MEMORANDUM
RM-3103-PR
AUGUST 1964

ON DISTRIBUTED COMMUNICATIONS:
II. DIGITAL SIMULATION OF HOT-POTATO ROUTING IN A BROADBAND DISTRIBUTED COMMUNICATIONS NETWORK

Sharla P. Boehm and Paul Baran

PREPARED FOR:
UNITED STATES AIR FORCE PROJECT RAND

The RAND Corporation
SANTA MONICA • CALIFORNIA

RD

MEMORANDUM
RM-3103-PR
AUGUST 1964

ON DISTRIBUTED COMMUNICATIONS:
II. DIGITAL SIMULATION OF HOT-POTATO ROUTING IN A BROADBAND DISTRIBUTED COMMUNICATIONS NETWORK

Sharla P. Boehm and Paul Baran

This research is sponsored by the United States Air Force under Project RAND—Contract No. AF 49(638)-700 monitored by the Directorate of Development Plans, Deputy Chief of Staff, Research and Development, Hq USAF. Views or conclusions contained in this Memorandum should not be interpreted as representing the official opinion or policy of the United States Air Force.

The RAND Corporation
1700 MAIN ST • SANTA MONICA • CALIFORNIA • 90406

PREFACE

This Memorandum is one in a series of eleven RAND Memoranda detailing the Distributed Adaptive Message Block Network, a proposed digital data communications system based on a distributed network concept, as presented in Vol. I in the series.* Various other items in the series deal with specific features of the concept, results of experimental modelings, engineering design considerations, and background and future implications.

The series, entitled On Distributed Communications, is a part of The RAND Corporation's continuing program of research under U.S. Air Force Project RAND, and is related to research in the field of command and control and in governmental and military planning and policy making.

The present Memorandum, the second in the series, describes preliminary computer simulation of a message routing scheme investigated as part of a study of ways of reducing the vulnerability of command and control communications networks. This routing doctrine, referred to as the "hot-potato" switching doctrine, differs from normal store-and-forward switching in that it permits an apparent "real-time" transmission of data even though it itself uses store-and-forward techniques.

The purpose of the simulation was primarily to determine whether the doctrine contained any unforeseen problems, and to define first-cut design parameters. A

---

*A list of all items in the series is found on p. 47.

further simulation permitting faster operating speeds and a larger array of nodes is reported in Vol. III in the series.

SUMMARY

The small-memory, stochastic, hot-potato routing scheme described in Vol. I in this series has been simulated on the IBM 7090 computer, using FORTRAN language. The simulation has shown that an efficient store-and-forward routing doctrine can be performed in a distributed network using a relatively simple policy at each Switching Node.

The computer program simulated real-time operation of the network, with a time factor of about 540 seconds of computer time to one second of real-time operation. Network learning time was determined as a function of traffic volume, redundancy of connection, and maximum allowable handover number. It was found that, from a dead start, a 49-node network can learn to route messages very effectively within about one second of scaled real-world time. The traffic-handling capacity of the network was examined, and it was found that loadings on the order of about 50 per cent of peak capacity were possible without excessively increasing the transmission path length or delay time.

The mean handover number (path length) for messages traveling through the network was studied as a function of maximum allowable handover number and of redundancy. It was found that the mean handover number is quite insensitive to maximum allowable handover number; and, the greater the link redundancy, the lower the mean handover number.

During the simulation it was found that a certain percentage of the messages were dropped when the handover

number exceeded the allowable maximum. This problem has been solved by J. W. Smith and is described in Vol. III which is a continuation of the work started in the present Memorandum. With this exception, no major difficulties were encountered. It appears that the basic routing doctrine examined can permit a network to suffer a large number of breaks, then quickly reconstitute itself by rapidly relearning to make "optimum" use of the surviving links.

CONTENTS

I. INTRODUCTION

This study examines a communications network which is comprised of an array of stations interconnected by links. A dynamically changing routing doctrine is used, such that if any paths exist between the i^{th} station and the j^{th} station, the network will seek to route messages by the shortest such path. Such a network routing doctrine provides a very much higher degree of alternate routing capability than other networks now in use.

As it is difficult to visualize in a straightforward analysis how such networks behave, we have resorted to a "Monte Carlo" simulation. Hypothetical messages are "generated" at many stations and "transmitted" to other stations using simple pre-arranged switching rules at each node. From time to time, parameters, such as the number of messages per unit time, are varied and the behavior of the network is observed. The approach is like that of the psychologist who regards a human being as a black box: he applies inputs and observes the outputs.

All traffic to be transmitted within the model is generated at the stations or nodes. Traffic to be transmitted is first chopped into small blocks, called Message Blocks, or simply messages. These messages are then relayed from station to station through the network with each station acting as a small "post office" connected to adjacent "post offices." Messages will eventually arrive at the desired destination. In the proposed system, the transmission time and storage time at the nodes is so short that to the user there appears to be a direct link between

the originating station and the end station. In effect, a slow digital-stream of bits inserted into the i^{th} station has been stored in a buffer until enough bits are received to form the Message Block.* Each Message Block is whipped into the network and transmitted over high-speed transmission links, to finally emerge at the j^{th} receiving station. At the j^{th} station, the bit-stream is reconstructed into its original slow digital-stream.

Each node starts with complete ignorance of the state of its neighbors; i.e., whether its neighbors are alive or dead, and which links exist and which do not. With the routing doctrine used, each node soon learns enough about its external environment to be able to operate independently, creating a reasonably efficient overall switching flow. The term "efficient switching" is used to describe the case where messages are transmitted over paths which approximate the shortest possible path between the transmitting and receiving stations and allow high average throughput rates.

Messages are assumed to originate uniformly in the network at the early stage of the simulation. The end-destination stations are chosen at random, but a message cannot have the same origin and destination. Delays are assumed for transmission and processing time at each node and link.

---

*In practice, this would be done at a substation called the Multiplexing Station, as described in ODC-VIII. (ODC is an abbreviation of the series title, On Distributed Communications; the number following refers to the particular volume within the series. A list of all items in the series is found on p. 47).

A decision is made at each node as to the best non-busy link over which each Message Block shall be relayed. "Best" is defined as the shortest measured path length from the desired end station. Rather than wait for the best link to be free, the second, third, etc., best link that is not busy (or down) is taken. If at a particular node or station, several links of the node equally fulfill this requirement, one is chosen at random. In general, the link chosen is the one that appears to be the next one on the shortest path to the end station. This path choice is determined by an information tag affixed to each message, called a handover number. Each station or node notes the handover number (number of steps) of each incoming message. It keeps track of these numbers on a handover number table, which it uses to determine which of its neighbors appears to be most directly connected with each originating station. Later, the node uses this recorded information to select the preferential link over which messages to each end station should be directed.

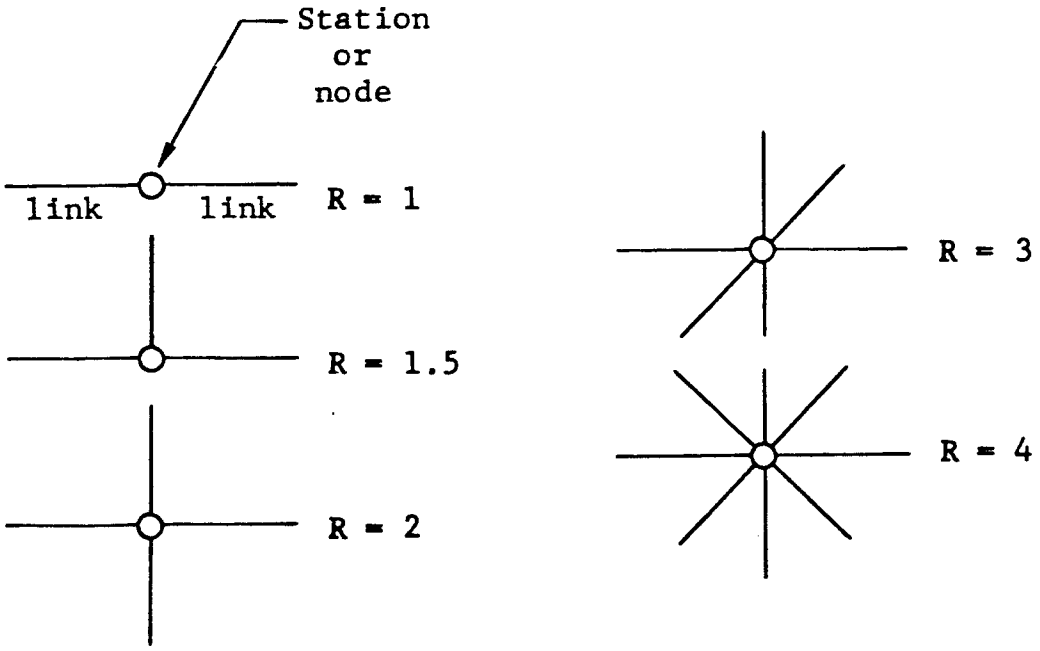
As links may be destroyed and other links may be repaired, and as traffic loads vary throughout the network, it is necessary to provide some means at the node for learning and forgetting the "best" direction in which to send messages going to any destination.

II. THE SIMULATED NETWORK

DESCRIPTION

The size of the network simulated was limited by the amount of storage available in the IBM 7090 computer using FORTRAN. A heavy storage requirement was dictated by the need for each simulated node or station to maintain a table of recorded handover numbers--the tag appended to each message indicating the number of times that message has been relayed. For each node, a table containing handover numbers to every other node via every one of up to a maximum of eight links is needed.

The choice of a maximum of eight links was made by limiting the simulation to networks with a redundancy level, R, of four or less, defined as follows:



Specifically, the configurations considered were for redundancy levels of 1, 1.5, 2, 3, and 4. In a large array, the number of links emerging from a node is equal to two times the redundancy level. The network under consideration was limited to a seven-by-seven array.*

PURPOSE

This simulation was written to examine and to test the properties of a network of a type which has never been built. It was hoped that the choice of such hitherto unspecified parameters as the maximum handover number and the allowable traffic volume could be narrowed. Results vary decisively as these parameters change. We sought first a reasonably good set of basic key parameters before adding additional refinements necessary to simulate a real-world system. Thus, the simulation changed in midstream as modifications were made from time to time to improve the routing doctrine. For example, while tracing a message as it moved through the system it was found that certain conditions would cause two nodes to relay messages in a ping-pong fashion. Instead of moving out into the system, the message simply bounced back and forth between the two nodes. This was straightened out by a modification to the switching rules before continuing subsequent runs.

---

*A larger array of stations (> 11 x 11) has been simulated by J. W. Smith, as described in ODC-III.

THE ROUTING DOCTRINE IN THE FIRST SIMULATION

The initial constraints were:

a) Each node accepts new traffic messages if, and only if, open storage capacity exists at the node.

b) A "from" address and a handover number tag is appended to each message inserted into the network.

c) Every time a message is relayed from one node to another, the handover number is incremented by one.

d) A table of the "minimum" handover number encountered relative to each originating station over each link in the network is stored at each node.

e) Before the simulation is started, the entries on the tables of handover numbers are set to predetermined maximum values.

f) Messages found in the network with a handover number exceeding a maximum allowable value are dropped. (On some runs, the location, the source, and the desired end point of such dropped messages are output to facilitate isolation of trouble spots.)

g) Messages correctly delivered are erased. (On some runs, the handover number, station of origin, and terminal or "sink" station are printed out for analysis.)

h) Messages are routed by examining the handover number table and choosing the outgoing "non-busy" line bearing the lowest handover number table entry.

i) If all lines are busy, a message waits at the node until one is free.*

---

*This has since been changed to prevent the "all-lines-busy" case from ever occurring; the procedure is called "choking," and is described in detail in ODC-III, -IV.

III. DESCRIPTION OF THE PROGRAM: SUBROUTINES

The program consists of a set of subroutines written in FORTRAN. As the program was constantly changed, certain parts of the program listing in the Appendix are residuals from earlier runs.*

MAIN

MAIN reads the input and calls up most of the subroutines in the basic program.

SETTAB

SETTAB is used at the beginning of each run to establish an initial maximum handover number table for each link of each node. The handover number from each node to every other node via all possible links is set to the value KMAX, which is simply a starting value. The border links of the network are set to IFIN (32,000) to keep messages within the chosen size network.

RANTIM

RANTIM sets values of random-length time delays. It has changed since the first draft of the program, when it was thought necessary to maintain separate values for switching delays in the nodes and in the links. However, as it now seems possible to build hardware to switch all messages within an almost constant time interval (see

*This section presumes familiarity with FORTRAN. The reader who is not so conversant may wish to advance directly to Sec. IV, p. 11.

ODC-VII), we have chosen to combine these delays into a single "pipe filling" and message-length combination delay. As real-world networks have varying-length paths, each separate transmission delay is chosen by use of a random number generator test per network run. As each link in the system can send and receive simultaneously (full duplex), this subroutine sets the delay time on both legs of each link to the same value.

MOVE

MOVE is the heart of the program. It contains an iterative FORTRAN "DO" loop to manage all of the message propagation actions of the system. Snapshot printouts which indicate network status behavior are performed within this subroutine. Each message in the network is tested to determine its location and whether sufficient propagation time has elapsed to change its status. If, at the time of examination, the message is at a node and is free to move to an output link, subroutine SELINK (see below) is called. Similarly, if the message is traveling down a link and exceeds the propagation time delay, SELNOD (see below) is called into play.

NEW

Subroutine NEW is used to generate messages. If the message has either reached its desired terminal destination, or its maximum allowable handover number has been exceeded, subroutine NEW is called to generate a new message. Since we are simulating an array of 49 stations, it was convenient to make the "from" address of a message

exactly equal to the number of the message modulo 49. (This explains why the number of messages in the system at any one time in most of the runs is a multiple of 49.) The "to" address of each message is chosen at random. For example, when a message from Node 3 has been delivered, another message will start from Node 3. One set of trial runs was made with all messages starting from a central node, while all other runs generated traffic uniformly at all nodes.

SELINK

SELINK selects the link or path to be taken to convey each message through the network. Each link is tested to see if it is usable. The output link chosen is that link which is not busy and which has the lowest-value entry on the local handover number table.

SELNOD

Subroutine SELNOD serves two purposes. It checks to see if messages have arrived at end destinations, and it updates the learning table. Subroutine SELNOD checks the handover number and the destination of the message. If either the maximum allowable handover number has been exceeded or the desired end destination has been reached, a signal is set to call subroutine NEW; otherwise, the handover number is merely increased by one. Next, the lowest previously-found entry in the handover table is compared to that of the handover number of the newly-arrived message. Whichever is smaller replaces the former value on the handover table. This permits the node to

determine the best directions for sending future traffic. This minimum-value substitution is called "perfect learning," as it provides a record of the shortest path a message ever took to arrive at the node. In order to make the network insensitive to possible errors and to allow for rapid change of routing, "imperfect learning" is used. Imperfect learning includes weighting factors to allow for error, failure of the network, and repair of the network.

PRINT and ERROR

These routines control the various forms of printouts needed to evaluate network performance.

DAMAGE

Subroutine DAMAGE is used to destroy nodes in order to simulate failure or damage effects. The basic DAMAGE subroutine can destroy nodes in a controlled random fashion where the percentage of nodes to be destroyed has been specified.

IV. TIME SCALE AND SCALING FACTORS

The network is a highly parallel system in which separate simultaneous transactions are taking place, as in a three-ring circus. In order to simulate this network on a computer, which is a pure serial device capable of processing only one event at a time, it is necessary to examine each link and each node sequentially. Perfect simulation of events can only be performed if the assumed sequential time intervals of examination are so short that not more than one interacting change of state takes place during any single time interval. This approach, of course, is quite wasteful of computer time.

Fortunately, it was found that good simulation could be performed with a very coarse "grain size" of time. We have standardized upon this time unit as a standard "time-frame." All other times are scaled in terms of this basic value. The following is a scaling of real-world times in time-frame units.

t_f = 0.33 milliseconds = 1 time-frame.

t_s = station processing delay time; variable from 0 to 0.66 ms ≃ 0.33 milliseconds; average = 1 time-frame.

t_p = pipe filling plus message-length time; variable from 0.66 to 2.64 ms; 2 to 8 time-frames.

t_e = message length; 0.66 ms; 2 time-frames.

The values assumed here are for links up to about 150 miles in length, transmitting 1024-bit Message Blocks at a data rate of about 1,500,000 bits per second. A period of 3000 time-frames is equivalent to one second

of real-world time. Since it takes about nine minutes to simulate one second of real-world time, the ratio between real-world time and simulation time is about 1:540. Each message is held at each node for a single time-frame. If a link is free, the message may move after the count. It takes from two to eight time-frame counts to move from one node to the next. The random value within the limits of t_p is re-chosen whenever a new network is defined; the greater the value chosen, the greater the geographical length of the assumed communication link.

V. DETERMINATION OF LEARNING TIME

We define "learning" as the utilization of knowledge acquired by each node of the system as to the network status so as to best determine the link that lies on the shortest route to each end station. If messages are routed over paths whose mean lengths approach the shortest possible path lengths, we say the network has "learned" well." A useful metric for evaluating learning is the total value of all entries in the handover number tables of all nodes. This value is called the "table total": the lower the value, the better the learning. This learning curve approaches a theoretical minimum with network use.

TRAFFIC VOLUME

The network learns quickly as the volume of traffic inserted into the virgin network is increased and levels off as the capacity of the system is approached. It can be seen in Fig. 1 that starting off a new network with dummy traffic is desirable if the absolute shortest learning time is desired; otherwise, many links that may lie on short paths would never be tested.

In Fig. 1, the maximum allowable handover number was set at 40, and the number of messages per unit time was varied. With a small volume of message traffic initially, the system took longer to learn, as expected.

In Fig. 2 it can be seen that the number of messages delivered per unit time increased with network load, but leveled off at 60 per cent capacity, providing an upper

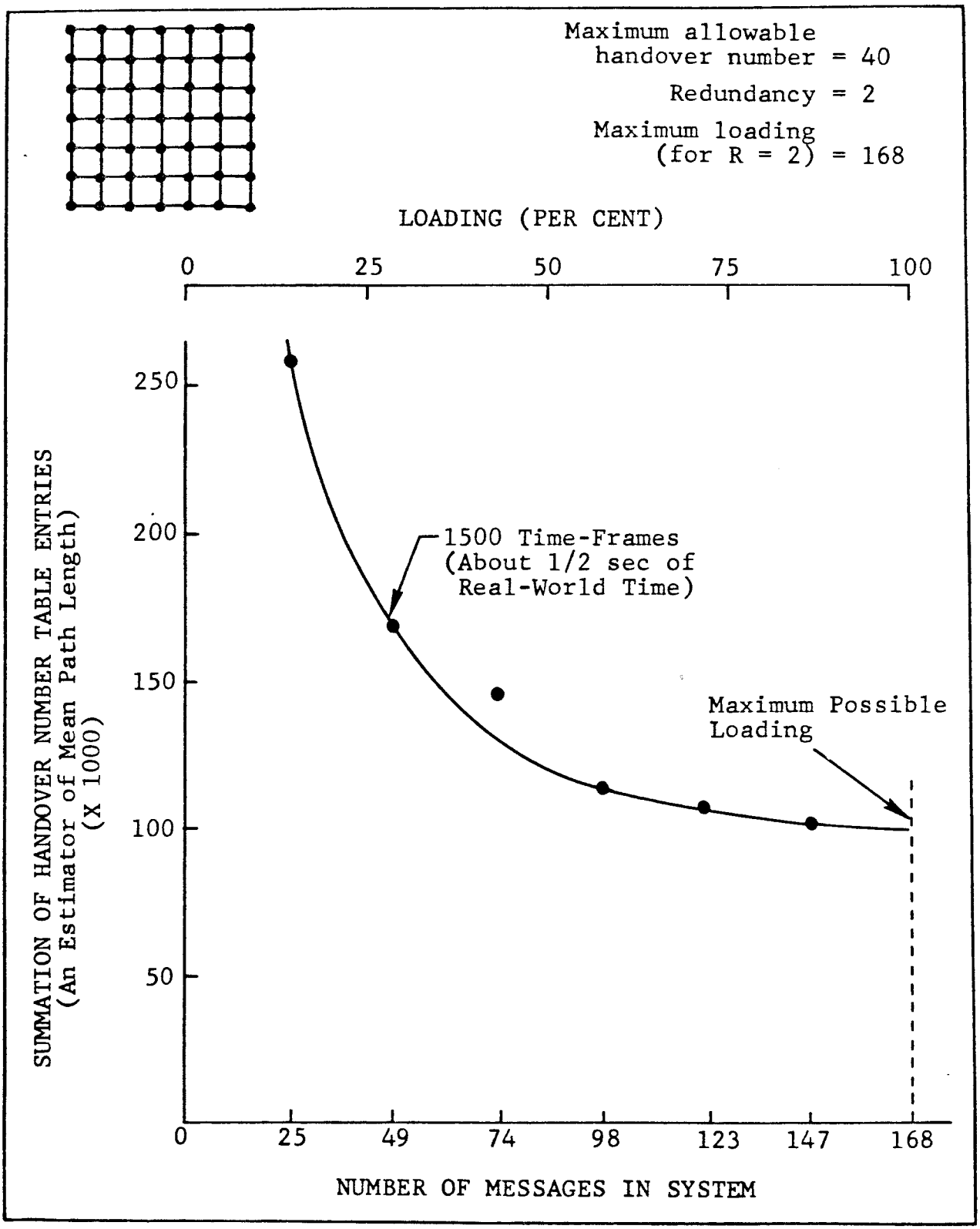


Fig. 1--Learning as a Function of Traffic Volume (From a Cold Start)

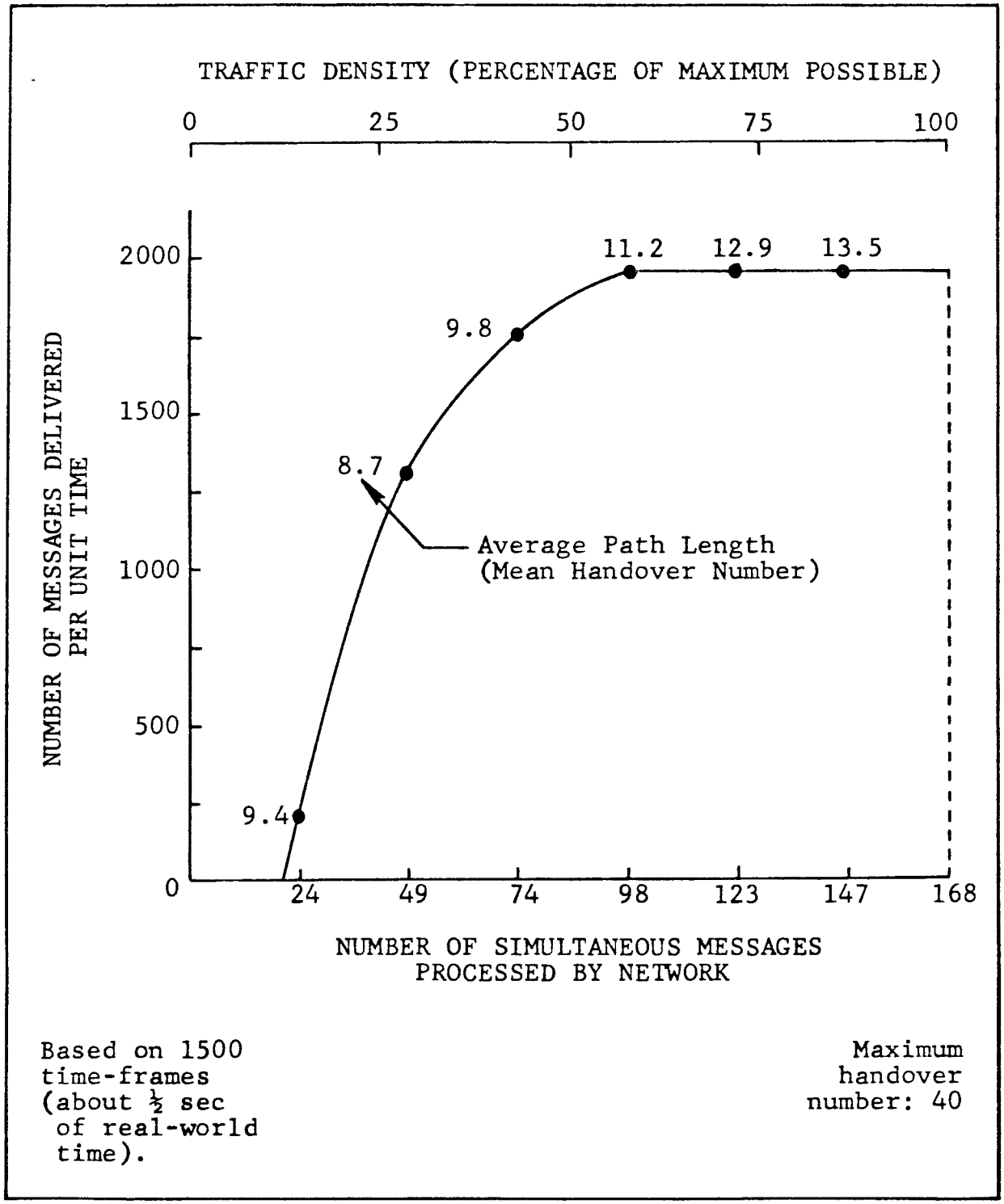


Fig. 2--Learning as a Function of Traffic Density

bound for system load-carrying capacity under the assumed loading conditions. The mean handover number increased as the number of messages processed per unit time increased. Figure 2 shows that the average path length did not increase sharply, even up to 50 per cent loading.

The definition of the term "capacity" as used here refers to a comparison of the number of messages in the system times 100 divided by the number of links.

REDUNDANCY

The rate of initial learning improved as the redundancy of the network was increased. Keeping other factors constant, twice as many messages could be delivered per unit time for a network of redundancy level 4, as for one with R = 1.5. The slopes of Fig. 3 indicate the overall learning rate.

LOCATION

From a preliminary examination of the learning power relative to the location of a node, it appears that the nodes on the border do the best (see Fig. 4). These nodes have fewer links to "educate." The most centralized nodes do next best, as they have the most traffic. The intermediate zone between these groups showed the poorest initial learning rate.

THE LEARNING RATE CONSTANT

The formula for learning or modifying the handover number table was chosen to be $a_{t+1} = a_t + K(b_t - a_t)$, where

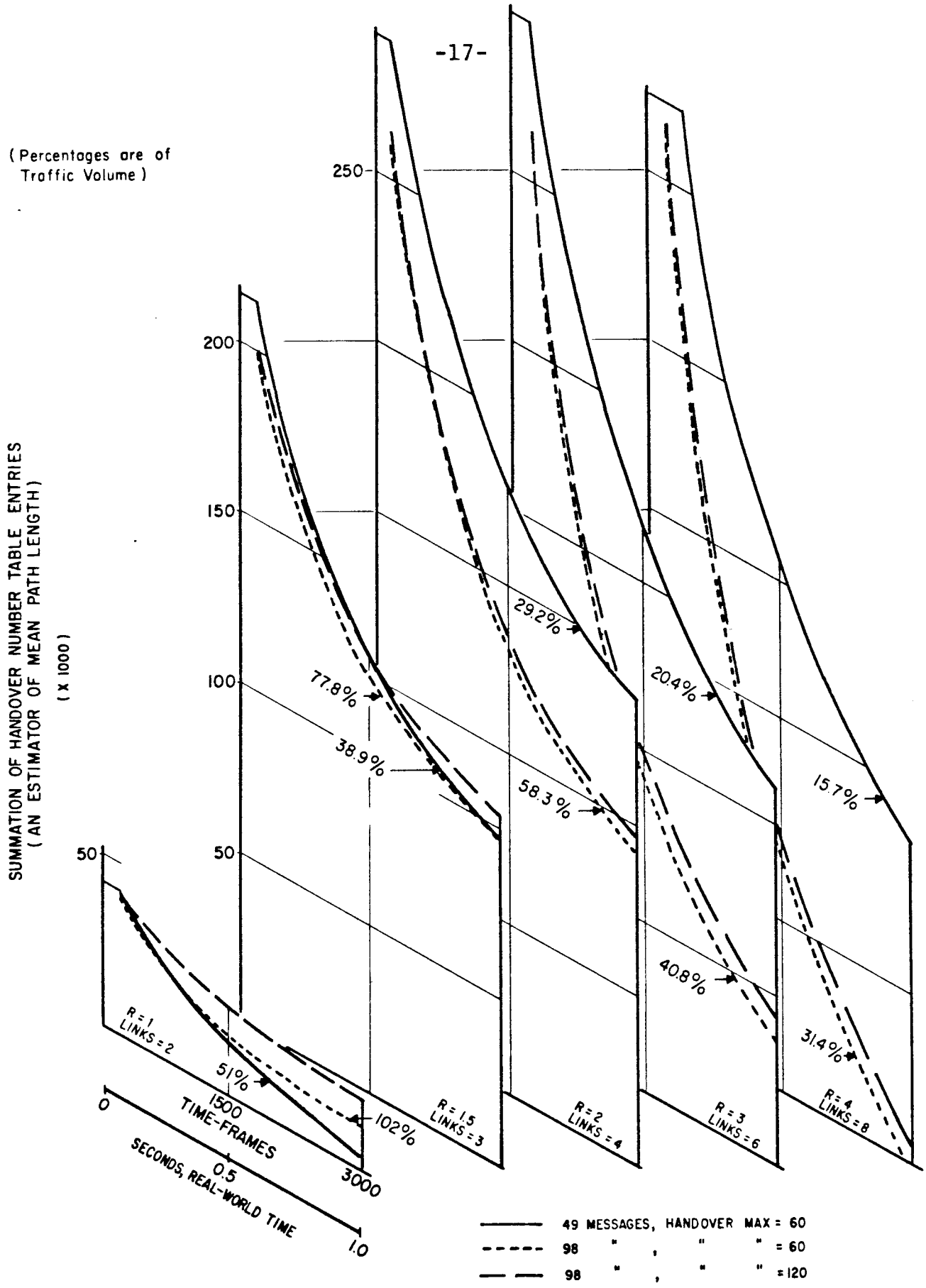


FIG. 3--LEARNING AS A FUNCTION OF TRAFFIC VOLUME, REDUNDANCY, AND MAXIMUM ALLOWABLE HANDOVER NUMBER

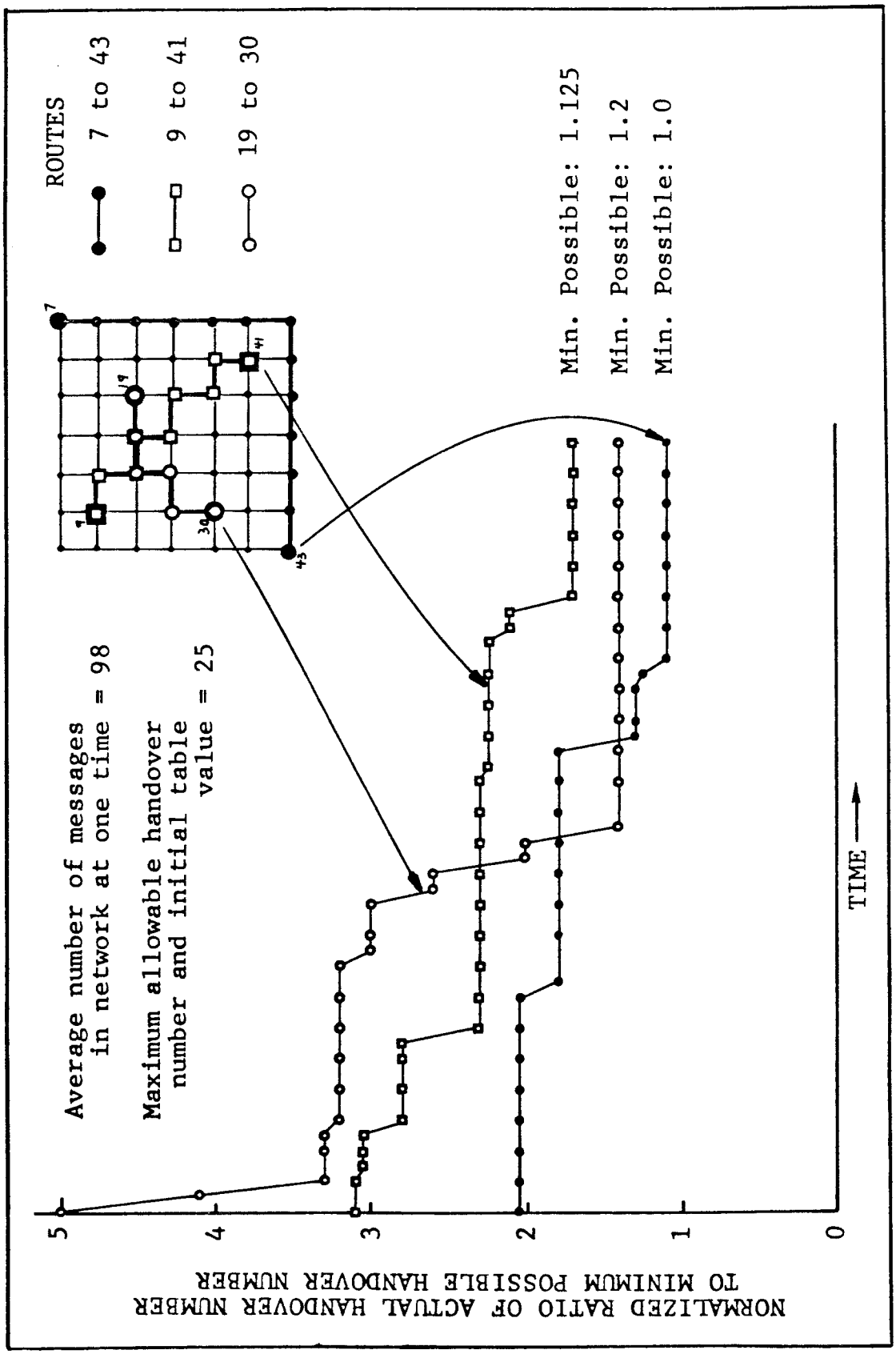


Fig. 4--Learning as a Function of Node Location

a is the handover value in the table, b is the handover number of the newly delivered message, and K is a constant. The choice of learning function was partially based upon the requirement that in the real-world it should be capable of being implemented by simple circuitry. Figure 5 shows the effect of varying the learning constant, K = 0.6, 0.8, and 1.0. The initial value of table total was the same in the three cases.

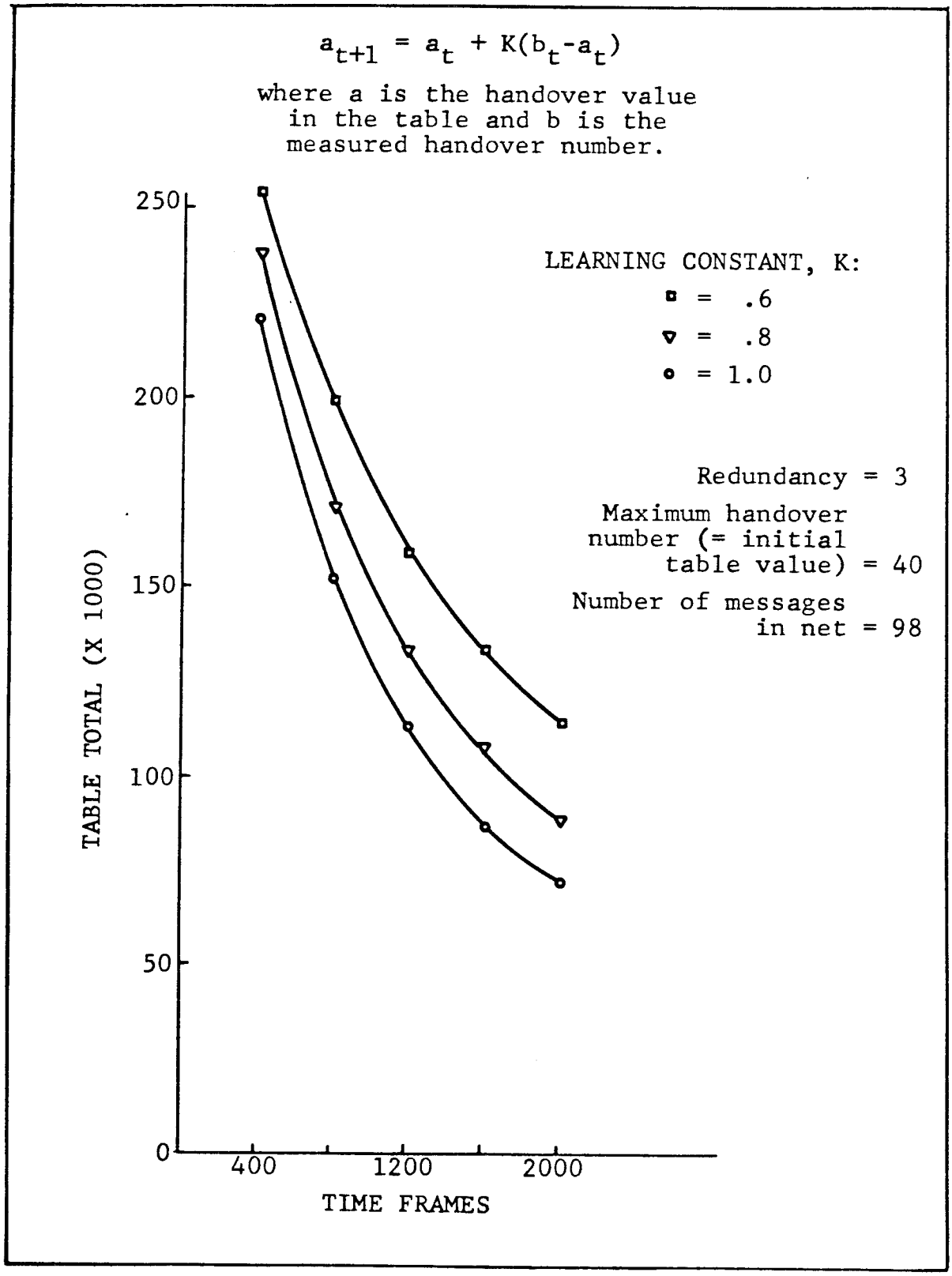


Fig. 5--Learning as a Function of Learning Constant, K

VI. DETERMINATION OF TRAFFIC HANDLING CAPACITY

As expected, more traffic can be handled with an increase in the number of links per node. If the average number of messages delivered for two links were to be set with a base of unity, the following table could be formed:

Redundancy	Links/ Node	No. of Messages Delivered Per Fixed Time Interval
1	2	1
1.5	3	4.2
2	4	15
3	6	21.9
4	8	28.9

These numbers would vary a bit if more messages were introduced.

However, the number of messages delivered per unit time does not appear to increase significantly as the maximum allowable handover number increases. Therefore, one can place a bound on the maximum allowable handover number. If this value is set too high, it could permit consecutive data blocks to arrive out of sequence at high user-input data rates.*

---

*Methods to prevent such an occurrence are described in ODC-VII, -VIII.

VII. DETERMINATION OF PATH LENGTH

REDUNDANCY

The measured mean handover numbers appeared to settle at about the following equilibrium values:

Redundancy	No. of Messages in System		
	49	98	196
1	19	24	-
1.5	10	21	-
2	6	9.5	-
3	5	5	10
4	4	4	-

TRAFFIC

For a test with R = 2 and the handover maximum at 40, the mean handover number increased as the traffic increased.

No. of Messages	Mean Handover Number
49	8.7
98	11.2
147	13.5

Appendix A

SAMPLE OUTPUT

LOADING MAP

A sample computer output containing the map used to indicate the location of all the subroutines of the simulation program is shown in Fig. 6. The values shown are the addresses of the subroutines in the core memory and are used for checkout purposes only.

INITIAL INPUT PARAMETERS

Under the printed heading "EXECUTION", twelve separate parameters that can be selected as input quantities and inserted into the program as a control card are shown. These separate quantities, shown in Fig. 6, are:

1) An arbitrary number used to start the random number at a different selectable starting point for each run (5351 in the example shown).
2) The number of columns in the rectangular array of nodes (7).
3) The number of rows in the rectangular array of nodes (7).
4) The number of links connected to each node (8). The value of eight links corresponds to a redundancy level of R = 4.
5) The number of time-frames between computer printout snapshots (200).
6) If the program is used to observe the learning occurring between two selected nodes, this

```
•     XEJ
LOADING MAP.  (000000=MAIN PROGRAM.  -ADDRESS INDICATES UNUSED ENTRY POINT.)
     SELINK 00152      SELNOD 03507      NEW    00701      000000 01016      MOVE   01224      RANTIM 01653      SETTAB 02220
     ROUTE  03305      PRINT  03526      (FPT)  03662      (TSHM) 03760      (TSH) -03755      (STHM) 04020      (STH) -04015
     (WTC)  04144      (WER)  04113      (ROC)  04223      (RER)  04201      (TCO)  04405      (TEF)  04404      (RCH)  04403
     (ETT)  04402      (REW)  04401      (WEF)  04400      (BSR)  04377      (WRS)  04376      (RDS)  04375      (IOS)  04243
     (TRC)  04406      (IOU)  04425      (TES)  04500      PDUMP -04504      DUMP   04503      RAN    -04756      RANDOM 04760
     RSTART 04747      (RTN)  06471      (FIL)  06460      (IOH)  05023      (WOT)  06622      (TALL)-06667      (ACTMR-06693
     (EXEM) 06674      CLKOUT-07512      ERRDMP-07512      ENDJOB-07512      RETURN-07512      EXIT   07512
        EXECUTION                                                  TIME 203700
     5351          7          7          8        200          1          0          0         40         98
     3000         60
        129417 TABLE TOTAL
```

Fig. 6--Computer Subroutine Location Map and Input Parameters

parameter would indicate the "from" node (not examined in the attached printouts).

7) Same as the previous input, but for the "to" node.
8) Minimum path between the inputs of 6) and 7) (not used in the following printouts).
9) Initial table value.
10) Number of messages flowing around in the system at any time (98).
11) Number of time-frames the system is to be run (3000).
12) Maximum allowable handover number (60).

OUTPUT

The next form of output, Fig. 7, presents a snapshot summary of traffic status in the network each 200 (in this case) time-frames.

1) Number of messages delivered during the last 200 time-frames (271, 503, 592, etc.).
2) Mean value of all handover number table entries during the last 200 time-frames (10.324723, 7.109344, 5.500000, ...).
3) Percentage of messages dropped because handover numbers exceeded maximum allowable values (0, 0.001984, 0.01033, 0.004458, ...).
4) Total value of the sum (modulo 131,172) of all entries on the handover number table (69422, 26960, 126426, 101382).

It is to be noted that this table summation is fixed point and modulo 131,172. Thus, the first few entries of

```
                                                PERCENT OF MESSAGES DROPPED      0.
NUMBER OF MESSAGES DEL   271 MEAN HANDOVER NUMBER      10.324723
        69422 TABLE TOTAL
                                                PERCENT OF MESSAGES DROPPED      0.001984
NUMBER OF MESSAGES DEL   503 MEAN HANDOVER NUMBER       7.109344
        26960 TABLE TOTAL
                                                PERCENT OF MESSAGES DROPPED      0.010033
NUMBER OF MESSAGES DEL   592 MEAN HANDOVER NUMBER       5.500000
       126426 TABLE TOTAL
                                                PERCENT OF MESSAGES DROPPED      0.004458
NUMBER OF MESSAGES DEL   670 MEAN HANDOVER NUMBER       4.743284
       101382 TABLE TOTAL
                                                PERCENT OF MESSAGES DROPPED      0.011111
NUMBER OF MESSAGES DEL   623 MEAN HANDOVER NUMBER       4.632424
        80662 TABLE TOTAL
                                                PERCENT OF MESSAGES DROPPED      0.013294
NUMBER OF MESSAGES DEL   668 MEAN HANDOVER NUMBER       4.652695
        63549 TABLE TOTAL
                                                PERCENT OF MESSAGES DROPPED      0.007246
NUMBER OF MESSAGES DEL   685 MEAN HANDOVER NUMBER       4.049635
        50456 TABLE TOTAL
                                                PERCENT OF MESSAGES DROPPED      0.022727
NUMBER OF MESSAGES DEL   731 MEAN HANDOVER NUMBER       3.987688
        37483 TABLE TOTAL
                                                PERCENT OF MESSAGES DROPPED      0.005674
NUMBER OF MESSAGES DEL   701 MEAN HANDOVER NUMBER       4.186876
        27134 TABLE TOTAL
                                                PERCENT OF MESSAGES DROPPED      0.015428
NUMBER OF MESSAGES DEL   702 MEAN HANDOVER NUMBER       4.116809
        17743 TABLE TOTAL
                                                PERCENT OF MESSAGES DROPPED      0.010652
NUMBER OF MESSAGES DEL   743 MEAN HANDOVER NUMBER       3.969044
         8352 TABLE TOTAL
                                                PERCENT OF MESSAGES DROPPED      0.011412
NUMBER OF MESSAGES DEL   693 MEAN HANDOVER NUMBER       4.190476
          447 TABLE TOTAL
                                                PERCENT OF MESSAGES DROPPED      0.005420
NUMBER OF MESSAGES DEL   734 MEAN HANDOVER NUMBER       4.192098
       124178 TABLE TOTAL
                                                PERCENT OF MESSAGES DROPPED      0.009763
NUMBER OF MESSAGES DEL   710 MEAN HANDOVER NUMBER       4.240845
       117383 TABLE TOTAL
                                                PERCENT OF MESSAGES DROPPED      0.014570
NUMBER OF MESSAGES DEL   744 MEAN HANDOVER NUMBER       4.063172
```

Fig. 7--Snapshots of Traffic Summary

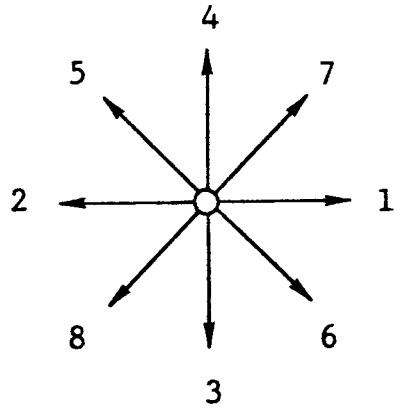
the table totals are in excess of 131,172. However, after only several cycles the values drop below 131,172 and remain below this point for the period of interest.

HANDOVER NUMBER TABLES

The following two illustrations define the numbering procedure for node and link used in the simulation:

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	32	33	34	35
36	37	38	39	40	41	42
43	44	45	46	47	48	49

NODE NUMBER DEFINITION



LINK NUMBER DEFINITION

In the distributed network being simulated, a separate handover number table is stored at each node. Each node's table is an ordered list of desired nodes sought. For example, Node 1 would have its table ordered in terms of to Node 11, to Node 25, to Node 49, etc.

However, in the simulation it was more convenient to combine all the tables and to order it in terms of the "from" stations. Hence, Fig. 8 shows the handover number table for Node 1, which is in the upper left-hand corner of the network. Thus, it can be seen that traffic from all stations to the east of Node 1 arrived with relatively low handover numbers over Link 2 of Node 2.

TO NODE 1								
FROM NODE	LINK 1	LINK 2	LINK 3	LINK 4	LINK 5	LINK 6	LINK 7	LINK 8
1	3	32000	3	32000	32000	8	32000	32000
2	6	1	40	32000	32000	4	32000	2
3	38	2	3	32000	32000	40	32000	2
4	15	3	4	32000	32000	40	32000	3
5	40	4	7	32000	32000	19	32000	4
6	23	8	40	32000	32000	17	32000	5
7	32000	6	7	32000	32000	32000	32000	6
8	2	32000	4	1	32000	4	2	32000
9	3	2	36	2	1	10	39	3
10	40	2	3	3	2	40	40	4
11	10	3	4	4	3	19	40	3
12	40	4	5	13	4	21	40	4
13	40	9	21	15	5	40	9	5
14	32000	6	40	8	6	32000	32000	6
15	5	32000	40	2	32000	40	2	32000
16	4	3	14	2	2	40	3	4
17	40	3	4	3	2	22	8	5
18	40	3	4	4	3	7	7	4
19	18	4	12	6	4	27	40	4
20	40	5	40	11	5	20	9	20
21	32000	6	40	7	40	32000	32000	40
22	7	32000	17	3	32000	40	3	32000
23	4	40	8	3	3	40	3	13
24	4	6	40	3	3	40	40	6
25	6	4	40	21	3	40	5	40
26	40	4	7	5	4	19	11	5
27	40	7	40	6	5	40	40	5
28	32000	8	40	13	6	32000	32000	6
29	5	32000	12	4	32000	40	4	32000
30	5	40	40	4	4	40	4	40
31	30	5	40	4	4	27	40	40
32	9	6	7	40	4	40	5	40
33	8	7	40	6	4	40	40	6
34	40	6	18	10	5	40	14	6
35	32000	7	40	40	6	32000	32000	10
36	7	32000	11	5	32000	40	5	32000
37	40	40	39	5	5	40	40	40
38	7	40	40	25	5	40	40	40
39	40	7	40	5	5	40	6	10
40	40	6	40	5	10	40	7	40
41	40	40	40	40	5	40	25	17
42	32000	6	40	40	9	32000	32000	40
43	10	32000	32000	6	32000	32000	6	32000
44	39	7	32000	6	6	32000	6	32000
45	28	8	32000	32	7	32000	7	32000
46	36	40	32000	8	6	32000	27	32000
47	11	16	32000	35	7	32000	40	32000
48	40	8	32000	10	40	32000	16	32000
49	32000	27	32000	40	40	32000	32000	32000
TO NODE 2								
FROM NODE	LINK 1	LINK 2	LINK 3	LINK 4	LINK 5	LINK 6	LINK 7	LINK 8

Fig. 8--Handover Number Table

Appendix B

PROGRAM LISTING

```
*    CARDSCOLUMN
*    LIST
*    LABEL
CMAIN5
      DIMENSION KTS(49),KTP(49,8),ITABLE(49,49,8),KFROM(400),KTO(400),KH
     1AND(400),KAT(400),KK(400),ISTATE(400),KSTATE(49,8),KKF(400)
      COMMON K1,K2,K3,K4,TSMIN,TSMAX,TPMIN,TPMAX,TF,KTS,KTP,IJ,ITABLE,IF
     1IN,MM,KFROM,KTO,KHAND,KAT,KK,ISTATE,KSTATE,KAM,KKM,KTM,KMAX,KTT,K
     2HMAX,KFM,KKF,K5,K6,K7,K8
    1 INPUT 41,100,KMAX,MM,KTT,KHMAX
      INPUT 41,101,TPMIN,TPMAX,TSMIN,TSMAX,TF
C     RESET RNDM NO GEN//I-DIM//J-DIM//REDUN//TO//FROM//MIN//HANDOVER
      INPUT 41,102,K1,K2,K3,K4,K5,K6,K7,K8
      OUTPUT 42,103,K1,K2,K3,K4,K5,K6,K7,K8,KMAX,MM,KTT,KHMAX
      IFIN=32000
      CALL SETTAB
      CALL RANTIM
      DO 5  M=1,MM
    5 CALL NEW(M)
      CALL MOVE
      DO 6 I=1,400
    6 KKF(I)=0
      DO 7 I=1,49
      DO 7 J=1,8
    7 KSTATE(I,J)=0
      GO TO 1
  100 FORMAT(4110)
  101 FORMAT(5E14.6)
  102 FORMAT(816)
  103 FORMAT(10110)
      END
```

CARD COUNT 31

FINAL CARD COUNT 31

```
*      CARDSCOLUMN
*      LIST
*      LABEL
       SUBROUTINE SETTAB
       DIMENSION KTS(49),KTP(49,8),ITABLE(49,49,8),KFROM(400),KTO(400),KH
      1AND(400),KAT(400),KK(400),ISTATE(400),KSTATE(49,8),KKF(400)
       COMMON K1,K2,K3,K4,TSMIN,TSMAX,TPMIN,TPMAX,TF,KTS,KTP,IJ,ITAPLE,IF
      1IN,MM,KFROM,KTO,KHAND,KAT,KK,ISTATE,KSTATE,KAM,KKM,KTM,KMAX,KTT,K
      2HMAX,KFM,KKF,K5,K6,K7,K8
       IJ=K2*K3
       IF(K4-3)2,6,2
     2 DO   3 I=1,IJ
       DO   3 J=1,IJ
       DO   3 K=1,K4
     3 ITABLE(I,J,K)=KMAX
       IF(K4-2)20,4,6
     4 DO   5 I=1,IJ
       ITABLE(I,1,2)=IFIN
     5 ITABLE(I,IJ,1)=IFIN
       GO TO 21
     6 IF(K4-3)20,22,23
    22 DO 13 I=1,IJ
       DO 10 J=1,IJ
       DO   7 K=1,2
     7 ITABLE(I,J,K)=KMAX
       KEVEN=XMODF(J,2)
       IF(KEVEN) 9,8,9
     8 ITABLE(I,J,3)=KMAX
       ITABLE(I,J,4)=IFIN
       GO TO 10
     9 ITABLE(I,J,3)=IFIN
       ITABLE(I,J,4)=KMAX
    10 CONTINUE
       DO 11 J=1,K3
       ITABLE(I,J,4)=IFIN
       KON=IJ-J+1
    11 ITABLE(I,KON,3)=IFIN
```

```
      DO 13 KJ=1,K2
      KON=KJ*K3
      ITABLE(I,KON,1)=IFIN
      KON=K3*(KJ-1)+1
      ITABLE(I,KON,2)=IFIN
   13 CONTINUE
      K4=4
      GO TO 21
   23 IF(K4-4)20,24,25
   24 DO 15 I=1,IJ
      DO 14 J=1,K3
      ITABLE(I,J,4)=IFIN
      KON=IJ-J+1
      ITABLE(I,KON,3)=IFIN
   14 CONTINUE
      DO 15 KJ=1,K2
      KON=KJ*K3
      ITABLE(I,KON,1)=IFIN
      KON=K3*(KJ-1)+1
      ITABLE(I,KON,2)=IFIN
   15 CONTINUE
      GO TO 21
   25 IF(K4-6)20,26,27
   26 DO 17 I=1,IJ
      DO 16 J=1,K3
      ITABLE(I,J,4)=IFIN
      ITABLE(I,J,5)=IFIN
      KON=IJ-J+1
      ITABLE(I,KON,3)=IFIN
      ITABLE(I,KON,6)=IFIN
   16 CONTINUE
      DO 17 KJ=1,K2
      KON=KJ*K3
      ITABLE(I,KON,1)=IFIN
      ITABLE(I,KON,6)=IFIN
      KON=K3*(KJ-1)+1
```

```
      ITABLE(I,KON,2)=IFIN
      ITABLE(I,KON,5)=IFIN
   17 CONTINUE
      GO TO 21
   27 IF(K4-8)20,28,20
   28 DO 19 I=1,IJ
      DO 18 J=1,K3
      ITABLE(I,J,4)=IFIN
      ITABLE(I,J,5)=IFIN
      ITABLE(I,J,7)=IFIN
      KON=IJ-J+1
      ITABLE(I,KON,3)=IFIN
      ITABLE(I,KON,6)=IFIN
      ITABLE(I,KON,8)=IFIN
   18 CONTINUE
      DO 19 KJ=1,K2
      KON=KJ*K3
      ITABLE(I,KON,1)=IFIN
      ITABLE(I,KON,6)=IFIN
      ITABLE(I,KON,7)=IFIN
      KON=K3*(KJ-1)+1
      ITABLE(I,KON,2)=IFIN
      ITABLE(I,KON,5)=IFIN
      ITABLE(I,KON,8)=IFIN
   19 CONTINUE
      GO TO 21
   20 CALL DUMP(K1,K8,2)
   21 RETURN
      END
```

CARD COUNT 102

FINAL CARD COUNT 102

```
*      CARDSCOLUMN
*      LIST
*      LABEL
       SUBROUTINE ERROR
       DIMENSION KTS(49),KTP(49,8),ITABLE(49,49,8),KFROM(400),KTO(400),KH
      1AND(400),KAT(400),KK(400),ISTATE(400),KSTATE(49,8),KKF(400)
       COMMON K1,K2,K3,K4,TSMIN,TSMAX,TPMIN,TPMAX,TF,KTS,KTP,IJ,ITABLE,IF
      1IN,MM,KFROM,KTO,KHAND,KAT,KK,ISTATE,KSTATE,KAM,KKM,KTM,KMAX,KTT,K
      2HMAX,KFM,KKF,K5,K6,K7,K8
       DO 10 J=1,IJ
       OUTPUT 42,80,J
       DO 10 K=1,K4
       OUTPUT 42,90,K
       OUTPUT 42,100,(ITABLE(J,I,K),I=1,IJ)
   10 CONTINUE
       RETURN
   80 FORMAT(10H TO NODE   I3)
   90 FORMAT(10H LINK NO   I3)
  100 FORMAT(7I10)
       END
```

CARD COUNT 20

FINAL CARD COUNT 20

```
*      CARDSCOLUMN
*      LIST
*      LABEL
       SUBROUTINE ROUTE(L,M)
       DIMENSION KTS(49),KTP(49,8),ITABLE(49,49,8),KFROM(400),KTO(400),KH
      1AND(400),KAT(400),KK(400),ISTATE(400),KSTATE(49,8),KKF(400)
       COMMON K1,K2,K3,K4,TSMIN,TSMAX,TPMIN,TPMAX,TF,KTS,KTP,IJ,ITABLE,IF
      1IN,MM,KFROM,KTO,KHAND,KAT,KK,ISTATE,KSTATE,KAM,KKM,KTM,KMAX,KTT,K
      2HMAX,KFM,KKF,K5,K6,K7,K8
       IF(L-1) 16,1,2
     1 KAT(M)=KAT(M)+1
       L=2
       GO TO 16
     2 IF(L-2) 16,3,4
     3 KAT(M)=KAT(M)-1
       L=1
       GO TO 16
     4 IF(L-3) 16,5,6
     5 KAT(M)=KAT(M)+K3
       L=4
       GO TO 16
     6 IF(L-4) 16,7,8
     7 KAT(M)=KAT(M)-K3
       L=3
       GO TO 16
     8 IF(L-5) 16,9,10
     9 KAT(M)=KAT(M)-(K3+1)
       L=6
       GO TO 16
    10 IF(L-6) 16,11,12
    11 KAT(M)=KAT(M)+(K3+1)
       L=5
```

```
      GO TO 16
   12 IF(L-7) 16,13,14
   13 KAT(M)=KAT(M)-(K3-1)
      L=8
      GO TO 16
   14 IF(L-8) 16,15,16
   15 KAT(M)=KAT(M)+(K3-1)
      L=7
   16 RETURN
      END
```

CARD COUNT 42

FINAL CARD COUNT 42

```
*      CARDSCOLUMN
*      LIST
*      LABEL
       SUBROUTINE PRINT
       DIMENSION KTS(49),KTP(49,8),ITABLE(49,49,8),KFROM(400),KTO(400),KH
      1AND(400),KAT(400),KK(400),ISTATE(400),KSTATE(49,8),KKF(400)
       COMMON K1,K2,K3,K4,TSMIN,TSMAX,TPMIN,TPMAX,TF,KTS,KTP,IJ,ITABLE,IF
      1IN,MM,KFROM,KTO,KHAND,KAT,KK,ISTATE,KSTATE,KAM,KKM,KTM,KMAX,KTT,K
      2HMAX,KFM,KKF,K5,K6,K7,K8
       DO 20 J=1,IJ
       OUTPUT 42,80,J
       OUTPUT 42,110
       DO 20 I=1,IJ
       OUTPUT 42,120,I,(ITABLE(J,I,K),K=1,8)
   20 CONTINUE
       RETURN
   80 FORMAT(10H TO NODE  I3)
  110 FORMAT(103H FROM NODE   LINK 1       LINK 2       LINK 3       LINK 4
     1       LINK 5       LINK 6       LINK 7       LINK 8)
  120 FORMAT(I6,8I12)
       END
```

CARD COUNT 21

FINAL CARD COUNT 21

```
*      CARDSCOLUMN
*      LIST
*      LABEL
CSELND8
       SUBROUTINE SELNOD(M)
       DIMENSION KTS(49),KTP(49,8),ITABLE(49,49,8),KFROM(400),KTO(400),KH
      1AND(400),KAT(400),KK(400),ISTATE(400),KSTATE(49,8),KKF(400)
       COMMON K1,K2,K3,K4,TSMIN,TSMAX,TPMIN,TPMAX,TF,KTS,KTP,IJ,ITABLE,IF
      1IN,MM,KFROM,KTO,KHAND,KAT,KK,ISTATE,KSTATE,KAM,KKM,KTM,KMAX,KTT,K
      2HMAX,KFM,KKF,K5,K6,K7,K8,K9,K10
       KTS(KAM)=XABSF(KTS(KAM))
       KKF(M)=KAT(M)
       CALL ROUTE(KKM,M)
       KHAND(M)=KHAND(M)+1
       KAM=KAT(M)
       KTEMP=XMINOF(KHAND(M),ITABLE(KFM,KAM,KKM))
       ITABLE(KFM,KAM,KKM)=KTEMP
       KK(M)=0
       ISTATE(M)=XABSF(KTS(KAM))
       IF(KAT(M)-KTO(M)) 4,5,4
    5  K9=K9+1
       K10=K10+KHAND(M)
       KK(M) =-IFIN
       GO TO 6
    4  IF(KHAND(M)-KHMAX) 6,3,3
    3  K8=K8+1
       KK(M)=-IFIN
    6  RETURN
       END
```

CARD COUNT 29

FINAL CARD COUNT 29

```
*      CARDSCOLUMN
*      LIST
*      LABEL
CSELNK6
       SUBROUTINE SELINK(M)
       DIMENSION KTS(49),KTP(49,8),ITABLE(49,49,8),KFROM(400),KTO(400),KH
      1AND(400),KAT(400),KK(400),ISTATE(400),KSTATE(49,8),KKF(400)
       COMMON K1,K2,K3,K4,TSMIN,TSMAX,TPMIN,TPMAX,TF,KTS,KTP,IJ,ITABLE,IF
      1IN,MM,KFROM,KTO,KHAND,KAT,KK,ISTATE,KSTATE,KAM,KKM,KTM,KMAX,KTT,K
      2HMAX,KFM,KKF,K5,K6,K7,K8,K9,K10
       KTMP=0
       KMIN=IFIN
       DO 2  K=1,K4
       IF(KSTATE(KAM,K)) 2,1,2
     1 KTEMP=K
       CALL ROUTE(KTEMP,M)
       KTEM=KAT(M)
       KAT(M)=KAM
       IF(KTEM-KKF(M)) 9,5,9
     5 KTMP=K
       GO TO 2
     9 KMIN=XMINOF(KMIN,ITABLE(KTM,KAM,K))
     2 CONTINUE
       IF(KMIN-IFIN) 3,7,7
     3 CALL RANDOM(R)
       K=XINTF(10.0*R)
       IF(K) 3,3,11
    11 IF(K-K4) 10,10,3
    10 IF(KMIN-ITABLE(KTM,KAM,K)) 3,4,3
     4 IF(KSTATE(KAM,K)) 3,12,3
    12 IF(K-KTMP) 13,3,13
```

```
13 KK(M)=K
   KKM=K
 6 KSTATE(KAM,KKM)=KTP(KAM,KKM)
   GO TO 8
 7 KK(M)=0
   ISTATE(M)=1
   KTS(KAM)=-KTS(KAM)
 8 RETURN
   END
```

CARD COUNT 40

FINAL CARD COUNT 40

```
*      CARDSCOLUMN
*      LIST
*      LABEL
CNEW3
       SUBROUTINE NEW(M)
       DIMENSION KTS(49),KTP(49,8),ITABLE(49,49,8),KFROM(400),KTO(400),KH
      1AND(400),KAT(400),KK(400),ISTATE(400),KSTATE(49,8),KKF(400)
       COMMON K1,K2,K3,K4,TSMIN,TSMAX,TPMIN,TPMAX,TF,KTS,KTP,IJ,ITABLE,IF
      1IN,MM,KFROM,KTO,KHAND,KAT,KK,ISTATE,KSTATE,KAM,KKM,KTM,KMAX,KTT,K
      2HMAX,KFM,KKF,K5,K6,K7,K8
       KFROM(M)=XMODF(M,IJ)
     3 CALL RANDOM(R)
       K=XINTF(100.0*R)
       IF(K) 3,3,4
     4 IF(K-IJ) 6,6,3
     6 IF(KFROM(M)-K) 5,3,5
     5 KTO(M)=K
       KHAND(M)=0
       KAT(M)=KFROM(M)
       KAM=KAT(M)
       ISTATE(M)=KTS(KAM)
       KK(M)=0
       RETURN
       END
```

CARD COUNT 24

FINAL CARD COUNT 24

```
*      CARDSCOLUMN
*      LIST
*      LABEL
CNEW3
       SUBROUTINE NEW(M)
       DIMENSION KTS(49),KTP(49,8),ITABLE(49,49,8),KFROM(400),KTO(400),KH
      1AND(400),KAT(400),KK(400),ISTATE(400),KSTATE(49,8),KKF(400)
       COMMON K1,K2,K3,K4,TSMIN,TSMAX,TPMIN,TPMAX,TF,KTS,KTP,IJ,ITABLE,IF
      1IN,MM,KFROM,KTO,KHAND,KAT,KK,ISTATE,KSTATE,KAM,KKM,KTM,KMAX,KTT,K
      2HMAX,KFM,KKF,K5,K6,K7,K8
       KFROM(M)=XMODF(M,IJ)
       IF(KFROM(M)) 3,7,3
     7 KFROM(M)=IJ
     3 CALL RANDOM(R)
       K=XINTF(100.0*R)
       IF(K) 3,3,4
     4 IF(K-IJ) 6,6,3
     6 IF(KFROM(M)-K) 5,3,5
     5 KTO(M)=K
       KHAND(M)=0
       KAT(M)=KFROM(M)
       KAM=KAT(M)
       ISTATE(M)=KTS(KAM)
       KK(M)=0
       RETURN
       END
```

FINAL CARD COUNT 26

```
*       CARDSCOLUMN
*       LIST
*       LABEL
CRANTM2
        SUBROUTINE RANTIM
        DIMENSION KTS(49),KTP(49,8),ITABLE(49,49,8),KFROM(400),KTO(400),KH
       1AND(400),KAT(400),KK(400),ISTATE(400),KSTATE(49,8),KKF(400)
        COMMON K1,K2,K3,K4,TSMIN,TSMAX,TPMIN,TPMAX,TF,KTS,KTP,IJ,ITABLE,IF
       1IN,MM,KFROM,KTO,KHAND,KAT,KK,ISTATE,KSTATE,KAM,KKM,KTM,KMAX,KTT,K
       2HMAX,KFM,KKF,K5,K6,K7,K8
        CALL RSTART(K1)
        IJ=K2*K3
        DO 9  I=1,IJ
     5  KTS(I)=1
        DO 9 K=2,K4,2
     6  CALL RANDOM(R)
        IF(10.*R-TPMIN) 6,7,7
     7  IF(10.*R-TPMAX) 8,8,6
     8  KTEMP=XINTF(10.*R)
        KTP(I,K)=KTEMP
        IF(K-2) 17,10,11
    10  J=I-1
        L=1
        GO TO 3
    11  IF(K-4) 17,12,13
    12  J=I-K3
        L=3
        GO TO 3
    13  IF(K-6) 17,14,15
    14  J=I+K3+1
```

```
      L=5
      GO TO 3
   15 IF(K-8) 17,16,17
   16 J=I+K3-1
      L=7
    3 IF(J) 9,9,4
    4 KTP(J,L)=KTEMP
    9 CONTINUE
      GO TO 18
   17 CALL DUMP(KTP,IJ,2)
   18 RETURN
      END
```

CARD COUNT 42

FINAL CARD COUNT 42

```
*      CARDSCOLUMN
*      LIST
*      LABEL
CMOVE5
       SUBROUTINE MOVE
       DIMENSION KTS(49),KTP(49,8),ITABLE(49,49,8),KFROM(400),KTO(400),KH
      1AND(400),KAT(400),KK(400),ISTATE(400),KSTATE(49,8),KKF(400)
       COMMON K1,K2,K3,K4,TSMIN,TSMAX,TPMIN,TPMAX,TF,KTS,KTP,IJ,ITABLE,IF
      1IN,MM,KFROM,KTO,KHAND,KAT,KK,ISTATE,KSTATE,KAM,KKM,KTM,KMAX,KTT,K
      2HMAX,KFM,KKF,K5,K6,K7,K8,K9,K10
       HAND=FLOATF(MM)
       DO 11 KT=1,KTT
       KTEMP=XMODF(KT,K5)
       IF(KTEMP) 20,21,20
   21  KTOT=0
       DO 30 I=1,IJ
       DO 30 J=1,IJ
       DO 30 K=1,K4
       IF(ITABLE(I,J,K)-20000) 25,25,30
   25  KTOT=KTOT+ITABLE(I,J,K)
   30  CONTINUE
       OUTPUT 42,180,KTOT
       HANDO=FLOATF(K8)
       TK9=FLOATF(K9)
       PCH=HANDO/(HANDO+TK9)
       K8=0
       OUTPUT 42,150,PCH
       TK10=FLOATF(K10)
       AVEHND=TK10/TK9
       OUTPUT 42,170,K9,AVEHND
       K9=0
       K10=0
   20  DO 11 M=1,MM
       KAM=KAT(M)
       KKM=KK(M)
```

```
      KTM=KTO(M)
      KFM=KFROM(M)
      IF(KK(M)) 12,7,9
    7 ISTATE(M)=ISTATE(M)-1
      IF(ISTATE(M)) 12,8,11
    8 CALL SELINK(M)
      GO TO 11
    9 KSTATE(KAM,KKM)=KSTATE(KAM,KKM)-1
      IF(KSTATE(KAM,KKM))18,10,11
   10 CALL SELNOD(M)
      GO TO 11
   12 CALL NEW(M)
   11 CONTINUE
   13 RETURN
   18 CALL DUMP(KSTATE,KAM,2)
  150 FORMAT(76H                                                    PERCENT O
     1F MESSAGES DROPPED  F12.6)
  170 FORMAT(25H NUMBER OF MESSAGES DEL  I4,23H MEAN HANDOVER NUMBER  F1
     12.6)
  180 FORMAT(2X,I13,12H TABLE TOTAL)
      END
```

CARD COUNT 56

FINAL CARD COUNT 56

ON DISTRIBUTED COMMUNICATIONS:

List of Publications in the Series

I. Introduction to Distributed Communications Networks, Paul Baran, RM-3420-PR.

Introduces the system concept and outlines the requirements for and design considerations of the distributed digital data communications network. Considers especially the use of redundancy as a means of withstanding heavy enemy attacks. A general understanding of the proposal may be obtained by reading this volume and Vol. XI.

II. Digital Simulation of Hot-Potato Routing in a Broadband Distributed Communications Network, Sharla P. Boehm and Paul Baran, RM-3103-PR.

Describes a computer simulation of the message routing scheme proposed. The basic routing doctrine permitted a network to suffer a large number of breaks, then reconstitute itself by rapidly relearning to make best use of the surviving links.

III. Determination of Path-Lengths in a Distributed Network, J. W. Smith, RM-3578-PR.

Continues model simulation reported in Vol. II. The program was rewritten in a more powerful computer language allowing examination of larger networks. Modification of the routing doctrine by intermittently reducing the input data rate of local traffic reduced to a low level the number of message blocks taking excessively long paths. The level was so low that a deterministic equation was required in lieu of Monte Carlo to examine the now rare event of a long message block path. The results of both the simulation and the equation agreed in the area of overlapping validity.

IV. Priority, Precedence, and Overload, Paul Baran. RM-3638-PR.

The creation of dynamic or flexible priority and precedence structures within a communication system handling a mixture of traffic with different data rate, urgency, and importance levels is discussed. The goal chosen is optimum utilization of the communications resource within a seriously degraded and overloaded network.

V. History, Alternative Approaches, and Comparisons, Paul Baran, RM-3097-PR.

A background paper acknowledging the efforts of people in many fields working toward the development of large communications systems where system reliability and survivability are mandatory. A consideration of terminology is designed to acquaint the reader with the diverse, sometimes conflicting, definitions used. The evolution of the distributed network is traced, and a number of earlier hardware proposals are outlined.

VI. Mini-Cost Microwave, Paul Baran, RM-3762-PR.

The technical feasibility of constructing an extremely low-cost, all-digital, X- or K_u-band microwave relay system, operating at a multi-megabit per second data rate, is examined. The use of newly developed varactor multipliers permits the design of a miniature, all-solid-state microwave repeater powered by a thermo-electric converter burning L-P fuel.

VII. Tentative Engineering Specifications and Preliminary Design for a High-Data-Rate Distributed Network Switching Node, Paul Baran, RM-3763-PR.

High-speed, or "hot-potato," store-and-forward message block relaying forms the heart of the proposed information transmission system. The Switching Nodes are the units in which the complex processing takes place. The node is described in sufficient engineering detail to estimate the components required. Timing calculations, together with a projected implementation

scheme, provide a strong roundation for the belief that the construction and use of the node is practical.

VIII. The Multiplexing Station, Paul Baran, RM-3764-PR.

A description of the Multiplexing Stations which connect subscribers to the Switching Nodes. The presentation is in engineering detail, demonstrating how the network will simultaneously process traffic from up to 1024 separate users sending a mixture of start-stop teletypewriter, digital voice, and other synchronous signals at various rates.

IX. Security, Secrecy, and Tamper-Free Considerations, Paul Baran, RM-3765-PR.

Considers the security aspects of a system of the type proposed, in which secrecy is of paramount importance. Describes the safeguards to be built into the network, and evaluates the premise that the existence of "spies" within the supposedly secure system must be anticipated. Security provisions are based on the belief that protection is best obtained by raising the "price" of espied information to a level which becomes excessive. The treatment of the subject is itself unclassified.

X. Cost Estimate, Paul Baran, RM-3766-PR.

A detailed cost estimate for the entire proposed system, based on an arbitrary network configuration of 400 Switching Nodes, servicing 100,000 simultaneous users via 200 Multiplexing Stations. Assuming a usable life of ten years, all costs, including operating costs, are estimated at about $60,000,000 per year.

XI. Summary Overview, Paul Baran, RM-3767-PR.

Summarizes the system proposal, highlighting the more important features. Considers the particular advantages of the distributed network, and comments on disadvantages. An outline is given of the manner in which future research aimed at an actual implementation of the network might be conducted. Together with the introductory volume, it provides a general description of the entire system concept.