms a minimum delay vector $\underline{V}_J = (\hat{T}(1), \hat{T}(2), \ldots)$, where the K^{th} component $\hat{T}(K) = \min_{\{I_N\}} \hat{T}_J(K, I_N)$ and transmits $\underline{V}_J$ to its neighbors

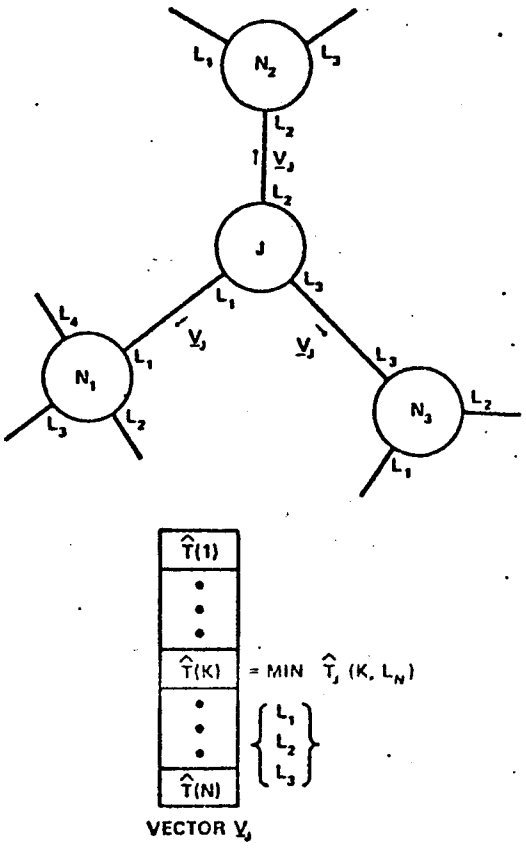


Figure 6. Minimum Delay Vector Transmission

(N_1, N_2 and N_3). Upon receipt of a minimum delay vector, a node (for example N_1) adds its current output line queue length (line L_1 for this example) plus a constant D_p to all entries in the vector $\underline{V}_J$ and replaces colum 1 (corresponding to L_1) in its delay table with these new values. Mathematically, the updated delay table entries are, in general

$$\hat{T}_M(D, I_N) = Q(M, I_N) + D_p + \hat{T}(D) \qquad (4)$$

where $Q(M, I_N)$ is the queue (in sec) of line I_N at node M. The constant D_p can be interpreted in two ways. First, if its value equals the average time to transmit a packet over an outgoing channel, then neglecting channel propagation delays, D_p represents the minimum average delay to reach a neighbor node. Secondly, if the delay tables are updated rapidly in a lightly loaded net, then the delay table estimate $\hat{T}_J(D, I_N) = N^*(J, D, I_N) \cdot D_p$ where $N^*(J, D, I_N)$ is the number of lines encountered in the path $\pi(J, .., D)$ when a packet leaves node J on line I_N. Thus, by varying D_p, we can control the degree of alternate routing and sensitivity of the algorithm to small variations in queue lengths. That is, if D_p is large compared to the average queueing delay in a node, then the path chosen for a message will tend to be one of the paths with smallest $N^*(S, D, \cdot)$.

There are two methods which can be employed to cause the transmission of the delay table update vectors $\underline{V}_J$: (1) The periodic updating algorithm forces these transmissions at a periodic rate R_U (as is currently done in the ARPA network) and (2) the asynchronous updating algorithm allows these transmissions asynchronously; this transmission can occur after the routing of a packet (via Eq. (2)) on line $OL_N(i)$ if $\hat{T}_i(i, OL_N(i))$ has changed by more than a specified amount (a threshold) since the last update occurred. Thus, the delay vectors can percolate throughout the net in a short time period. If the threshold value is excessively large, updating ceases and the asynchronous routing schemes reduce to the shortest queue + bias algorithms (with bias $D_p N^*(\cdot, \cdot, \cdot)$).

The choice of routes is determined as follows: If the update mechanism is periodic, then the set of routes obtained via Eq. (2) is held fixed until the tables are again updated; in the asynchronous case, Eq. (2) is used to determine the route of each packet dynamically.

Of all the stochastic techniques, the distributed routing algorithms are the most efficient for handling line and node failures. Once a failure is determined (see Ref. 9 for procedures utilized in the ARPA network), the proper entries in the node delay tables can be forced to remain excessively large as long as the failure persists.

Returning to Figure 3, the arrows on the right-hand side represent (from tail to head) increasing complexity and expected performance of the algorithms. Of all the routing techniques shown, we feel that the distributed stochastic routing techniques have the best potential performance to offer in operational store-and-forward computer-communication networks. These techniques operate essentially as distributed network routing control centers and can adapt rapidly to link and node failures as well as to changing traffic conditions.

IV. NETWORK PERFORMANCE

In order to design optimal computer-communication networks or to assess their performance, one requires quantitative measures of network performance. There are basically two classes of performance measures. The first class does not relate in any simple way to individual messages in the network, but rather to the performance of particular components that compose the network. Examples of such performance measures are: average channel utilization; nodal storage utilization; and channel error rates. Many of these performance measures can be computed analytically. The second class of performance measures relate more directly to individual messages and more definitive statements about overall network perform-

ance can be made. An example of such a performance measure is the measured distribution of time to transmit a message through the net. However, among the possible performance measures, the average message delay is one that has yielded nicely to analysis. In addition, it also reflects the following network phenomena in its computation:

- Message delay due to formation of queues within the nodes
- Nodal processing delays
- The decrease in effective channel capacity due to the transmission of acknowledgments and routing information within the network
- Negative acknowledgments causing packet retransmission
- Adaptability of the routing algorithm to varying traffic loads and channel and node failures
- Packet looping caused by momentary errors in estimation of the required routes by the routing algorithm, and
- Nodal storage blockage

In earlier works on communication nets[11] and computer-communication networks[12,13], Kleinrock studied such nets using methods from queueing theory[17] which he showed provide an effective method for the computation of the average delay of single packet messages using fixed routing procedures. He and Fultz[8] have modified these models to better predict the single packet message delays. In addition, they have removed some of the independence assumptions discussed in Ref. 11 in order to handle the multipacket message case. Figures 7 and 8 show a comparison of simulation[10,12] and analytical results[8] for a

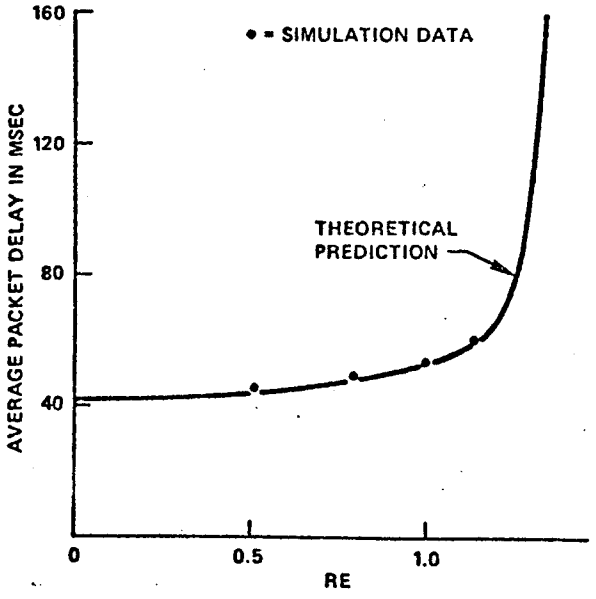


Figure 7. Single-Packet Message Delay

fixed routing procedure utilizing the network configuration shown in Figure 1. Both the analytic and simulation models reflect an assumed traffic matrix [TM] whose entries give the average traffic flow requirements in bits/second between source-destination pairs of nodes. In Figure 7 we have scaled all entries in [TM] by a factor RE (called the Effective Data Rate) which allows us to study the average message delay as a function of network loading. Figure 8 shows the average message delay

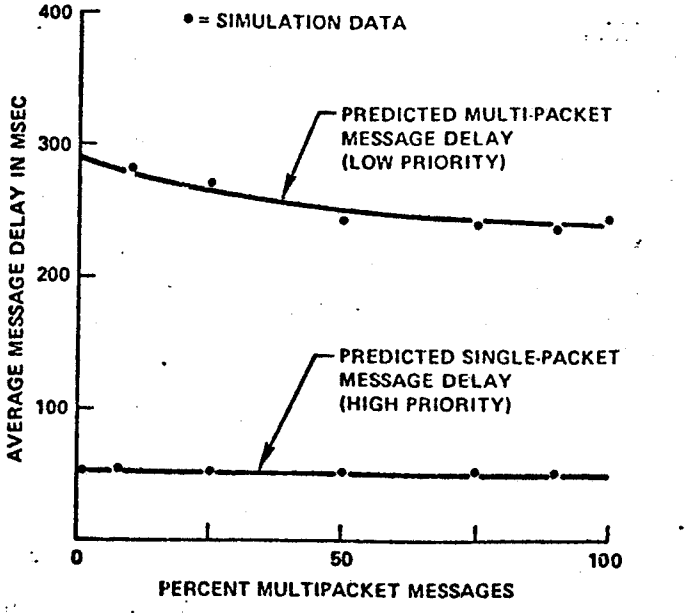


Figure 8. Message Delay Versus Mix (RE = 1)

for the two priority classes as we vary the mix of short (single-packet) high-priority messages and long (multi-packet) low-priority messages, while maintaining a constant average input data rate to the entire net. For the fixed routing procedure, we see that the average message delay is adequately predicted by the analytic results. However, when one assesses the performance of stochastic routing techniques, these curves do not indicate typical network performance. Kleinrock[11] and Prosser[14,15] have given methods to analyze random routing procedures. Here we give a method of estimating average single-packet message delay for the isolated and distributed stochastic routing procedures.

We begin by noticing that the isolated and distributed algorithms operate as fixed routing procedures over small periods of time. As time progresses and the algorithms adapt, they utilize various combinations of fixed routing procedures. Of interest is that fixed routing procedure which minimizes the average message delay for a given network loading factor RE. Figure 9 portrays the average

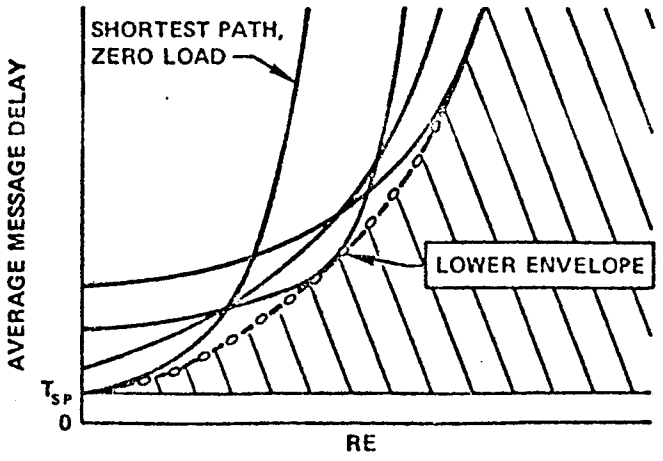


Figure 9. Average Message Delay Profile

single-packet message delay as a function of RE and various fixed routing procedures. The lower envelope of all these delay curves reflects the minimum average message delay utilizing fixed routing algorithms. We have a procedure for computing this lower envelope[8]. The horizontal line of value T_{SP} is the theoretical minimum average message delay and represents a solution of the shortest path problem[5] for RE = 0. The shaded portion of the figure represents a region of operation which can only be penetrated if the stochastic routing algorithm happens to take exquisite advantage of the instantaneous characteristics of message flow within the network to produce a smaller average message delay than the best fixed routing algorithm. To date, none of our simulation results has penetrated this region. This indicates that the lower envelope delay curve is a good measure of attainable performance for stochastic routing algorithms.

For the periodic updating algorithm, there are two parameters which may be adjusted for performance optimization (D_P and the periodic update rate R_U) for any value of RE. Figure 10 shows this performance as a function of D_P for various values of R_U with RE = 1. These delay curves reflect the additional message delay caused by the presence of the routing update traffic flow within the net. For each routing update packet (which contains the vector $\underline{V}_J$), its line transmission time, T_U, utilized in the simulation program was $0.8T_P$, where T_P (= 12.6 msec) is the average line transmission time for a single-packet message (average packet length in bits divided by the line capacity in bits per sec).

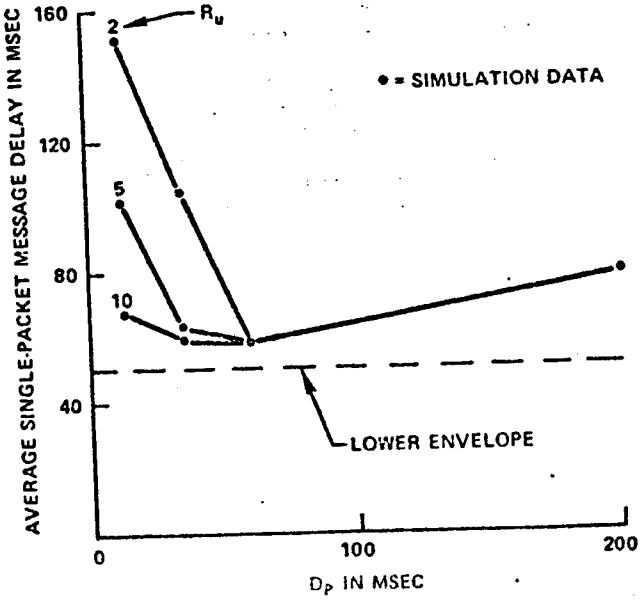


Figure 10. Periodic Updating Algorithm (RE = 1)

For small D_P, the simulation program shows that many loops exist in the fixed routing procedure utilized between delay table updates and thus produces a large average message delay as shown in the figure; the higher values of R_U shown permit better adaptation to the traffic, even offsetting the increase in traffic due to these updates. Although not shown in the figure, limited simulation data indicates that the average delay for R_U = 20 updates/sec is larger than for R_U = 10 updates/sec; thus R_U cannot be increased indefinitely without suffering a loss in performance.

For large D_P (60 msec and greater), little evidence of looping is found. The average message delay for D_P = 200 msec is within two msec of the simulation result at RE = 1 for the fixed routing procedure based upon the solution of the shortest path, zero-load problem. This indicates that the delay table updating cannot, for this value of D_P, adapt to the fluctuations in network traffic so as to lower the average message delay. However, the algorithm can still adapt to line and node failures and the delay and routing tables would reflect these failures. For the simulation data plotted in Figure 10, the minimum average delay occurs at $D_P \approx 60$ msec, which is approximately five times as large as the average line transmission time T_P for a single packet. In the solution of T_{SP} for this network, the longest route also contains five lines. Further investigation is required to determine if there is a similar observable pattern for other values of RE and for variations in the traffic matrix [TM] and network topology.

For the asynchronous updating algorithm, there are also two parameters which can be adjusted for performance optimization (D_P and the threshold values). Here we consider constant thresholds (adaptive thresholds will be considered in the future). The simulated updating procedure operates as follows: A copy of the new minimum delay vector, $\underline{V}_J$, is retained in node J each time it is formed for updating. As packets are routed at node J via Eq. (2), the minimum delay corresponding to $OL_N(i)$ is compared to its corresponding entry $\hat{T}(i)$ in the stored vector. $\underline{V}_J$ as shown below.

$$\left|\hat{T}_J(i) - \hat{T}_J(i, OL_N(i))\right| = \Delta\hat{T}_J(i) \qquad (5)$$

If $\Delta\hat{T}_J(i) \geq$ threshold, then the update procedure is invoked as shown in Figure 6. Otherwise, no update occurs. The motivation, of course, for utilizing thresholds is to sense changes in the traffic distribution (delay) and only update when these changes are pertinent as opposed to the periodic updating algorithm which forces updates even when the delay tables remain static. Figure 11 shows the algorithm performance as a function of D_P

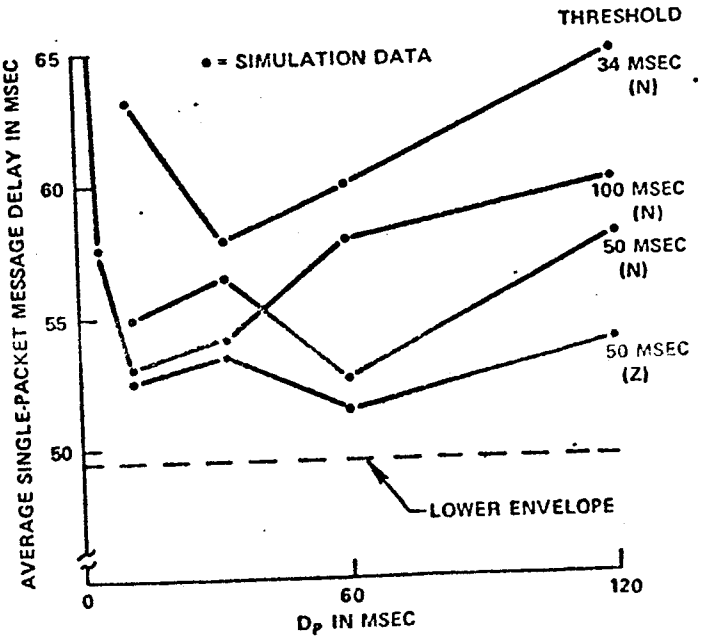


Figure 11. Asynchronous Updating Algorithm (RE = 1)

for various threshold values with RE = 1. The curves labeled (N) indicate normal operation of the algorithm ($T_U = 0.8T_P$), while the curve labeled (Z) corresponds to $T_U = 0$. Thus, the difference between the two 50 msec threshold curves represents the increase in average message delay due to the presence of the update traffic within the net.

The asynchronous update algorithm does not exhibit the distinct minimum message delay as a function of D_P as found for the periodic update algorithm. Also, no correlation was found between the number of updates and message delay for a fixed threshold value, even though the number of updates increased as D_P increased (except for the dip in the 50 msec threshold curves at D_P = 60 msec). For a fixed $D_P \geq 60$ msec, there is a correlation between average message delay and threshold, the minimum being at approximately the 50 msec threshold, which lies between the 34 and 100 msec thresholds.

Perhaps the most interesting delay curve shown in Figure 11 is that for a threshold of 100 msec. For $D_P \leq 34$ msec, no updates occurred during the simulation; thus the algorithm operation reduced to the shortest queue + bias class. However, line and node failures would cause the algorithm to update. It is quite possible that the threshold test (Eq. (5)) could be eliminated and updating forced only when a line or node failure is recognized. This requires further investigation. For $D_P \leq T_P$ msec, the algorithm becomes highly unstable and many loops appear in the routing. This accounts for the large increase in delay, as the figure indicates.

Finally, Figure 12 shows the best simulated performance of three routing algorithms (periodic updating, asynchronous updating and shortest queue + bias) as a function of the network traffic loading factor RE. The lower envelope represents the achievable average message delay for the best fixed routing scheme (although it would vary with RE), but has not as yet been simulated. However, the results shown in Figure 7 indicate that simulation should agree with this theoretical curve.

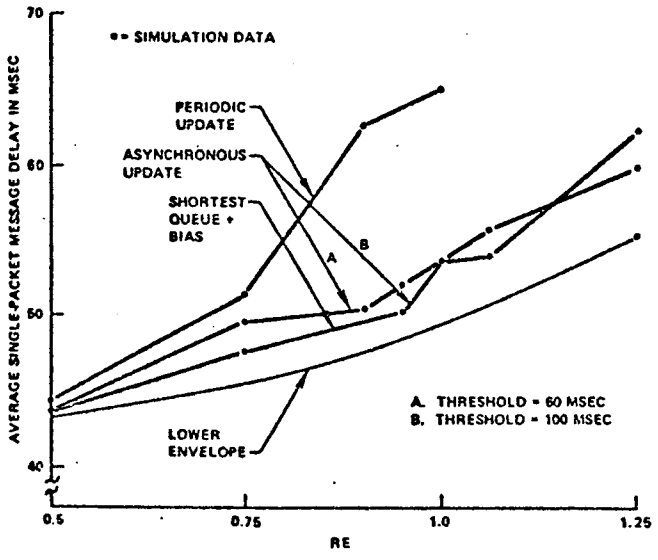


Figure 12. Comparison of Routing Algorithms

The periodic updating algorithm is shown to be inferior in performance to the other algorithms simulated. Apparently, utilizing a fixed routing procedure between updates causes increased congestion within the network and thus increases message delay.

For a moderate threshold value (100 msec), the asynchronous updating algorithm achieves the same performance as the shortest queue + bias algorithm (because no delay table updates were initiated during the simulation). The 60 msec threshold value produces a very interesting result. As RE approaches 1.25, the asynchronous updating algorithm performs better than the shortest queue + bias algorithm. This shows that the algorithm is utilizing the information contained in the minimum delay vectors V_J to adapt to the fluctuations in network traffic flow. Further, it indicates that the presence of the delay table updating traffic within the net does not necessarily cause an increase in the average message delay.

Before a more detailed comparison can be made among the algorithms, further understanding of the relationship between D_P and R_U or the threshold value must be gained. In addition, line and node failures must be simulated in order to determine how rapidly the algorithms adapt and what average message delay they produce.

V. CONCLUSIONS

We have presented a meaningful overview of routing techniques available for computer-communication networks and have developed the structure of routing algorithms which appear to be the most promising for operational networks. The main thrust of our research has been to develop models of network performance and routing algorithms and compare their performance via computer simulation. Moreover, preliminary measurement data (time delay measurements, degree of alternate routing, etc.), collected by Cole[4] on the ARPA network, indicates general agreement with our simulation results. We are now in a position to compare our analytic and simulation models with real network performance data.

We have demonstrated that fixed routing procedures perform most effectively from among our many comparisons; however, such procedures cannot adapt to variations in network traffic and topology. The adaptability of our distributed stochastic algorithms provides efficient performance under such variations and they appear as strong candidates for use in store-and-forward computer-communication nets.

REFERENCES

1. Baran, P., "On Distributed Communications," The Rand Corporation, Series of 11 Memoranda, August 1964.

2. Boehm, S., and P. Baran, "Digital Simulation of Hot-Potato Routing in a Broadband Distributed Communication Network," The Rand Corporation, Memorandum, Rm-3103-PR, August 1964.

3. Boehm, B. W., and R. L. Mobley, "Adaptive Routing Techniques for Distributed Communications," The Rand Corporation, Memorandum, Rm-4781-PR, 1966.

4. Cole, G., "Computer Network Measurements: Theory and Design," Ph.D. Dissertation, Computer Science Department, University of California at Los Angeles, to be published, 1971.

5. Dijkstra, E. W., "A Note on Two Problems in Connection with Graphs," Numerische Mathematik, pp. 269-271, Vol. 1, 1959.

6. Ferrell, C. W., and P. J. Knobe, "The Impact of Satellite Communications on Computer Networks," Proceedings of the Computers and Communications Conference, Rome, New York, September 30-October 2, 1969.

7. Frank, H., I. T. Frisch and W. Chou, "Topological Considerations in the Design of the ARPA Computer Network," AFIPS Conference Proceedings, Spring Joint Computer Conference, May 1970.

8. Fultz, G. L., "Adaptive Routing Techniques for Store-and-Forward Message-Switching Computer-Communication Networks," Ph.D. Dissertation, Computer Science Department, University of California at Los Angeles, to be published, 1971.

9. Heart, F. E., R. E. Kahn, S. M. Ornstein, W. R. Crowther and D. C. Walden, "The Interface Message Processor for the ARPA Computer Network," AFIPS Conference Proceedings, Spring Joint Computer Conference, May 1970.

10. IBM Corporation, "General Purpose Systems Simulator III, User's Manual," Form H20-0163.

11. Kleinrock, L., Communication Nets; Stochastic Message Flow and Delay, New York: McGraw-Hill, 1964.

12. Kleinrock, L., "Models for Computer Networks," Proceedings of the International Communications Conference, Boulder, Colorado, pp. 21-9 to 21-16, June 1969.

13. Kleinrock, L., "Analytic and Simulation Methods in Computer Network Design," AFIPS Conference Proceedings, pp. 569-579, Spring Joint Computer Conference, May 1970.

14. Prosser, R. J., "Routing Procedures in Communication Networks, Part I: Random Procedures," IRE Transactions on Communication Systems, CS-10, pp. 322-329, 1962.

15. Prosser, R. J., "Routing Procedures in Communication Networks, Part II: Director Procedures," IRE Transactions on Communications Systems, CS-10, pp. 329-335, 1962.

16. Roberts, L. G., and B. D. Wessler, "Computer Network Development to Achieve Resource Sharing," AFIPS Conference Proceedings, pp. 543-549, Spring Joint Computer Conference, May 1970.

17. Saaty, T. L., Elements of Queueing Theory with Applications, New York: McGraw-Hill, 1961.