State Criminal Justice Telecommunications (STACOM) Final Report

Volume IV: Network Design Software User's Guide

Law Enforcement Assistance Administration
U. S. Department of Justice

State Criminal Justice Telecommunications (STACOM) Final Report

Volume IV: Network Design Software User's Guide

Jun-Ji Lee

October 31, 1977

Law Enforcement Assistance Administration
U. S. Department of Justice

U. S. DEPARTMENT OF JUSTICE
Law Enforcement Assistance Administration

James M. H. Gregg, Acting Administrator

Harry Bratt, Assistant Administrator
National Criminal Justice Information
and Statistics Service

Wayne P. Holtzman, Director
Systems Development Division

This report was prepared by the Jet Propulsion Laboratory, California Institute of Technology for the Law Enforcement Assistance Administration, Department of Justice by agreement with the National Aeronautics and Space Administration. Opinions expressed are those of the author and do not necessarily reflect the official position or policies of the United States Department of Justice.

For sale by the Superintendent of Documents, U.S. Government Printing Office
Washington, D.C. 20402

Stock No. 027-000-00676-1

FOREWORD

The State Criminal Justice Telecommunications (STACOM) Project consists of two major study tasks. The first entails a study of criminal justice telecommunication system user requirements and system traffic requirements through the year 1985. The second investigates the least cost network alternatives to meet these specified traffic requirements.

Major documentation of the STACOM Project is organized in four volumes as follows:

State Criminal Justice Telecommunications (STACOM) Final Report - Volume I: Executive Summary	77-53 Vol. I
State Criminal Justice Telecommunications (STACOM) Final Report - Volume II: Requirements Analysis and Design of Ohio Criminal Justice Telecommunications Network	77-53 Vol. II
State Criminal Justice Telecommunications (STACOM) Final Report - Volume III: Requirements Analysis and Design of Texas Criminal Justice Telecommunications Network	77-53 Vol. III
State Criminal Justice Telecommunications (STACOM) Final Report - Volume IV: Network Design Software Users' Guide	77-53 Vol. IV

The above material is also organized in an additional four volumes which provide a slightly different reader orientation as follows:

Title	Document No.
State Criminal Justice Telecommunications (STACOM) Functional Requirements - State of Ohio	5030-43*
State Criminal Justice Telecommunications (STACOM) Functional Requirements - State of Texas	5030-61*
State Criminal Justice Telecommunications (STACOM) User Requirements Analysis	5030-80*
State Criminal Justice Telecommunications (STACOM) Network Design and Performance Analysis Techniques	5030-99*

*Jet Propulsion Laboratory internal document.

This document, No. 77-53, Volume IV, entitled, "Network Design Software Users' Guide," describes techniques that are implemented in the STACOM program. It then illustrates the application of this program by providing a run example with detailed input/output listing.

It presents the results of one phase of research carried out jointly by the Jet Propulsion Laboratory, California Institute of Technology, and the States of Texas and Ohio. The project is sponsored by the Law Enforcement Assistance Administration, Department of Justice, through the National Aeronautics and Space Administration (Contract NAS7-100).

ABSTRACT

A users' guide is provided in this volume for the network design software developed during the State Criminal Justice Telecommunications (STACOM) project sponsored by the Law Enforcement Assistance Administration (LEAA).

The network design program is written in FORTRAN V and implemented on a UNIVAC 1108 computer under the EXEC-8 operating system which enables the user to construct least-cost network topologies for criminal justice digital telecommunications networks. A complete description of program features, inputs, processing logic, and outputs is presented. Also included is a sample run and a program listing.

CONTENTS

Figures

Tables

SECTION 1

INTRODUCTION

1.1 PURPOSE AND SCOPE

The STACOM (STAte Criminal Justice COMmunication) network topology program is a software tool which has been developed and utilized during the STACOM project. This Software Users' Guide provides:

(1) A detailed description of the program, i.e., what it does and how it does it.

(2) Details of the STACOM storage structure and of its program structure so that a user can easily comprehend its capabilities and limitations.

(3) Details of the options available, a functional block diagram, and a program listing with comment statements so that a user can expand/improve the program capabilities by either changing parameter values or modifying the program itself.

(4) Details of a sample run stream used as a reference run for correct operation, and an input/output example, so that a user can easily operate the program as a tool for network design.

The STACOM program was developed and implemented with the FORTRAN-V programming language, which is one of several high-level languages available in the UNIVAC 1108 computer systems at the Jet Propulsion Laboratory. EXEC-8 is the operating system used in these systems. With this in mind, usage of this program in a similar UNIVAC system may require some degree of conversion effort. For a facility with computers other than the UNIVAC type, a considerable effort would be required in converting this program into one compatible with the operating system of that facility.

The balance of this document consists essentially of two parts. The first deals with the functional design portion of the STACOM topology program (Section 2); the other is concerned with the operational aspect (Section 3).

1.2 SUMMARY

1.2.1 The STACOM Program

The development of the STACOM (STAte Criminal Justice COMmunication) network topology program was performed to support the primary STACOM objective of providing the tools needed for designing and evaluating intrastate communication networks. The STACOM project goals are to:

(1) Develop and document techniques for intrastate traffic measurement, analysis of measured data, and prediction of traffic growth.

(2) Develop and document techniques for intrastate network design, performance analysis, modeling, and simulation.

(3) Illustrate applications of network design and analysis techniques to typical existing network configurations and new or improved configurations.

(4) Develop and illustrate a methodology for establishing priorities for cost-effective expenditures to improve capabilities in deficient areas.

A task involving the development of a software package for the synthesis and analysis of alternate network topologies was undertaken.

In the following subsections, we describe a typical law enforcement communication network, what the STACOM program does, how it does it, and a general operating procedure for using the program.

1.2.2 State Criminal Justice Communication Network and its Optimization

A State law enforcement communication network is defined as a network which contains a set of system terminations connected by a set of links. Each system termination consists of one or more physical terminals or computers located at the same city, called a terminal city. The main purpose of the communication network is to provide to the terminal users rapid access to and response from the data base system, and rapid response time for intra-agency communication.

Various ways of connecting a given set of terminals may be used, depending on different requirements. Because the operating costs for a given communication network depend very much on its layout, some cost reduction is possible through an initial investment in a configuration analysis.

The activity of designing a network with the lowest costs which satisfy loading requirements, called network optimization, uses various existing techniques which provide means for such purposes.

1.2.3 Functions Performed by the STACOM Program

The STACOM program is a software tool which has been developed to design optimal networks that will achieve lower operating costs. It utilizes a modified Esau-Williams technique to search for those direct links between system terminations and a regional switching center (RSC) which may be eliminated in order to reduce operating costs without impairing system performance. The RSC provides either a switching capability, a data base center, or both.

Inputs for the STACOM program contain data such as traffic, terminal locations, and functional requirements. The network may be divided into any number of desired regions in any given program run. Each region has a Regional Switching Center (RSC) which serves terminals in its region. RSCs are, finally, interconnected to form the complete network. Upon receipt of a complete set of input data, the STACOM program first performs the formation of regions and, if needed, the selection of RSCs. The program then builds a regional network in which only system terminations in the region are connected. The program subsequently optimizes the regional network for each region requested by the user.

The formation of regions is performed by the program on the basis of attempting to arrive at near-equal amounts of traffic for all regions. After finding the farthest unassigned system termination from the system centroid (a geographical center), the program starts formation of the first region by selecting unassigned system terminations close to this system termination until the total amount of traffic for that region is greater than a certain percentage (90% in this implementation) of the average regional traffic. The average regional traffic is simply the total network traffic divided by the number of desired regions. The same process is repeated by the program in forming the rest of the regions.

The selection of an RSC is based on the minimal traffic-distance product sum. In the selection process, each system termination is chosen as a trial RSC, and the sum of traffic-distance products is then calculated. The location of the system termination which provides the minimal sum is then selected as the RSC, although the location of the RSC for a given region may also be specified by the user. The optimization process consists of two basic steps, i.e., searching for lines whose elimination yields the best cost saving, and updating the network. The two steps are repeated until no further saving is possible.

Before performing network optimization, the STACOM program constructs an initial star network in which each system termination is directly connected to the regional center. It then starts the optimization process. At the termination of this process, a multidrop network is generally developed. In a multidrop network, some lines have more than one system termination; these are called multidrop lines.

When needed, the STACOM program will continue to form an optimized interregional network, which consists of inter-connections between regional centers.

The process for interregional network optimization involves the same two steps: searching and updating. However, the searching step is primarily to find the alternate route, for diverting traffic between two regional switching centers, that provides the best saving.

Based on the data provided, a successful run of the STACOM program generates a regular printer output and, if requested, a CalComp plot. The printer output contains data such as initial regional network and optimized network costs, assignments of system terminations, etc. The CalComp plot shows the geographical connections of the optimized network detailing multidrop line connections to all of the system terminations.

1.2.4 Operational Procedure

1.2.4.1 Initialization and Setup. When the STACOM program is executed from an 80-character/line demand terminal, an alternate file, 100, to be used as a printer output file, must be defined. Otherwise, all printout data will be directed to the terminal which will produce interleaving output. The file is defined by the statement @ASG,UP 100.

In addition to the redirection of output file destination, the user must direct the punch card file to a proper unit for a CalComp plotter. As an example, the statement @SYM,P PUNCH$,,G9PLTF will direct the punch card images to a CalComp plotter designated with G9PLTF.

1.2.4.2 Starting a Run.

1.2.4.2.1 Batch Mode. Following is a list of control statements required when running the STACOM program as a batch run:

```
@RUN run-ID, account-no., project-ID, SUP-time, pages/cards
@ASG,UP 100
@SYM,P PUNCH$,,plotter-ID
@XQT file.STACOM
      (INPUT DATA)
@BRKPT 100
@FREE 100
@SYM 100,,printer-ID
@FIN
```

The RUN card gives the following information: designated run ID, user's account number, project-ID, expected SUP-time usage (sum of CPU time, I/O time, and control/execute request time), limited number of printer pages, and number of cards which may be generated from the run. Plotter-ID gives the logical ID of the CalComp pen plotter and file is the file which contains the absolute element of the STACOM program. Printer-ID gives the logical ID of the line printer. INPUT DATA as shown is the input data required. When all of these data items are in order and ready, the deck can be submitted to the operator for processing.

1.2.4.2.2 Demand Mode. If program execution is to be performed via a demand terminal, the user can converse interactively with the program. The user may also run the program as a batch job by having all input data prepared and added after the @XQT statement.

Under the conversational mode, the user acts as a respondent who answers the requests for data made by the program. This mode of operation provides the user with an understanding of how the program is progressing. A user can very often terminate a run before a complete set of input data is given if he has some knowledge of the progress being made. This capability can prevent the user from an unnecessary waste of time. For example, if a run encounters a system which has more oversized distance data than allowed, a message from the program will be printed out

on the terminal. This will force the user to modify the program in order to handle the large number of oversized distance data.

1.2.4.3 Normal Termination. When a STACOM program run proceeds successfully and terminates normally, the normal file unit 6 will contain messages for each successful regional network optimization. After a normal termination, the user can direct the output file 100 to a printer device, and the CalComp plot will be generated by the designated CalComp pen plotter.

1.2.5 Aborting and Recovering a Run

When a run encounters trouble resulting from incorrect input data, the user can use the normal aborting procedure to terminate its execution if it is a demand job. A statement of @@X after interrupting the line communication by pressing the BREAK key, will terminate a program execution at any time. On the other hand, the EXEC-8 may abort a run when certain serious violations occur during its execution.

If a program run has been interrupted because of a system outage, no recovery of the run is possible.

SECTION 2

THE STACOM PROGRAM

2.1 INTRODUCTION

Two types of analysis are involved in designing a communication network. The first is concerned with arriving at acceptable line loadings; the second involves the achievement of optimal line configurations. The STACOM program was developed to accomplish both of these types of analysis.

Before describing the STACOM program itself, a State criminal justice information system with its communication network is examined as a typical existing communication network. The goal of the STACOM program is then discussed.

2.1.1 State Criminal Justice Information System

An information system is usually developed to provide a systematic exchange of information between a group of organizations. The information system is used to accept (as inputs), store (in files or a data base), and display (as outputs) strings of symbols that are grouped in various ways. While an information system may exist without a digital computer, we will consider only systems which contain digital computers as integral parts.

Information systems can be classified in various ways for various purposes. If classification is by the type of service rendered, the type of information system which serves a criminal justice community within a State can be considered as an information storage and retrieval system. This type of information system is the subject of our interest. For example, the State of Ohio has an information system with a data base located at Columbus. The data base contains records on wanted persons, stolen vehicles, and stolen license plates. Also included in the same computer are files of the Bureau of Motor Vehicles (BMV) which contain records on all licensed drivers and motor vehicles in that State.

2.1.2 State Digital Communication Network

For a given State information system, the storage and retrieval of data to/from the data base can be accomplished in various ways for different user requirements. In general, the users of a State criminal justice information system are geographically distant from the central data base computer. Because a fast turn-around time is a necessity for this particular user community, direct in-line access to the central data base by each criminal justice agency constitutes the most important of the user's requirements. In addition, it is required to move message data quickly from one agency to another at a different location. These goals require the establishment of a data communication network. Because the computer deals only with digital data, only digital data communication networks are considered here.

A digital communication network consists mainly of a set of nodes connected by a set of links. The nodes may be computers, terminals, or other types of communication control units that are placed in various locations, and the links are the communication channels providing data paths between the nodes. These channels are usually private or switched lines that are leased from a common carrier. A simple example of a network is given in Figure 2-1, where the links between modems are communication lines leased from a common carrier. The communication control unit in city E is used to multiplex or concentrate several low-speed terminals onto a high-speed line. The line which connects cities C, D, and others is called a multidrop line, and this line connects several terminals to the data base computer.

2.1.3 A STACOM Communication Network

For the purposes of the STACOM study, a communication network was defined as a set of system terminations connected by a set of links. Each system termination consists of one or more physical terminals or computers located at the same city.

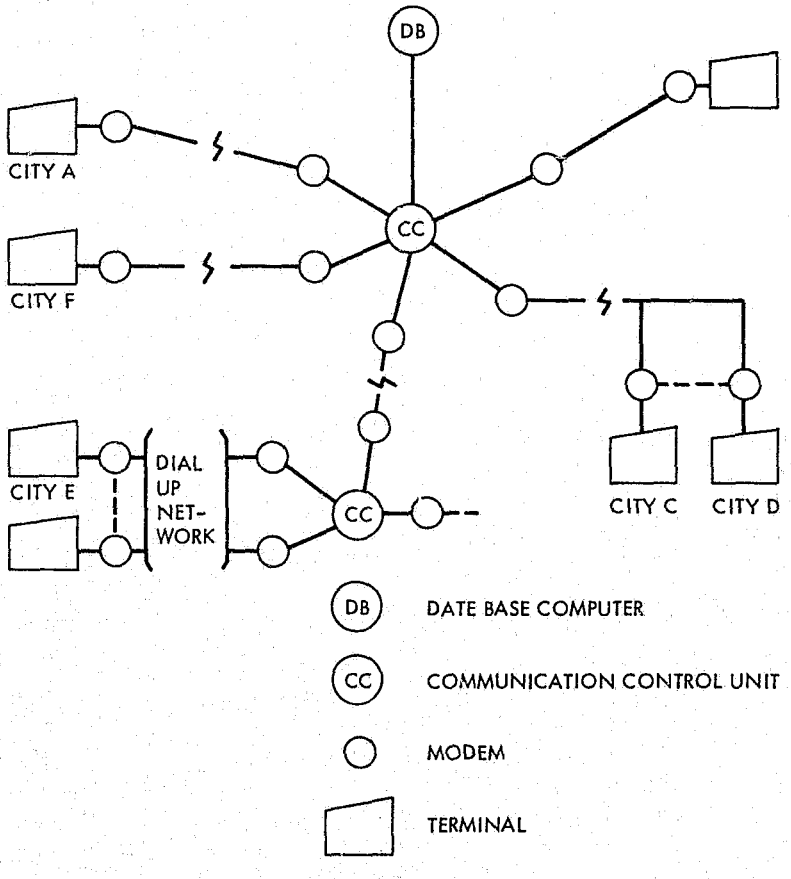


Figure 2-1. Example of a Digital Communication Network

2.1.4 Communication Network Configurations

The communication network for an information system with a central data base computer is one of three basic network configurations: the star, the multidrop, or distributed connection. These three types are shown in Figure 2-2.

As shown in Figure 2-2, the star network consists of four direct connections, one for each system termination. Each connection is called a central link. The multidrop network has one line with two system terminations and two central links. In the distributed network shown, more than one path exists between each individual system termination and the central data base.

2.1.5 Network Optimization

Given a communication network, the operating costs for the various types of lines or common carrier facilities required are governed by tariffs based upon location, circuit length, and type of line. Experience suggests that the operating cost of a network can often be substantially reduced by an initial investment in a configuration analysis. In other words, some efforts in network optimization generally provide cost-saving.

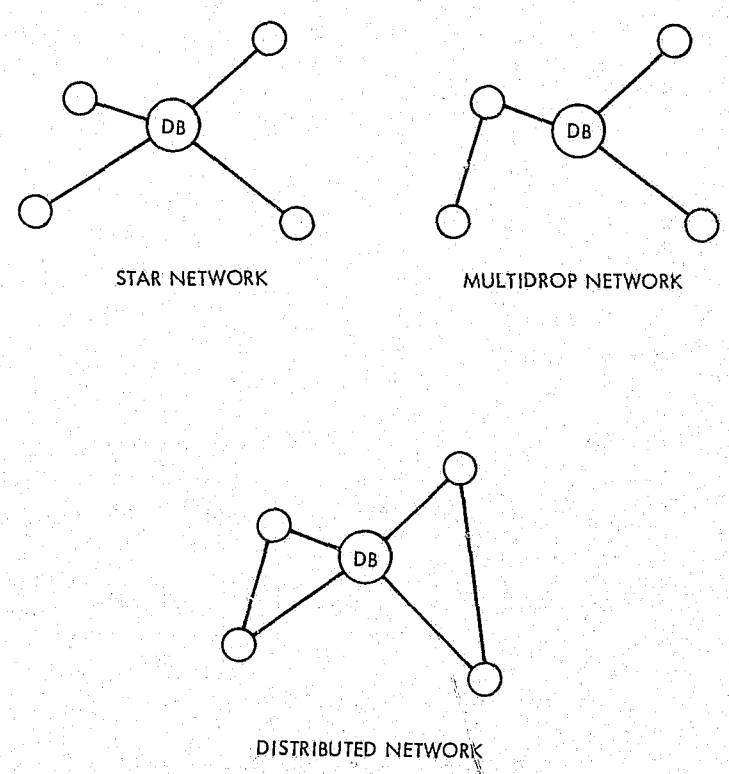


Figure 2-2. Basic Communication Network Configurations

There are two ways of constructing a communication network in a geometrical sense. One can divide a communication system into several regions, construct an optimal regional communication network for each region, and then build an inter-regional network connecting all of the regional centers to the central data base center. Each regional center is responsible for switching messages issued from and returned to each system termination in the region. Alternatively, one can consider the whole system as a region which is entirely made up of system terminations, and perform the optimization for that region.

2.1.6 The STACOM Program and its Purposes

One of the objectives in the STACOM study is to design optimal and effective communication networks which will satisfy predicted future traffic loads for both selected model states, Ohio and Texas. In order to achieve this objective, the STACOM program was developed and utilized for the analysis and synthesis of alternative network topologies. It is also the project's goal that the final product be a portable software package which can be used as a network design tool by any user.

In network design, two major problems are the selection of a cost-effective line configuration for given traffic, and the design of an optimal network to arrive at lower operating costs.

The goal of the STACOM program is to provide a user with a systematic method for solving both problems. In other words, the main purpose of the STACOM program is to provide the network designer with a tool which he can use for line selection and for obtaining optimal line connections.

2.1.7 Functions Performed by the STACOM Program

The STACOM program can be used to generate an optimal network configuration for a communication system if traffic to/from each system termination is provided. In addition to performing the normal input/output functions, the program will:

(1) Define regions, based on equal traffic distribution.

(2) Select regional centers, based on minimal traffic-distance product sum.

(3) Form a regional star network with the selected regional center as the regional switching center (RSC).

(4) Perform regional network optimization.

(5) Form an optimized inter-regional network if required.

In performing initial network formation and subsequent optimization, line selection is done by the STACOM program to satisfy the following conditions:

(1) The line utilization factor does not exceed a specific number

(2) The average terminal-response time is less than a preselected unit of time

(3) The number of terminals on a multidrop line is less than a preselected number.

In the process of regional network optimization, the STACOM program utilizes a modified Esau-Williams method (Reference 1). Starting with a star network, in which each system termination has a central link to the regional center, the optimization process searches for a central link, the elimination of which will provide the best savings in cost; the program then provides an alternate route for the traffic that would have been carried by the link eliminated. The process is repeated until no further cost saving is possible. The result of this process is a multidrop network.

When a communication system has more than two regions, the STACOM program can also be used to generate an optimal inter-regional network. It first constructs an initial inter-regional network in which every Regional Switching Center (RSC) has a direct link to every other RSC, it then performs line elimination by diverting traffic through other routes.

Figure 2-3 gives examples of regional star networks and an initial inter-regional network; Figure 2-4 gives examples of optimized regional networks and inter-regional network obtained from Figure 2-3.

2.2 MAIN FEATURES

As described in Paragraph 2.1, the STACOM program has been developed for the purpose of performing analysis and synthesis of alternative network topologies. The following is a list of features which characterize the STACOM program:

(1) The Esau-Williams routine has been modified, tested and utilized for determining near optimal network topology.

(2) A tree type structure is used as the storage structure in the program.

(3) The program execution has been made flexible; for example, constraint on response time for a multidrop line is now an input parameter.

(4) A response-time algorithm has been implemented in the program.

(5) A CalComp plotting routine has been included for drawing resulting multidropped networks.

In the rest of this subsection, these main features are discussed in detail.

2.2.1 Structure

2.2.1.1 Storage. Since a multidrop network can be viewed as a tree composed of sub-trees, it was determined that a tree-type data structure would be appropriate and convenient for representing a multidrop network.

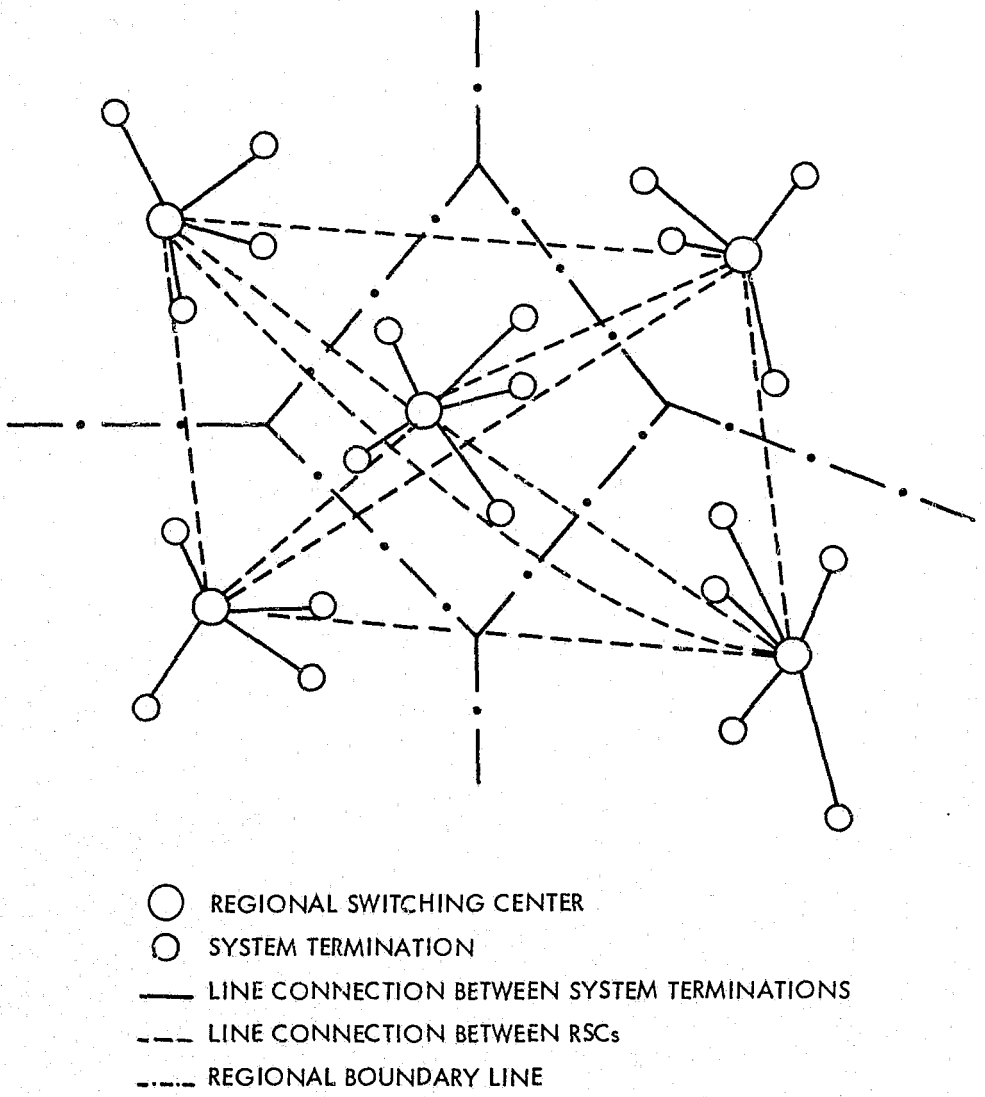


Figure 2-3. Example of Initial Regional Networks and an Initial Interregional Network

A tree-type storage structure is therefore needed in the program. This tree-type storage structure is implemented by defining a set of storage cells.

Each system termination (data) is represented internally by a storge cell in the program. Each cell consists of five fields and each field occupies one word (i.e., a 36-bit word for UNIVAC 1108 computers).

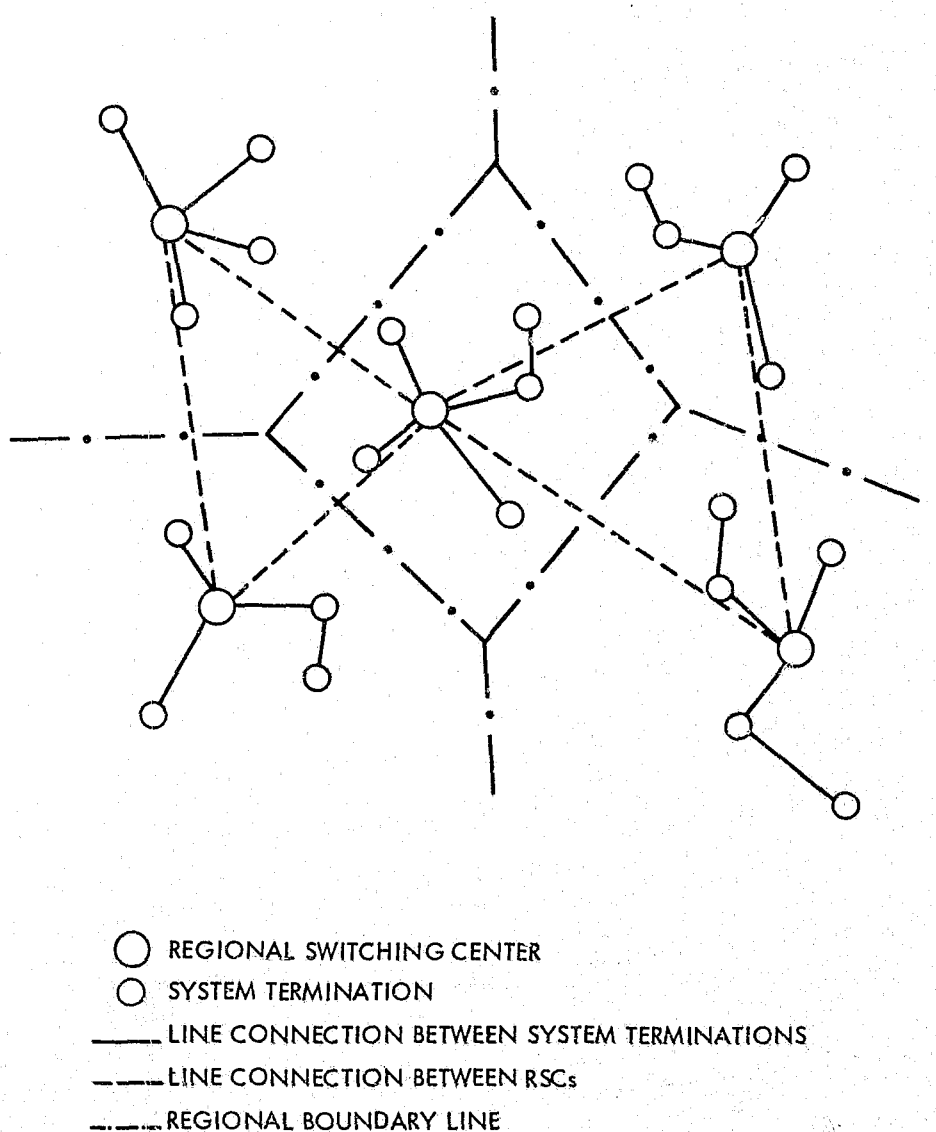
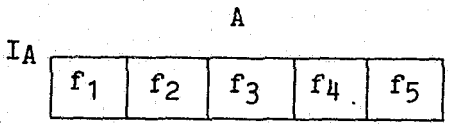


Figure 2-4. Example of Optimized Regional Networks and an Optimized Interregional Network

Defining that system termination X is a successor of Y and Y a predecessor of X if X branches out from Y, and X is the root of a tree if it has no predecessor before it, then the basic storage cell for system termination A can be described as follows:



Let $c(f_i)$ = content of i-th field in a storage cell I_A, where IA is an internal index for a system termination A (data), then

$c(f_1)$ = the number of system terminations under A

$c(f_2)$ = a pointer which points to the first successor of A

$c(f_3)$ = a pointer which points to the next system termination whose predecessor is the same as A's

$c(f_4)$ = a pointer which points back to the previous system termination whose predecessor is the same as A's

$c(f_5)$ = a pointer which points to A's predecessor

When there is a 'zero' in a field, this indicates there is no one relating to A under that specific relationship. Given a tree as Figure 2-5, A is root of the tree; it has 4 successors, i.e., B, C, D, and E. Figure 2-6 is the internal representation of that relationship among indices I_A, I_B, I_C, I_D, and I_E which are internal cardinal numbers for system terminations A, B, C, D, and E.

The first field of storage cell I_A indicates that there are four system terminations under I_A; the pointer to I_B says that I_B is its first successor. Since I_A is the root of the tree, the other three fields are left with zeros.

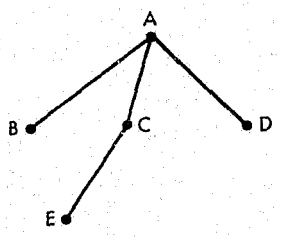


Figure 2-5. A Tree with A as its Root

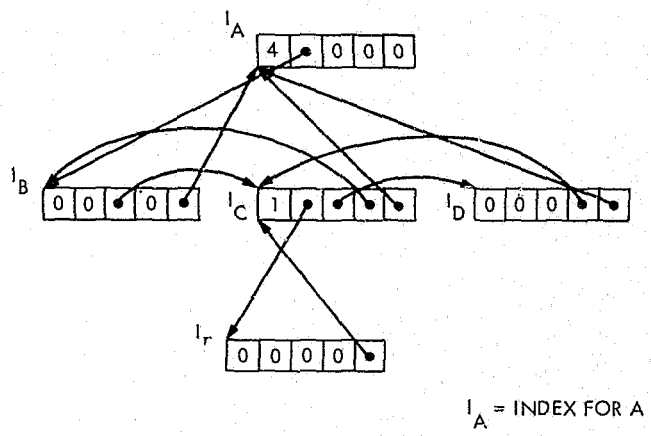


Figure 2-6. Internal Representation of the Tree in Figure 2-5

In the case of I_C, I_D is the next successor of I_A and the previous successor of I_A is I_B. Its third field has a pointer pointing to I_D, and its fourth field a pointer pointing to I_B.

2.2.1.2 Program. The STACOM program consists of twelve functionally independent routines. Figure 2-7 shows the basic structure of the program. The functional interrelationship is indicated by arrows.

An arrow from routine A to routine B indicates that routine B will be called upon by routine A during its execution. All of these routines communicate to each other through the COMMON block in addition to the normal subroutine arguments.

Major functions of eleven of these routines are given below. RSPNSE Routine is described in the following paragraph.

(1) MAIN Routine

This is the master routine of the STACOM program. In its execution, it reads in all the data required from an input device (card reader or demand terminal) and performs calculations of distances between any two system terminations. It assigns system terminations to regions, and, if necessary, selects the regional switching center by finding the system termination in the region with the minimal traffic-distance product sum. It calls upon routine RGNNET to build a star network and then performs network optimization, if required, for each of these regions.

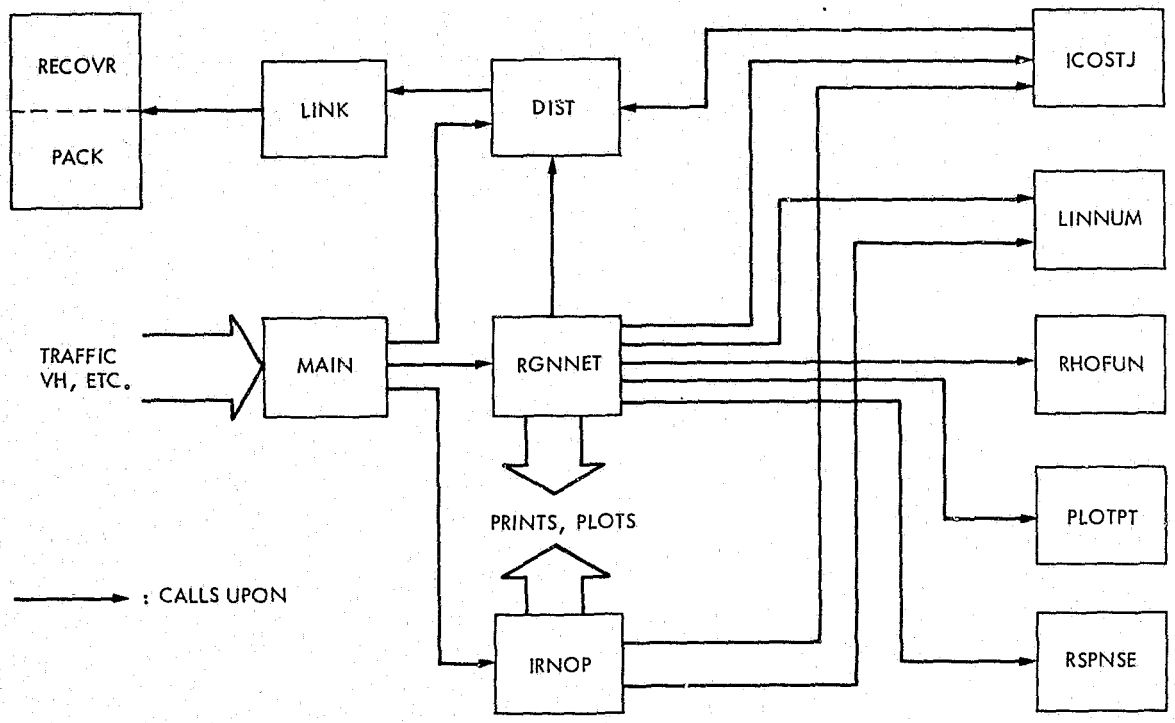


Figure 2-7. STACOM Program Structure

It also performs the construction of an inter-regional network and its optimization by calling subroutine IRNOP.

In addition to these processings, the MAIN routine also prints out distance matrix, traffic matrix, and lists of system terminations by region.

(2) RGNNET Routine

This routine is called upon only by the MAIN routine. Its main functions are the formation and optimization of regional star networks. During the formation of a regional star network, each system termination is linked directly to the designated or selected Regional Switching Center (RSC) by assigning the RSC index to the last field of each associated storage cell. Tree relationships are built among system terminations by assigning pointers to the third and fourth fields of each storage cell. The resulting star network is then printed on the printer.

The optimization process utilizes the Esau-Williams algorithm (Reference 1) with some modifications. It consists of two steps: searching for a central link (a direct link from a system termination to RSC) with best cost savings under constraints (such as response-time requirement), and subsequent network updating. This network optimization process is executed only upon request. When no further cost improvement is possible, this routine prints a resulting network with data such as number of system terminations and the response time, traffic, cost, etc., associated with each multidrop line. Routine PLOTPT is then called upon to plot the resulting network layout.

(3) IRNOP Routine

This routine is called upon to act by routine MAIN. It forms an interregional network and then performs its optimization. The interregional lines are assumed to be full-duplex lines. During the optimization process, no line between two RSCs can be eliminated if traffic between them cannot be handled through only one intermediate RSC. Also, each RSC requires at least two lines to other RSCs.

(4) LINNUM Routine

This routine provides an estimated line configuration required to satisfy a given traffic load and is mainly called upon by routine RGNNET. During its execution, utilization of selected lines are calculated against the given traffic load by calling RHOFUN so that effective line utilization is less than the pre-determined number.

(5) RHOFUN Routine

This routine calculates the line effective utilization for a given traffic and line configuration.

(6) ICOSTJ Routine

Given the line configuration and indices for any two system terminations, this routine calculates the installation costs and annual recurring costs for the line and other chargeable items required. In calculating line costs, it calls upon routine DIST for distance data between two given system terminations. Resulting cost data are arranged by chargeable item type.

(7) DIST Routine

This routine retrieves distance data between any two system terminations by calling routine PACK. When the

distance is greater than 510 miles, it retrieves distance data by calling routine RECOVR.

(8) PACK Routine

This routine stores or retrieves distance data between any two system terminations. It is called upon by routine MAIN for distance data depositing, and called upon by routine DIST for its retrieval. For the purpose of saving storage, distance data has been compressed, and each 36-bit word has been divided into four sub-words of 9 bits. Therefore, any distance datum with value equal to or greater than 511 is stored in another specified area; its retrieval calls upon routine RECOVR.

(9) RECOVR Routine

During distance data retrieval in the execution of the DIST routine, if the return value from routine PACK is 511, this routine will be called upon to provide the actual distance data, which is equal to or greater than 511.

(10) LINK Routine

Since the distance between any two system terminations I and J is independent of how I and J are referred to, the routine LINK provides a mechanism for preserving such an independency by mapping I and J into an absolute index.

(11) PLOTPT Routine

This routine provides instructions for plotting a given point on a CalComp plotter. Location of a point is calculated by its associated Vertical-Horizontal (V-H) coordinates (defined under Paragraph 2.4.2).

2.2.2 Response Time Algorithm -- RSPNSE Routine

There is a limit on the number of terminals which can be linked together by a multidrop line due to constraints on reliability and response time. However, it would be an oversimplification to just use a particular number as the main constraint in determining how many terminals a multidrop line can have. In reality, the response time of a given multidrop line depends on the amount of traffic, the number of terminals on the line, and very heavily, on the number of transactions to be processed in the data base computer system.

In the STACOM program, a response time algorithm is implemented in such a way that during the network optimization process it is used to accept or reject the addition of a given terminal to a multidrop line. This response time routine calculates the average response time

on the given multidrop line, given the number of terminals and amount of peak traffic on the line. This average response time accounts for the following types of delays; the wait-for-line time and line service time for the inquiry message from a terminal to the central switcher (i.e., a switcher which either contains data bases or communicates directly with the data base computer), the computer turnaround time at the switcher, and the wait-for line time and line service time for the returned message to the terminal. When there is an RSC between a terminal and the central switcher, the turnaround time at the RSC and the line service time between the RSC and the central switcher are counted as part of the average response time. Before its inclusion in the STACOM program, the fidelity of this algorithm was evaluated by simulation and found to be acceptable.

2.2.3 Flexibility

At the outset of the STACOM project it was anticipated that the STACOM program would be used for states with varying traffic requirements; it was decided that the resulting program should be as flexible and general as possible. With this in mind, the STACOM program has been implemented with the following features which make it flexible and thereby enhance its capabilities:

(1) Rate Structures, Line Types, and Chargeable Items

Because a State can have more than one rate structure (tariff) applicable at any one time, the STACOM program has been designed to accommodate this.

Under a specific rate structure, any combination of line types with their names, line capacities, and basic cost figures can be prescribed to the program. In addition to the line cost, any number of chargeable items associated with each line type can be prescribed to the program. For example, any combination of cost items such as service terminals, drops, modem and others can be used. Furthermore, under the Multischedule Private Line (MPL) tariffs given by AT&T for interstate communication lines, the monthly line charge between any two terminals is now a function of both the inter-city distance and the traffic densities of both terminal cities. The STACOM program has been implemented in such a way that it can take line-cost figures based on MPL tariffs or other tariffs.

(2) Region Formation, Switcher Selection, and Network Optimization.

Given a set of system terminations dividing them into regions can be performed in either of the following ways: the user can pre-assign some or all of the terminations into preselected regions, alternatively, the user can let the program perform the region

formation by simply providing the system centroid. Following the formation process, the STACOM program will start selecting regional switching centers for regions without a preassigned switching center. The process of regional network formation and its optimization will then follow.

(3) Number of Terminals per Multidrop Line.

It may be desirable to set a limit on the number of terminals on a multidrop line. In its implementation, the STACOM program takes this number from the user's input data as a constraint during its optimization process.

(4) Average Terminal Response Time.

Besides the limit on the number of terminals allowed on a multidrop line, a good network design also requires a constraint on the average terminal response time on a multidrop line. The STACOM program allows a user to specify the limit on a run basis.

2.2.4 Programming Language

The STACOM program is implemented with the FORTRAN V language of UNIVAC systems, compiled with the EXEC-8 FORTRAN processor, and mapped by its MAP processor.

Detailed features of FORTRAN V programming language are described in Reference 2.

2.2.5 Operating System Requirements

Because the EXEC-8 operating system of the UNIVAC 1108 computer was used in the development of the STACOM program, the current edition of the STACOM program can only be executed under the EXEC-8 system. Furthermore, since a CalComp routine is linked with the program, the plotter must be part of the operating system. If such a hardware unit is not included in the system, the STACOM program must be updated to reflect this environment.

In addition, the current STACOM program was designed with the feature that all the desired output be put into a FORTRAN file designated as 100. Before executing this program, a file with the name 100 must be assigned. Otherwise, regular WRITE unit 6 will be the destination output file, e.g., the print output will go the user's demand terminal when it is run as a demand job.

As an example, the following is a complete list of EXEC-8 control statements which need to be prepared or typed in after the run card for properly executing the STACOM program.

```
@ASG,UP 100
@SYM,P PUNCH$,,G9PLTF
@XQT  File.STACOM
        .
        .
      (data)
        .
        .
@BRKPT 100
@FREE  100
@SYM   100,,T4
```

The @SYM,P command directs the resulting plot card images to a CalComp plotter designated G9PLTF. The last @SYM command directs print output to a slow hardcopy printer designated T4.

2.2.6 Functional Limitations

While the STACOM program was designed and implemented with the intention that it be applicable as widely as possible, it does have certain limitations. These are due mainly to the limit of the program size (sum of I and D bank) allowed under the EXEC-8 system for simplistic programs. The maximum program size allowed is 65k words per program. Although it is more convenient for later use to assign all parameters with maximum values (as long as the overall program size is within the 65K-word limit) this results in greater expense in use of the program due to the higher core-time product. Therefore, it is recommended that all parameters be set at values just high enough for anticipated use.

After setting parameter values, the STACOM program capabilities are then limited to these assigned values. If a run requires that a certain parameter value be exceeded, the STACOM program must be recompiled and remapped.

2.3 INPUT

2.3.1 Data Requirements

A setup of input data is needed before starting a STACOM program run. The list of data items which need to be provided by the user are given here in temporal order and explained briefly. Detailed FORTRAN V formats for these are described in Table 3-1 of Section 3.

2.3.1.1 Number of Regions. The first datum needed by the STACOM program is the exact number of regions under consideration. This number (designated internally as NR1) instructs the program to divide all of the system terminations into NR1 regions.

2.3.1.2 Number of System Terminations, Number of Data Bases, and Number of Terminal Cities. The number of system terminations is the actual number of system terminations to be operated on by the STACOM program, and is designated internally as N1. In anticipation of possible multiple data bases at different locations, the number of data bases (designated internally as N7) informs the program that each system termination has N7 pairs of data (one pair per data base).

The number of terminal cities (NCITY) informs the program that NCITY V-H coordinates are to be provided later.

2.3.1.3 Identification of Data Bases and V-H Coordinates. N7 identifications provides the exact locations of data bases under consideration. All of the V-H coordinates for NCITY terminal cities are needed for calculating distances between any two cities.

2.3.1.4 Descriptions of System Terminations. For each of the system terminations under consideration, the set of data, i.e., identification, name, city location index, and traffic to all of N7 data bases are needed in order to properly execute the STACOM program.

2.3.1.5 Rate Structure and its Application Rule. There may exist one or more line tariffs applicable to different portions of any given state. The STACOM program has been designed with a capability to handle this situation. The number of applicable rate structures (line tariffs) and the rule governing their applications have to be input to the program by the user.

2.3.1.6 Traffic Density and Applicable Rate Structure for each System Termination. In order to accommodate the fact that costs for lines between high traffic density cities are much lower than for others, (e.g., TELPAK lines), the traffic density index and applicable rate structure for each system termination informs and directs the program to properly perform costing on lines connected to this termination.

2.3.1.7 Descriptions of Applicable Lines. The user dictates to the STACOM program the types of applicable communication lines by providing number of lines, their names and capacities, their desired maximum utilizations and their uses.

2.3.1.8 Descriptions of Chargeable Items. In addition to costs for lines, there are several other chargeable items such as modems, service terminals and drop charges. The user must provide the number of chargeable items and their names. Furthermore, the user has to provide the STACOM program with installation and monthly recurring costs for each chargeable item as a function of rate structure, line type, traffic density, and duplexing mode. This costing information is required to estimate overall cost of the to-be-designed communication network.

2.3.1.9 Line Cost Data. Installation and monthly recurring costs for lines for each applicable line type as a function of rate structure, traffic density, and duplexing mode are also required.

2.3.1.10 Constraints on Formation of Regions. The user can preload any number of system terminations to preselected regions if so desired by assigning them to their specific destinations (regions). He can also put constraints on preselected regions by not allowing any insertion of system terminations to these regions.

2.3.1.11 Options on Regional Network Optimization. The user can direct the STACOM program to perform regional network optimization on regions if required. This is done by simply specifying such requests to the program.

2.3.1.12 Protocol Characteristics for Multidrop Lines. The user must provide characteristics of line protocol to the program. For example, characteristics such as number of polling characters, NAK response characters, and message overhead characters are required. These data, along with the other line traffic characteristics data, enable the STACOM program to estimate the average terminal response time for a given multidrop line.

2.3.1.13 Characteristics of Future Traffic. Characteristics for future line traffic are also required. Data such as number of message types, their ratios, and average lengths allow the program to compute line service time and line utilization, which, in turn, are used to estimate the average terminal response time.

2.3.1.14 Preloading System Terminations to Preselected Regions and Pre-Assigning Regional Switching Centers. If the user wishes to assign certain system terminations to preselected regions and to pre-assigned regional switching centers, he can now proceed to do so. Otherwise, the program will perform these functions automatically.

2.3.1.15 Assigning System Centroid. If the STACOM program is required to divide system terminations into regions and to select regional switching centers, the system centroid is required so that the program can divide them properly (in a geographical sense).

2.3.1.16 Descriptions of the Central Switcher. Data describing the central switcher are needed to compute switcher turn-around time for a given transaction. These data include the estimated message rate at the switcher, number of transactions entering the switcher for completing a message, average service time per transaction, and number of processors available.

2.3.1.17 Constraints on Multidropped Lines and Average System Response Time. The user can impose a constraint on the number of terminals allowed on a multidrop line either by limiting the number of terminals on a multidrop line, or by setting up a maximum average response time limit to the multidrop line or both.

2.3.1.18 CalComp Plot. The user can request a CalComp plot of the final multidrop communication network if so desired. Of course, some installations may not have such a device and the STACOM needs to be recompiled without plotting routine.

2.4 PROCESSING LOGIC

The previous section described the type of input data needed by the STACOM program. This subsection will be devoted to the processing logic implemented in the program.

2.4.1 Traffic Calculation

2.4.1.1 Traffic Conversion. In the STACOM program, each system termination is provided with a set of traffic figures which represent outgoing traffic to and incoming traffic from each data base in the system. The unit of traffic is specified as characters per minute.

The traffic data for all system terminations are read into the matrix TRAFD(N1, 2, N7) during the data input phase, where N1 is the number of system terminations and N7 is the number of data bases. While the input traffic data are given in characters per minute, the STACOM program is designed to deal with traffic in terms of bits per second (BPS). Thus, at the time of program execution, all traffic data are converted into units of bits per second by multiplying them by a factor of 8/60. Here, we assume that synchronous communication is to be used.

2.4.1.2 Origi· and Destination Traffic by System Terminations. Summations across the last subscript of the TRAFD matrix are performed to give total traffic originating from and destined for each system termination. The resulting data are stored in TRAFIT (N1) and TRAFDN (N1), respectively. More specifically, originating and destination traffic totals are given by

$$\text{TRAFIT}(i) = \sum_{j=1}^{N7} \text{TRAFD}\ (i, 2, j)$$

and

$$\text{TRAFDN}(i) = \sum_{j=1}^{N7} \text{TRAFD}\ (i, 1, j)$$

2.4.2 Distance Calculation

2.4.2.1 V-H Coordinates. The length of the line plays a major role in determining line costs on communication networks. While the common carrier is free to route the line over any desired path, and may switch the line to different paths to circumnavigate breakdowns or overloads, the line charges are normally independent of actual line layout and are based on the straight line distance between the points connected.

The AT&T has a system in which they have divided the United States by horizontal and vertical grid lines. By means of these lines, they give almost every city/location a vertical (V) and horizontal (H) coordinate, these coordinates provide the layout-free way of distance calculation.

2.4.2.2 Distances between System Terminations. With V-H coordinates as defined by the AT&T, the distance between any two locations is calculated as follows (Reference 3):

(1) Obtain the V and H coordinates for these two locations.

(2) Obtain the difference between the V coordinates and the difference between the H coordinates of these two locations.

(3) Square each difference obtained in 2 and take a summation of both squares.

(4) Divide the sum obtained in 3 above by 10. Round to next integer number if any fraction is obtained.

(5) Obtain the square root of the result obtained in 4 above. This is the distance between the given locations in miles. (fractional miles being considered as full miles.)

For example, to calculate the distance between Austin and Dallas, Texas, we proceed as follows:

	V	H
Austin	9005	3996
Dallas	8436	4034
Difference	569	38

$$\text{Distance} = \sqrt{\frac{(569)^2 + (38)^2}{10}} = \sqrt{\frac{323761 + 1444}{10}}$$

$$= \sqrt{32521} = 181 \text{ miles}$$

When a specific location in the United States is not designated with specific V and H coordinates, it is normally assigned with the same V and H coordinates as the closest location.

Following the procedures as given above, the distance between any given two system terminations is calculated and stored in arrays DSTNCE or IVRD.

2.4.2.3 Distance Data Compression and Overflow Table. Given N system terminations, there are N(N-1)/2 combinations in choosing two system terminations from them. Furthermore in any given state, there exist only a few large inter-terminal distances. These two facts indicate that some reduction in resulting STACOM program size can be made by performing compression of distance data. Two efforts have been undertaken for that purpose.

Under the UNIVAC system, each computer word is 36 bits long. We divide each word into four 9-bit segments. Each segment is used to store one distance datum with values ranging from 0 to 511. To compensate for the fact that some distances data may be greater than 511, an overflow table IVRD is provided to collect oversized distance data. In other words, given two system terminations with indices I and J, its distance is recorded into DSTNCE as follows:

(1) Find corresponding V-H coordinates of locations for both system terminations.

(2) Calculate distance D according to the procedure given in Paragraph 2.4.2.2.

(3) Find a unique and absolute location L in DSTNCE, by using the following equation:

$$L = I*NPC+J - \Delta(I)$$

where

$$\Delta(I) = \sum_{i=1}^{I} i,$$

and

$$I < J$$

NPC = number of distinctive locations in the system

This mapping function is performed by subroutine LINK,

(4) Define

$$L1 = \lfloor (L-1)/4 \rfloor + 1$$
$$S1 = (L-1) \text{ Modulo } 4 + 1$$

where $\lfloor x \rfloor$ = the integer part of X and

$$D1 = \begin{cases} D \text{ if } D < 511 \\ 511 \text{ if } D \geq 511 \end{cases}$$

(5) Store D1 in segment S1 of entry L1 of table DSTNCE.

(6) If $D \geq 511$, store L and D in next available space of table IVRD.

On the other hand, given two system terminations with indices I and J, the retrieval of distance is performed as follows:

(1) Calculate L, L1 and S1 as described above.

(2) Retrieve the content D1 in segment S1 of entry L1 of table DSTNCE. If $D1 < 511$, it is the distance.

(3) If $D1 = 511$, retrieve the second element of the row of table IVRD, whose first element contains value L. The retrieval value is the distance.

2.4.3 Formation of Regions

After traffic summations and distance table formation are completed, the STACOM program starts to form regions. It assigns all of the non-preloaded system terminations to regions which can accommodate them. Figure 2-8 illustrates the process of such a function.

The process begins with an estimation of the traffic per region, called TPR, which is obtained by averaging the total non-binding traffic, i.e.,

$$TPR = TPR1/ANR1$$

with

$$TPR1 = \sum_{\substack{i \notin I \\ 1 \leq i \leq N1}} [TRAFIT(i) + TRAFDN(i)]$$

where

I = the set of system terminations in preloaded regions which do not allow other system terminations to be inserted to them

ANRI = NR1 - [number of preloaded regions which do not allow any insertions]

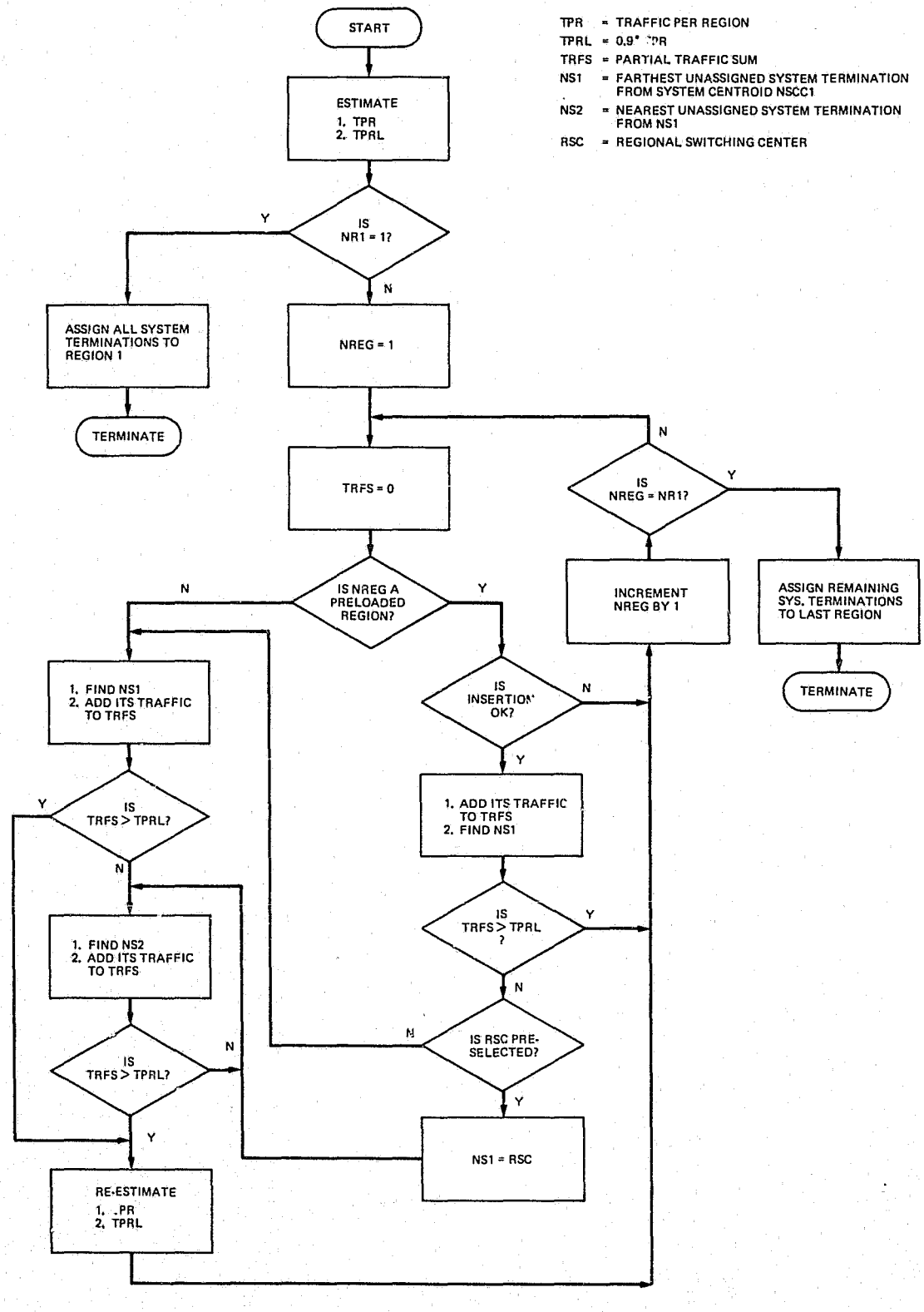


Figure 2-8. Flow Chart for Formations of Regions

When the number of regions is 1, all of the system terminations are assigned to the region and no other region formation process is performed. Otherwise, the program starts assigning system terminations to regions (in a cardinal order) which allow their entries.

The following two subsections describe the detailed processes for assigning system terminations to a region either with preloading or without preloading.

2.4.3.1 Assigning System Terminations to a Region without Preloading. When a region NREG is not preloaded with any system termination, processing continues with the finding of the farthest unassigned system termination (NS1) from the system centroid (NSCC1). This system termination is then assigned to the region NREG; its incoming and outgoing traffic is added to the partial sum traffic, called TRFS. The resulting TRFS is then tested. If it is greater than TPRL, (lower bound), which is equal to 0.9 x TPR, assignment processing for region NREG ends with re-estimating TPR and TPRL which are obtained as follows:

$$TPR1 = TPR1 - TPFS$$

$$TPR = TPR1/(ANR1 - 1.)$$

$$TPRL = 0.9 * TPR$$

On the other hand, if TPRS is less than or equal to TPRL, additional system terminations can be assigned to this region. The next system termination for addition to this region is selected by finding the nearest unassigned system termination, called NS2, from NS1. NS2 is then assigned to region NREG and its traffic added to TRFS. The value of TRF is again tested against TPRL to determine if other additions are possible.

This process is repeated until partial regional traffic sum TRFS is greater than TPRL. At this point, the region is considered full and addition of system terminations to this region stops. However, if the region being filled is the last one, all remaining system terminations are placed into this last region. Otherwise, the program continues to work on the next region. Before leaving region NREG, it re-estimates TPR and TPRL as shown before.

2.4.3.2 Assigning System Terminations to a Region with Preloading. If the region NREG is a preloaded region, i.e., it has been preloaded with system terminations, the program continues with a test. The test is needed to determine whether region NREG will accept any additional system terminations. If other insertions to the region are not allowed, the processing on this region stops and continues to the next region.

Otherwise, the program starts adding traffic to all preloaded system terminations to TRFS and finding the farthest unassigned system termination NS1 from the system centroid. It then tests whether TRFS is

greater than TPRL. If it is greater, the program stops here and continues to process the next region.

When TRFS is less than TPRL, the program checks whether there is a preselected RSC for the region NREG. If there is, the program uses the RSC as the NS1. Then it follows the same procedure as described in paragraph 2.4.3.1 to add more system terminations to the region.

It should be noted that STACOM has been implemented in such a way that when it is desired to preload some or all regions, the last one need not be specified. The program will assign the rest of the unassigned system terminations to the last region.

2.4.3.3 Example for Formation of Regions. Figure 2-9 illustrates the results of applying the formation of region logic to a Texas communication system with 265 system terminations. In this example run, neither preloading of system terminations nor preselection of regional switching centers are requested. In other words, the program is asked to perform automatic regional formations and to select the regional switching centers. System termination Austin is chosen as the system centroid.

The total amount of traffic, TPR1 is at a rate of 1585.02/bps, and the number of regions is 2. Therefore, at the beginning, TPR is given as 1585.02/2=792.51 bps, and TPRL = 713.26 bps. In the process of assigning system terminations to region 1, El Paso is found to be the farthest location from Austin, i.e., NS1 = the internal index for system termination El Paso. With NS1 available, the program starts the procedure of searching for NS2, adding its traffic to partial sum TRFS and testing whether TRFS is greater than TPRL. It repeats the same procedure 123 times until TRFS has reached the value of 750.08 bps which is greater than TPRL.

2.4.4 Selection of Regional Switchers

Selection of regional switching centers follows formation of regions as described in Paragraph 2.4.3. For a given region, its regional switching center (RSC) can be either preselected by the user or be chosen by the program. In the latter case, the program selects the system termination within the region such that total intra-region traffic-distance products are minimized.

The functional flow chart of RSC selection is depicted in Figure 2-10. Processing begins with assigning 10^{12} to WCASE (as base for traffic-distance product sum). It then calculates the estimated sum of all traffic-distance products with each system termination in the region as an RSC site. The sum, called SUMT, is obtained as follows:

$$SUMT = \sum_{i=1}^{NMBR} [TRAFDN(i) + TRAFIT(i)] * DIST(i,K)$$

where

NMBR = number of system terminations in the region under consideration

K = the index of the system termination considered as the trial RSC site

DIST(i,K) = the distance between system termination i and the RSC trial site K

The resulting SUMT is then compared with WCASE. If SUMT is found to be less than WCASE, the value for WCASE is replaced by the value of SUMT and the corresponding index for the RSC trial site is the updated RSC, called NRSC.

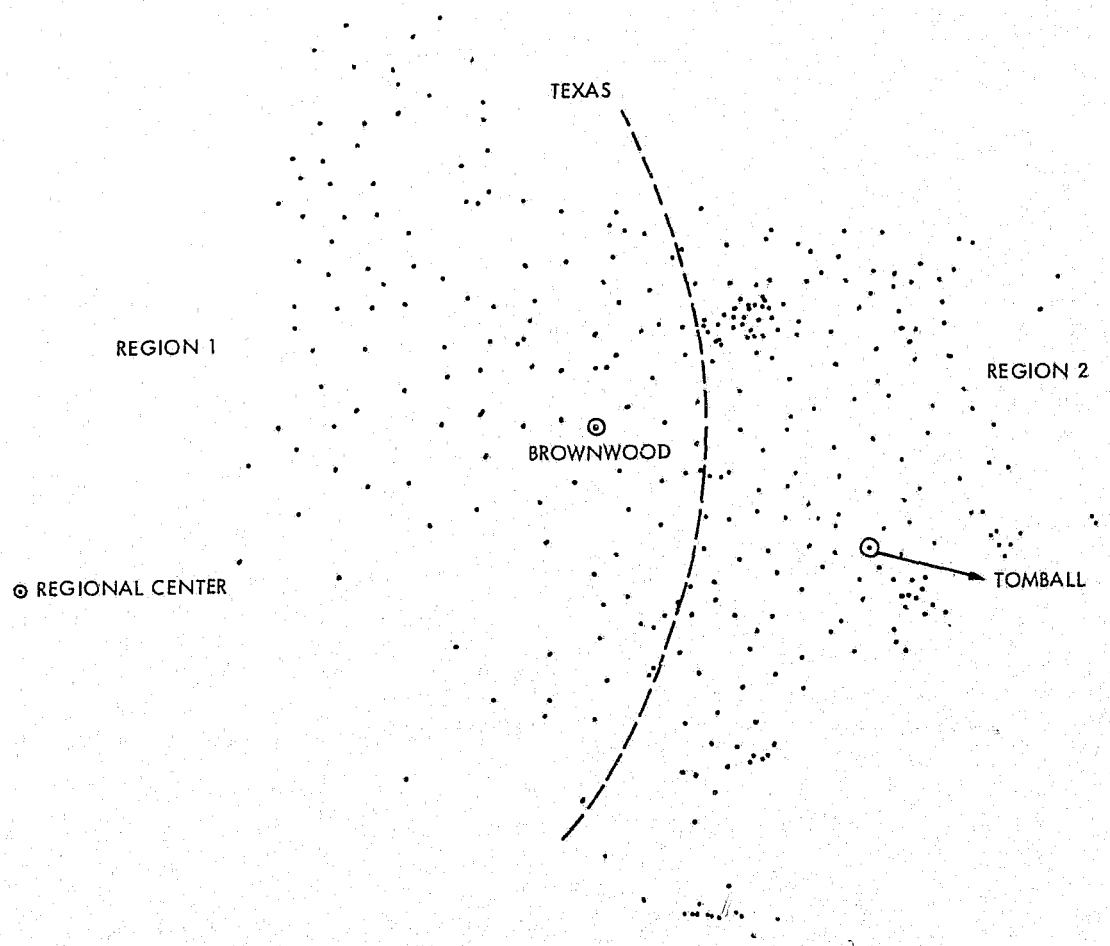


Figure 2-9. Example of Region-Network Formation and Regional Switcher Selection

NRSC = INDEX FOR REGIONAL SWITCHING CENTER

$SUMT = \sum_{i=1}^{NMBR} [TRAFDN(i) + TRAFIT(i)] * DIST(i, K)$

NMBR = NUMBER OF SYSTEM TERMINATIONS IN THE REGION

DIST(i, K) = DISTANCE BETWEEN SYSTEM TERMINATIONS i AND K

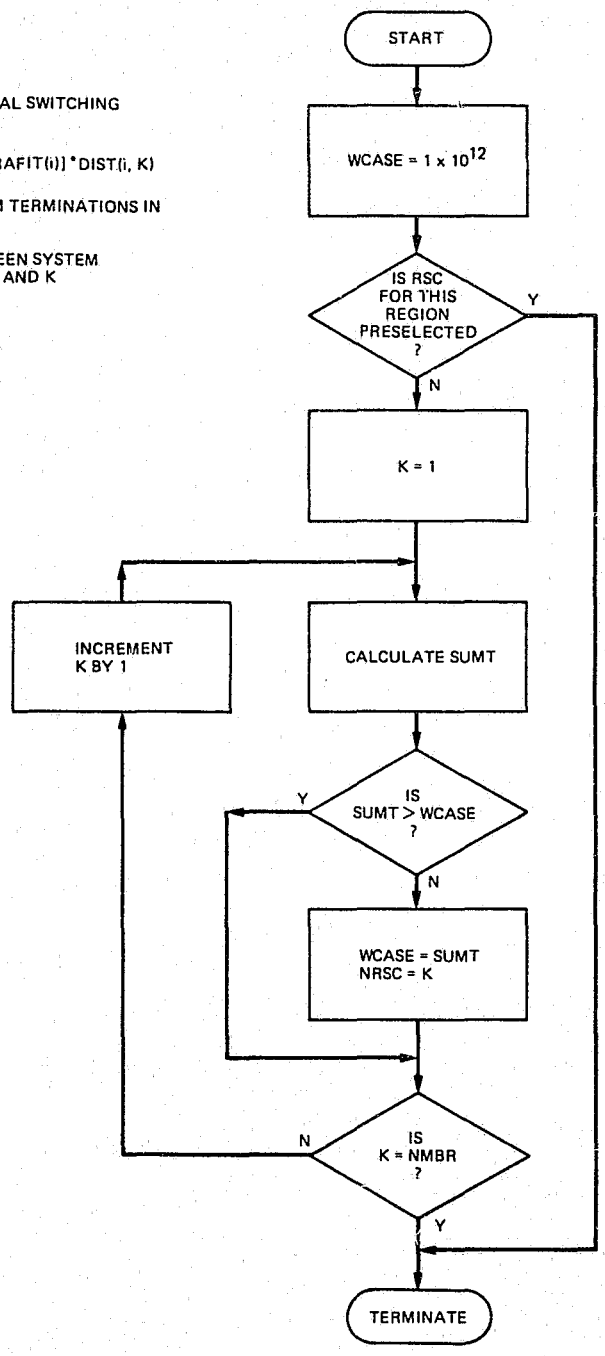


Figure 2-10. Flow Chart for RSC Selection

After the above processing has been repeated NMBR times, the resulting NRSC is the index for the selected RSC and WCASE the region's minimal traffic-distance product sum.

When a regional switching center is preselected by the user, the program skips the process as described here.

Following the selection of a regional switching center for a given region, the program continues to perform regional network formation and network optimization before it repeats the selection of regional switching centers for remaining regions.

The process of regional network formation and optimization is discussed in Paragraphs 2.4.5 and 2.4.6.

2.4.4.1 Example for Selecting a Regional Switching Center. Following the formation of regions in the example given in Paragraph 2.4.3.3, the program has chosen Brownwood of Brown county as the switcher location for Region 1 and Tomball of Harris county as the switcher location for Region 2. Both locations have been found to provide the minimal traffic-distance product sums for respective regions. These two cities are shown in Figure 2-9.

2.4.5 Formation of Regional Star Networks

Formation of a regional network starts with a star network and then continues with an optimization process which, most of the time, results in a cost-saving multidrop network. This subsection describes the process of forming a star network, which is depicted in Figure 2-11. The initial regional network is formed by directly connecting each system termination to the regional switching center. Selection of these intra-region lines is constrained by the rule that each selected line should maintain the line utilization factor, called RHO, at a value less or equal to a preselected number, say, 0.7.

For each system termination in the region, the program finds incoming and outgoing traffic, TRFOUT and TRFIN, and also its distance, DSTN, from the RSC for each system termination in the region. The program calls subroutine LINNUM, which constructs a line configuration LDUMMY and calculates its line utilization, based on the values of TRFIN and TRFOUT provided. The processing continues to calculate both the cost, COST, for the derived line configuration LDUMMY and its response time RSPTIM. Finally, all these data are stored for later printout and comparisons.

The derivation of line configuration LDUMMY by subroutine LINNUM and the associated cost, COST, deserves more explanation. The program assumes that the duplexing mode for all line types under consideration to be half-duplexed. Therefore, subroutine LINNUM will sum up TRFIN and TRFOUT and find an applicable line with the least capacity which assures less than 0.7 of utilization. When the highest capacity line cannot handle the traffic, the routine will try to add one additional line with least capacity until the constraint of 0.7 utilization factor is satisfied. With line configuration LDUMMY obtained, calculation of cost,

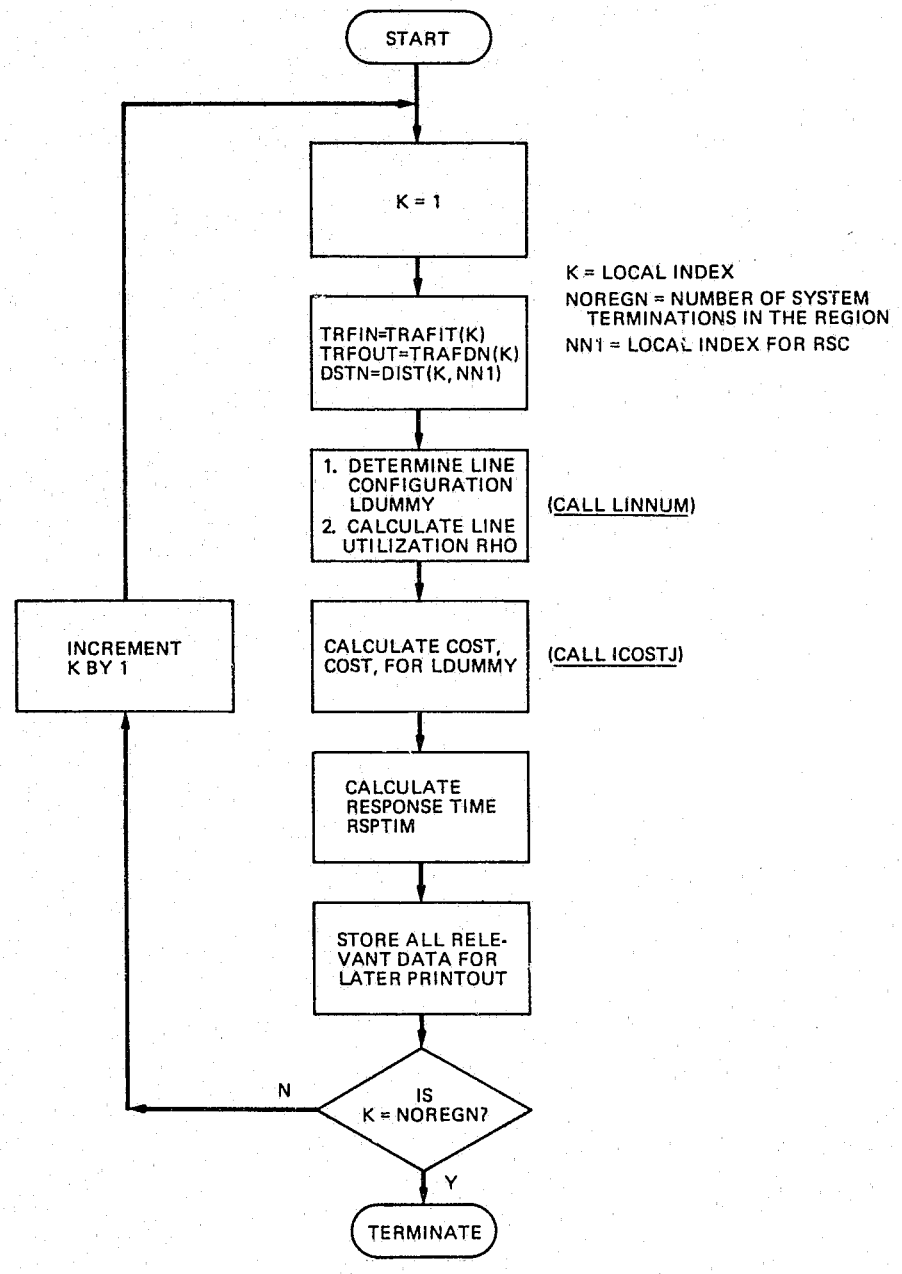


Figure 2-11. Flow Chart for Regional Star Network Formation

COST, for the direct link between system termination K and the RSC is performed by ICOSTJ. The routine ICOSTJ calculates all of the related installation and annual recurring costs for lines and other chargeable items. All of these itemized costs are then summarized as COST. Cost calculations are performed on the basis of the rate structures applicable to system terminations at both ends.

2.4.5.1 Examples of Line Selections. Table 2-1 lists some examples of line configurations results obtained by LINNUM, and illustrates how the LINNUM subroutine selects lines for given traffic. The first column of the table represents total traffic (sum of TRFIN and TRFOUT). In this example, it is assumed that only line types with capacities of 300 bps, 1200 bps and 4800 bps are under consideration. Line utilization factor has been constrained to not greater than 0.7.

2.4.6 Optimization of Regional Networks

After completing the formation of a regional star network, the program proceeds to the optimization process, if requested. The optimization process basically utilizes a technique developed by L. R. Esau and K. C. Williams (Reference 1) and is used to minimize line operating costs. The actual implementation of the technique has been made with several additional constraints for practical reasons.

Before going into detail, here is a brief explanation of the goal and process of network optimization of a regional star network. Figure 2-12 depicts a typical star network in which each system termination has a direct link, called central link, to the central regional center. The goal of optimization is to reduce line costs by eliminating as many central links by connecting the associated system terminations to their nearby system terminations as possible, until it is no longer cost-effective to do so. Figure 2-13 shows a typical multidrop network

Table 2-1. Examples of Line Configurations Obtained by Subroutine LINNUM

Traffic (bps)	Line Type			
	300 bps	1200 bps	2400 bps	4800 bps
200	1	0	0	0
500	0	1	0	0
850	0	0	1	0
1300	0	0	1	0
2000	0	0	0	1
3500	1	0	0	1

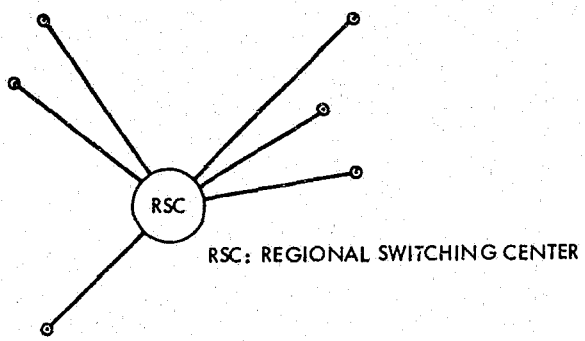


Figure 2-12. Typical Star Network

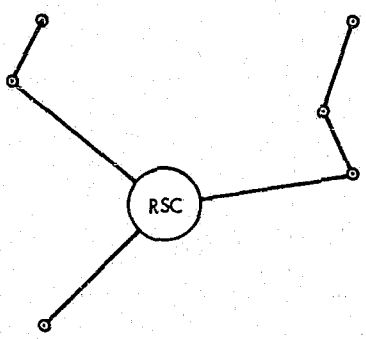


Figure 2-13. Typical Multidrop Network of Optimization

after this goal is met, assuming that the line cost is only a function of distance. While this example as given depicts the main concept of network optimization, it does not illustrate the process of sizing each newly formed multidrop line to reflect the increase of traffic resulting from the addition of new system terminations.

The following section describes the logic implemented in this program.

2.4.6.1 Network Optimization by Esau-Williams Technique. Before explaining the logic for network optimization implemented in the STACOM program, a brief explanation of the Esau-Williams network optimization process is appropriate. With a given star network, the basic process of the Esau-William technique is to repeat two basic steps until it is no longer possible to derive any cost saving.

For the convenience of the following discussion we define a sub-network (subnet) as a tree-type multidrop line consisting of one or more system terminations and having a central link connected to the regional center. Each central link of a given star type network is a simple sub-network by definition.

The first step involves searching for the best central link of a system termination, K, so that its elimination and the subsequent reconnection of the rest of the sub-network to a nearby sub-network L provides the best cost saving. In other words, for each system termination, i, with a central link to the regional center, this routine estimates the best saving, S_i, resulting from eliminating the given central link and reconnecting the rest of the subnetwork to a nearby subnetwork beginning with L_i. If we express it as a formula, then

$$K = i \text{ such that } S_i = \underset{j \in C}{\text{Max}} \{S_j\}$$

L_i = j for which the integration of K and L sub-networks provides S_K which is the best saving

where

C = the set of system terminations with central links to regional center

j = the first system termination of sub-network L

The other step involves network updates after it has been determined that the central link from system termination K is to be eliminated; this step will integrate remaining subnet K with subnet L utilizing an alternate route.

It should be noted that although this network optimization process will generate the best network most of the time, it does not always provide the best one. In other words, this technique generates the local optimal solution rather than the global solution. This is because the first selection of a central link for elimination dictates the final network to be created by repeating the process as described above. However, as shown in Reference 4, the process does provide a solution which is always close to, if not, the best.

2.4.6.2 Network Optimization Logic in STACOM Program. The optimization logic as implemented in the STACOM program basically utilizes the Esau-Williams technique. However, constraints have been incorporated into it in order to satisfy project requirements and to eliminate unnecessary searching. Figure 2-14 shows the functional flow chart for the overall logic.

The optimization process starts with the test to see whether there is only one sub-network left. If this is true, it stops. Otherwise, the program, utilizing four variables K, L, M and KI, starts evaluating possible cost saving by eliminating central link K and

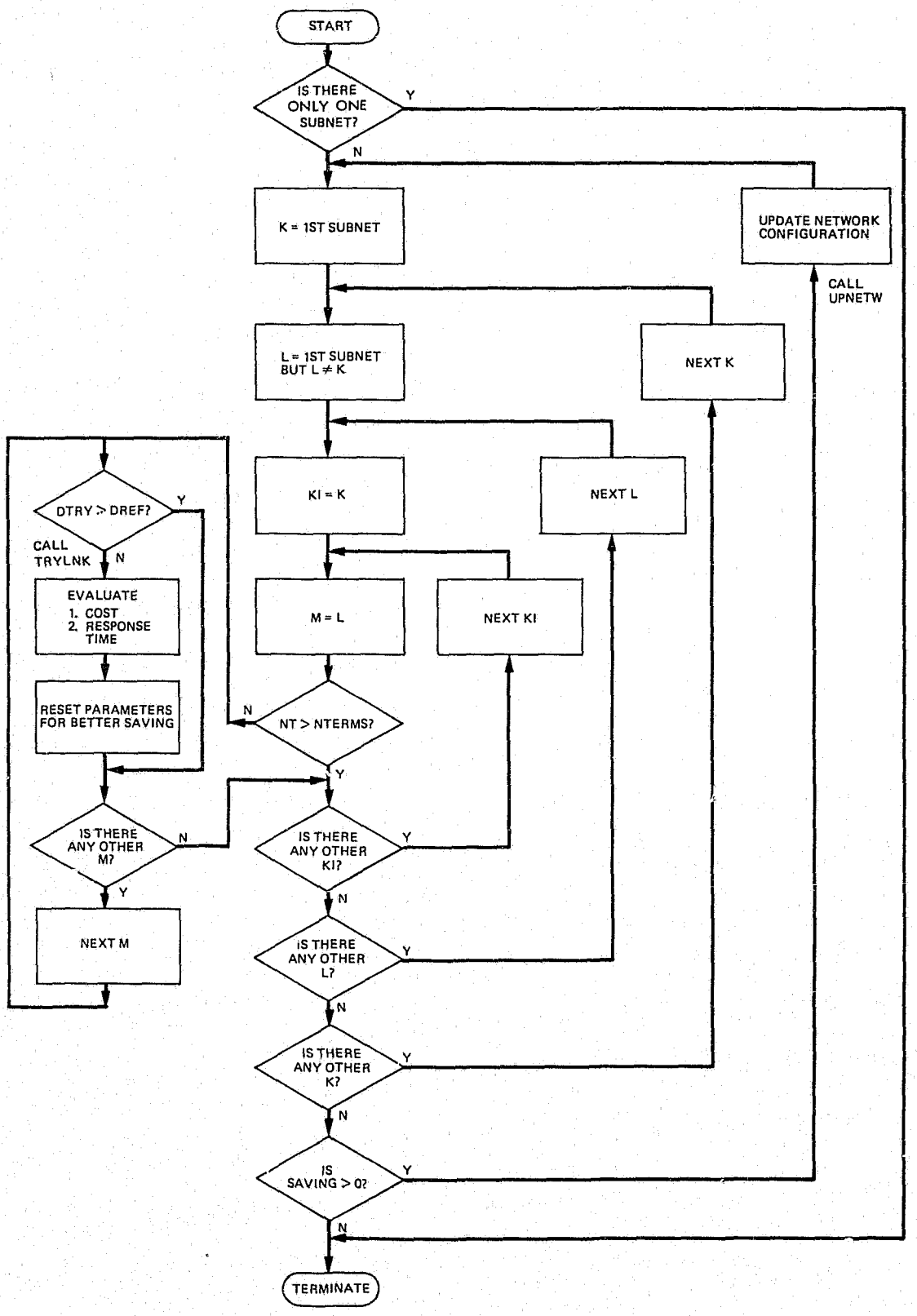


Figure 2-14. Flow Chart for Subroutine ESSWIL

reconnecting the rest of sub-network K to sub-network L through system terminations M of L and KI of K as shown in Figure 2-15.

Selections of values for variables K, L, M, and KI are in the following way. For each processing cycle, searching and updating, K is assigned the index values from the first sub-network to the last one of the existing network. For each K, L is assigned index values from the first sub-network to the last one except $K \neq L$. With values for K and L chosen, M is assigned the index values of all the system terminations on sub-network L and KI the index values of all the system terminations on the sub-network K.

For each given set of K and L, the program tests whether the sum, NT, of numbers of system terminations for both sub-networks exceeds the value of NTERMS which constrains the number of system terminations on a multidrop line. If this is true, it skips the process of calling on subroutine TRYLNK, because it is not possible to integrate both sub-networks without violating the said constraint. Otherwise, it continues to the distance test.

K = THE SUBNETWORK BEGINNING WITH SYSTEM TERMINATION K

L = THE SUBNETWORK BEGINNING WITH SYSTEM TERMINATION L

M = THE SYSTEM TERMINATION ON SUBNET L TO WHICH KI IS TO BE CONNECTED

KI = THE SYSTEM TERMINATION ON SUBNET K FROM WHICH SUBNET K IS CONNECTED TO M OF SUBNET L

DREF = THE DISTANCE BETWEEN SYSTEM TERMINATIONS K AND THE RSC

DTRY = (THE DISTANCE BETWEEN SYSTEM TERMINATIONS KI AND M)/2

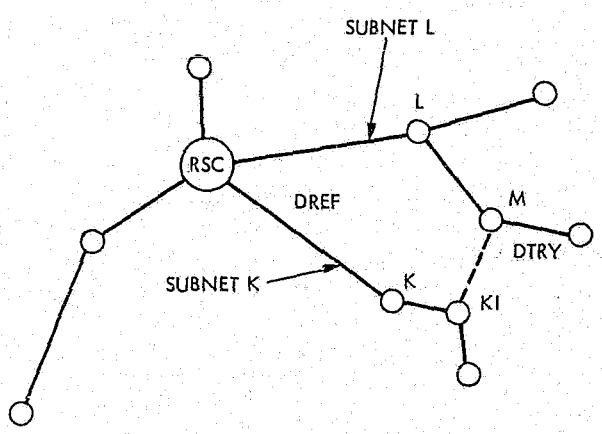


Figure 2-15. Relationship among K, L, KI, and M Parameters

The program first calculates the distance DREF between system termination K and the region switching center for each K, and then the DTRY which is half of the distance between system terminations KI and M for each combination.

If DTRY is greater than DREF, the program skips the process of calling on subroutine TRYLNK. Otherwise, it calls on subroutine TRYLNK. The purpose of subroutine TRYLNK is to estimate the possible cost saving resulting from eliminating central link K, and integrating sub-networks K and L by connecting system terminations KI of K and M of L. If the saving is better than the maximum saving obtained so far, it is used as the up-to-date best cost saving under the set of values for K, L, KI, and M. A detailed description of functions performed by subroutine TRYLNK is given in Paragraph 2.4.6.3. After all possible combinations for K, L, KI, and M have been tested and it has been found that the up-to-date best cost saving is positive, the program performs the second function of network optimization, i.e., updating the network. It then repeats the whole process on the newly updated network which happens to have one less central link.

If the up-to-date maximum cost saving is non-positive, the optimization process stops here.

2.4.6.3 Function Performed by Subroutine TRYLNK for a Given Set of Values K, KI, L, and M. The processing, as shown in Figure 2-16, starts with estimating the total amount of traffic that a single multidrop line (sub-network) of integrating subnetworks K and L needs to handle. It then estimates the required line configuration, LDUMMY, by calling subroutine LINNUM which has been described in Paragraph 2.4.5.1. Based on LDUMMY, the program estimates the average response time and tests it against the user-provided response time limit by calling subroutine RSPNSE. If the estimated response time is not satisfied, the program updates the line configuration LDUMMY to the next higher line type and repeats the process of estimating its average response time and testing it against the given constraint. This process ends when either there is a satisfied line configuration or it is not possible to upgrade any further.

When a satisfied line configuration is obtained, the program continues to estimate its cost saving, based on the assumed integrated sub-network. If the resulting cost saving is better than the up-to-date best cost saving, it replaces all of the maximum saving parameters, which are used to keep tracking the up-to-date best network changes; it then returns to its calling routine. If there is no line configuration satisfying the response time constraint, the process stops and the program returns to its calling routine.

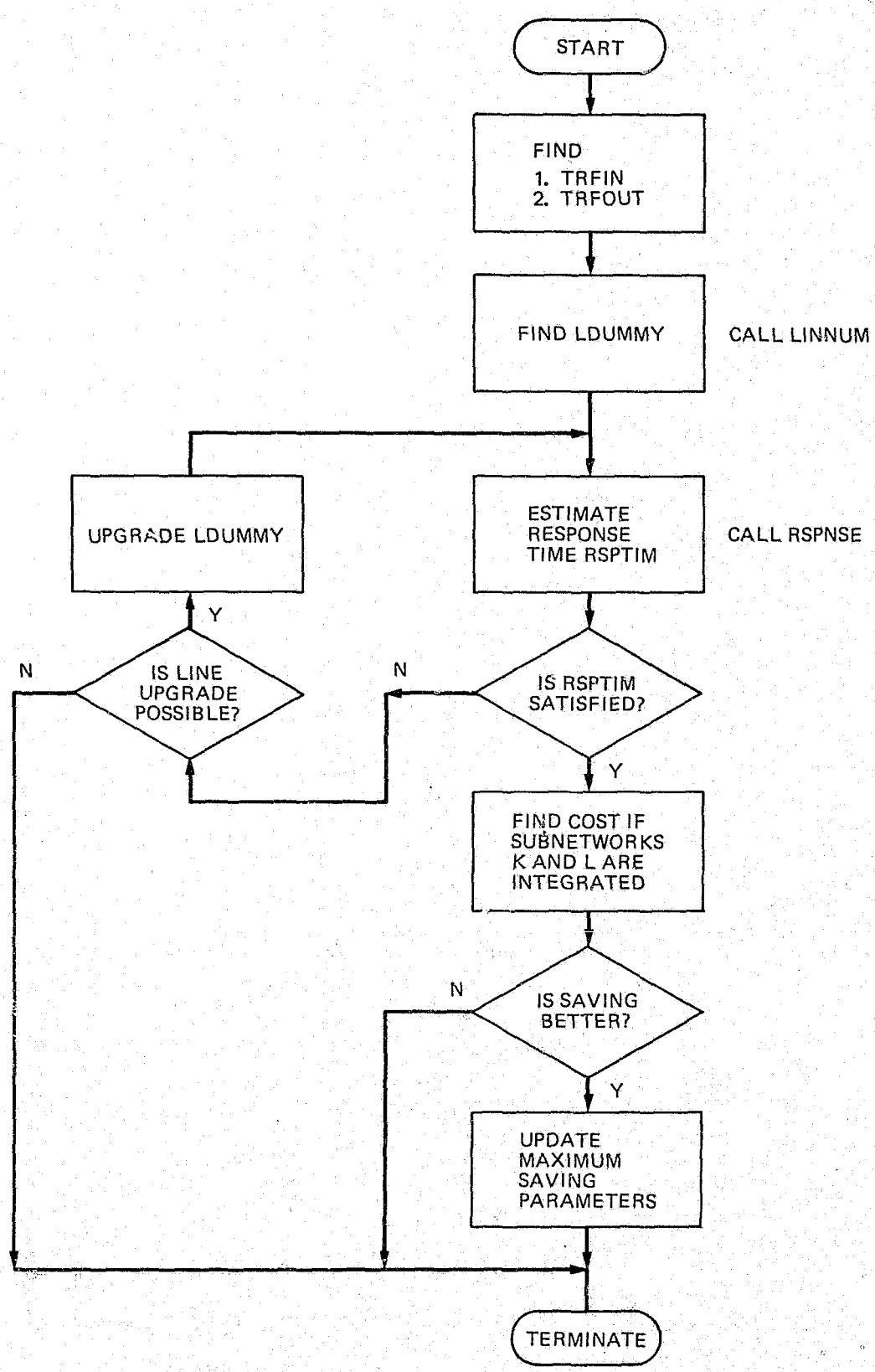


Figure 2-16. Flow Chart for Subroutine TRYLNK

2.4.6.4 Functions Performed by Subroutine RSPNSE. Figure 2-17 shows the flow chart of the subroutine RSPNSE. This subroutine calculates six items of delays: polling, message transmission time from a terminal to the central switches, input buffer queue time, service time, output buffer queue time, and returned message transmission time from the central switcher to the same terminal.

After summing up these delays as RSPTIM, this subroutine compares its value with the upper bound response time as set up by the user. It assigns 1 to IOK as an indication of satisfying response time requirement and returns.

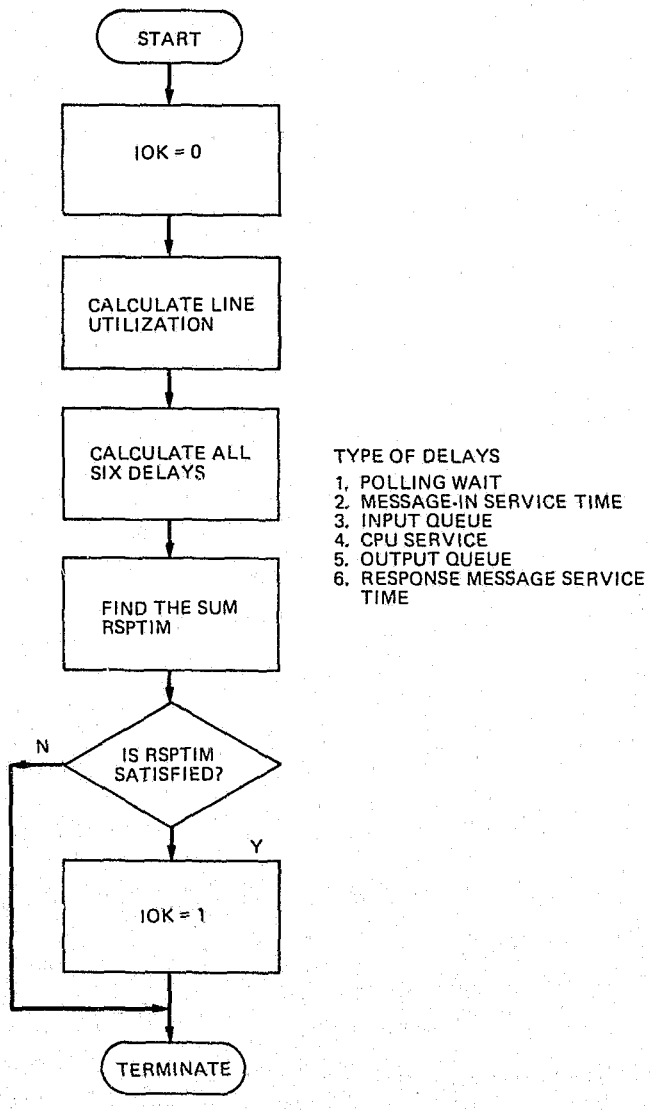


Figure 2-17. Flow Chart for Subroutine RSPNSE

2.4.6.5 Network Updates by Subroutine UPNETW. If there is a positive cost saving after trying all possible combinations for parameters K, KI, L, and M, subroutine UPNETW is called upon to perform the other function for each cycle of the network optimization process as described in Paragrapn 2.4.6.1.

In the STACOM program, subroutine UPNETW performs the following main functions: (1) updating of network descriptions, (2) revision of relevant accounting data (such as the number of terminals on the new L sub-network, its average response time, and total traffic).

2.4.7 Formation of an Interregional Network

The interregional network is formed by erecting communication lines between the regional switching centers (RSCs). The initial network has a direct line between any two RSCs.

As shown in Figure 2-18, for each combination of two RSCs I and J, the maximum traffic in either direction is considered as the design traffic between these two RSCs. This is different from intraregional line selection because it is assumed that full duplex lines are to be used. The traffic matrix TRM contains traffic data between RSCs. With this information, line configuration LINEQU between RSCs I and J is obtained by calling subroutine LINNUM.

Cost of line configuration LINEQU is then estimated and added to the total cost.

2.4.8 Optimization of an Interregional Network

After the initial interregional network is completed, the program starts a line elimination process in order to obtain a cost-effective network.

Figure 2-19 shows the basic topological consideration involved in line elimination. In considering whether line I-J can be eliminated, the algorithm tries to divert I-J traffic to other lines with excess capacity, for example, over route 1-4-3. If there is no alternate route with enough excessive capacity to handle I-J traffic, the program begins adding capacity to alternate routes in order to accommodate the required traffic. It then estimates the cost saving under the proposed modifications.

The algorithm iterates the above described process for all combinations and records the best cost saving and the best line elimination. It then updates the network.

This cycle of searching for the best cost saving and updating the network repeats continuously until cost savings can no longer be realized.

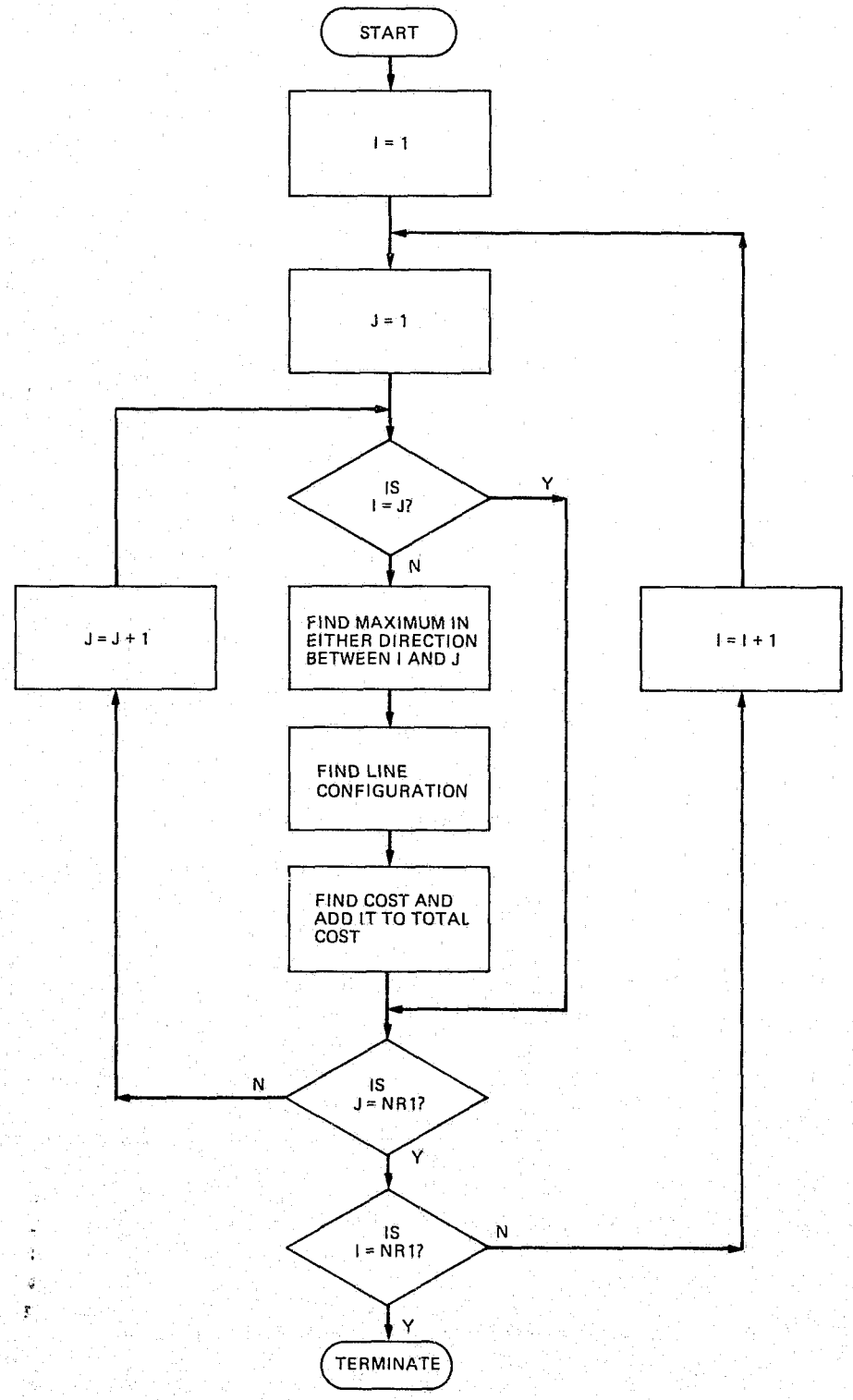


Figure 2-18. Flow Chart for Intraregional Line Selection

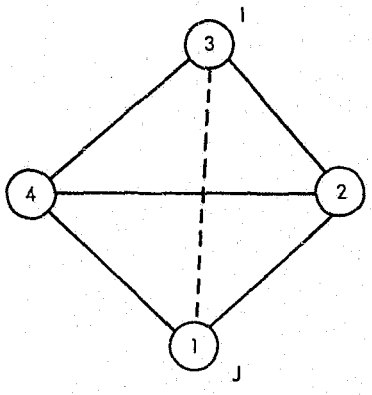


Figure 2-19. Basic Topology of Line Elimination

2.4.8.1 Interregion Network Optimization Logic Implemented. Figure 2-20 depicts the functional flow chart for the interregional network optimization as implemented in the STACOM program.

A parameter, I, is used to select one of the RSC nodes to be considered for line elimination. A test is then made on RSC I to insure that at least three links to other RSCs exist. If I has at least three links, another parameter, J, is used to select any other RSC node for trying to eliminate its link to I. J is tested to insure that it has three links to other RSCs and J is different from I. Another test is made to insure that I and J are connected to each other. If any of these conditions are not met, RSC node J + 1 is selected and these three tests are repeated.

If these conditions are met, a test is carried out to see if sufficient network connectivity will still be maintained if connection I-J is removed. Due to the consideration of availability, the program is designed in such a way that each RSC node will have at least two communication links to other RSCs and each RSC node will be connected to every other RSC node through no more than one intermediate node.

If the network connectivity requirement can be maintained with the removal of link I-J, the program searches for alternate routes with excess capacity in an effort to re-route the I-J traffic load without increasing network capacities. If all I-J traffic can be successfully diverted in this manner, the I-J link is eliminated and the network traffic matrix and costs are re-calculated; the process then begins anew.

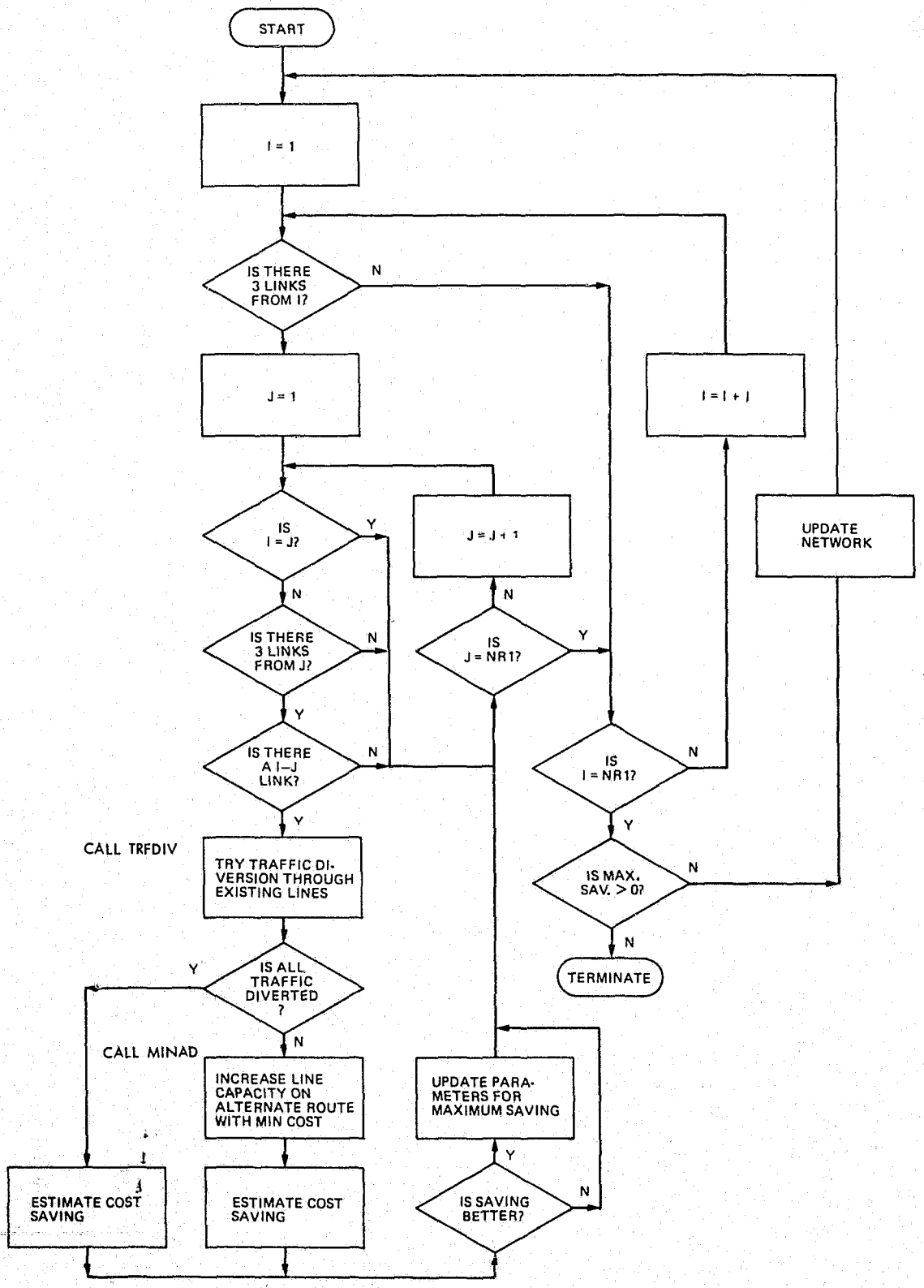


Figure 2-20. Flow Chart for Interregion Network Optimization

If all I-J traffic cannot be diverted through existing network routes with excess capacities, the capacity of the first available alternate route is increased to handle the remaining traffic. The cost saving is determined as equal to the original cost of the line removed minus the cost for the capacity increase. If the cost saving is an improvement over previous trials, line and traffic data are saved to reflect the up-to-date best modification of the network.

At the conclusion of each cycle, if the cost saving is positive, the line and traffic data associated with the best saving are used to eliminate the line, update the network, and recost the network.

The process is continually iterated for each updated network configuration until cost savings are no longer positive.

2.5 OUTPUT

The STACOM program generates a regular printer output and a CalComp plotter output. In addition, when the program is run as a demand job, run-status output will show up on the interactive terminal. This part of the printout provides information on the progress of the run.

Details of data contained in the regular printer output are given in Paragraph 2.5.1; Paragraph 2.5.2 describes the CalComp plot.

2.5.1 Printer

The printer output contains all the data resulting from the running of the STACOM program. The amount of printout data depends upon the number of system terminations operated and also upon the number of functions executed in each specific run.

Following is a list, in temporal order, of the data items which a run may produce.

2.5.1.1 Line Type and Transmission Line Characteristics. The first set of data are the line type and transmission line characteristics as used in the run. for each line type, the polling protocol data and modem turn-around time data, etc., are provided.

2.5.1.2 Message Characteristics. Message characteristics are the next set of data output from the program. They include average input message length, average output message length and overall average message length.

2.5.1.3 Preloading of System Terminations and Preselection of Regional Switcher Locations. If there are any preloadings of system terminations and/or pre-selections of switcher locations, this information will be provided in the printout. Otherwise, no data will be shown in this regard.

2.5.1.4 Traffic and Distance Tables. These are tables which show both traffic from/to all system terminations and distances between system terminations.

The first table gives the traffic data from each system termination to/from each data base; the next one gives the traffic data destined to and originating from each system termination. The last table shows the distance data between any two system terminations.

2.5.1.5 System Centroid and the Utilization Factor of the Central Switcher. The system centroid as designated by the user is printed next as a reminder. After this, the CPU utilization factor of the central switcher as calculated by the program is printed to indicate the load.

2.5.1.6 System Terminations in a Region and its Regional Switching Center. For each region, the program prints out the identification and name of each system termination in the region. These system terminations may have been pre-loaded or assigned to the region by the program. The program also prints out the location of the RSC for the region, which is either pre-assigned by the user or selected by the program.

2.5.1.7 Star Network and its Costs. After showing what system terminations are in the region, the program prints out the regional star networks and costs associated with each central link. It also provides summarized costs. Detailed descriptions for each central link are given below.

2.5.1.7.1 Line Configuration and Effective Utilization. The line configuration for each central link is printed as a column vector, which has the same number of line types used in the run. The effective line utilization is also printed to show the traffic load from the system termination.

2.5.1.7.2 Distance. The distance from the system termination to the regional switching center is printed.

2.5.1.7.3 Line Traffic and Effective Response Time. The amount of traffic from/to the system termination is printed before the effective line response time as calculated by the program is printed. The calculation is based on the line configuration and traffic as shown and should be better than the response time requirement.

2.5.1.7.4 Installation and Annual Recurring Costs. The installation and annual recurring costs for providing the central link are given in terms of chargeable items such as service terminal, modem, line and drop. Partial sums for the line are also printed. Finally, total installation and annual recurring costs for each chargeable item and for the overall star network are printed.

2.5.1.8 Final Optimized Network and its Costs. After performing optimizations on the star network, the program prints out descriptions for each multidrop line in the final optimized network. The following list shows the data items which may be printed.

2.5.1.8.1 Multidrop Line Configuration. Each multidrop line has an index, the beginning terminal and number of terminals on the line. The exact line configuration is printed as a column vector, with only one non-zero element. The content of that non-zero element must be one, due to the fact that multidropped terminals can only perform on one line.

2.5.1.8.2 Line Utilization, Mileage, Traffic, and Response Time. The line utilization, total mileage and incoming/outgoing traffic on each multidrop line are printed. The program next prints the average response time, which should be better than that required by design, to be expected by each user terminal on the line.

2.5.1.8.3 Installation and Annual Recurring Costs. The amount of installation and recurring costs are then listed in terms of chargeable items as explained in Section 2.5.1.7.4.

Finally, total installation and annual recurring costs for each chargeable item and for the overall network are printed.

2.5.1.9 Network Drawing. A network diagram in terms of tree-type relationship is last printed. It uses the system termination identification as nodal notation.

2.5.1.10 Initial Interregional Network. If formation and optimization of the interregion network is required, the program will perform these functions and print its initial and optimized network. For each pair of RSCs, the program prints out line names, configuration, utilization, and installation and recurring costs. Total network cost is also provided.

2.5.1.11 Optimized Interregional Network. The program prints out similar data for the final optimized interregional network after completing the network optimization.

2.5.2 CalComp Plot

A CalComp plot subroutine has been incorporated into the STACOM program for the purpose of providing a visual plot of each optimized regional network obtained by the optimization process. The subroutine converts each final optimized regional network into a two dimensional plot, utilizing the CalComp plotter. It should be noted that the CalComp plot is an optional product. If desired the user can command the STACOM program not to generate the plot.

2.6 SYSTEM CONFIGURATION

In this section, we will describe the basic computer system required to run the STACOM program.

2.6.1 Hardware

The following list describes the hardware units that should be part of the computer system on which the STACOM program is run.

2.6.1.1 Central Processing Unit. Due to the fact that the STACOM program is coded with the FORTRAN V language and compiled and mapped under the EXEC-8 operating system of the UNIVAC 1108 systems, a UNIVAC 1108 CPU or one equivalent to it is a prerequisite of the use of the STACOM program. When this type of CPU is not available, some conversion efforts on the STACOM program may be required.

2.6.1.2 Main Core Storage. Although the core size required by the STACOM program varies by parameter values assigned, it is generally true that 65K words would be a minimal requirement.

2.6.1.3 CalComp Pen Plotter. A CalComp pen plotter is required for the use of the STACOM program. If other types of CalComp plotters, e.g., CalComp Model 1675 are to be used, the plotting subroutine of the STACOM program needs to be revised.

2.6.1.4 Line Printer. A regular printer to receive FORTRAN output files is needed. It will print out all run results collected by file 100.

2.6.1.5 Demand Terminal. A demand terminal provides the user with an alternate way of running the STACOM program, although the program can be run as a batch job. With the demand terminal, a user can interactively perform the program execution.

SECTION 3

PROGRAM OPERATIONS

3.1 INTRODUCTION

This section is intended for use as a reference manual for the user, both to prepare input data and to operate the STACOM program. With this in mind, this section is devoted to an explanation of how input data are prepared, how the program is executed, and what the input/output of the program is to be.

3.2 ENVIRONMENT

3.2.1 Hardware

The following list describes the hardware units that should be part of the computer system on which the STACOM program is run.

3.2.1.1 Central Processing Unit (CPU). Because the STACOM program is coded with the FORTRAN language and compiled and mapped under the EXEC-8 operating system of the UNIVAC 1108 systems (see Paragraph 1.1), a UNIVAC 1108 CPU or one equivalent to it is a prerequisite for using the STACOM program. When this type of CPU is not available, some conversion effort on the STACOM program may be required.

3.2.1.2 Main Core Storage. Although the core size required by the STACOM program varies with the parameter values assigned, it is generally true that 65k words would be a minimal requirement.

3.2.1.3 CalComp Pen Plotter. A CalComp pen plotter is required for the use of the STACOM program. If other types of CalComp plotters, e.g., CalComp Model 1675, are to be used, the plotting subroutine of the STACOM program has to be revised.

3.2.1.4 Line Printer. A regular line printer to receive FORTRAN output files is needed. It is to print out all run results collected by file 100.

3.2.1.5 Demand Terminal. A demand terminal provides the user with an alternate way of running the STACOM program, although the program can be run as a batch job. With the demand terminal, a user can interactively perform the program execution.

3.2.2 Software

3.2.2.1 Programming Language. The STACOM Program is implemented with the FORTRAN V language of the UNIVAC system, compiled by the EXEC-8 FORTRAN Processor FOR, and mapped by the mapping processor MAP. Because of the inclusion of a plotting subroutine, the system library file LIB*PLOT$ is required during mapping.

An understanding of the FORTRAN V features is available in Reference 2.

3.2.2.2 Operating System. The EXEC-8 operating system of the UNIVAC 1108 computer system is used in the development of the STACOM program. As this operating system has been used for executing regular FORTRAN V programs this same operating system must be used for executing the current edition of the STACOM program.

The STACOM program has been designed so that all of the desired printer output will be dumped to file 100. Therefore, before executing the STACOM program, an alternate file 100 must be assigned. Otherwise, regular WRITE unit 6 will be the destination device; this will make it awkward when runs are performed via a demand terminal since most of the output from the program uses 132 characters per line.

Furthermore, an execution of the program will generate a punch-card image file. It is, therefore, recommended that a file be assigned to store the punch-card file, and that this later be directed to a CalComp plotter. An alternative is to have a command statement which requests the operating system to @SYM the output punch-card file to a CalComp pen plotter.

3.2.3 Functional Limitations

While the STACOM program has been designed and implemented with the intent that it be as widely applicable as possible, it does have certain limitations. Following is a list of functional limitations that exist in the program.

3.2.3.1 Program Size. Under the EXEC-8 operating system, the size for regular programs is limited to 65k words per program. Because of this, assignments of parameter values during the compilation stage are conditioned to this limit of the overall program size when mapped. Although it will be more convenient for later uses of the STACOM program if all of the parameters are assigned with maximum values within the limit of 65k words, this will increase the run cost. This is because of the core-time product charge.

3.2.3.2 Parameter Variables. The PARAMETER statement of the FORTRAN language is one of those commands which make the language a powerful tool in problem solving.

To accomplish the goal of making the STACOM program a widely usable tool for network design, it has been implemented with several parameter variables. For each compilation of the program, a set of values is assigned to the parameter variables. Therefore, any subsequent use of the STACOM program will be limited to cases where the actual values assigned to the variables are within the parameter values defined during compilation. Any run whose input data violates this rule will need modification of the parameter values of the program, recompilation, and remapping. For example, NP1 is a parameter variable which is used to make the number of system terminations allowed in a system a variable. A choice of NP1 as 105, for example, dictates that the STACOM program can only be used in systems where 105 or less system terminations are under consideration. Any run which has a number of system terminations greater than 105 will result in either an abnormal run termination or a normal run termination with unwanted output.

3.2.3.3 Response Time. The response time algorithm implemented in the program is based on the model (Reference 5). In applying this program to a given system, some consideration of the applicability of the response time algorithm is required. If the central switcher does not behave similar to this model the response time subroutine RSPNSE has to be revised and recompiled and the STACOM program has to be re-mapped.

3.2.3.4 CalComp Plot. The graphic output portion of the STACOM program has been implemented with the plotting routines designed for the CalComp pen plotter. If other types of CalComp plotters, e.g., CalComp Model 1675, are to be used, the plotting subroutine of this program needs to be revised and recompiled and the STACOM program has to be re-mapped.

3.3 RUN DESCRIPTION

3.3.1 Initialization and Setup

When the STACOM program is executed from an 80-character/line demand terminal, an alternate file, 100, to be used as a printer output file, must be defined. Otherwise, all printout data will be directed to the terminal which will produce interleaving output. The statement @ASG,UP 100 defines the alternate file.

In addition to the redirection of output file destination, the user has to direct the punch-card file to a proper unit for a CalComp plotter. As an example, the statement @SYM,P PUNCH$,,G9PLTF will direct the punch-card images to a CalComp plotter designated with G9PLTF.

The preparation of input data can be best described by referring to Table 3-1 which shows all of the data items with their required formats. The table is self-explanatory, but some of the data items deserve additional description.

Because the exact number of data bases varies from State to State, the format for item 5 allows a maximum of 5 data bases wherein the last three pairs of entries must be given on a separate card.

The notation $\lceil X \rceil$ for item 8 indicates that the exact value is equal to the next integer which is greater than or equal to X. The format for line recurring costs has been designed with the assumption that both linear and nonlinear functions will be used in tariffs for line services. Because of this, the STACOM program provides options for either scheme. When a cost function is nonlinear, it is assumed to be stepwise and only eight steps are allowed. If eight are not enough, the program has to be updated.

The amount of input data for item 15 varies from one run to another. The program has default values of zeros for all entries in IACTN (NR1,2). A zero for the first element indicates the acceptance of additional system terminations into a region when it is a preloaded region; a zero for the second element indicates that the optimization process for the region is not needed.

When a user decides either to exclude the addition of other system terminations into a preloaded region, or to request an optimization process performed upon a specific region he must so inform the STACOM program by adding data cards with two integer numbers. The first number gives the region index; the second number indicates the action: 1 indicates insertion exclusion, and 2 indicates optimization. When all requests for actions have been made, a card with two zeroes is required to indicate that fact.

Finally, item 20 provides the tool for a user to preload system terminations to certain regions, and/or preselect the regional switching center. Three numbers are needed for each action. The first number, called NCODE, directs the specified action: 1 assigns a system termination to a specific region; 2 assigns a system termination as the RSC for a specific region. The second number, called NSTATE, gives the identification number for a system termination to be assigned to a region or to be selected as an RSC. The third number, called NREGQ, designates the region to be acted upon. When the first number has a value of three, the assignment selection activity terminates.

3.3.2 Run Options

As indicated in Table 3-1, there are several independent variables provided only at the time of execution. This provides additional capabilities to the STACOM program.

Table 3-1. Formats for Input Data

Item No.	Item Description	Names of Internal Variables/Arrays	Number of Cards Needed	Formats
1.	No. of regions under consideration	NR1	1	(I3)
2.	No. of system terminations, no. of data bases, and no. of distinctive cities under consideration.	N1, N7, NCITY	1	(3I5)
3.	IDs for data bases	NBASE(N7)	1	(3(1X,A4))
4.	V-H Coordinates for cities	IVERT(NCITY), IHORZN(NCITY)	NCITY	(33X,I5,2X,I5)
5.	ID, name, city index, additional no. of terminals and traffic to/from each data base for each system termination.	INDXPT(N1),NAMEST(H,N1),IADD(N1) MAPADR(N1),TRAFD(N1,2,N7)	a. N1 if N7≤2 b. 2N1 if N7>2	(A4,1X,3A6,A4, I2,I4,4F10.2/ 6F10.2
6.	No. of rate structures	N2	1	(I3)
7.	Rate application matrix	IRATEJ(N2,N2)	N2	(10I2)
8.	Traffic density index and applicable rate structure for each city	IRAND(NCITY,2)	⌈NCITY/40⌉	(80I1)
9.	No. of applicable line types	N3	1	(I3)

Table 3-1. Formats for Input Data
(Continuation 1)

Item No.	Item Description	Names of Internal Variables/Arrays	Number of Cards Needed	Formats
10.	Name, capacity, utilization limit, usage and duplexing mode for each line	LINAME(N3), LINCAP(N3), LINMIX(N3), IDUPLX(N3)	N3	(A6,1X,I6,1X, F3.2,2(1X,I1))
11.	No. of chargeable items	N4	1	(I3)
12.	Names of chargeable items	NAMEHW(N4)	1	(10(A6,1X))
13.	Installation and recurring costs for chargeable items WRT rate structure, traffic density and duplexing mode for each line type	AINSTC(N2,N3,N4,3,2,2), RECRC (N2,N3,N4,3,2,2)	2xN2xN3xN4x3x2	(2F9.2)
14.	Linear installation and recurring costs for lines WRT rate structure type, density, and duplex mode	IFLAG(N2,N3),ANSTLN (N2,N3,3,2,2), RECRLN (N2,N3,3,2,16)	a. N2x(2+N3x3x2) b. 2xN2(2+N3x 3x2) if non linear	a. (4F9.2/I1/10F8.3) b. (4F9.2/I1/10F8.3/ 10F8.3)
15.	Action indices for regions	NREG,NCODE for IACTN (NR1,2)	Variable	(2I2)

Table 3-1. Formats for Input Data
(Continuation 2)

Item No.	Item Description	Names of Internal Variables/Arrays	Number of Cards Needed	Formats
16.	No. of polling characters no. of NAK characters, no. of polling overhead characters, no of NAK overhead characters, message overhead characters, Modem turnaround time, and other delay for each line type	NPL(N3), NAK(N3), NPLOH (N3) NAKOH(N3), MOH (N3) TAMDM(N3), TAPD (N3)	N3	(5I4,2F7.5)
17.	No. of message types	NTYP	1	(I4)
18.	Message name, input message length, output message length, input percentage and output percentage with priority 1 and 2	MSGNAM (NTYP), MSLIN (NTYP) MSGOVT(NTYP), RATIOI(NTYP,2) RATIO(NTYP,2)	NTYP	(A6,2(2I4,2F6.3))
19.	Average CPU service time per transaction	CPUAVG	1	(F7.4)
20.	Preloading system terminations and/or preselecting regional centers	NCODE, NSTATE, NREGQ	Variable	(I1,1X,A4,A5)
21.	System Centroid	NSCC1	1	(A4)

Table 3-1. Formats for Input Data
(Continuation 3)

Item No.	Item Description	Names of Internal Variables/Arrays	Number of Cards Needed	Formats
22.	Total no. of messages per second and no. of requests made at the central switcher	XSAC, NREQSW	1	(F8.5,I3)
23.	Limit on no. of terminals on a multi-dropped line, response time requirement and no. of CPU processors for computer	NTERMS, TIMREQ, MPROC	1	(I3,F5.2,I2)
24.	Plot request	MPLOT	1	(I3)

Following is a list of run options for the STACOM program.

(1) The user can preload system terminations to regions and/or preselect regional switching centers.

(2) The user can select certain regions for which the optimization process will be performed.

(3) Type of lines and chargeable items can be selectively chosen.

(4) The user can put a limit on the number of terminals on a multidrop line as described and can limit the average terminal response time.

(5) The number of central processor units in the central computer system can be 1, 2, or 4.

(6) The CalComp plot can be skipped.

3.3.3 Control Instruction and Sequences

3.3.3.1 Starting a Run.

3.3.3.1.1 Batch Mode. Following is a list of control statements required when running the STACOM program as a batch run:

```
@RUN run-ID, account-no., project-ID, SUP-time, pages/cards
@ASG,UP 100
@SYM,P PUNCH$,,plotter-ID
@XQT file.STACOM
      (DATA)
@BRKPT 100
@FREE 100
@SYM 100,,printer-ID
@FIN
```

The RUN card gives the following information: designated run-ID, user's account number, project-ID, expected SUP-time usage and limited number of printer pages, and number of cards which may be generated from the run. Plotter-ID gives the logical ID of the CalComp pen plotter and file is the file which contains the absolute element of the STACOM program. Printer-ID gives the logical ID of the regular printer. DATA as shown is the input data described in Paragraph 3.2.1; the user should arrange the data in the same order. When all of these data items are in order and ready, the deck can be submitted to the operator at the computer site for processing.

3.3.3.1.2 Demand Mode. If program execution is to be performed via a demand terminal, the user can converse interactively with the program. The user may also run the program as a batch job by having all input data prepared and added after the @XQT statement.

Under the conversation mode, the user acts as a respondent who answers the requests for data made by the program. This mode of operation provides the user with some understanding of program progress A user can very often terminate a run before a complete set of input data is given. This is possible because the user has some knowledge of the progress being made. This capability can prevent the user from an unnecessary waste of time. For example, if a run encounters a system which has more oversized distance data than allowed, a message from the program will be printed out on the terminal. The user will be alarmed by this fact and may decide to terminate the program run.

3.3.3.2 Run Progression. After receiving all of the required data, the program will perform all functions as designed and requested by the user whether a batch or a demand job has been executed. The program will perform formation of regions, selection of regional switching centers, formation of a regional star network and its optimization if requested, and finally, formation and optimization of an interregional network. All of the desired output data will go to the alternate file, 100.

3.3.3.3 Normal Termination. When a STACOM program run proceeds successfully and terminates normally, the normal file unit 6 will contain two lines of messages for each successful regional network optimization. These two lines are:

(1) TRYLNK has been accessed for xxxxx times.

(2) UPNETW has been accessed for xxxxx times.

The first message indicates the number of subroutine calls to TRYLNK that have been made during the process of searching for a better network. The second message indicates the number of optimization cycles which the run has gone through before the optimization process stops. After a normal termination, the user can direct the output file 100 to a printer device and the punch card file to a CalComp pen plotter if file PUNCH$ has been directed to an alternate file.

3.3.3.4 Aborting and Recovering a Run. When a run encounters trouble resulting from incorrect input data, the user can use the normal aborting procedure to terminate its execution if it is a demand job. A statement of @@x after interrupting the line communication by pressing the BREAK key will terminate a program execution at any time. On the other hand, the EXEC-8 may abort a program execution when certain serious violations occur during its execution, e.g., number of punch cards exceeding the limit on the run card.

If a program run has been interrupted due to system outage, no recovery of the run is possible.

3.3.4 Program Listing

A listing of the STACOM program elements is given in Appendix A.

3.4 SAMPLE RUN

To illustrate how STACOM can be run, a sample run is provided in the following subsections along with detailed explanations.

3.4.1 Run Stream

The following list of control statements shows the typical batch mode run stream used to execute the STACOM program.

```
@RUN JJL,J6G3YL,51928,20,90/1000
@ASG,UP 100
@SYM,P PUNCH$,,G9PLTF
@XQT LEE.STACOM
@ADD LEE.DATA
@BRKPT 100
@FREE 100
@SYM,U 100,,T4
@FIN
```

The first control statement is a run request which specifies its run ID as JJL, identifies its account number as J6G3YL, assigns project ID as 51928, requests a maximum of 20 minutes of SUP-time and finally asks for a limit of 90 printer pages and 1,000 punch cards. The limits on SUP-time, number of printer pages and number of punch cards deserve some attention when making a run. If there is an underestimate in any of these three limits, the run may abort due to insufficient resource assignment.

The second statement is used to assign an alternate FORTRAN output file as required by the program. It is intended to be a one-day file.

Statement 3 requests the system to direct the punch card image file to the CalComp pen plotter with the name G9PLTF.

Statement 4 is a command for executing the STACOM program which is designated with the element name STACOM in file LEE.

The next statement asks the operating system to use the content of element DATA as its input data.

Statements 6 and 7 are used to close file 100 and catalog it for later use.

Statement 8 asks the operating system to send the printer file 100 to an on-site low speed printer with ID T4. The U option retains the FORTRAN print file after a copy is printed.

The last statement terminates the run with a request for a detailed description of run charges and run history. The number of pages in the print file and the number of punch-card images are part of the data given by the accounting subsystem when a run terminates.

When the same program is to be executed via a demand terminal, the content of element LEE.DATA can be divided into several individual elements plus certain key-in control statements. Essentially, however, the same amount of input data must be provided to succeed in running the program.

3.4.2 Input

As a specific example, Table 3-2 gives the list of data which have been used in analyzing the South Plains portion of Texas under the Council-of-Governments structure.

Encircled numbers have been written on the left hand side so that Table 3-2 and Table 3-1 are made compatible. Data associated with each encircled number in Table 3-2 corresponds to the data item with the same index in Table 3-1. Items 1 and 2 indicate that the run is concerned with 1 region case, a total of 25 system terminations, 4 data bases, and a total of 358 distinctive cities. Items 3 gives the IDs for locations of those four data bases, and item 4 lists the names of all 358 cities which have distinctive V-H coordinates (four digit integers). Since the number of data bases is greater than 2, two input cards are needed for each system termination; therefore a total of 50 cards are needed as listed under item 5. Since there is only one rate structure, one card is needed for rate application matrix (see items 6 and 7).

Item 8 shows the traffic density and rate application table for which 9 cards are required. Items 9 and 10 indicate that only 3 type of lines (with rates 1200 bps, 2400 bps, and 4800 bps) are considered; 0.7 is the line utilization limit for all of them. Three chargeable items are applicable as shown in items 11 and 12. Item 13 is somewhat complicated, the following explanation should enable the reader to understand it. These 108 data cards are divided into 3 groups with the first group given to the first line type, i.e., 1200 bps, and so on.

Table 3-2. Input Data for the Example Run

```
51928*STACOM(0).INPUT/0777
     1 (1)---->1
     2 (2)---->25    4  358
     3 (3)---->AAAA DDDD SSSS HHHH
     4 (4)---->PALESTINE              ANDERSON    8558   3750
     5         ANDREWS                ANDREWS     8897   4993
     6         LUFKIN                 ANGELINA    8575   3561
     7         ROCKPORT               ARANSAS     9405   3694
     8         ARCHER CITY            ARCHER      8396   4410
     9         JOURDANTON             ATASCOSA    9332   4032
    10         BELLVILLE              AUSTIN      8964   3710
    11         MULESHOE               BAILEY      8518   5157
    12         SEYMOUR                BAYLOR      8437   4518
    13         BEEVILLE               BEE         9378   3850
    14         BELTON                 BELL        8827   4010
    15         FORT HOOD              BELL        8832   4070
    16         HARKER HEIGHTS         BELL        8832   4063
    17         KILLEEN                BELL        8832   4063
    18         NOLANVILLE             BELL        8832   4038
    19         TEMPLE                 BELL        8812   3992
    20         ALAMO HEIGHTS          BEXAR       9225   4062
    21         FT SAM HOUSTON         BEXAR       9225   4062
    22         LEON VALLEY            BEXAR       9223   4092
    23         SAN ANTONIO            BEXAR       9225   4062
    24         UNIVERSAL CITY         BEXAR       9187   4037
    25         CLIFTON                BOSQUE      8690   4089
    26         MERIDIAN               BOSQUE      8668   4112
    27         TEXARKANA              BOWIE       8111   3626
    28         ALVIN                  BRAZORIA    8996   3488
    29         ANGLETON               BRAZORIA    9059   3499
    30         CLUTE                  BRAZORIA    9081   3487
    31         FREEPORT               BRAZORIA    9096   3466
    32         LAKE JACKSON           BRAZORIA    9081   3487
    33         PEARLAND               BRAZORIA    8970   3506
    34         BRYAN                  BRAZOS      8827   3788
    35         COLLEGE STATION        BRAZOS      8827   3788
    36         ALPINE                 BREWSTER    9364   5057
    37         FALFURRIAS             BROOKS      9645   3827
    38         BROWNWOOD              BROWN       8797   4327
    39         CALDWELL               BURLESON    8880   3834
    40         PORT LAVACA            CALHOUN     9258   3665
    41         BROWNSVILLE            CAMERON     9861   3606
    42         HARLINGEN              CAMERON     9820   3663
    43         PORT ISABEL            CAMERON     9807   3565
    44         SAN BENITO             CAMERON     9826   3648
    45         LINDEN                 CASS        8217   3643
    46         DIMMIT                 CASTRO      8427   5109
    47         ANAHUAC                CHAMBERS    8884   3418
    48         JACKSONVILLE           CHEROKEE    8492   3709
    49         CHILDRESS              CHILDRESS   8328   4743
    50         MORTON                 COCHRAN     8622   5129
    51         ROBERT LEE             COKE        8857   4603
    52         COLEMAN                COLEMAN     8804   4413
    53         FRISCO                 COLLIN      8354   4069
    54         MCKINNEY               COLLIN      8340   4038
    55         PLANO                  COLLIN      8383   4037
    56         WELLINGTON             COLLNGSWH   8240   4776
```

Table 3-2. Input Data for the Example Run
(Continuation 1)

57	COLUMBUS	COLORADO	9032	3740
58	NEW BRAUNFELS	COMAL	9145	4018
59	COMANCHE	COMANCHE	8735	4275
60	GAINESVILLE	COOKE	8289	4162
61	COPPERAS COVE	CORYELL	8844	4092
62	GATESVILLE	CORYELL	8771	4089
63	CRANE	CRANE	9073	4896
64	OZONA	CROCKETT	9144	4642
65	DALHART	DALLAM	8129	5249
66	ADDISON	DALLAS	8404	4048
67	CEDAR HILL	DALLAS	8485	4047
68	DALLAS	DALLAS	8436	4034
69	DESOTO	DALLAS	8478	4030
70	DUNCANVILLE	DALLAS	8469	4044
71	FARMERS BRANCH	DALLAS	8414	4062
72	GARLAND	DALLAS	8400	4018
73	GRAND PRAIRIE	DALLAS	8458	4066
74	HIGHLAND PARK	DALLAS	8436	4034
75	IRVING	DALLAS	8440	4064
76	LANCASTER	DALLAS	8470	4013
77	MESQUITE	DALLAS	8426	4000
78	RICHARDSON	DALLAS	8399	4035
79	SEAGOVILLE	DALLAS	8447	3980
80	UNIVERSITY PARK	DALLAS	8436	4034
81	LAMESA	DAWSON	8779	4919
82	HEREFORD	DEAF SMTH	8378	5143
83	DENTON	DENTON	8372	4127
84	LEWISVILLE	DENTON	8398	4089
85	CUERO	DEWITT	9209	3823
86	SPUR	DICKENS	8560	4784
87	SAN DIEGO	DUVAL	9542	3688
88	EASTLAND	EASTLAND	8649	4352
89	ODESSA	ECTOR	8982	4930
90	ENNIS	ELLIS	8515	3970
91	WAXAHACHIE	ELLIS	8517	4011
92	EL PASO	EL PASO	9231	5655
93	STEPHENVILLE	ERATH	8645	4232
94	MARLIN	FALLS	8739	3931
95	BONHAM	FANNIN	8234	3996
96	RORY	FISHER	8679	4646
97	FLOYDADA	FLOYD	8486	4902
98	FAIRFIELD	FREESTONE	8602	3839
99	PEARSALL	FRIO	9374	4129
100	RICHMOND	FT BEND	9009	3598
101	ROSENBERG	FT BEND	9009	3598
102	SEMINOLE	GAINES	8822	5040
103	FRIENDSWOOD	GALVESTON	8969	3489
104	GALVESTON	GALVESTON	8985	3397
105	HITCHCOCK	GALVESTON	8992	3441
106	LA MARQUE	GALVESTON	8975	3424
107	LEAGUE CITY	GALVESTON	8967	3468
108	TEXAS CITY	GALVESTON	8975	3424
109	POST	GARZA	8650	4854
110	FREDERICKSBURG	GILLESPIE	9079	4196
111	GOLIAD	GOLIAD	9301	3807
112	GONZALES	GONZALES	9137	3884
113	PAMPA	GRAY	8148	4952

Table 3-2. Input Data for the Example Run (Continuation 2)

114	DENISON	GRAYSON	8225	4069
115	SHERMAN	GRAYSON	8253	4072
116	GLADEWATER	GREGG	8354	3698
117	KILGORE	GREGG	8379	3674
118	LONGVIEW	GREGG	8348	3660
119	NAVASOTA	GRIMES	8865	3715
120	SEGUIN	GUADALUPE	9161	3981
121	PLAINVIEW	HALE	8465	4981
122	HAMILTON	HAMILTON	8744	4177
123	SPEARMAN	HANSFORD	8026	5037
124	QUANAH	HARDEMAN	8324	4654
125	KOUNTZE	HARDIN	8735	3405
126	SILSBEE	HARDIN	8730	3380
127	BAYTOWN	HARRIS	8916	3466
128	BELLAIRE	HARRIS	8938	3536
129	DEER PARK	HARRIS	8929	3491
130	GALENA PARK	HARRIS	8938	3536
131	HOUSTON	HARRIS	8938	3536
132	HUMBLE	HARRIS	8881	3540
133	JACINTO CITY	HARRIS	8938	3536
134	JERSEY VILLAGE	HARRIS	8925	3581
135	KATY	HARRIS	8965	3618
136	LA PORTE	HARRIS	8829	3470
137	PASADENA	HARRIS	8938	3536
138	SEABROOK	HARRIS	8945	3462
139	SOUTH HOUSTON	HARRIS	8938	3536
140	SOUTHSIDE PLACE	HARRIS	8938	3536
141	SPRING VALLEY	HARRIS	8938	3536
142	TOMBALL	HARRIS	8889	3609
143	VILLAGE	HARRIS	8938	3536
144	WEBSTER	HARRIS	8967	3468
145	WEST UNIV PL	HARRIS	8938	3536
146	MARSHALL	HARRSN	8311	3602
147	HASKELL	HASKELL	8555	4567
148	SAN MARCOS	HAYS	9096	4001
149	CANADIAN	HEMPHILL	8036	4882
150	ATHENS	HENDERSON	8484	3826
151	DONNA	HIDALGO	9849	3728
152	EDINBURG	HIDALGO	9830	3758
153	HIDALGO	HIDALGO	9856	3764
154	MCALLEN	HIDALGO	9856	3764
155	MERCEDES	HIDALGO	9845	3701
156	MISSION	HIDALGO	9861	3781
157	PHARR	HIDALGO	9854	3754
158	WESLACO	HIDALGO	9847	3716
159	HILLSBORO	HILL	8612	4026
160	LEVELLAND	HOCKLEY	8629	5053
161	SULPHUR SPRINGS	HOPKINS	8281	3861
162	BIG SPRING	HOWARD	8847	4800
163	COMMERCE	HUNT	8280	3921
164	GREENVILLE	HUNT	8317	3949
165	BORGER	HUTCHNSON	8146	5033
166	JACKSBORO	JACK	8442	4303
167	EDNA	JACKSON	9186	3698
168	BEAUMONT	JEFFERSON	8777	3344
169	NEDERLAND	JEFFERSON	8789	3316
170	PORT ARTHUR	JEFFERSON	8806	3298

Table 3-2. Input Data for the Example Run
(Continuation 3)

171	ALICE	JIM WELLS	9533	3855
172	BURLESON	JOHNSON	8522	4103
173	CLEBURNE	JOHNSON	8563	4102
174	ANSON	JONES	8647	4563
175	STAMFORD	JONES	8603	4562
176	KARNES CITY	KARNES	9294	3915
177	KAUFMAN	KAUFMAN	8442	3936
178	TERRELL	KAUFMAN	8410	3943
179	BOERNE	KENDALL	9168	4133
180	JAYTON	KENT	8589	4718
181	KERRVILLE	KERR	9143	4226
182	JUNCTION	KIMBLE	9097	4373
183	KINGSVILLE	KLEBERG	9566	3801
184	BENJAMIN	KNOX	8472	4609
185	PARIS	LAMAR	8173	3897
186	LITTLEFIELD	LAMB	8558	5069
187	OLTON	LAMB	8490	5054
188	LAMPASAS	LAMPASAS	8875	4137
189	HALLETTSVILLE	LAVACA	9114	3789
190	YOAKUM	LAVACA	9157	3814
191	CENTERVILLE	LEON	8682	3768
192	CLEVELAND	LIBERTY	8801	3540
193	LIBERTY	LIBERTY	8835	3463
194	MEXIA	LIMESTONE	8635	3889
195	GEORGE WEST	LIVE OAK	9419	3910
196	LUBBOCK	LUBBOCK	8598	4962
197	SLATON	LUBBOCK	8616	4916
198	TAHOKA	LYNN	8680	4924
199	MADISONVILLE	MADISON	8740	3733
200	JEFFERSON	MARION	8267	3618
201	BAY CITY	MATAGORDA	9135	3578
202	EAGLE PASS	MAVERICK	9505	4370
203	BRADY	MCCULLOCH	8938	4344
204	BELLMEAD	MCLENNAN	8706	3993
205	BEVERLY HILLS	MCLENNAN	8706	3993
206	WACO	MCLENNAN	8706	3993
207	WOODWAY	MCLENNAN	8706	3993
208	HONDO	MEDINA	9285	4174
209	MENARD	MENARD	9011	4407
210	MIDLAND	MIDLAND	8934	4888
211	CAMERON	MILAM	8835	3910
212	ROCKDALE	MILAM	8877	3898
213	COLORADO CITY	MITCHELL	8781	4706
214	BOWIE	MONTAGUE	8351	4275
215	MONTAGUE	MONTAGUE	8323	4261
216	CONROE	MONTGOMRY	8832	3600
217	DUMAS	MOORE	8141	5144
218	DAINGERFIELD	MORRIS	8240	3704
219	NACOGDOCHES	NACOGDCHS	8515	3569
220	CORSICANA	NAVARRO	8553	3921
221	SWEETWATER	NOLAN	8737	4632
222	CORPUS CHRISTI	NUECES	9475	3739
223	ROBSTOWN	NUECES	9496	3786
224	PERRYTON	OCHILTREE	7962	4987
225	VEGA	OLDHAM	8292	5177
226	ORANGE	ORANGE	8746	3281
227	MINERAL WELLS	PARKER	8520	4261

Table 3-2. Input Data for the Example Run
(Continuation 4)

228	CARTHAGE	PANOLA	8385	3564
229	WEATHERFORD	PARKER	8508	4206
230	FARWELL	PARMER	8503	5221
231	FRIONA	PARMER	8432	5185
232	FORT STOCKTON	PECOS	9207	4954
233	AMARILLO	POTTER	8266	5076
234	CANYON	RANDALL	8317	5075
235	CLARKSVILLE	RED RIVER	8147	3809
236	PECOS	REEVES	9136	5101
237	BIG LAKE	REGAN	9062	4723
238	HEARNE	ROBERTSON	8802	3846
239	ROCKWALL	ROCKWALL	8384	3989
240	BALLINGER	RUNNELS	8855	4498
241	HENDERSON	RUSK	8420	3640
242	ARANSAS PASS	SAN PTRCO	9437	3700
243	GREGORY	SAN PTRCO	9455	3731
244	INGLESIDE	SAN PTRCO	9447	3711
245	PORTLAND	SAN PTRCO	9455	3731
246	SINTON	SAN PTRCO	9436	3777
247	ELDORADO	SCHLEICHR	9076	4547
248	SNYDER	SCURRY	8718	4737
249	STRATFORD	SHERMAN	8049	5194
250	TYLER	SMITH	8417	3744
251	BRECKENRIDGE	STEPHENS	8582	4394
252	STERLING CITY	STERLING	8900	4686
253	ASPERMONT	STONEWALL	8589	4650
254	TULIA	SWISHER	8397	5016
255	ARLINGTON	TARRANT	8472	4085
256	BEDFORD	TARRANT	8447	4092
257	COLLEYVILLE	TARRANT	8447	4117
258	CROWLEY	TARRANT	8518	4118
259	EULESS	TARRANT	8447	4092
260	FORREST HILL	TARRANT	8479	4122
261	FORT WORTH	TARRANT	8479	4122
262	GRAPEVINE	TARRANT	8425	4094
263	HALTOM CITY	TARRANT	8479	4122
264	HURST	TARRANT	8447	4117
265	LAKE WORTH	TARRANT	8471	4158
266	NO RICHLAND HLS	TARRANT	8447	4117
267	RICHLAND HILLS	TARRANT	8479	4122
268	SOUTHLAKE	TARRANT	8425	4094
269	WHITE SETTLMNT	TARRANT	8485	4153
270	ABILENE	TAYLOR	8698	4513
271	SANDERSON	TERRELL	9333	4816
272	BROWNFIELD	TERRY	8705	5007
273	MT PLEASANT	TITUS	8234	3755
274	SAN ANGELO	TOM GREEN	8944	4563
275	AUSTIN	TRAVIS	9005	3996
276	GILMER	UPSHUR	8317	3716
277	RANKIN	UPTON	9084	4811
278	UVALDE	UVALDE	9357	4279
279	DEL RIO	VAL VERDE	9399	4490
280	CANTON	VAN ZANDT	8414	3858
281	VICTORIA	VICTORIA	9245	3748
282	HUNTSVILLE	WALKER	8758	3652
283	HEMPSTEAD	WALLER	8923	3691
284	MONAHANS	WARD	9066	5005

Table 3-2. Input Data for the Example Run
(Continuation 5)

285	BRENHAM	WASHINGTN	8932	3752
286	LAREDO	WEBB	9681	4099
287	PIERCE	WHARTON	9115	3649
288	WHARTON	WHARTON	9078	3630
289	SHAMROCK	WHEELER	8170	4808
290	BURKBURNETT	WICHITA	8290	4440
291	WICHITA FALLS	WICHITA	8326	4413
292	VERNON	WILBARGER	8326	4567
293	RAYMONDVILLE	WILLACY	9768	3703
294	FLORESVILLE	WILSON	9261	3979
295	KERMIT	WINKLER	9024	5060
296	DECATUR	WISE	8399	4205
297	WINNSBORO	WOOD	8295	3794
298	DENVER CITY	YOAKUM	8781	5088
299	GRAHAM	YOUNG	8492	4365
300	OLNEY	YOUNG	8450	4414
301	CRYSTAL CITY	ZAVALA	9466	4246
302	ARANSAS PASS	ARANSAS	9437	3700
303	PLEASANTON	ATASCOSA	9320	4027
304	BANDERA	BANDERA	9205	4190
305	BASTROP	BASTROP	9007	3909
306	MARBLE FALLS	BURNET	8980	4115
307	LOCKHART	CALDWELL	9077	3954
308	LULING	CALDWELL	9117	3933
309	BAIRD	CALLAHAN	8688	4450
310	PITTSBURG	CAMP	8264	3742
311	PANHANDLE	CARSON	8210	5009
312	ATLANTA	CASS	8182	3618
313	RUSK	CHEROKEE	8515	3672
314	HENRIETTA	CLAY	8323	4354
315	CROSBYTON	CROSBY	8548	4862
316	CARROLLTON	DALLAS	8410	4066
317	COOPER	DELTA	8241	3896
318	CARRIZO SPRS.	DIMMIT	9500	4240
319	CISCO	EASTLAND	8662	4377
320	LA GRANGE	FAYETTE	9016	3813
321	MT VERNON	FRANKLIN	8246	3801
322	ANDERSON	GRIMES	8836	3708
323	MEMPHIS	HALL	8287	4821
324	ALAMO	HIDALGO	9854	3754
325	GRANBURY	HOOD	8572	4178
326	CROCKETT	HOUSTON	8634	3685
327	STINNETT	HUTCHNSON	8117	5054
328	JASPER	JASPER	8603	3399
329	GROVES	JEFFERSON	8789	3316
330	PORT NECHES	JEFFERSON	8789	3316
331	COTULLA	LASALLE	9476	4120
332	GIDDINGS	LEE	8968	3848
333	GROESBECK	LIMESTONE	8671	3886
334	LLANO	LLANO	8970	4199
335	NEWTON	NEWTON	8600	3353
336	BRIDGE CITY	ORANGE	8774	3295
337	VIDOR	ORANGE	8761	3334
338	PALO PINTO	PLO PINTO	8541	4291
339	LIVINGSTON	POLK	8716	3543
340	REFUGIO	REFUGIO	9365	3757
341	FRANKLIN	ROBERTSON	8766	3839

Table 3-2. Input Data for the Example Run
(Continuation 6)

```
342       HEMPHILL              SABINE      8511   3413
343       SAN AUGUSTINE         SN AUGUST   8491   3471
344       COLDSPRING            SN JACINT   8754   3567
345       SAN SABA              SAN SABA    8886   4242
346       MATHIS                SAN PTRCO   9448   3840
347       CENTER                SHELBY      8443   3505
348       RIO GRANDE CTY        STARR       9861   3887
349       BENBROOK              TARRANT     8499   4140
350       EVERMAN               TARRANT     8505   4110
351       RIVEROAKS             TARRANT     8479   4122
352       GROVETON              TRINITY     8661   3605
353       WOODVILLE             TYLER       8664   3458
354       EL CAMPO              WHARTON     9115   3649
355       WHEELER               WHEELER     8126   4829
356       IOWA PARK             WICHITA     8327   4445
357       GEORGETOWN            WILLIAMSO   8927   4014
358       TAYLOR                WILLIAMSO   8922   3962
359       QUITMAN               WOOD        8340   3806
360       PLAINES               YOAKUM      8735   5105
361       ZAPATA                ZAPATA      9786   4009
362 (5)-> AZLI MULESHOE PD                    8      7.35     19.23     .00       .00
363             .00       .00       .00       .00
364       AZKK MORTON SO                     47      4.38     10.58     .00       .00
365             .00       .00       .00       .00
366       AZKW SPUR PD                       83      6.28     17.80     .00       .00
367             .00       .00       .00       .00
368       AZX7 FLOYDADA SO                   94      5.60     11.91     .00       .00
369             .00       .00       .00       .00
370       AZLQ POST SO                      106      2.86      8.16     .00       .00
371             .00       .00       .00       .00
372       AZLA PLAINVIEW PD                 118     25.27     55.61     .00       .00
373             .00       .00       .00       .00
374       AZLB PLAINVIEW SO                 118      7.67     14.67     .00       .00
375             .00       .00       .00       .00
376       AZLD LEVELLAND PD                 157     10.45     27.80     .00       .00
377             .00       .00       .00       .00
378       AZLC LITTLEFIELD PD               183      8.87     21.94     .00       .00
379             .00       .00       .00       .00
380       AZKA LITTLEFIELD SO               183      4.57     11.05     .00       .00
381             .00       .00       .00       .00
382       AZTI OLTON PD                     184      5.36     16.01     .00       .00
383             .00       .00       .00       .00
384       AZGL LUBBOCK DPS                  193     31.12     98.55     .00       .00
385             .00       .00       .00       .00
386       AZLK LUBBOCK PD                   193    242.28    340.68     .00       .00
387             .00       .00       .00       .00
388       AZLL LUBBOCK SO                   193     39.61     59.13     .00       .00
389             .00       .00       .00       .00
390       AZLP SLATON PD                    194     10.01     23.71     .00       .00
391             .00       .00       .00       .00
392       AZLJ TAHOKA PD                    195      5.72     17.21     .00       .00
393             .00       .00       .00       .00
394       AZKS TAHOKA SO                    195      4.03      9.94     .00       .00
395             .00       .00       .00       .00
396       AZLF BROWNFIELD PD                269     17.09     40.89     .00       .00
397             .00       .00       .00       .00
398       AZLF BROWNFIELD SO                269      4.03      8.36     .00       .00
```

Table 3-2. Input Data for the Example Run
(Continuation 7)

```
399              .00        .00        .00        .00
400      AZLN DENVER CITY PD              295       7.39      20.87       .00       .00
401              .00        .00        .00        .00
402      NAAE MULESHOE S.O.                 8       1.66       5.26       .00       .00
403              .00        .00        .00        .00
404      NABG CROSBYTON S.O.              312       1.66       5.26       .00       .00
405              .00        .00        .00        .00
406      NACG LEVELLAND S.O.              157       2.68       8.48       .00       .00
407              .00        .00        .00        .00
408      NAFA PLAINES S.O.                357       1.53       4.86       .00       .00
409              .00        .00        .00        .00
410      AAA, AUSTIN SWITCHER             272         .0         .0         .0        .0
411              .0         .0         .0         .0         .0
412  (6)→1
413  (7)→1
414  (8)→10101010101010101010111010111011101010111010101110111010101011101110111010111110
415      10101010101110101010101010101010101010101010101011101010101010101010101010101011
416      10101010101110101011010101010101011101010111010101110101010110101110111111101111101
417      10101010101010111010101010101010101010101010101010101011110111010111110101011111
418      11101010111011101010101010101011010101011101110101010111010101010111011101101010101
419      10101110101011101011101011101011111111110101010101010101010111111101010101010101010
420      10101010101011101010101011101011011010101010101010111010111111110101010101011110
421      10111110111010111110101010101010101010101010101010101010111110101010101010101010
422      10101010101010101010101011010101010101010101010101010101010101010101010101010
423  (9)→3    @N3=3, I.E., 150,1200,2400
424 (10)→1.2KB    1200 .7  1 1
425      2.4KB    2400 .7  1 1
426      4.8KB    4800 .7  1 1
427 (11)→3   @N4=3 AS NO. OF H/W UNITS ASSOCIATE WITH EACH SYS. TEMN.
428 (12)→SER.T. MODEM  DROP   @NAMES
429 (13)→10.      10.
430          10.      10.
431          15.      15.
432          15.      15.
433          10.      10.
434          10.      10.
435          15.      15.
436          15.      15.
437          10.      10.
438          10.      10.
439          15.      15.
440          15.      15.
441          50.      50.
442          50.      50.
443          22.      22.
444          22.      22.
445          50.      50.
446          50.      50.
447          22.      22.
448          22.      22.
449          50.      50.
450          50.      50.
451          22.      22.
452          22.      22.
453           0.       0.
454           0.       0.
455          10.      10.
```

Table 3-2. Input Data for the Example Run
(Continuation 8)

456	10.	10.
457	0.	0.
458	0.	0.
459	10.	10.
460	10.	10.
461	0.	0.
462	0.	0.
463	10.	10.
464	10.	10.
465	10.	10.
466	10.	10.
467	15.	15.
468	15.	15.
469	10.	10.
470	10.	10.
471	15.	15.
472	15.	15.
473	10.	10.
474	10.	10.
475	15.	15.
476	15.	15.
477	100.	100.
478	100.	100.
479	54.	54.
480	54.	54.
481	100.	100.
482	100.	100.
483	54.	54.
484	54.	54.
485	100.	100.
486	100.	100.
487	54.	54.
488	54.	54.
489	0.	0.
490	0.	0.
491	10.	10.
492	10.	10.
493	0.	0.
494	0.	0.
495	10.	10.
496	10.	10.
497	0.	0.
498	0.	0.
499	10.	10.
500	10.	10.
501	10.	10.
502	10.	10.
503	15.	15.
504	15.	15.
505	10.	10.
506	10.	10.
507	15.	15.
508	15.	15.
509	10.	10.
510	10.	10.
511	15.	15.
512	15.	15.

Table 3-2. Input Data for the Example Run
(Continuation 9)

```
513          100.      100.
514          100.      100.
515          135.      135.
516          135.      135.
517          100.      100.
518          100.      100.
519          135.      135.
520          135.      135.
521          100.      100.
522          100.      100.
523          135.      135.
524          135.      135.
525            0.        0.
526            0.        0.
527           10.       10.
528           10.       10.
529            0.        0.
530            0.        0.
531           10.       10.
532           10.       10.
533            0.        0.
534            0.        0.
535           10.       10.
536           10.       10.
537 (14)-->1      @LINEAR FOR 1200 BAUD LINE
538          900.         .0    900.         .0
539            1.        3.       1.        3.
540          900.         .0    900.         .0
541            1.        3.       1.        3.
542          900.         .0    900.         .0
543            1.         .6      1.         .0
544       1      @LINEAR FOR 2400 BAUD LINE
545          900.         .0    900.         .0
546            1.        3.       1.        3.
547          900.         .0    900.         .0
548            1.        3.       1.        3.
549          900.         .0    900.         .0
550            1.         .6      1.         .0
551       1      @LINEAR FOR 4800 BAUD LINE
552          900.         .0    900.         .0
553            1.        3.       1.        3.
554          900.         .0    900.         .0
555            1.        3.       1.        3.
556          900.         .0    900.         .0
557            1.         .6      1.         .0
558 (15)-->1 2
559       0 0
560  (16)-->3   2   0   0   8 .008   .0
561          3   2   0   0   8 .008   .0
562          3   2   0   0   8 .050   .0
563 (17)-->11
564  (18)-->LIDR   35 300 4.26  3.94    0   0  .0    .0
565       TCIC   60  861 5.25 14.47    0   0  .0    .0
566       ADM   500 500 3.62  3.62    0   0  .0    .0
567      G-CODE 300 300  .13  8.43    0   0  .0    .0
568       CCH   426 459 5.78  5.78    0   0  .0    .0
569       MVD    50 175 9.34  9.34    0   0  .0    .0
```

Table 3-2. Input Data for the Example Run
(Continuation 10)

```
570        INQ/NL  50 200  .45   .45    0   0  .0    .0
571        ADM/NL 300 300  .45   .0     0   0  .0    .0
572         NCIC   50  90  .0   9.21    0   0  .0    .0
573        DB/DLS  90  50 3.49  1.1     0   0  .0    .0
574        DB/ANT  90  50  .45   .45    0   0  .0    .0
575   (19)-->.110
576 (20)-->2 AAAA     1   @SWITCHER ASSINGMENT WITH I1,1X,A4,I5
577        3          @TERMINATE SWITCHER ASSIGNMENT
578   (21)->AAAA    @STATE CENTER
579 (22)-->6.43      2     @TOTAL XSAC & REQ. AT THE AUSTIN SWITCHER WITH F8.5 &I3
580   (23)->20 7.0  1  @TERM./LINE,RESP. TIME, MPROC WITH I3,F5.2,I2
581 (24)-->1          @1 FOR PLOTTING AND 0 FOR SKIPPING IT WITH I3

CPU:.787   CTP:.091   SUPS:17.904

@BRKPT PRINT$
```

Each group is then divided into three subgroups of 12 cards, one for each chargeable item. Each subgroup is then divided into 3 units, 4 cards per unit, according to the three types of traffic density combinations: high-high, high-low and low-low. Each specific unit is then divided into two subunits of 2 cards. The first subunit is for installation costs, and the second for recurring costs. The first card of each subunit is for costs under half duplexing mode, and the second for costs under full duplexing mode. The first number of each card is the cost for the initial unit; the second for each additional unit at the same location.

Item 14 indicates that, in Texas, a linear costing function is used for all of the line service charges. The first card gives the installation charge as a function of distance, and the second the monthly recurring charge as a function of distance. Under each line type, the line cost is also given as a function of traffic density mix between two terminals.

Item 15 indicates that an optimization process is requested after a star network is formed.

Item 16 shows the line protocol characteristics for those three line types under consideration by providing data such as no. of polling characters, modem turn-around time, while item 17 and 18 give the message statistics. Item 19 indicates that a 110 milli-second is used as the average transaction service time needed in the central switcher of the system being studied.

Item 20 pre-selects system termination AAAA as the RSC, and item 21 designates AAAA as the system centroid.

The three remaining cards define the total traffic load at the central switcher, the multidrop line constraints, and a request for a CalComp plot at the end of each regional network optimization.

3.4.3 Output

After a normal termination from a STACOM program run, outputs from the printer and the CalComp plotter should contain all data desired. This subsection describes the contents of these outputs obtained from the example run.

3.4.3.1 Printer Output. Data showing results from a normal program execution of the STACOM program are printed on a regular printer. Table 3-3 shows the exact output obtained from running the STACOM program utilizing the set of input data as given in Paragraph 3.4.2.

To facilitate the following discussions, the contents of Table 3-3 are itemized as shown.

Item 1 reminds the user that only one region has been considered in this specific run. Item 2 shows the line protocol for each

CONTINUED

1 OF 2

Table 3-3. Printer Output from the Example Run

```
(1) ------> THERE ARE    1 REGIONS

(2) --> 1200   POLL CHAR.=    3 NAK CHAR.=   2 POLL O/H=    0
               NAK O/H=    0  MSG O/H=    8
               MPSEM=     .008 PPSEM=     .000
        2400   POLL CHAR.=    3 NAK CHAR.=   2 POLL O/H=    0
               NAK O/H=    0  MSG O/H=    8
               MPSEM=     .008 PPSEM=     .000
        4800   POLL CHAR.=    3 NAK CHAR.=   2 POLL O/H=    0
               NAK O/H=    0  MSG O/H=    8
               MPSEM=     .050 PPSEM=     .000

(3) ------> AVG. INPUT MSG WITH PRIO 1= 147.0 CHARS
            AVG. INPUT MSG              147.0 CHARS
            AVG. OUTPUT MSG W/H PRIO 1= 212.2 CHARS.
            AVG. OUTPUT MSG WITH PRIO2=    .0 CHARS
                 OVERALL AVG. MSG     = 184.0 CHARS
(4) ------------> 2  AAAA     1
                  3           0
```

(5) ------------>

TRAFFIC MATRIX(BPS)

TERM.	AZLI	AZKK	AZKW	AZXZ	AZLQ	AZLA	AZLB	AZLD	AZLC	AZKA	AZTI	AZGL	AZLK	AZLL	AZLR
TRAFIN	1.0	.6	.8	.7	.4	3.4	1.0	1.4	1.2	.6	.7	4.1	32.3	5.3	1.3
TRFOUT	2.6	1.4	2.4	1.6	1.1	7.4	2.0	3.7	2.9	1.5	2.1	13.1	45.4	7.9	3.2
TRAFIN	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0
TRFOUT	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0
TRAFIN	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0
TRFOUT	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0
TRAFIN	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0
TRFOUT	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0

Table 3-3. Printer Output from the Example Run
(Continuation 1)

⑥ → TOTAL TRAFFIC ORIGINATED FROM SYS. TERMN. (BITS/SEC)

AZLI	.980	AZKK	.584	AZKW	.837	AZXZ	.747	AZLQ	.381
AZLA	3.369	AZLB	1.023	AZLD	1.393	AZLC	1.183	AZKA	.609
AZTI	.715	AZGL	4.149	AZLK	32.304	AZLL	5.281	AZLR	1.335
AZLJ	.763	AZKS	.537	AZLE	2.279	AZLF	.537	AZLN	.985
NAAE	.221	NABG	.221	NACG	.357	NAFA	.204	AAAA	.000

TOTAL TRAFFIC= 61.00

TOTAL TRAFFIC DESTINATED TO SYS. TERMN. (BITS/SEC)

AZLI	2.564	AZKK	1.411	AZKW	2.373	AZXZ	1.588	AZLQ	1.088
AZLA	7.415	AZLB	1.956	AZLD	3.707	AZLC	2.925	AZKA	1.473
AZTI	2.135	AZGL	13.140	AZLK	85.424	AZLL	7.884	AZLR	3.161
AZLJ	2.295	AZKS	1.325	AZLE	5.452	AZLF	1.115	AZLN	2.783
NAAE	.701	NABG	.701	NACG	1.131	NAFA	.648	AAAA	.000

TOTAL TRAFFIC= 114.39

TOTAL SYSTEM TRAFFIC= 175.39

Table 3-3. Printer Output from the Example Run (Continuation 2)

```
(7)                                  POINT TO POINT DISTANCE MATRIX

TERM.    AZLR    AZLL    AZLK    AZGL    AZTI    AZKA    AZLC    AZLD    AZLH    AZLA    AZLQ    AZXZ    AZKW    AZKK    AZLI
AZLI     83.     67.     67.     67.     34.     31.     31.     49.     59.     69.    105.     82.    119.     35.
AZKK     68.     54.     54.     54.     49.     28.     28.     25.     69.     69.     88.     84.    111.
AZKW     46.     58.     58.     58.     89.     91.     91.     88.     70.     70.     37.     45.
AZXZ     42.     41.     41.     41.     49.     58.     58.     66.     26.     26.     55.
AZLQ     23.     38.     38.     38.     81.     74.     74.     64.     71.     71.
AZLA     52.     43.     43.     43.     25.     41.     41.     57.      0.
AZLB     52.     43.     43.     43.     25.     41.     41.     57.
AZLD     44.     31.     31.     31.     44.     24.     24.
AZLC     52.     37.     37.     37.     25.      0.
AZKA     52.     37.     37.     37.     25.
AZTI     60.     45.     45.     45.
AZGL     16.      0.      0.
AZLK     16.      0.
AZLL     16.
AZLR      0.

(8)  ---> NCC=   AAAA

(9)  ---> CPU UTILIZATION PER PROCESSOR IS  .707
```

```
(10) ---> REG=  1 , SYS. TERMN.=

AZLI MULESHOE PD          AZKK MORTON SO          AZKW SPUR PD            AZXZ FLOYDADA SO
AZLQ POST SO              AZLA PLAINVIEW PD       AZLB PLAINVIEW SO       AZLD LEVELLAND PD
AZLC LITTLEFIELD PD       AZKA LITTLEFIELD SO     AZTI OLTON PD           AZGL LUBBOCK DPS
AZLK LUBBOCK PD           AZLL LUBBOCK SO         AZLR SLATON PD          AZLJ TAHOKA PD
AZKS TAHOKA SO            AZLE BROWNFIELD PD      AZLF BROWNFIELD SO      AZLN DENVER CITY PD
NAAE MULESHOE S.O.        NABG CROSBYTON S.O.     NACG LEVELLAND S.O.     NAFA PLAINES S.O.
AAAA AUSTIN SWITCHER

INDICES FOR SYS. TERM.=
  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

(11) ---> RSC=  AAAA FOR REGION    1
```

Table 3-3. Printer Output from the Example Run (Continuation 3)

(12)

REGIONAL STAR NETWORK AND ITS COSTS- 1

SYSTEM TERMN.		AZLI	AZKK	AZKW	AZXZ	AZLB	AZLA	AZLH	AZLD	AZLC	AZKA
NO. OF LINES REQ.											
1.2KB		1	1	1	1	1	1	1	1	1	1
2.4KB		0	0	0	0	0	0	0	0	0	0
4.8KB		0	0	0	0	0	0	0	0	0	0
LINE UTILIZATION		.004	.003	.004	.003	.002	.010	.003	.005	.004	.003
DSTNCE FROM RSC		399	379	287	331	294	356	356	355	368	368
TRAFFIC											
LINE TO CPU		.980	.584	.837	.747	.381	3.369	1.023	1.343	1.183	.609
CPU TO LINE		2.564	1.411	2.373	1.588	1.088	7.415	1.956	3.707	2.925	1.473
LINE RESPONSE TIME		3.279	3.276	3.278	3.277	3.275	3.296	3.278	3.283	3.281	3.276
SUBTOTAL											
INST. COSTS											
LINES		0	0	0	0	0	0	0	0	0	0
SER.T.	480	20	20	20	20	20	20	20	20	20	20
MODEM	2400	100	100	100	100	100	100	100	100	100	100
DROP		0	0	0	0	0	0	0	0	0	0
ANNUAL RECURR. COST											
LINES	229750	14364	13644	10332	11916	10584	2563	2563	12780	13248	13248
SER.T.	8640	360	360	360	360	360	360	360	360	360	360
MODEM	12672	528	528	528	528	528	528	528	528	528	528
DROP	5760	240	240	240	240	240	240	240	240	240	240
TOTAL COST											
INST. COST	2880	120	120	120	120	120	120	120	120	120	120
RECUR. COST	256822	15492	14772	11460	13044	11712	3691	3691	13908	14376	14376

REGIONAL STAR NETWORK AND ITS COSTS- 2

SYSTEM TERMN.		AZII	AZGL	AZLK	AZLL	AZLR	AZLJ	AZKS	AZLE	AZLF	AZLN
NO. OF LINES REQ.											
1.2KB		1	1	1	1	1	1	1	1	1	1
2.4KB		0	0	0	0	0	0	0	0	0	0
4.8KB		0	0	0	0	0	0	0	0	0	0
LINE UTILIZATION		.003	.016	.069	.012	.005	.004	.003	.008	.002	.004
DSTNCE FROM RSC		373	332	332	332	316	311	311	334	334	353
TRAFFIC											
LINE TO CPU		.715	4.149	32.304	5.281	1.335	.763	.537	2.279	.537	.985
CPU TO LINE		2.135	13.140	45.424	7.884	3.161	2.295	1.325	5.452	1.115	2.783
LINE RESPONSE TIME		3.278	3.311	3.461	3.301	3.281	3.278	3.275	3.289	3.275	3.280
SUBTOTAL											
INST. COSTS											
LINES		0	0	0	0	0	0	0	0	0	0
SER.T.		20	20	20	20	20	20	20	20	20	20
MODEM		100	100	100	100	100	100	100	100	100	100
DROP		0	0	0	0	0	0	0	0	0	0
ANNUAL RECURR. COST											
LINES		13428	2390	2390	2390	11376	2239	2239	12024	12024	12708
SER.T.		360	360	360	360	360	360	360	360	360	360
MODEM		528	528	528	528	528	528	528	528	528	528

Table 3-3. Printer Output from the Example Run
(Continuation 4)

DROP	240	240	240	240	240	240	240	240	240	240
TOTAL COST										
INST. COST	120	120	120	120	120	120	120	120	120	120
RECUR. COST	14556	3518	3518	3518	12504	3367	3367	13152	13152	13836

REGIONAL STAR NETWORK AND TTS COSTS- 3

SYSTEM TERMN.	NAAE	NABG	NACG	NAFA	AAAA
NO. OF LINES REQ.					
1.2KB	1	1	1	1	0
2.4KB	0	0	0	0	0
4.8KB	0	0	0	0	0
LINE UTILIZATION	.002	.002	.002	.002	.000
DSTNCE FROM RSC	399	310	355	361	0
TRAFFIC					
LINE TO CPU	.221	.221	.357	.204	.000
CPU TO LINE	.701	.701	1.131	.648	.000
LINE RESPONSE TIME	3.273	3.273	3.275	3.273	.000
SUBTOTAL					
INST. COSTS					
LINES	0	0	0	0	0
SER.T.	20	20	20	20	0
MODEM	100	100	100	100	0
DROP	0	0	0	0	0
ANNUAL RECURR. COST					
LINES	14364	11160	12780	12996	0
SER.T.	360	360	360	360	0
MODEM	528	528	528	528	0
DROP	240	240	240	240	0
TOTAL COST					
INST. COST	120	120	120	120	0
RECUR. COST	15492	12288	13908	14124	0

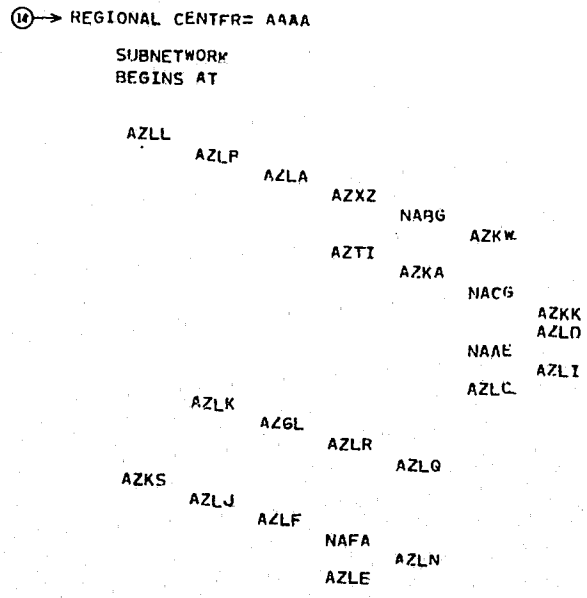
TOTAL COST= 259702

Table 3-3. Printer Output from the Example Run
(Continuation 5)

(13) → FINAL MULTIDROP NETWORK AND ITS COSTS- 1

SUBNET NO.		1	2
BEGINNNING NODE		AZLL	AZKS
NO. OF TERM.		18	6
NO. OF LINES			
1.2KB		1	1
2.4KB		0	0
4.8KB		0	0
LINE UTILIZATION		.138	.017
TOTAL MILEAGE		617	388
TRAFFIC			
LINE TO CPU		55.691	5.305
CPU TO LINE		100.777	13.617
LINE RESPONSE TIME		4.038	3.416
	SUBTOTAL		
INST. COSTS			
LINES		0	0
SER.T.	260	190	70
MODEM	1300	950	350
DROP		0	0
ANNUAL RECURR. COST			
LINES	16422	11411	5011
SER.T.	4680	3420	1260
MODEM	6864	5016	1848
DROP	3120	2280	840
TOTAL COST			
INST. COST	1560	1140	420
RECUR. COST	31086	22127	8959
	TOTAL COST=		32646

Table 3-3. Printer Output from the Example Run
(Continuation 6)



individual line type under consideration. For example, a modem turn-around time of 50 milli-seconds has been used in the run.

Item 3 shows the traffic characteristic as calculated by the STACOM program and item 4 prints out the pre-assignment activities. In this example run, the system termination AAAA is preselected as the regional switching center; since only one region is under consideration, all of the remaining system terminations are assigned to region 1.

Item 5 shows a small portion of a traffic matrix from each system termination to four data bases calculated by the program. Item 6 prints the total incoming/outgoing traffic in bps to/from each individual system termination. Also included is total incoming/outgoing traffic to/from the system.

Item 7 gives a short list of point-to-point distances between system terminations as calculated by the program.

Item 8 gives the system centroid as designated from the input. Item 9 shows the CPU utilization at the central switcher of the system being studied.

Item 10 gives the IDs and names of all system terminations in the region and their internal indices. Item 11 prints the regional switching center for the region which has been preselected. In this run, the RSC turns out to be the central switcher.

Item 12 provides the details of the star network developed by the program. For example, the system termination AZLI is linked to the regional switching center AAAA by a 1200 bps line. With the traffic as shown, its line utilization is only .004 and response time 3.279 seconds. It is 399 miles away from AAAA. Based on the tariff applicable for Texas, its installation costs are $20 for service terminal and $100 for modems. Annual recurring costs are $892 for lines, $360 for service terminals, $528 for modems and $240 for the drop charges. After the printout for the star network, the multidrop network (as generated by the STACOM program) is printed as given by item 13. In this example run, two distinctive subnetworks have been generated. Both subnetworks require only the 1200 bps lines. In addition to data similar to item 12, it also includes the total number of terminals on each multidrop line and the total connection milage. Summarized costs are also provided.

Finally, the actual structure of the final multidrop network is printed as item 14. It is printed in a tree-type form, relating each individual termination to others.

The above described printer output is a copy of the FORTRAN output alternate file, 100. In addition to this, a regular FORTRAN output file, 6, is generated by the program. For this example run, Table 3-4 is the copy of output file, 6. It indicates all of the request messages go by the program during its input phase. The last two lines are an indication that the program has been successfully executed.

Table 3-4. Unit 6 Printer Output from the Example Run

```
ASSUME NUMBER OF REGIONS
ENTER NR AND STRIKE RETURN KEY
TYPE IN NO. OF SYS. TERMNS, DATA BASES AND CITIES WITH FORMAT 3I5
THERE ARE    25SYS. TERMNS.,    4 DATA BASES  35R CITIES
TYPE IN DATA BASE LOCATIONS WITH FORMAT 6(1X,A4)
   4 DATA BASES ARE AT  AAAA  DDDD  SSSS  HHHH
TYPE IN CITY V-H WITH FORMAT (33X,I5,2X,I5)
TYPE IN PID NO., NAME, MAPPING ADR. AND TRAFFIC
WITH FORMAT I4,1X,4A6,I4,6F8.2
TYPE IN NO. OF RATE STRUCTURES UNDER
CONSIDERATION WITH FORMAT I3
TYPE IN RATE APPLICATION TO EACH COMBN.
WRT EACH SYS. TERM. WITH FORMAT 10I2
READ IN TRAFFIC DENSITY TYPE AND RATE STRUCTURE
FOR EACH CITY WITH FORMAT 80I1
TYPE IN NO. OF LINE TYPES APPICABLE WITH FORMAT I3
TYPE IN NAME, CAPACITY,  UTIL. FACTOR AVAIL. FOR
EACH LINE TYPE WITH FORMAT A6,1X,I6,1X,F3.2,2(1X,I1)
TYPE IN NO. OF DEVICES  AND NAMES FOR EACH LINE TYPE
WITH FORMAT I3/10(A6,1X)
TYPE IN INST. AND RECURR. COSTS WRT
RATE STRUCTURE, LINE TYPE, DEVICE, TRAFFIC DENSTIY
AND DUPLEXING MODE WITH FORMAT 2F9.2/2F9.2
TYPE IN INST. COSTS FOR LINES WRT
RATE, LINE, DENSITY, AND DUPLEXING MODE
WITH FORMAT 4F9.2
TYPE IN INDEX FOR LINEARITY OF LINE RECUR. COST
FUNCTION WITH 1=LINEAR AND NONLINEAR OTHERWISE
WITH FORMAT I1 FOR EACH LINE TYPE
TYPE IN RECUR. COSTS WITH FORMAT 4F9.2 IF LINEAR
WITH FORMAT 10F8.3/10F8.3 IF NONLINEAR
IF NONLINEAR, USE 10F8.2
TYPE IN ACTION INDICES FOR EACH REGION
1ST ELEMENT: 1= INSERTION TO THIS PRELOADED REGION IS OK
2ND ELEMENT: 1= OPTIMIZATION IS NEEDED
TYPE IN REGION INDEX AND ACTION NUMBER NEEDED
WITH FORMAT 2I2 AND END IT WITH A 0 0
TYPE IN NPL, NAK, NPLOH, NAKOH, MOH,
TAMDM, TAD IN FORMAT (5I4,2F7.5)
TYPE IN NO. OF MSG TYPES, AND TRAFFIC STATISTICS
SUCH AS MSGNAM, MSLIN, MSLOUT, RATIO WITH
FORMAT I4/(A6,2(2I4,2F6.3))
TYPE IN PRELOADED SYSTEM TERMN. AND RSC WITH
FORMAT I1,1X,A4,A5
    3283  DISTANCE ITEMS ARE OVERSIZED
ASSUME A SYSTEM  CENTROID
ENTER CODE FOR NSCC AND STRIKE RETURN KEY
INPUT TOTAL NO. OF TRANSACTIONS AND NO. OF ACCESS AT THE SWITCHER
ENTER WITH F8.5 AND I3 UNDER XSAC/SEC
READ IN LIMITS ON NO. OF SYS. TERMS. ON A LINE
,RESPONSE TIME REQD AND NO. OF PROCESSORS WITH FORMAT
I3,F5.2,I2
IF PLOTTING IS REQUIRED, TYPE 1 WITH FORMAT I3
 TRYLNK HAS BEEN ACCESSED FOR      10052 TIMES
 UPNETW HAS BEEN ACCESSED FOR         22 TIMES

QBRKPT PRINT$
```

3.4.3.2 CalComp Plot. Figure 3-1 is the actual network graph as plotted by the CalComp plotter. It reflects the network as printed in the last part of printer output. It should be noted that because of the existence of identical V-H coordinates associated with system terminations in the example run, fewer distinctive nodes are shown in the plot. The root node is for the system termination, AAAA, which is the location of the Austin central switcher as used in the example run.

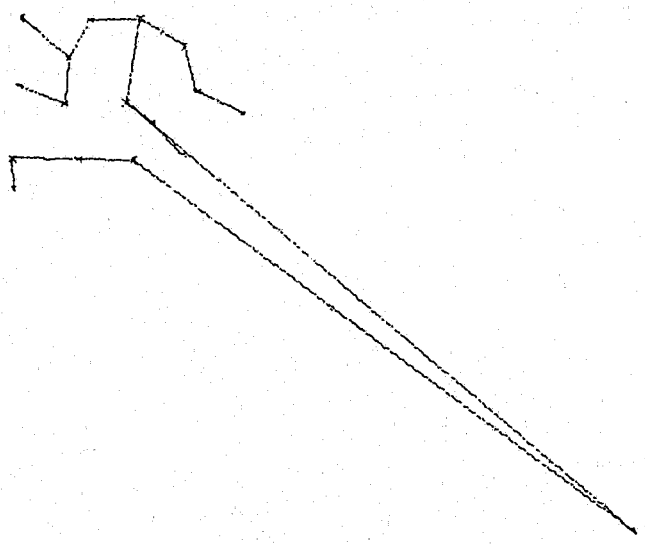


Figure 3-1. CalComp Plot from the Example Run

REFERENCES

1. Esau, L. and Williams, K., "On Teleprocessing System Design: Part II-A, Method for Approximating the Optimal Network", IBM System Journal, Vol. 5, No. 3, 1966.

2. "FORTRAN V Programmer's Reference", UNIVAC Series, UP-4060, Rev. 2, UNIVAC, A Division of Sperry Rand Corporation, Blue Bell, Pa.

3. Martin, J., System Analysis for Data Transmission, Prentice-Hall, Inc., Englewood Cliffs, New Jersey

4. Karnaugh, M., Multipoint Network Layout Program, Internal Document RC (#16892), IBM Thomas J. Watson Research Center, New York

5. Fielding, J., Frewing, K., and Reilly, N., "Requirements Analysis and Design of Ohio Criminal Justice Telecommunications Network", State Criminal Justice Telecommunications (STACOM) Final Report, JPL Document 77-53, Vol. II, Jet Propulsion Laboratory, Pasadena, Calif.

APPENDIX A

STACOM PROGRAM LISTING

```
51928*STACOM(1),MAIN/0777
      1      C*************************************************************C
      2      C*                                                           *C
      3      C*                      STACOM TOPOLOGY PROGRAM              *C
      4      C*                    JET PROPULSION LABORATORY              *C
      5      C*                      4800 OAK GROVE DRIVE                 *C
      6      C*                   PASADENA, CALIFORNIA  91103             *C
      7      C*                                                           *C
      8      C*************************************************************C
      9      C
     10      C THIS PROGRAM IS DESIGNED TO PERFORM FORMATIONS OF REGIONS, SELECTIONS
     11      C OF REGIONAL SWITCHING CENTERS, FORMATIONS OF INITIAL REGIONAL NETWORKS,
     12      C OPTIMIZATION OF REGIONAL NETWORKS USING THE ESAU-WILLIAMS METHOD IF
     13      C REQUESTED, AND FINALLY FORMATION OF AN INTERREGION NETWORK AND ITS
     14      C OPTIMIZATION
     15      C
     16      C          ********************************************
     17      C
     18      C THIS TOPOLOGY PROGRAM CONTAINS ONE MAIN PROGRAM AND ELEVEN SUBPROGRAMS.
     19      C THEY ARE AS FOLLOWS:
     20      C    MAIN PROGRAM : MAIN (REGION ASSIGNMENTS OF SYSTEM TERMINATIONS)
     21      C    SUBPROGRAM-1 : RGNNET (REGIONAL NETWORK FORMATION AND ITS OPTIMIZATION)
     22      C    SUBPROGRAM-2 : IRNOP (INTER-REGION NETWORK OPTIMIZATION)
     23      C    SUBPROGRAM-3 : ICOSTJ (COSTING FUNCTION)
     24      C    SUBPROGRAM-4 : RHOFUN (LINE UTILIZATION FUNCTION)
     25      C    SUBPROGRAM-5 : LINNUM (LINE CONFN. DEFINITION BASED ON TRAFFIC)
     26      C    SUBPROGRAM-6 : PACK (STORING OR RETRIEVING DISTANCE DATA)
     27      C    SUBPROGRAM-7 : DIST (FINDING DISTANCE BETWEEN TWO GIVEN TERMINALS)
     28      C    SUBPROGRAM-8 : LINK (FINDING COMPRESSED INDEX FOR DIST)
     29      C    SUBPROGRAM-9 : RECOVR (RECOVERING COMPRESSED DISTANCE DATA)
     30      C    SUBPROGRAM-10: PLOTPT (PLOTTING EACH DROP ON A MULTIDROP NETWORK)
     31      C    SUBPROGRAM-11: RSPNSE (ESTIMATING RESPONSE TIME)
     32      C
     33      C          ************************************************
     34            PARAMETER MW=4,IWT=100,NLIMIT=2,NPC=360,NP9=18
     35            PARAMETER NP1=130,  NP2=1, NP3=4, NP4=3
     36            PARAMETER NP6=(NPC*NPC/2-NPC+1)/4+1
     37            PARAMETER NP7=4,NP0=10*NPC
     38            COMMON /EIN/ SVR(NP1),NRSC(MW),NUMPR(MW),TRAFDN(NP1),
     39           *       TRAFIT(NP1)
     40           *   /VH/ IVERT(NPC), IHORZN(NPC)
     41           *   /CONST/ N1,N2,N3,N4,N7,NCITY
     42           *   /INF/ IRATEJ(NP2,NP2),IRAND(NPC,2),IFLAG(NP2,NP3)
     43           *   /BCOST/ AINSTC(NP2,NP3,NP4,3,2,2), RECRC(NP2,NP3,NP4,3,2,2),
     44           *    ANSTLN(NP2,NP3,3,2,2), RECRLN(NP2,NP3,3,2,16),IDUPLX(NP3)
     45           *   /LINCHR/ LINMIX(NP3), LINCAP(NP3), UTILI7(NP3)
     46           *   /REF/ IREF(NPC),TRAFD(NP1,2,NP7), DSTNCE(NP6),MAFADR(NP1)
     47           *   /OVFR/ IVRD(NP0,2),IOVER1
     48           *   /NAME/ NAMEST(NP1,4),LINAME(NP3),NAMEHW(NP4)
     49           *   /SUM/ ASUM(4),BSUM
     50           *   /XMT/ TIMXMT(7,NP3), WAIT(6)
     51           *   /MSLA/ AMSL(7)
     52           *   /ROUND/ NTERMS,TIMREQ,MPROC,MPLOT
     53           *   /ADD/ IADD(NP1),KCHG,KADD  @KCHG=FIRST DROP, KADD=JUST FOR LINE
     54            INTEGER DSTNCE
     55            DIMENSION IACTN(MW,2),INDXPT(NP1)
     56            DIMENSION NUMRR(NP1),ITRAFC(2),NBASE(NP7)
```

```
      DATA ITRAFC/'TRAFINTREOUT'/
      DIMENSION TRM(MW,MW), DRM(MW,MW), NUMR(NP1), NUMR(NP1,4)
      INTEGER SVR
      DIMENSION OUTPRT(NP1)
      NMAX=NPO   @MAXIMUM SIZE FOR OVERFLOW DISTANCE DATA TABLE
      CPUAVG=0.
C
C SELECT NUMBER OF REGIONS
C
  225 WRITE(6,220)
      READ(5,735) NR1
      WRITE(IWT,1011) NR1
      ANR1=NR1
C
C READ IN TRAFFIC DENSITY INDEX AND RATE STRUCTURE FOR EACH SYSTEM
C TERMINATION IN THE SYSTEM
C
      CALL CREADA(N1)
C
C READ IN RATE APPLICATION MATRIX
C
      CALL CREADB(N2)
C
C READ IN NAMES, CAPACITIES, UTILIZATION FACTORS AND AVAILABILITIES
C FOR LINES APPLICABLE IN THE SYSTEM
C
      CALL CREADC(N3)
C
C READ IN INSTALLATION AND RECURRING COSTS FOR CHARGEABLE ITEMS
C REQUIRED FOR COMMUNICATION LINES
C
      CALL CREADD(N4)
C
C READ IN INSTALLATION AND RECURRING COSTS FOR LINES
C
      CALL CREADF
C
C READ IN ACTIONS TO BE PERFORMED ON EACH REGIONAL NETWORK
C 1ST ELEMENT : 1=INSERTIONS TO PRELOADED REGIONS ARE ALLOWED
C               0= SUCH AN ACTION IS NOT ALLOWED
C 2ND ELEMENT : 1=NETWORK OPTIMIZATION IS TO BE PERFORMED
C               0=NO OPTIMIZATION IS NEEDED
C
      CALL CREADK
C
C READ IN LINE AND LINE PROTOCOL CHARACTERISTICS
C
      CALL CREADR
C
C CONVERT TRAFFIC FROM CHARACTERS/MIN TO BITS/SEC
C
      DO 85 K=1,2
      DO 85 I=1,N1
      DO 85 L=1,N7
      TRAFD(I,K,L)=TRAFD(I,K,L)*8./60.
85    CONTINUE
      ISUM=0
```

```
114             DO 25 I=1,NCITY
115               ISUM=ISUM+I
116               IREF(I)=ISUM
117       25    CONTINUE
118             DO 701 I=1,NR1
119               NUMPR(I) = 0 @NO. OF SYSTEM TERMINATIONS AT EACH REGION
120       701   CONTINUE
121             WRITE(6,888)
122       805   READ(5,800) NCODE,NSTATE,NREGO
123             WRITE(IWT,804) NCODE, NSTATE, NREGO
124             NSTATE=LOCAL(NSTATE)   @FIND CARDINAL INDEX
125             GO TO(801,802,240),NCODE
126       802   CONTINUE
127             SVR(NSTATE) = NREGO
128             NUMPR(NREGO) = NUMPR(NREGO) + 1
129       801   CONTINUE
130             NRSC(NREGO) = NSTATE
131             GO TO 805
132       240   CONTINUE
133             DO 70 L=1,N1
134               TRAFON(L)=0.
135          70 CONTINUE
136             IOVER1=1  @COUNTER FOR OVERSIZED TRAFFIC DATA
137       C
138       C CALCULATE DISTANCE DATA BETWEEN SYSTEM TERMINATIONS
139       C
140             DO 20 J=1,NCITY
141               DO 30 K=1,NCITY
142                 IF(J-K) 51,30,30
143       51        CONTINUE
144                 ISQ1=(IVERT(J)-IVERT(K))**2
145                 ISQ1=ISQ1+(IHORZN(J)-IHORZN(K))**2
146                 IF(ISQ1 .EQ. 0)GOTO  22
147                 SQ1=ISQ1/10.
148                 NSQ1=INT(SQ1)
149                 DIFF=SQ1-NSQ1
150                 IF(DIFF .GT. 0.) SQ1=NSQ1+1.
151                 BDIST= SQRT(SQ1)
152                 KDIST=INT(BDIST)
153                 DIFF=BDIST-KDIST
154                 IF(DIFF .GT. 0.) KDIST=KDIST+1
155                 GOTO 23
156       22        CONTINUE
157                 KDIST=0
158       23      CONTINUE
159                 JKL=LINK(J,K)
160                 IF(KDIST .LE. 510) GOTO 5
161                 CALL OVERFL(JKL,KDIST)
162                 GOTO 30
163       5         CONTINUE
164                 CALL PACK(JKL,KDIST,1,DSTNCE)
165       30      CONTINUE
166       20    CONTINUE
167             IOVER1=IOVER1-1
168             WRITE(6,3) IOVER1
169       C
170       C TOTAL INPUT TRAFFIC BY EACH SYS. TERMN.
```

```
171       C
172             TRFALL=0.0
173             TALLIT=0.
174             TALLDN=0.
175             DO 41 L=1,N1
176               TRAFIT(L)= 0.0
177               TRAFDN(L)= 0.0
178               DO 42 J=1,N7
179                 TRAFIT(L) = TRAFIT(L) + TRAFD(L,2,J)
180                 TRAFDN(L) = TRAFDN(L) + TRAFD(L,1,J)
181       42        CONTINUE
182               TALLDN=TALLDN+TRAFDN(L)
183               TALLIT=TALLIT+TRAFIT(L)
184       41      CONTINUE
185             TRFALL=TALLDN+TALLIT
186       C
187       C PRINT OUT TRAFFIC DATA BETWEEN SYSTEM TERMINATIONS
188       C
189             NTURN=N1/15 + 1
190             NREM=MOD(N1,15)
191             IF(NREM .EQ. 0) NTURN=NTURN-1
192             WRITE(IWT,111)
193             DO 1100 KK=1,1    @FOR TEST ONLY
194               KK1=(KK-1)*15 + 1
195               KK2=KK*15
196               IF(KK2 .GT. N1) KK2=N1
197               WRITE(IWT,113) (INDXPT(J), J=KK1,KK2)
198               DO 99 J=1,N7
199                 DO 97 KT=1,2
200                   DO  28  KR=KK1,KK2
201                   OUTPRT(KR)=TRAFD(KR,KT,J)
202       28          CONTINUE
203                   WRITE(IWT,110) ITRAFC(KT), (OUTPRT(K), K=KK1,KK2)
204       97        CONTINUE
205       99      CONTINUE
206       1100  CONTINUE
207       C
208       C PRINT OUT TRAFFIC ORIGINATED FROM EACH SYSTEM TERMINATION
209       C
210             WRITE(IWT,1013)
211             WRITE(IWT,1001) (INDXPT(NJ),TRAFDN(NJ),NJ=1,N1)
212             WRITE(IWT,74) TALLDN
213       C
214       C PRINT OUT TRAFFIC DESTINATED TO EACH SYSTEM TERMINATION
215       C
216             WRITE(IWT,1014)
217             WRITE(IWT,1001) (INDXPT(NJ),TRAFIT(NJ),NJ=1,N1)
218             WRITE(IWT,74) TALLIT
219             WRITE(IWT,75) TRFALL
220       C
221       C PRINT OUT DISTANCE DATA BETWEEN SYSTEM TERMINATIONS
222       C
223             NTURN=N1/15+1
224             NREM=MOD(N1,15)
225             IF(NREM.EQ.0) NTURN=NTURN-1
226             NTURN=1          @FOR SHORT OUTPUT
227             DO 101 KK=1, NTURN
```

```
228               KK1=(KK-1)*15 + 1
229               KK2=KK*15
230               IF(KK2 .GT. N1) KK2=N1
231               WRITE(IWT,109) (INDXPT(J),J=KK2,KK1,-1)
232               DO 98 J=1,KK2
233                 IF(J.GE.KK1) KK1=J+1
234                 DO 27 KR=KK2,KK1,-1
235                   OUTPRT(KR)=DIST(J,KR)
236         27        CONTINUE
237                   WRITE(IWT,112) INDXPT(J),(OUTPRT(K),K=KK2,KK1,-1)
238         98      CONTINUE
239         101   CONTINUE
240               WRITE(6,210)
241         4005  CONTINUE
242               READ(5,734) NSCC1
243               WRITE(IWT,1015) NSCC1
244               NSCC1=LOCAL(NSCC1)
245               IF(NSCC1.NE.0) GOTO 4003
246               WRITE(6,4013)
247               GOTO 4005
248         4003  CONTINUE
249               TPR1 = TRFALL
250               WRITE(6,2101)
251               READ(5,2102) XSAC, NREQSW  @NREQSW=NO. OF REQUESTS/TRANS AT SWITHER
252               WRITE(6,2103)
253               READ(5,2104) NTERMS,TIMREQ,MPROC
254               WRITE(6,2105)
255               READ(5,2104) MPLOT  @MPLOT=1 IF PLOT IS NEEDED
256         C
257         C PRE-CALCULATE CPU TURNAROUND TIME
258         C
259               CALL CWAITC
260         C
261         C SUM UP TOTAL TRAFFIC FOR PRELOADED SYSTEM TERMINATIONS IN REGIONS
262         C WHICH DO NOT ALLOW ANY INSERTIONS OF OTHER SYSTEM TERMINATIONS
263         C
264               TPR2=0
265               DO 77 N=1,N1
266                 NK=SVR(N)
267                 IF(NK .EQ. 0) GOTO 77 @NOT PRELOADED
268                 IF(IACTN(NK,1) .EQ. 0) GOTO 77  @INSERTIONS ARE ALLOWED
269                 TPR2=TRAFDN(N) + TRAFIT(N) + TPR2
270         77    CONTINUE
271               DO 76 L=1,NR1
272                 IF(IACTN(L,1).EQ.0.OR.NUMPR(L).EQ.0) GOTO 76
273                 ANR1=ANR1-1.
274         76    CONTINUE
275               TPR1=TPR1-TPR2
276               IF(NR1 .EQ. 1) GOTO 726  @ONE REGION CASE
277         C
278         C DETERMINE LOWER LIMIT FOR AVERAGE REGIONAL TRAFFIC
279         C
280               ZETA=.1
281               IF(ANR1.EQ.0.) GOTO 340
282               TPR=TPR1/ANR1
283               GOTO 350
284         340   CONTINUE
```

```
285           TPR=TPR1
286       350 CONTINUE
287           TPRL=TPR*(1.-ZETA)
288           DO 909 NREG=1,NR1
289             TRFS=0.
290             AMAXD=0.
291             II=0
292             IF(NUMPR(NREG).NE.0) GOTO 5000 @NREG IS A PRELOADED REGION
293       C
294       C ASSIGN SYSTEM TERMINATIONS TO A REGION WITHOUT ANY PRELOADING
295       C
296             DO 400 NI=1,N1
297               IF(SVR(NI) .NE. 0) GOTO 400 @NI IS PRELOADED
298               ADIST=DIST(NSCC1,NI)
299               IF(ADIST .LE. AMAXD) GOTO 400
300               AMAXD=ADIST  @UPDATE LONGEST DIST. FROM NCCC
301               II=NI  @UPDATE FARTHEST SYS. TERMN.
302       400     CONTINUE
303             NS1=II  @THE FARTHEST SYSTEM TERMINATION
304             TRFS=TRFS + TRAFDN(NS1) + TRAFIT(NS1)
305             SVR(NS1)=NREG
306             NUMPR(NREG)=NUMPR(NREG)+1
307             IF(TRFS .GT. TPRL) GOTO 707
308             GOTO 7021
309       5000  CONTINUE
310             IF(IACTN(NREG,1) .EQ. 1) GOTO 909 @INSERTIONS ARE NOT ALLOWED
311       C
312       C SUM UP TRAFFIC IN THIS REGION
313       C
314             DO 702 I=1,N1
315               IF(SVR(I) .NE. NREG) GOTO 702
316               TRFS=TRFS+TRAFDN(I)+TRAFIT(I)
317               ADIST=DIST(NSCC1,I)
318               IF(ADIST .GT. AMAXD) II=I
319       702     CONTINUE
320             IF(TRFS .GT. TPRL) GOTO 707  @ENOUGH TRAFFIC IN THIS REGION
321             NS1=II  @THE FARTHEST SYS. TERMN. IN THE REGION
322             IF(NRSC(NREG) .NE.0) NS1=NRSC(NREG)
323       7021  CONTINUE
324             CALL FINDD(NS1,NS2)
325             IF(NS2 .EQ. 0) GOTO 909
326             SVR(NS2)=NREG
327             NUMPR(NREG)=NUMPR(NREG)+1
328             TRFS=TRFS+TRAFDN(NS2)+TRAFIT(NS2)
329             IF(NREG .EQ. NR1) GOTO 7021
330             IF(TRFS .GT. TPRL) GOTO 707
331             GOTO 7021
332       707   CONTINUE
333             TPR1= TPR1-TRFS   @UPDATE REMAINING TRAFFIC
334             ANR1=ANR1- 1.
335             TPR=TPR1/ANR1  @UPDATE AVERAGE TRAFFIC PER REGION
336             TPRL=TPR*(1. -ZETA)  @UPDATE LOWER LIMIT
337       909   CONTINUE
338           GOTO 703
339       726 CONTINUE
340       C
341       C ONE REGION CASE
```

```
342       C
343             DO 727 NN=1,N1
344               SVR(NN) = 1
345       727   CONTINUE
346             NUMPR(1) = N1
347       703   CONTINUE
348       C
349       C SELECT REGIONAL SWITCHING CENTER
350       C
351             DO 500 J=1,NR1
352               WCASE = 1.0E12
353               NMBR = 0
354               DO 505 K=1,N1
355                 IF(SVR(K) .NE. J) GO TO 505
356                 NMBR = NMBR + 1
357                 NUMR(NMBR) = K
358                 NUMRR(NMBR)=INDXPT(K)
359                 DO 490 I=1,4
360                   NUMB(NMBR,I)=NAMEST(K,I)
361       490       CONTINUE
362       505     CONTINUE
363       C
364       C PRINT OUT PID AND NAMES FOR SYSTEM TERMINATIONS IN THE REGION J
365       C
366             WRITE(IWT,1018) J,(NUMRR(I),(NUMB(I,I1),I1=1,4),I=1,NMBR)
367       C
368       C PRINT OUT INDICES OF SYSTEM TERMINATIONS IN THE REGION J
369       C
370             WRITE(IWT,1028) (NUMR(I),I=1,NMBR)
371       C
372             IF(NRSC(J) .NE. 0) GO TO 501  @PPE-SELECTED
373             DO 520 K=1,NMBR
374               NN1 = NUMR(K)          @ASSUMED RSC
375               SUMT = 0.0
376               DO 530 L=1,NMBR
377                 NN2 = NUMR(L)
378                 SUMT=SUMT+(TRAFON(NN2)+TRAFIT(NN2))*DIST(NN2,NN1)
379       530       CONTINUE
380               IF(SUMT .GT. WCASE) GO TO 520
381               WCASE = SUMT
382               NRSC(J) = NN1
383       520     CONTINUE
384       501     CONTINUE
385              NN4=NRSC(J)
386              WRITE(IWT,1003) INDXPT(NN4),J
387              IGO=IACTH(J,2) @IF 1, OPTIMIZATION IS REQUIRED
388               CALL RGNNET(J,NMBR,NUMR,IGO,NUMRR)
389       500   CONTINUE
390       C
391       C GENERATE INTER-REGION ORIGIN-DESTINATION MATRIX
392       C
393       C INITIALIZATION
394       C
395             IF(NR1.LE.2) GOTO 551
396             DO 902 K1=1,N7
397               KKK=NBASE(K1)
398               KKK=LOCAL(KKK)
```

```
399             IF(KKK .EQ. 0) WRITE(6,7777) K1
400             NBASE(K1)=SVR(KKK)
401       902   CONTINUE
402             DO 699 K1=1,NR1
403             DO 699 K2=1,NR1
404               TRM(K1,K2)=0.
405               TRM(K2,K1)=0.
406       699   CONTINUE
407             DO 900 J=1,NR1
408               NMBR = 0
409               DO 905 K=1,N1
410                 IF(SVR(K) .NE. J) GO TO 905
411                 DO 915 KK=1,N7
412                   NN2=NBASE(KK)  @REGIONAL INDEX FOR KK'S DATA BASE
413                   TRM(J,NN2)=TRAFD(K,2,KK)+TRM(J,NN2)  @OUTGOING TRAFFIC
414                   TRM(NN2,J)=TRAFD(K,1,KK)+TRM(NN2,J)  @INCOMING TRAFFIC
415       915         CONTINUE
416       905       CONTINUE
417               DO 920 J1=1,NR1
418                 NN2 = NRSC(J1)
419                 DRM(J,J1) = DIST(NN1,NN2)
420       920       CONTINUE
421       900       CONTINUE
422             NTURN=NR1/10+1
423             DO 535 L=1,NTURN
424               LL=(L-1)*10+1
425               LU=L*10
426               IF(LU.GT.NR1) LU=NR1
427               WRITE(IWT,1030) NR1,NR1,(K,K=LL,LU)
428               DO 1022 I=1,NR1
429                 WRITE(IWT,1021) I,(TRM(I,J),J=LL,LU)
430       1022    CONTINUE
431       535   CONTINUE
432             DO 545 L=1,NTURN
433               LL=(L-1)*10 + 1
434               LU=L*10
435               IF(LU .GT. NR1) LU=NR1
436               WRITE(IWT,1031) NR1,NR1,(K,K=LL,LU)
437               DO 1024 I=1,NR1
438               WRITE(IWT,1021) I,(DRM(I,J),J=LL,LU)
439       1024    CONTINUE
440       545   CONTINUE
441       C     CALL IRNOP(NR1,NLIMIT,TRM)
442       74    FORMAT(//,40X,' TOTAL TRAFFIC=',F9.2)
443       75    FORMAT(//,35X,'TOTAL SYSTEM TRAFFIC=',F9.2)
444       220   FORMAT('1ASSUME NUMBER OF REGIONS',
445            *      /' ENTER NR AND STRIKE RETURN KEY')
446       735   FORMAT(I3)
447       888   FORMAT(1X,'TYPE IN PRELOADED SYSTEM TERMN. AND PSC WITH',
448            1   /,1X,'FORMAT I1,1X,A4,A5')
449       800   FORMAT(I1,1X,A4,I5)
450       804   FORMAT(10X,I1,2X,A4,I5)
451       3     FORMAT(1X,I8,'  DISTANCE ITEMS ARE OVERSIZED')
452       210   FORMAT(' ASSUME A SYSTEM  CENTROID',
453            *      /' ENTER CODE FOR NSCC AND STRIKE RETURN KEY')
454       734   FORMAT(A4)
455       4013  FORMAT(' THE GIVEN SYSTEM COMM. CENTROID IS NOT OK, RETYPE IT')
```

```
2101  FORMAT(1X,'INPUT TOTAL NO. OF TRANSACTIONS AND NO. OF ACCESS ',
     1  'AT THE SWITCHER',/,1X,'ENTER WITH F8.5 AND I3 (UNDER XSAC/SEC')
2102  FORMAT(F8.5,I3)
2103  FORMAT(1X,'READ IN LIMITS ON NO. OF SYS. TERMS. ON A LINE',/,
     1 ' ,RESPONSE TIME REQD AND NO. OF PROCESSORS WITH FORMAT ',
     2 /,' I3,F5.2,I2')
2104  FORMAT(I3,F5.2,I2)
2105  FORMAT(1X,'IF PLOTTING IS REQUIRED, TYPE 1 WITH FORMAT I3')
110   FORMAT((/,1X,A6,2X,15F8.1))
111   FORMAT(1H1,50X,'TRAFFIC MATRIX(BPS) ')
113   FORMAT( //,1X,'TERM. ',4X,15(4X,A4), / )
109   FORMAT(1H1,40X,'POINT TO POINT DISTANCE MATRIX ',
     * //,1X,'TERM.',15(4X,A4), / )
112   FORMAT((2X,A4,15F8.0))
1001  FORMAT((1X,5(6X,A4,2X,F10.3),/))
1003  FORMAT(//,1X,'RSC=',2X,A4,' FOR REGION',I5)
1011  FORMAT(/,10X,' THERE ARE',I5,' REGIONS',/)
1013  FORMAT(1H1,35X,' TOTAL TRAFFIC ORIGINATED FROM SYS. TERMN. ',
     * '(BITS/SEC)',/)
1014  FORMAT(/,35X,' TOTAL TRAFFIC DESTINATED TO SYS. TERMN. ',
     * '(BITS/SEC)',/)
1015  FORMAT(/,10X,' NCC= ',2X,A4,/)
1018  FORMAT(1H1,' REG=',I3,' , SYS. TERMN.=',//,(1X,4(A4,1X,4A6)))
7777  FORMAT(1X,'THE',I3,'TH DATA BASE IS NOT GVN AS A SYS. TERM.')
1021  FORMAT(//,(15X,I5,5X,10F10.3,/))
1028  FORMAT(//,' INDICES FOR SYS. TERM.=',(/,30I4))
1030  FORMAT('1',/,30X,'INITIAL INTERREGION TRAFFIC MATRIX (',
     * I2,' X',I2,')',///,21X,10(5X,I5))
1031  FORMAT('1',/,30X,'INTERREGION DISTANCE MATRIX (',I2,' X',I2,')',
     * 'MILES',///,21X,10(5X,I5))
551   CONTINUE
      STOP
      SUBROUTINE FINDD(N,M)
C          **********************
C
C FIND THE NEXT SYSTEM TERMINATION M WHICH IS CLOSEST TO N
C WHERE M HAS NOT BEEN ASSIGNED TO ANY REGION YET
C
C          **********************
      AMIN=20000.
      M=0
      DO 708 K=1,N1
        IF(SVR(K) .NE. 0) GOTO 708
        ADIST=DIST(N,K)
        IF(ADIST .GE. AMIN) GOTO 708
        AMIN= ADIST
        M=K
708   CONTINUE
      RETURN
      SUBROUTINE CREADK
C          **********************
C READ IN ACTIONS REGARDING INSERTIONS OF SYSTEM TERMINATIONS
C TO PRELOADED REGIONS AND REGIONAL NETWORK OPTIMIZATIONS
C          **********************
      WRITE(6,94)
94    FORMAT(' TYPE IN ACTION INDICES FOR EACH REGION ',
     * /,' 1ST ELEMENT: 1= INSERTION TO THIS PRELOADED REGION IS OK ',
```

```
513           * /,' 2ND ELEMENT: 1= OPTIMIZATION IS NEEDED')
514            DO 200 NN1=1,NR1
515            DO 200 NN2=1,2
516              IACTN(NN1,NN2)=0
517       200  CONTINUE
518            WRITE(6,206)
519       206  FORMAT(' TYPE IN REGION INDEX AND ACTION NUMBER NEEDED',
520           * /,' WITH FORMAT 2I2 AND END IT WITH A 0 0')
521       250  CONTINUE
522            READ(5,201) NREG, NCODE
523       201  FORMAT(2I2)
524            IF(NREG .EQ. 0) GOTO 265
525            IF(NREG.GT.NR1 .OR. NCODE .GT. 2) GOTO 260
526            IACTN(NREG,NCODE)=1
527            GOTO 250
528       260   CONTINUE
529            WRITE(6,202)
530       202   FORMAT(' PLEASE RETYPE THE INPUT')
531            GOTO 250
532       265  CONTINUE  @NO MORE INPUT
533            RETURN
534            SUBROUTINE CREADA(N1)
535       C           ********************
536       C
537       C FUNCTIONS OF THIS SUBROUTINE ARE TO
538       C     1. RECEIVE TOTAL NO. OF SYSTEM TERMINATIONS, DATA BASES AND CITIES
539       C     2. RECEIVE CITY LOCATIONS (V & H)
540       C     3. RECEIVE PID NO., SYS. TERM. NAMES, ADDR. MAPPING AND TRAFFICS
541       C
542       C           ********************
543            WRITE(6,81)
544       81   FORMAT(' TYPE IN NO. OF SYS. TERMNS, DATA BASES AND CITIES '
545           *   'WITH FORMAT 3I5')
546            READ (5,10) N1,N7,NCITY  @ NUMBER OF SYSTEM TERMINATIONS
547            WRITE(6,78) N1,N7,NCITY
548       78   FORMAT(' THERE ARE ',I5,'SYS. TERMNS.,',I4,' DATA BASES',I5,
549           *' CITIES',/,' TYPE IN DATA BASE LOCATIONS WITH FORMAT 6(1X,A4)')
550            READ(5,15) (NBASE(I),I=1,N7)
551       15   FORMAT(6(1X,A4))
552            WRITE(6,16) N7, (NBASE(I),I=1,N7)
553       16   FORMAT(I5,' DATA BASES ARE AT',6(2X,A4))
554            WRITE(6,161)
555       161  FORMAT(' TYPE IN CITY V-H WITH FORMAT (33X,I5,2X,I5)')
556            READ(5,17)(IVERT(I),IHORZN(I),I=1,NCITY)
557       17   FORMAT((33X,I5,2X,I5))
558            WRITE(6,76)
559       76   FORMAT(' TYPE IN PID NO., NAME, MAPPING ADR. ',
560           * 'AND TRAFFIC',/,' WITH FORMAT I4,1X,4A6,I4,6F8.2')
561            DO 79 I=1,N1
562            READ(5,80)II,(NAMEST(I,J),J=1,4),IADD(I),MAPADR(I),((TRAFD(I,K,L),
563           *   K=1,2),L=1,N7)
564              INDXPT(I)=II
565       79   CONTINUE
566       80   FORMAT((A4,1X,3A6,A4,I2,I4,4F10.2/4F10.2))
567       10   FORMAT (3I5)
568            RETURN
569            SUBROUTINE CREADR(N2)
```

```
570       C            ********************
571       C
572       C CREATE A RATE APPLICATION MATRIX IRATEJ(N2,N2)
573       C
574       C            ********************
575             WRITE(6,83)
576       83    FORMAT(' TYPE IN NO. OF RATE STRUCTURES UNDER',
577            * /,' CONSIDERATION WITH FORMAT I3')
578             READ (5,50) N2
579             WRITE(6,84)
580       84    FORMAT(' TYPE IN RATE APPLICATION TO EACH COMBN. ',
581            * /,' WRT EACH SYS. TERM. WITH FORMAT 10I2')
582             DO 11 IRATE=1,N2
583               READ (5,100) (IRATEJ (J,IRATE),J=1,N2)
584       11    CONTINUE
585             WRITE(6,71)
586       71    FORMAT(' READ IN TRAFFIC DENSITY TYPE AND RATE STRUCTURE',
587            *  /,' FOR EACH CITY WITH FORMAT 80I1')
588             READ(5,72) ((IRAND(I,J),J=1,2),I=1,NCITY)
589       72    FORMAT((80I1))
590       50    FORMAT (I3)
591       100   FORMAT (10I2)
592             RETURN
593             SUBROUTINE CREADC(N3)
594       C            ********************
595       C
596       C READ IN NAMES, UTILIZATION FACTORS AND CAPACITY FIGURES
597       C FOR LINES TO BE USED IN THE SYSTEM
598       C
599       C            ********************
600             WRITE(6,85)
601       85    FORMAT(' TYPE IN NO. OF LINE TYPES APPICABLE WITH FORMAT I3')
602             READ (5,50) N3
603             WRITE(6,86)
604       86    FORMAT(' TYPE IN NAME, CAPACITY,  UTIL. FACTOR AVAIL. FOR ',
605            * /,' EACH LINE TYPE WITH FORMAT A6,1X,I6,1X,F3.2,2(1X,I1)')
606             DO 12 I=1,N3
607               READ(5,100)LINAME(I),LINCAP(I),UTILIZ(I),LINMIX(I),IDUPLX(I)
608       12    CONTINUE
609       50    FORMAT (I3)
610       100   FORMAT(A6,1X,I6,1X,F3.2,2(1X,I1))
611             RETURN
612             SUBROUTINE CREADD (N4)
613       C            ********************
614       C
615       C CREATE A MATRIX OF BASIC INSTALLATION AND RECURRING COSTS
616       C FOR CHARGEABLE ITEMS, ASSUMING COST IS A LINEAR FUNCTION
617       C
618       C            ********************
619             WRITE(6,87)
620       87    FORMAT(' TYPE IN NO. OF DEVICES  AND NAMES FOR EACH LINE TYPE',
621            * /,' WITH FORMAT I3/10(A6,1X)')
622             READ (5,50) N4, (NAMEHW(I),I=1,N4)
623             WRITE(6,88)
624       88    FORMAT(' TYPE IN INST. AND RECURR. COSTS WRT ',
625            * /,' RATE STRUCTURE, LINE TYPE, DEVICE, TRAFFIC DENSTIY ',
626            * /,' AND DUPLEXING MODE WITH FORMAT 2F9.2/2F9.2')
```

```
627           DO 13 IRATE=1,N2
628           DO 13 ILINE=1,N3
629           DO 13 IDVICE=1,N4
630           DO 13 IDNSTY=1,3
631           READ(5,100)((AINSTC(IRATE,ILINE,IDVICE,IDNSTY,J,K),K=1,2),J=1,2)
632           READ (5,100)((RECRC(IRATE,ILINE,IDVICE,IDNSTY,J,K),K=1,2),J=1,2)
633     13    CONTINUE
634     100   FORMAT (2F9.2/2F9.2)
635     50    FORMAT(I3/,10(A6,1X))
636           RETURN
637           SUBROUTINE CREADE
638     C           **********************
639     C
640     C CREATE A MATRIX OF BASIC INSTALLATION AND RECURRING COSTS FOR
641     C LINES. COST MAY OR MAYNOT BE A LINEAR FUNCTION OF DISTANCE
642     C
643     C           **********************
644           WRITE(6,89)
645     89    FORMAT(' TYPE IN INST. COSTS FOR LINES WRT ',
646          1 /,' RATE, LINE, DENSITY, AND DUPLEXING MODE ',
647          2 /,' WITH FORMAT 4F9.2')
648           WRITE(6,90)
649     90    FORMAT(' TYPE IN INDEX FOR LINEARITY OF LINE RECUR. COST',
650          1 /,' FUNCTION WITH 1=LINEAR AND NONLINEAR OTHERWISE',
651          2 /,' WITH FORMAT I1 FOR EACH LINE TYPE')
652           WRITE(6,91)
653     91    FORMAT(' TYPE IN RECUR. COSTS WITH FORMAT 4F9.2 IF LINEAR ',
654          1 /,' WITH FORMAT 10F8.3/10F8.3 IF NONLINEAR',
655          2 /,' IF NONLINEAR, USE 10F8.2')
656           DO 14 IRATE=1,N2
657           DO 14 ILINE=1,N3
658             READ(5,200) INDEX
659             IFLAG(IRATE,ILINE)=INDEX @LINE COST LINEARITY INDICATOR
660           DO 14 IDNSTY=1,3
661             READ (5,100) ((ANSTLN(IRATE,ILINE,IDNSTY,I,J),J=1,2),I=1,2)
662             IF (INDEX.NE.1) GO TO 3
663     C     LINEAR COST FUNCTION
664             READ (5,100) ((RECRLN(IRATE,ILINE,IDNSTY,I,J),J=1,2),I=1,2)
665             GO TO 14
666     3     CONTINUE  @ NONLINEAR COST FUNCTION
667           READ (5,401) ((RECRLN(IRATE,ILINE,IDNSTY,I,J),J=1,16),I=1,2)
668     14    CONTINUE
669     100   FORMAT ((4F9.2))
670     200   FORMAT (I1)
671     401   FORMAT ((10F8.3/10F8.3))
672           RETURN
673           FUNCTION LOCAL(NL)
674     C           ********************
675     C
676     C FIND LOCAL INDEX FOR SYSTEM TERMINATION WITH TO NL
677     C
678     C           ********************
679           LOCAL=0
680           IF(NL.EQ.0) RETURN
681           DO 4001 NN=1,N1
682             IF(INDXPT(NN).EQ.NL) GOTO 4002
683     4001  CONTINUE
```

```
      RETURN
4002  LOCAL=NN
      RETURN
      SUBROUTINE OVERFL(J,K)
C           ********************
C
C STORE OVERFLOW ELEMENT (J,K) AT LOCATION IOVER OF TABLE
C LA AND PUT A MARK 511 AT LOCATION J OF TABLE MA (DSTNCE)
C
C           ********************
      IF(IOVER1 .GE. NMAX) GOTO 8000
      CALL PACK (J,511,1,DSTNCE)
      IVRD(IOVER1,1)=J
      IVRD(IOVER1,2)=K
      IOVER1=IOVER1+1
      RETURN
8000  CONTINUE
      WRITE(6,8001)
8001  FORMAT(2X,' THE OVERFLOW TABLE HAS BEEN FULLY LOADED',
     *   /,2X,' PLEASE INECREASE ITS SIZE')
      STOP
      SUBROUTINE CREADR
C           ********************
C
C RECEIVE DATA FOR RESPONSE TIME CALCULATION
C
C MSLIN(NP9,2)= INPUT MSG LENGTH  AS A FUNCTION OF TYPE AND PRIORITY
C MSLOUT(NP9,2)=OUTPUT MSG LENGTH AS A FUNCTION OF TYPE AND PRIORITY
C AMSL(7)= AVERAGE MSG LENTH FOR
C     1=POLLING   2=NAK RESPONSE   3=INPUT MSG WITH PRIORITY 1
C     4=INPUT MSGS    5=OUTPUT MSG WITH PRIO 1
C     6=OUTPUT MSGS WITH PRIO 2   7=ALL MSGS
C TIMXMT(7,NP3)=AVERAGE TRANSMISSION TIME FOR ABOVE ITEMS
C RATPRI(NP9,2,2)=OUTPUT MSG DISTRBUTN AND OUT-GOING MSG RATIO BY PRIO
C     N,1,1 = PERCENT OF OUTPUT MSG SENT'D WITH PRIO 1 IF ITS TYPE IS N
C     N,1,2 = PERECNT OF OUTPUT MSG WHOSE DESTINATION IS OUTSIDE OF
C RATIOI(NP9,2)=INPUT TRAFFIC DISTRI. AS A FUNCTION OF TYPE AND PRIORITY
C RATIO(NP9,2) =OUTPUT TRAFFIC DISTRI. AS A FUNCTION OF TYPE AND PRIORITY
C
C           ********************
      DIMENSION MSLIN(NP9,2),RATIO(NP9,2),RATIOI(NP9,2),
     1          MSLOUT(NP9,2),MSGNAM(NP9),
     2          NPL(NP3),NAK(NP3),NPLOH(NP3),NAKOH(NP3),
     3          MOH(NP3),TAMDM(NP3),TAPD(NP3)
      WRITE(6,771)
771   FORMAT(' TYPE IN NPL, NAK, NPLOH, NAKOH, MOH,',
     *  /,' TAMDM, TAD IN FORMAT (5I4,2F7.5)')
      READ(5,77) (NPL(I),NAK(I),NPLOH(I),NAKOH(I),
     * MOH(I),TAMDM(I),TAPD(I),I=1,N3)
      WRITE(IWT,73)(LINCAP(I),NPL(I),NAK(I),NPLOH(I),NAKOH(I),
     *  MOH(I),TAMDM(I),TAPD(I),I=1,N3)
73    FORMAT((2X,I5,3X,'POLL CHAR.=',I4,' NAK CHAR.=',I4,' POLL O/H=',
     1  I4,/,10X,'NAK O/H=',I4,'  MSG O/H=',I4,
     2  /,10X,'MPSEM=',F8.3,' PPSEM=',F8.3))
77    FORMAT((5I4,2F7.5))
      WRITE(6,772)
772   FORMAT(' TYPE IN NO. OF MSG TYPES, AND TRAFFIC STATISTICS',
```

```
741           1 ,/,' SUCH AS MSGNAM, MSLIN, MSLOUT, RATIO WITH',
742           2 ,/,' FORMAT I4/(A6,2(2I4,2F6.3))')
743            READ(5,77) NTYP
744            READ(5,179) (MSGNAM(I),(MSLIN(I,J),MSLOUT(I,J),
745           *  RATIOI(I,J),RATIO(I,J),J=1,2),I=1,NTYP)
746       179  FORMAT((A6,2(2I4,2F6.3)))
747            READ(5,81) CPUAVG
748       81   FORMAT(F7.4)
749       C
750       C CALCULATE AVERAGE MSG LENGTH
751       C
752            DO 61 I=3,7
753              AMSL(I)=0.
754       61   CONTINUE
755            DO 58 I=1,4
756              ASUM(I)=0.
757       58   CONTINUE
758            DO 62 I=1,NTYP
759            DO 66 J=1,2
760              J1=J+2
761              J2=J+4
762              AMSL(J1)=AMSL(J1)+MSLIN(I,J)*RATIOI(I,J)
763              AMSL(J2)=AMSL(J2)+MSLOUT(I,J)*RATIO(I,J)
764              ASUM(J)=ASUM(J)+RATIOI(I,J)
765              ASUM(J1)=ASUM(J1)+RATIO(I,J)
766       66     CONTINUE
767       62   CONTINUE
768            BSUM=0.
769            DO 67 I=1,4
770            J1=I+2
771            AMSL(7)=AMSL(7)+AMSL(J1)
772            BSUM=BSUM+ASUM(I)
773       67   CONTINUE
774            AMSL(7)=AMSL(7)/BSUM   @OVERALL AVG. MSG LENGTH
775            IF(ASUM(4) .EQ. 0.) GOTO 68
776            AMSL(6)=AMSL(6)/ASUM(4)
777       68     CONTINUE
778            AMSL(5)=AMSL(5)/ASUM(3)   @AVG. MSG LENTH FOR PRIO=1
779            AMSL(4)=(AMSL(3)+AMSL(4))/(ASUM(1)+ASUM(2)) @AVG INPUT MSG LEN.
780            AMSL(3)=AMSL(3)/ASUM(1)
781            WRITE(IWT,105) (AMSL(I),I=3,7)
782       105  FORMAT(/,5X,'AVG. INPUT MSG WITH PRIO 1=',F6.1,' CHARS',
783           1       /,5X,'AVG. INPUT MSG           ',F6.1,' CHARS',
784           2       /,5X,'AVG. OUTPUT MSG W/H PRIO 1=',F6.1,' CHARS.',
785           3       /,5X,'AVG. OUTPUT MSG WITH PRIO2=',F6.1,' CHARS',
786           4       /,5X,'     OVERALL AVG. MSG     =',F6.1,' CHARS')
787            DO 65 K=1,N3
788            AMSL(1)=NPL(K)
789            AMSL(2)=NAK(K)
790              TIMXMT(1,K)=(AMSL(1)+NPLOH(K))*8./LINCAP(K)+TAMDM(K)
791              TIMXMT(2,K)=(AMSL(2)+NAKOH(K))*8./LINCAP(K)+TAPD(K)
792              DO 63  J=3,7
793                TIMXMT(J,K)=(AMSL(J)+MOH(K))*8./LINCAP(K)+TAMDM(K)
794       63       CONTINUE
795       65   CONTINUE
796            BSUM=ASUM(3)+ASUM(4)
797            RETURN
```

```
      SUBROUTINE CWAITC
C           ******************
C
C PPF-CALCULATE CPU WAIT TIME
C
C           ******************
      RHOCPU=XSAC*CPUAVG/MPROC
      WRITE(100,850) RHOCPU
850   FORMAT(' CPU UTILIZATION PER PROCESSOR IS ', F5.3)
      IF(RHOCPU .LE. .8) GOTO 851
      WRITE(6,855)
855   FORMAT(' THE CPU IS OVERLOADED, THEREFORE IT IS NO USE TO ',
     * 'GO FURTHER.')
      STOP
851   CONTINUE
      BETA=RHOCPU
      IF( MPROC .EQ. 1) GOTO 700
      RH02=RHOCPU**2
      BETA=2.*RH02/(1+RHOCPU)
      IF( MPROC .EQ. 2) GOTO 700
      RH04=RHOCPU**4
      BETA=256*RH04/(24+72*RHOCPU+96*RH02
     *                +64*RH02*RHOCPU-240*RH04)
700   CONTINUE
      WAIT(4)=CPUAVG*(BETA/(MPROC*(1.-RHOCPU))+1)
      WAIT(4)=WAIT(4)*NREQSW
      RETURN
      END
@PRT STACOM.RGNNET/0777
```

```
51928*STACOM(1),RGNNET/0777
      1            SUBROUTINE RGNNET(JREGN, NOREGN, NUMP, IGO, NUMRR)
      2      C           ********************
      3      C
      4      C DEVELOP A REGIONAL MULTIDROP NETWORK, STARTING WITH A STAR
      5      C NETWORK AND THEN OPTIMIZE IT BY ESAU-WILLIAMS METHOD, GIVEN
      6      C THE FOLLOWING AUGUMENTS:
      7      C
      8      C  JREGN= THE INDEX FOR THE REGION UNDER CONSIDERATION
      9      C NOREGN= THE NUMBER OF SYSTEM TERMINATIONS IN REGION JREGN
     10      C   NUMR= AN ARRAY THAT CONTAINS INDICES FOR ALL SYSTEM
     11      C         TERMINATIONS IN REGION JREGN
     12      C    IGO= 1 IF NETWORK OPTIMIZATION IS TO BE PERFORMED
     13      C
     14      C    NOTE: NODE AND SYSTEM TERMINATION ARE EXCHANGEABLE
     15      C
     16      C           ********************
     17            PARAMETER NP1=130, NP2=1, NP3=4, NP4=3
     18            PARAMETER NP7=4
     19            PARAMETER IWT=100, MW=4, NPC=360
     20            PARAMETER NP6=(NPC*NPC/2-NPC+1)/4+1
     21            COMMON/CONST/ N1,N2,N3,N4,N7,NCITY
     22           *   /REF/IREF(NPC),TRAFD(NP1,2,NP7),DSTNCE(NP6),MAPADR(NP1)
     23           *   /LINCHR/ LINMIX(NP3), LINCAP(NP3), UTILIZ(NP3)
     24           *   /INF/ IRATEJ(NP2,NP2),IRAND(NPC,2),IFLAG(NP2,NP3)
     25           *   /EIN/ SVR(NP1),NRSC(MW),NUMPR(MW),TRAFON(NP1),TRAFTT(NP1)
     26           *   /NAME/NAMEST(NP1,4),LINAME(NP3),NAMEHW(NP4)
     27           *   /SUM/ ASUM(4),RSUM
     28           *   /MSLA/ AMSL(7)
     29           *   /ROUND/ NTERMS,TIMREQ,MPROC,MPLOT
     30           *   /ADD/ IADD (NP1),KCHG,KADD
     31           *   /RESP/ RHOLIN(4),RSPTIM
     32            DIMENSION COSTEW(NP1,4),IARRAY(NP1,5),ARRAY(NP1,2)
     33            DIMENSION TIMRSP(NP1),TRFSUM(NP1,2),TIMOUT(NP1)
     34            DIMENSION NLINES(NP1,NP3), LDUMMY(NP3), NUMR(1)
     35            DIMENSION IDST(NP1),LNKCHW(NP3,NP4,2),LNKCLN(NP3,2)
     36            DIMENSION ICSTHW(NP1,NP4,2),ICSTLN(NP1,2),ITCOST(NP1,2)
     37            DIMENSION LSUB(NP1),MSUB(NP1),NSUB(NP1)
     38            DIMENSION IBLANK(NP1),JCHAR(2)
     39            DIMENSION NUMRR(1)
     40            DIMENSION RHOF(NP1)
     41            EQUIVALENCE (JCHAR,ICHAR)
     42            DATA JBLANK/'      '/
     43            RSPTIM=0.
     44            IPOINT=0
     45            INTEGER TCOST1,TCOST2,COST,COSTEW
     46      C
     47      C INITIALIZE COST ARRAY, INDEPENDENT OF LINE TYPE
     48      C
     49            DO 399 K1=1,N1
     50            DO 399 K3=1,2
     51              ICSTLN(K1,K3)=0
     52            DO 399 K4=1,N4
     53              ICSTHW(K1,K4,K3)=0
     54      399   CONTINUE
     55            NN1=NRSC(JREGN) @GLOBAL INDEX FOR RSC
     56      C
```

```
C FIND THE LOCAL RSC INDEX IN THE REGION ARRAY
C
      DO 98 IND=1,NOREGN
        IF(NN1 .EQ. NUMR(IND)) GOTO 189
 98   CONTINUE
189   CONTINUE
      IRSC=IND
C
C BUILD A STAR NETWORK
C
      CALL STAREW
C
C PRINT OUT STAR NETWORK
C
      CALL SUMPRT(NOREGN,1)
      IF( IGO .EQ. 0) GOTO 979
C
C DEVELOP A MULTIDROP NETWORK UTILIZING THE
C ESAU-WILLIAMS ALGORITHM
C
      MAXSAV=0
      MAXM=0
      MAXL=0
      MAXK=0
      MAXKI=0
      MAXLIN=0
      MAXNOL=0
      LINNEW=0
      RSPMAX=0.
      RHOMAX=0.
      ICHAR=' '
      ITALLY=0
      JTALLY=0
      KCHG=1
      KADD=1
      IOK=0
      INTRY=0
      CALL ESSWIL
      WRITE(6,933) ITALLY,JTALLY
933   FORMAT(2X,'TRYLNK HAS BEEN ACCESSED FOR ',I9,' TIMES',
     1  /,2X,'UPNETW HAS BEEN ACCESSED FOR ',I9,' TIMES')
979   CONTINUE
      RETURN
      SUBROUTINE STAREW
C           ********************
C
C FORM THE INITIAL REGIONAL STAR NETWORK, IARRAY, AND FIND ITS
C COST, COSTEW
C NOREGN=NUMBER OF SYSTEM TERMINATIONS IN THE REGION
C
C           ********************
      INTEGER COST
      DO 100 K3=1, NOREGN
        DO 110 K4=2,4
          IARRAY(K3,K4)=0
110     CONTINUE
        KK=NUMR(K3)
```

```
114               IARRAY(K3,1)=IADD(KK)
115               IARRAY(K3,5)=IRSC   @ LOCAL INDEX FOR RSC
116               ARRAY(K3,1)=TRAFON(KK)
117               ARRAY(K3,2)=TRAFIT(KK)
118               TIMRSP(K3)=0.
119       100   CONTINUE
120             IARRAY(IRSC,1)=NOREGN - 1  @NO. OF NODES UNDER RSC
121             NM=1   @ASSUMING THE 1ST SUCCESSOR WITH INDEX 1
122             IF(IRSC .EQ. 1) NM=2  @1ST SUCCESSOR IS WITH INDEX 2
123             IARRAY(IRSC,2)=NM
124             IARRAY(IRSC,5)=0
125       C
126       C RELATE ALL OF RSC'S SUCCESSORS
127       C
128             DO 200 K5=1, NOREGN
129               IF(K5 .EQ. IRSC) GOTO 200
130               NM=NM + 1
131               IF(NM .EQ. IRSC) NM=NM + 1
132               IF(NM .GT. NOREGN) GOTO 200  @END OF SUCCESSORS' LINK
133               IARRAY(K5,3)=NM
134               IARRAY(NM,4)=K5
135       200   CONTINUE
136       C
137       C DETERMINE LINE TYPE FOR CENTRAL LINKS TO RSC
138       C
139             DO 550 NODE=1, NOREGN
140               IF(NODE .EQ. IRSC) GOTO 555
141               TRFIN=ARRAY(NODE,1)+0.5
142               TRFOUT=ARRAY(NODE,2)+.5
143               NN2=NUMR(NODE)
144               DSTN=DIST(NN1,NN2)
145               IDST(NODE)=DSTN
146               COSTEW(NODE,4)=DSTN
147       C
148       C TAKE A FIRST GUESS FOR LINE CONFIGURATION
149       C
150               COST=0
151               RHO=0.
152               MDROP=IARRAY(NODE,1)+1
153               CALL LINNUM(TRFIN,TRFOUT,LDUMMY,LINOLD,0,RHO)
154       781   CONTINUE
155       C
156       C COMPUTE INITIAL REPONSE TIME
157       C
158               IKONT=0
159               DO 783 I=1,N3
160               IF(LDUMMY(I) .NE. 0) IKONT=IKONT+1
161       783     CONTINUE
162               AINTRF=TRFIN
163               OUTTRF=TRFOUT
164               IF(IKONT .EQ. 1 .AND. LDUMMY(LINOLD) .EQ. 1) GOTO 772
165       C
166       C RESPONSE TIME CALCULATION NEEDS MODIFICATION
167       C
168               ACAP=0.
169               DO 771 NL=1,N3
170               ACAP=ACAP+LINCAP(NL)*LDUMMY(NL)
```

```
171                LDUMMY(NL)=0
172          771   CONTINUE
173                LDUMMY(LINOLD)=1
174                AINTRF=TRFIN*LINCAP(LINOLD)/ACAP @PERCENT TRAFFIC
175                OUTTRF=TRFOUT*LINCAP(LINOLD)/ACAP
176          772   CONTINUE
177                CALL RSPNSE(AINTRF,OUTTRF,LINOLD,MDROP,IOK)
178                IF(IOK .EQ. 1) GOTO 773
179                IF(LINOLD .EQ. N3) GOTO 774
180                LDUMMY(LINOLD)=0
181                LINOLD=LINOLD+1
182                LDUMMY(LINOLD)=1
183                GOTO 775
184          774   CONTINUE
185                NLL=0
186                N33=N3-1
187                DO 776 I=1,N33
188                  IF(LDUMMY(I) .EQ.0 ) GOTO 776
189                  NLL=I
190                  GOTO 780
191          776   CONTINUE
192                LDUMMY(1)=1
193                GOTO 775
194          780   CONTINUE
195                LDUMMY(NLL)=0
196                LDUMMY(NLL+1)=LDUMMY(NLL+1)+1
197          775   CONTINUE
198                CALL RHOFUN(TRFIN,TRFOUT,LDUMMY,LINOLD,RHOLIN,RHO)
199                GOTO 781
200          773   CONTINUE
201                TIMRSP(NODE)=RSPTIM
202                KCHG=2
203                CALL ISUMUP(IRSC,NODE,0,COST)
204                RHOF(NODE)=RHO
205                COSTFW(NODE,1)= COST
206                COSTFW(NODE,2)=LINOLD
207                DO 499 NL=1,N3
208                  NLINES(NODE,NL)=LDUMMY(NL)
209                DO 499 NM=1,2
210                ICSTLN(NODE,NM)=ICSTLN(NODE,NM)+LNKCLN(NL,NM)
211                DO 499 NK=1,N4
212                   ICSTHW(NODE,NK,NM)=LNKCHW(NL,NK,NM)+ICSTHW(NODE,NK,NM)
213          499   CONTINUE
214                JTRAF=TRFIN+TRFOUT
215                JTRAF=JTRAF/UTILIZ(LINOLD)
216                COSTFW(NODE,3)=JTRAF/LINCAP(LINOLD)+1
217                GOTO 550
218          555   CONTINUE
219          C
220          C ASSUMING TRAFFIC AT IRSC IS TAKEN CARE OF AUTOMATICALLY
221          C
222                DO 498 NL=1,N3
223                  NLINES(NODE,NL)=0
224          498   CONTINUE
225                COSTFW(NODE,1)=0
226                COSTFW(NODE,2)=0
227                COSTFW(NODE,3)=0
```

```
228               COSTFW(NODE,4)=0
229               RHOF(NODE)=0.
230         550   CONTINUE
231               RETURN
232               SUBROUTINE ISUMUP(L1,L2,LT,IC)
233         C           **********************
234         C
235         C CALCULATE COST BETWEEN NODES L1 AND L2 AND ADD IT TO
236         C TOTAL COST IC WHERE LT=LINE TYPE
237         C
238         C           **********************
239               LL1=NUMR(L1)
240               LL2=NUMR(L2)
241               CALL ICOSTJ(LDUMMY,LL1,LL2,LNKCHW,LNKCLN)
242               KK=N3
243               IF(LT.NE.0) KK=1
244               DO 211 LINTYP=1,KK
245                 LTYP=LINTYP
246                 IF(KK.EQ.1) LTYP=LT
247                 DO 221 I1=1,2
248                   IC=IC+LNKCLN(LTYP,I1)
249                   DO 222 I2=1,N4
250                     IC=IC+LNKCHW(LTYP,I2,I1)
251         222       CONTINUE
252         221     CONTINUE
253         211   CONTINUE
254               RETURN
255               SUBROUTINE ESSWIL
256         C           **********************
257         C
258         C TRY AGAIN TO OPTIMIZE THE NETWORK
259         C
260         C           **********************
261         5000  CONTINUE
262               K=IARRAY(IRSC,2)   @FIRST SUBNETWORK UNDER RSC
263               KNEXT=IARRAY(K,3)  @NEXT SUBNETWORK UNDER RSC
264               IF(KNEXT .EQ. 0) GOTO 599 @ONLY ONE SUBNETWORK
265         560   CONTINUE
266               IOK=0
267               L=IARRAY(IRSC,2) @K-SUBNET IS TO BE LINKED TO L-SUBNET
268         570   CONTINUE
269               IF(L .NE. K) GOTO 575
270               L=IARRAY(L,3)
271               IF(L.EQ.0) GOTO 660
272         575   CONTINUE
273               K1=NUMR(K)
274               DREF=DIST(NN1,K1)
275         C
276         C TEST TOTAL NO. OF TERMINALS IF K AND L ARE COMBINED
277         C
278               INTRY=0   @INDICATION OF ENTRY TO TRYLNK
279               LINE=COSTFW(K,2)
280               IF(LINCAP(LINE) .EQ. 9600) GO TO 585 @NO MULTIDROPPING ON 9600
281               NODET=IARRAY(K,1)+IARRAY(L,1)+2
282               IF(NODET .GT. NTERMS) GOTO 585 @TOO MANY TERMINALS
283               M=L
284               K1=K
```

```
      580   CONTINUE
            M1=NUMR(M)
            DTRY=DIST(K1,M1)/2.
            IF(DTRY .GT. DREF) GOTO 140
            CALL TRYLNK(K,KI,L,M) @ M IS THE INSERTION NODE
            IF(IOK .EQ. 0) GOTO 585
      140   CONTINUE
            M=NXTNOD(L,M)  @NEXT NODE UNDER M ON L-SUBNET
            IF(M .NE. 0) GOTO 580  @NO MORE NODES UNDER M ON L-SUBNET
            KI=NXTNOD(K,KI)  @START WITH NEXT NODE ON K-SUBNET
            IF(KI .EQ. 0) GOTO 585
            K1=NUMR(KI)
            M=L
            GOTO 580
      585   CONTINUE
            L=IARRAY(L,3)  @NEXT SUCCESSOR
            IF(L .NE. 0) GOTO 570
      660   CONTINUE
            K=IARRAY(K,3)
            IF(K .NE. 0) GOTO 560  @NOT AN END YET, REPEAT THE SEARCH
      C
      C ALL POSSIBLE COMBINATIONS HAVE BEEN TRIED
      C
            IF(MAXSAV .LE. 0) GOTO 599  @NO NEED TO GO FURTHER
      C
      C UPDATE NETWORK BASED ON UP-TO-DATE MAXIMUM COST SAVING
      C PARAMETERS
      C
            JTALLY=JTALLY+1
            CALL UPNETW
      C
      C REINITIALIZATION
      C
            RSPMAX=0.
            MAXSAV=0
            MAXK=0
            MAXL=0
            MAXM=0
            MAXKI=0
            MAXLIN=0
            LINNEW=0
            MAXNOL=0
            RHOMAX=0.
            GOTO 5000
      599   CONTINUE
      C
      C PRINT OUT COSTS FOR THE OPTIMIZED MULTIDROP NETWORK
      C
            CALL MUTDRP
      C
      C PRINT OUT THE OPTIMIZED MULTIDROP NETWORK
      C
            CALL NETPRT
            IF( MPLOT .NE. 1) GOTO 50
            CALL CALPLT
      50    CONTINUE
            RETURN
```

```
      SUBROUTINE TRYLNK(KL,KIL,LL,ML)
C           ********************
C
C TRY TO ELIMINATE CENTRAL LINK KL AND LINK IT TO THE SUBNETWORK
C LL THROUGH SYSTEM TERMINATIONS KIL AND ML.
C
C           ********************
      INTEGER  COSTKM,COST
      ITALLY=ITALLY+1
      IF(INTRY .EQ. 1) GOTO 719
      TRFIN=ARRAY(KL,1)+ARRAY(LL,1)+0.5
      TRFOUT=ARRAY(KL,2)+ARRAY(LL,2)+0.5
C
C FIND THE LINE WITH THE ENOUGH CAPACITY TO HANDLE
C THE TOTAL TRAFFIC ON THE PROPOSED SUBNETWORK  LL
C
      CALL LINNUM(TRFIN,TRFOUT,LDUMMY,LINNEW,1,RHO)
      IF(LINCAP(LINNEW) .EQ. 9600) GOTO 132
      LINUP=LINNEW-1
      IF(LINUP .EQ. 0) GOTO 712
      DO 711 NL=1,LINUP
        IF(LDUMMY(NL) .EQ. 0) GOTO 711
        GOTO 132
711   CONTINUE
712   CONTINUE
      NLNEW=LDUMMY(LINNEW)
      IF(NLNEW .GT. 1) GOTO 132 @MORE THAN 1 LINE NOT ALLOWED
      COST=COSTEW(LL,1)
      LINOLD=COSTEW(LL,2)
      NLOLD=COSTEW(LL,3)
      MCOSTL=COST
3000  CONTINUE
C
C TEST RESPONSE TIME, IF NOT SATISFIED, INCREASE LINE CAPACITY
C
      CALL RSPTST(KL,LL,LINNEW,IOK)
      IF(IOK .EQ. 1) GOTO 3001
C
C IF LINE TYPE IS THE HIGHEST, NO NEED TO GO FURTHER
C
      IF(LINNEW .EQ. N3) GOTO 132
      LDUMMY(LINNEW)=0
      LINNEW=LINNEW+1
      IF(LINCAP(LINNEW) .EQ. 9600) GOTO 132
      LDUMMY(LINNEW)=1
      NLNEW=1
      CALL RHOFUN(TRFIN,TRFOUT,LDUMMY,LINNEW,RHOLIN,RHO)
      GOTO 3000
3001  CONTINUE
      IF(LINNEW.EQ.LINOLD.AND.NLOLD.EQ.1) GOTO 131
      CALL LCOSTK(IRSC,LL,1,MCOSTL)   @NEW COST FOR SUBNET UNDER LL
131   CONTINUE
      LINOLD=COSTEW(KL,2)
      MCOSTK=COSTEW(KL,1)
      NLOLD =COSTEW(KL,3)
      IF( LINNEW .EQ. LINOLD .AND. NLOLD .EQ. 1) GOTO 133
      CALL LCOSTK(IRSC,KL,0,MCOSTK)  @NEW COST FOR SUBNET UNDER KL
```

```
399             GOTO 134
400       133   CONTINUE
401             ITEMP=0
402             KADD=0
403             KCHG=2
404             CALL ISUMUP(IRSC,KL,LINNEW,ITEMP)
405             MCOSTK=MCOSTK-ITEMP
406       134   CONTINUE
407             INTRY=1  @FLAG THAT INDICATES AN ENTRY TO TRYLNK
408             JSAV=COSTEW(LL,1)+COSTEW(KL,1)-(MCOSTL+MCOSTK)
409       719   CONTINUE
410             COSTKM=0
411             KADD=0
412             CALL ISUMUP(ML,KIL,LINNEW,COSTKM)
413             ISAV=JSAV-COSTKM
414             IF (ISAV .LE. MAXSAV) GO TO 132
415             RSPMAX=RSPTIM
416             MAXSAV=ISAV
417             MAXK=KL
418             MAXL=LL
419             MAXM=ML
420             MAXKI=KIL
421             MAXLIN= LINNEW
422             MAXNOL=NLNEW
423             RHOMAX=RHO
424       132   CONTINUE
425             RETURN
426             SUBROUTINE LCOSTK(I,NA,NB,TCOST)
427       C           ********************
428       C
429       C     FIND COST FOR A SUBNETWORK. NA=BEGINNING NODE FOR THE SUBNET
430       C     TO BE EVALUATED.
431       C     NB=1 WHEN COST FOR CENTRAL LINK NA IS TO BE INCLUDED
432       C     NB=0 WHEN COST FOR CENTRAL LINK NB IS NOT TO BE INCLUDED
433       C
434       C           **********************
435             INTEGER TCOST
436             TCOST=0
437             KCHG=2
438             CALL ISUMUP (I,NA,LINNEW,TCOST)
439       C
440       C     START COMPUTING SUBNET COST
441       C
442             JSON=IARRAY(NA,2)          @ FIRST SUCCESSOR
443             IF(JSON.EQ.0) GOTO 400
444       300   CONTINUE
445             JPA=IARRAY(JSON,5)
446             CALL ISUMUP(JPA,JSON,LINNEW,TCOST)
447             JSON=NXTNOD(NA,JSON)
448             IF (JSON .EQ. 0) GO TO 400        @ CALL IT AN END
449             GO TO 300
450       400   CONTINUE
451             IF(NB .EQ. 1) RETURN
452             ITEMP=0
453             KADD=0
454             CALL ISUMUP(I,NA,LINNEW,ITEMP)
455             TCOST=TCOST-ITEMP
```

```
      RETURN
      FUNCTION NXTNOD(L1,M1)
C           ********************
C
C     FIND THE NEXT NODE IN THE SUBNET L1 WHICH M1 BELONGS TO.
C     IN THE PROCESS, IF THE NEXT NODE IS L1, 0 IS RETURNED
C     OTHERWISE THE NEXT NODE IS RETURNED.
C
C           ********************
      NXTNOD=0
      MM=M1
      KSON=IARRAY(MM,2)
      IF(KSON .EQ. 0 .AND. MM .EQ. L1) RETURN  @A SINGLE NODE
      IF (KSON .EQ. 0) GO TO 1          @ NO SUCCESSOR
      NXTNOD=KSON
      RETURN
1     CONTINUE
C
C     LOOK FOR HIS NEXT BROTHER
C
      KBRO=IARRAY(MM,3)
      IF (KBRO .EQ. 0) GO TO 2   @NO MORE SUCCESSORS WITH SAME PREDECESSOR
      NXTNOD=KBRO
      RETURN
2     CONTINUE
C
C     GO TO HIS FATHER
C
      MM=IARRAY(MM,5)
      IF (MM .NE. L1) GO TO 1             @ BACK TO THE BEGINNING
      RETURN
      SUBROUTINE UPNETW
C           ********************
C
C UPDATE IARRAY AND COSTEW BASED ON MAXIMUM-SAVING
C PARAMETERS OBTAINED
C
C UPDATE TRAFFIC AND NO. OF TERMINALS FOR L-SUBNET
C
C           ********************
      NOK=IARRAY(MAXK,1)+1  @NO. OF NODES BELOW MAXK
      IARRAY(MAXL,1)=IARRAY(MAXL,1)+NOK
      ARRAY(MAXL,1)=ARRAY(MAXL,1)+ARRAY(MAXK,1)
      ARRAY(MAXL,2)=ARRAY(MAXL,2)+ARRAY(MAXK,2)
C
C UPDATE THE COSTEW
C
      COSTEW(MAXL,1)=COSTEW(MAXL,1)+COSTEW(MAXK,1)-MAXSAV
      COSTEW(MAXL,2)=MAXLIN
      COSTEW(MAXL,3)=MAXNOL
      COSTEW(MAXK,1)=0
      COSTEW(MAXK,2)=0
      COSTEW(MAXK,3)=0
      MAXKD=NUMR(MAXK)
      MAXMD=NUMR(MAXM)
      MAXKID=NUMR(MAXKI)
      COSTEW(MAXL,4)=COSTEW(MAXL,4)+COSTEW(MAXK,4)+DIST(MAXKID,MAXMD)
```

```
513           *        -DIST(MAXKD,NN1)
514            RHOF(MAXL)=RHOMAX
515            COSTEW(MAXK,4)=0
516       C
517       C UPDATE MULTIDROPPED-LINE RESPONSE TIME
518       C
519            TIMRSP(MAXL)=RSPMAX
520       91   CONTINUE
521            KIPA=IARRAY(MAXKI,5)  @REMEMBER KI'S PREDECESSOR
522            MSON=IARRAY(MAXM,2)  @M'S 1ST SUCCESSOR
523            CALL LNKOFF(MAXKI)  @DELETE KI AS A SUCCESSOR OF KIPA
524            IARRAY(MAXM,2)=MAXKI
525            IARRAY(MAXKI,5)=MAXM
526            IARRAY(MAXKI,3)=MSON
527            IF(MSON .NE. 0) IARRAY(MSON,4)=MAXKI
528            IARRAY(MAXKI,4)=0
529            MAXM=MAXKI
530            MAXKI=KIPA
531            IF(MAXM .NE. MAXK) GOTO 91
532            RETURN
533            FUNCTION JCOSTA(N,KREF)
534       C           ******************
535       C
536       C FIND PARTIAL SUM FOR ICSTLN
537       C
538       C           ******************
539            JCOSTA=0
540            DO 777 K1=1,KREF
541              JCOSTA=JCOSTA+ICSTLN(K1,N)
542       777  CONTINUE
543            RETURN
544            FUNCTION JCOSTB(N,M,KREF)
545       C           ******************
546       C
547       C FIND PARTIAL SUM FOR ICSTHW
548       C
549       C           ******************
550            JCOSTB=0
551            DO 778 KK=1,KREF
552              JCOSTB=JCOSTB+ICSTHW(KK,N,M)
553       778  CONTINUE
554            RETURN
555            SUBROUTINE NETPRT
556       C           ********************
557       C
558       C PRINT OUT CONFIGURATION OF THE MULTIDROP NETWORK
559       C
560       C           ********************
561            DO 196 KK=1,NP1
562              IBLANK(KK)=JBLANK
563       196  CONTINUE
564            NN2=NUMRR(IRSC)
565            WRITE(IWT,197) NN2
566       197  FORMAT(1H1,' REGIONAL CENTER= ',A4//,6X,'SUBNETWORK',/,6X,
567           * 'BEGINS AT',//)
568            KP=1
569            ISON=IARRAY(IRSC,2)
```

```
570           IPOINT=ISON
571           ISONR=NUMRR(ISON)
572           WRITE(IWT,192) (IBLANK(I),I=1,KP),ISONR
573     C
574     C LOOK FOR ITS FIRST SUCCESSOR
575     C
576     190   CONTINUE
577           ISON=IARRAY(IPOINT,2)   @CURRENT NODAL INDEX
578           IF(ISON .EQ. 0) GOTO 191 @NO MORE SON
579           KP=KP + 1 @A LEVEL DEEPER
580           ISONR=NUMRR(ISON)
581           WRITE(IWT,192)( IBLANK(I),I=1,KP),ISONR
582     192   FORMAT(1X,24(A6))
583           IPOINT=ISON
584           GOTO 190
585     191   CONTINUE
586     C
587     C LOOK FOR NEXT SUCCESSOR WITH THE SAME PREDECESSOR
588     C
589           IBRO=IARRAY(IPOINT,3)
590           IF(IBRO .EQ. 0) GOTO 193
591           IBRON=NUMRR(IBRO)
592           WRITE(IWT,192)(IBLANK(I),I=1,KP),IBRON
593           IPOINT=IBRO
594           GOTO 190
595     193   CONTINUE
596     C
597     C NEXT LEVEL UP
598     C
599           KP=KP-1
600           IPOINT=IARRAY(IPOINT,5)
601           IF(KP .EQ. 0) GOTO 194  @NO NEED TO GO FURTHER
602           GO TO 191
603     194   CONTINUE
604           RETURN
605           SUBROUTINE CONVRT(ICOST)
606     C           ********************
607     C
608     C CONVERT A NUMBER INTO ITS FIELD EQUIVALENT
609     C
610     C           ********************
611           JCHAR(1)=JBLANK
612           JCHAR(2)=JBLANK
613           IF(ICOST .EQ. 0) GOTO 916
614           ENCODE(198,JCHAR) ICOST
615     198   FORMAT(I8)
616     916   CONTINUE
617           RETURN
618           SUBROUTINE SUMPRT(NREF,NN)
619     C           ********************
620     C
621     C SUM UP COSTS AND PRINTS
622     C
623     C           ********************
624           TCOST1=0
625           TCOST2=0
626           DO 770 K=1,NREF
```

```
627               ITCOST(K,1)=ICSTLN(K,1)
628               ITCOST(K,2)=ICSTLN(K,2)
629               DO 7791 KK=1,N4
630                 ITCOST(K,1)=ITCOST(K,1)+ICSTHW(K,KK,1)
631                 ITCOST(K,2)=ITCOST(K,2)+ICSTHW(K,KK,2)
632       7791    CONTINUE
633               TCOST1=TCOST1+ITCOST(K,1)
634               TCOST2=TCOST2+ITCOST(K,2)
635       779   CONTINUE
636             KCOST=TCOST1+TCOST2
637       C
638       C PRINT OUT COST
639       C
640             NTURN=NREF/10+1
641             IREM=MOD(NREF,10)
642             IF(IREM .EQ. 0) NTURN=NTURN-1
643             LPAGE=1
644             DO 919 KW=1,NTURN
645               KWL=10*(KW-1)+I
646               KWU=10*KW
647               IF(KWU.GT.NREF) KWU=NREF
648               IF(NN .EQ. 0) GOTO 879
649               IF(LPAGE.NE.1) GOTO 9033
650               WRITE(IWT,9031) KW
651       9031    FORMAT('1',40X,'REGIONAL STAR NETWORK AND ITS COSTS-',I2)
652                 GOTO 9035
653       9033    CONTINUE
654                 WRITE(IWT,9034)KW
655       9034    FORMAT(/,40X,'REGIONAL STAR NETWORK AND ITS COSTS-',I2)
656       9035    CONTINUE
657                 WRITE(IWT,9032) (NUMRR(I),I=KWL,KWU)
658       9032    FORMAT(/,1X,'SYSTEM TERMN.',13X,10(4X,A4,1X))
659                 WRITE(IWT,903)
660       903     FORMAT(/,1X,'NO. OF LINES REQ.')
661                 DO 1903 NJ=1,N3
662                   IF(LINMIX(NJ) .EQ. 0) GOTO 1903
663                   WRITE(IWT,904) LINAME(NJ),(NLINES(K,NJ),K=KWL,KWU)
664       904     FORMAT(7X,A6,14X,10(I8,1X))
665       1903    CONTINUE
666                 WRITE(IWT,9036) (RHOF(NJ),NJ=KWL,KWU)
667       9036    FORMAT(/,1X,,'LINE UTILIZATION',11X,10(F8.3,1X))
668                 WRITE(IWT,906) (IDST(N),N=KWL,KWU)
669       906     FORMAT(/,1X,'DSTNCE FROM RSC',11X,10(I8,1X))
670                 GO TO 806
671       879     CONTINUE
672                 IF(LPAGE.NE.1) GOTO 8033
673                 WRITE(IWT,8031) KW
674       8031    FORMAT('1',40X,'FINAL MULTIDROP NETWORK AND ITS COSTS-',I2)
675                 GOTO 8035
676       8033      CONTINUE
677                 WRITE(IWT,8034) KW
678       8034    FORMAT(/,40X,'FINAL MULTIDROP NETWORK AND ITS COSTS-',I2)
679       8035    CONTINUE
680                 WRITE(IWT,803)(I,I=KWL,KWU)
681       803       FORMAT(/,1X,'SUBNET NO.',16X,10(I8,1X))
682                 DO 1803 N=KWL,KWU
683                   ID=NSUR(N)
```

```
684                 MSUB(N)=NUMRR(ID)
685                 LSUB(N)=IARRAY(ID,1)+1
686           1803  CONTINUE
687                 WRITE(IWT,1806) (MSUB(N),N=KWL,KWU)
688           1806  FORMAT(/,1X,'BEGINNNING NODE',11X,10(4X,A4,1X))
689                 WRITE(IWT,1807) (LSUB(N),N=KWL,KWU)
690           1807  FORMAT(3X,'NO. OF TERM.',12X,10(I8,1X))
691                 WRITE(IWT,811)
692           811   FORMAT(3X,'NO. OF LINES')
693                   DO 1808 NJ=1,N3
694                     IF(LINMIX(NJ).EQ.0) GOTO 1808
695                     WRITE(IWT,904) LINAME(NJ),(NLINES(K,NJ),K=KWL,KWU)
696           1808  CONTINUE
697                 WRITE(IWT,8036)(RHOF(NJ),NJ=KWL,KWU)
698           8036  FORMAT(3X,'LINE UTILIZATION',9X,10(F8.3,1X))
699                 WRITE(IWT,808) (IDST(N),N=KWL,KWU)
700           808   FORMAT(3X,'TOTAL MILAGE',12X,10(I8,1X))
701           806   CONTINUE
702                   DO 1101 N=KWL,KWU
703                     ID=N
704                     IF(NN.EQ.0) ID=NSUB(J)
705                     TRFSUM(N,1)=ARRAY(ID,1)
706                     TRFSUM(N,2)=ARRAY(ID,2)
707                     TIMOUT(N)=TIMRSP(ID)
708           1101    CONTINUE
709                 WRITE(IWT,1102) (TRFSUM(N,1),N=KWL,KWU)
710           1102  FORMAT(3X,'TRAFFIC',/,3X,'   LINE TO CPU',11X,10(F8.3,1X))
711                 WRITE(IWT,1103)(TRFSUM(N,2),N=KWL,KWU)
712           1103  FORMAT(3X,'   CPU TO LINE ',10X,10(F8.3,1X))
713                 WRITE(IWT,1104) (TIMOUT(N),N=KWL,KWU)
714           1104  FORMAT(3X,'LINE RESPONSE TIME',7X,10(F8.3,1X))
715                 WRITE(IWT,907)
716           907   FORMAT(21X,'SUBTOTAL',/,1X,'INST. COSTS')
717                   COST=JCOSTA(1,NREF)
718                   IF(KW.NE.1) COST=0
719                   CALL CONVRT(COST)
720                 WRITE(IWT,908) (JCHAR(L),L=1,2),(ICSTLN(NODE,1),NODE=KWL,KWU)
721           908   FORMAT(5X,'LINES',8X,A6,A2,1X,10(I8,1X))
722                   DO 1909 K=1,N4
723                     COST=JCOSTB(K,1,NREF)
724                     IF(KW.NE.1) COST=0
725                     CALL CONVRT(COST)
726                 WRITE(IWT,909) NAMEHW(K),(JCHAR(L),L=1,2),(ICSTHW(NODE,K,1),
727                * NODE=KWL,KWU)
728           909   FORMAT(5X,A6,7X,A6,A2,1X,10(I8,1X))
729           1909  CONTINUE
730                 WRITE(IWT,910)
731           910   FORMAT(1X,'ANNUAL RECURR. COST')
732                   COST=JCOSTA(2,NREF)
733                   IF(KW.NE.1) COST=0
734                   CALL CONVRT(COST)
735                 WRITE(IWT,908) (JCHAR(L),L=1,2),(ICSTLN(NODE,2),NODE=KWL,KWU)
736                   DO 1911 K=1,N4
737                   COST=JCOSTB(K,2,NREF)
738                   IF(KW.NE.1) COST=0
739                   CALL CONVRT(COST)
740                 WRITE(IWT,909) NAMEHW(K),(JCHAR(L),L=1,2),(ICSTHW(NODE,K,2),
```

```
741           * NODE=KWL,KWU)
742       1911  CONTINUE
743             WRITE(IWT,912)
744       912   FORMAT(1X,'TOTAL COST')
745               IF(KW.NE.1) TCOST1=0
746               CALL CONVRT(TCOST1)
747             WRITE(IWT,913) (JCHAR(L),L=1,2),(ITCOST(K,1),K=KWL,KWU)
748               IF(KW.NE.1) TCOST2=0
749               CALL CONVRT(TCOST2)
750             WRITE(IWT,914)(JCHAR(L),L=1,2),(ITCOST(K,2),K=KWL,KWU)
751       913   FORMAT(4X,'INST. COST',4X,A6,A2,1X,10(I8,1X))
752       914   FORMAT(4X,'RECUR. COST',3X,A6,A2,1X,10(I8,1X))
753               LPAGE=LPAGE+1
754               LPAGE=MOD(LPAGE,2)
755       919   CONTINUE
756             WRITE(IWT,695) KCOST
757       695   FORMAT(/,25X,'TOTAL COST=',I8)
758             RETURN
759             SUBROUTINE MUTDRP
760       C           ********************
761       C
762       C PRINT OUT FINAL MULTIDROP NETWORK WITH ITS COSTS
763       C
764       C           ********************
765             DO 590 NL=1,N3
766               LDUMMY(NL)=0
767       590   CONTINUE
768             IBRO=IARRAY(IRSC,2)  @FIRST SUCCESSOR
769             K1=1
770       699   CONTINUE
771             IF(IBRO .EQ. 0) GOTO 698
772             NK2=NUMR(IBRO)
773             NK1=NN1
774             NSUB(K1)=IBRO
775             LINE=COSTEW(IBRO,2)
776             LDUMMY(LINE)=COSTEW(IBRO,3)
777             JSON=IARRAY(IBRO,2)
778             IF(JSON.EQ.0) GOTO 694
779             DO 592 NM=1,2
780               ICSTLN(K1,NM)=0
781               DO 592 NK=1,N4
782                 ICSTHW(K1,NK,NM)=0
783       592     CONTINUE
784             DO 596 NL=1,N3
785               NLINES(K1,NL)=LDUMMY(NL)
786       596   CONTINUE
787             KCHG=2
788       312   CONTINUE
789             CALL ICOSTJ(LDUMMY,NK1,NK2,LNKCHW,LNKCLN)
790             DO 595 NL=1,N3
791             DO 595 NM=1,2
792               ICSTLN(K1,NM)=ICSTLN(K1,NM)+LNKCLN(NL,NM)
793             DO 595 NK=1,N4
794               ICSTHW(K1,NK,NM)=ICSTHW(K1,NK,NM)+LNKCHW(NL,NK,NM)
795       595   CONTINUE
796             IF(JSON.EQ.0) GOTO 311
797             NK2=NUMR(JSON) @GLOBAL INDEX FOR NEXT NODE
```

```
798           NK1=IARRAY(JSON,5) @PREDECESSOR
799           NK1=NUMR(NK1) @GLOBAL INDEX FOR PREDECESSOR
800           JSON=NXTNOD(IBRO,JSON)
801           GOTO 312
802     311   CONTINUE
803           LDUMMY(LINE)=0
804           GOTO 591
805     694   CONTINUE
806     C
807     C USE PREVIOUS DATA
808     C
809           DO 597 NL=1,N3
810             NLINES(K1,NL)=NLINES(IBRO,NL)
811     597   CONTINUE
812           DO 598 NM=1,2
813             ICSTLN(K1,NM)=ICSTLN(IBRO,NM)
814           DO 598 NK=1,N4
815             ICSTHW(K1,NK,NM)=ICSTHW(IBRO,NK,NM)
816     598   CONTINUE
817           LDUMMY(LINE)=0
818     591   CONTINUE
819           IDST(K1)=COSTEW(IBRO,4)
820           RHOF(K1)=RHOF(IBRO) @SHUFFLING RHO'S DUE TO RE-INDEXING
821           IBRO=IARRAY(IBRO,3)
822           K1=K1+1
823           GOTO 699
824     698   CONTINUE
825           NOSUB=K1-1
826           CALL SUMPRT(NOSUB,0)
827           RETURN
828           SUBROUTINE CALPLT
829     C           ********************
830     C
831     C PLOT A MULTIDROP NETWOEK
832     C
833     C           ********************
834           KP=1
835           IPOINT=IRSC
836           CALL TRSFRM(2)
837           ISON=IARRAY(IRSC,2)   @FIRST SUCCESSOR
838           IPOINT=ISON
839           CALL TRSFRM(1)
840     C
841     C LOOK FOR ITS FIRST SUCCESSOR
842     C
843     190   CONTINUE
844           ISON=IARRAY(IPOINT,2) @FIRST SUCCESSOR
845            IF(ISON .EQ. 0) GOTO 191
846           KP=KP+1
847           IPOINT=ISON
848           CALL TRSFRM(1)
849           GOTO 190
850     191   CONTINUE
851     C
852     C LOOK FOR ITS NEXT SUCCESSOR
853     C
854           IBRO=IARRAY(IPOINT,3)
```

```
      IPOINT=IARRAY(IPOINT,5)  @NOW ITS PREDECESSOR
      CALL TRSFRM(2)
      IF(IBRO .EQ.0) GOTO 193
      IPOINT=IBRO
      CALL TRSFRM(1)
      GOTO 190
193   CONTINUE
C
C GO BACK TO ITS PREDECESSOR
C
      KP=KP-1
      IF(KP .EQ. 0) GOTO 194
      GOTO 191
194   CONTINUE
      CALL TRSFRM(3)
      RETURN
      SUBROUTINE TRSFRM(LK)
C           ********************
C
C FIND GLOBAL MADADR INDEX FOR V-H COORDINATES AND PID NO'
C
C           ********************
      DATA IP/0/
      IF(LK .EQ. 3) GOTO 666
      LK1=NUMR(IPOINT)   @GLOBAL INDEX
      IDD=MAPADR(LK1)   @MAPADR INDEX FOR LK1
      IF(IDD .EQ. IP) RETURN
      IP=IDD
666   CONTINUE
      CALL PLOTPT(IDD,LK)
      RETURN
      SUBROUTINE LNKOFF(MP)
C           **********H*********
C
C DELETE MP AS A SUCCESSOR OF NODE PA
C
C           ********************
      IFRONT=IARRAY(MP,4)  @THE SUCCESSOR BEFORE MP
      IBACK =IARRAY(MP,3)  @THE SUCCESSOR AFTER MP
      IF(IFRONT .NE. 0) GOTO 92
      MPA=IARRAY(MP,5)
      IARRAY(MPA,2)= IBACK  @1ST SUCCESSOR UNDER NEW MPA
      GOTO 99
92    CONTINUE
      IARRAY(IFRONT,3)=IBACK
99    CONTINUE
      IF(IBACK .EQ. 0) RETURN
      IARRAY(IBACK,4)=IFRONT
      RETURN
      SUBROUTINE RSPTST(KKK,LLL,LINMAX,IOK)
C           ********************
C
C TEST RSPONSE TIME, SATISFIED WHEN IOK=1
C
C           ********************
      MDROP=IARRAY(LLL,1)+IARRAY(KKK,1)+2
      TRFIN=ARRAY(LLL,1)+ARRAY(KKK,1)
      TRFOUT=ARRAY(LLL,2)+ARRAY(KKK,2)
      CALL RSPNSE(TRFIN,TRFOUT,LINMAX,MDROP,IOK)
      RETURN
      END
@PRT STACOM.IRNOP/0777
```

```
51928*STACOM(1),IRNOP/0777
      1             SUBROUTINE IRNOP(NR,LIMIT,TRM)
      2       C           **********************
      3       C
      4       C     SUBPROGRAM FOR THE INTER-REGION NETWORK OPTIMIZATION
      5       C     LIMIT=MINIMAL NUMBER OF PATHS NEEDED PER REGIONAL
      6       C           SWITCHING CENTER
      7       C
      8       C           **********************
      9             PARAMETER NP1=130,NP2=1,NP3=4,NP4=3
     10             PARAMETER NPC=360, NP6=(NPC*NPC/2-NPC+1)/4+1
     11             PARAMETER NP7=4, IWT=100, MW=4
     12             COMMON/CONST/N1,N2,N3,N4,N7,NCITY
     13             COMMON/LINCHR/LINMIX(NP3),LINCAP(NP3), UTILIZ(NP3)
     14            *  /BCOST/AINSTC(NP2,NP3,NP4,3,2,2),RECRC(NP2,NP3,NP4,3,2,2),
     15            *    ANSTLN(NP2,NP3,3,2,2),RECRLN(NP2,NP3,3,2,16),IDUPLX(NP3)
     16            *  /NAME/INDXPT(NP1), NAMEST(NP1),LINAME(NP3),NAMEHW(NP4)
     17            *  /EIN/SVR(NP1),NRSC(MW),NUMPR(MW),TRAFDN(NP1),TRAFIT(NP1)
     18            *  /REF/IREF(NPC),TRAFD(NP1,2,NP7),DSTNCE(NP6),MAPADR(NP1)
     19             DIMENSION NETSUM(NP3,2),ORINET(MW,MW,NP3)
     20             DIMENSION NLINK(NP3),LNKCHW(NP3,NP4,2),LNKCLN(NP3,2)
     21             INTEGER SUMCST
     22             INTEGER ORINET
     23             DIMENSION TRRM(MW,MW),TR(MW,MW)
     24             INTEGER ORICST,ORICS1,ORICS2
     25             INTEGER DIVTRI(MW),DIVTRJ(MW)
     26             DIMENSION TRR(MW,MW),NETCNF(MW,MW,NP3),LINEQU(NP3),
     27            *   LINADI(NP3),LINADJ(NP3),ILINAD(NP3),JLINAD(NP3),
     28            *   LINEQ(NP3),LINEQA(NP3),LINEQB(NP3)
     29             DIMENSION RHOF2(MW,MW)
     30             DIMENSION TRM(MW,MW)
     31             EQUIVALENCE (LINEQ,LINEQA),(LINEQU,LINEQB)
     32       C
     33       C RESET UTILIZATION FACTOR TO .5
     34       C
     35             DO 70 NN1=1,N3
     36             UTILIZ(NN1)=.5
     37       70    CONTINUE
     38       C
     39       C COMPUTE ORINET(MW,MW,N3) FOR INITIAL TOPOLOGY WHERE N3 IS
     40       C THE NUMBER OF CHARGEABLE ITEMS
     41       C
     42             ORICST=0
     43             ORICS1=0
     44             ORICS2=0
     45             NR1=NR-1
     46             DO 203 NN1=1,N3
     47             DO 203 NN2=1,2
     48             NETSUM(NN1,NN2)=0 @COST SUM
     49       203   CONTINUE
     50       C
     51       C MODIFY DUPLEXING MODE FROM HALF TO FULL DUPLEX
     52       C
     53             DO 667 K1=1,N3
     54             IDUPLX(K1)=2
     55       667   CONTINUE
     56             DO 101 I=1,NR1
```

```
57            NLINK(I)=NR1  @NR1 LINKS AT THE BEGINNING
58            I1=I+1
59            DO 102 J=I1,NR
60            II=NRSC(I)
61            JJ=NRSC(J)
62            ATRMAX=AMAX1(TR(I,J), TR(J,I))  @ASSUMMING FULL DUPLEX
63            CALL LINNUM (ATRMAX,0.,LINEQ,LINUP,0,RHO)
64            RHOF2(I,J)=RHO
65            RHOF2(J,I)=RHO
66            CALL ICOSTJ(LINEQ,II,JJ, LNKCHW,LNKCLN)
67            DO 104 NN=1,N3
68            ORINET(I,J,NN)= LINEQ(NN)
69            ORINET(J,I,NN)= LINEQ(NN)
70            DO 105  NM=1,2     @ LINE COST
71            NETSUM(NN,NM)= NETSUM(NN,NM)+ LNKCLN(NN,NM)
72            DO 106  NK=1,N4    @ HARDWARE COSTS
73            NETSUM(NN,NM)= NETSUM(NN,NM)+ LNKCHW(NN,NK,NM)
74      106   CONTINUE
75      105   CONTINUE
76      104   CONTINUE
77      102   CONTINUE
78      101   CONTINUE
79            DO 107 K1=1,NR
80            DO 107 NN=1,N3
81            ORINET(K1,K1,NN)=0
82      107   CONTINUE
83            CALL OUTPRT(1)
84            ITALLY=0
85      999   CONTINUE
86            MAXSAV=0
87            DO 777  I=1,NR1
88            IF (NLINK(I) .LE. LIMIT) GO TO 777
89            I1=I+1
90            DO 788  J=I1,NR
91            IF (NLINK(J) .LE. LIMIT) GO TO 788
92            IN=NTEST(ORINET,I,J)
93            IF (IN .EQ. 0) GO TO 788  @NO LINK TO BE DELETED
94      C
95      C DETERMINE WHETHER THERE IS A LINK CONNECTED BY AT MOST ONE INDIRECT
96      C ROUTE BETWEEN ANY TWO REGIONS IN THE NETWORK WHEN THE DIRECT LINK
97      C BETWEEN I AND J IS ELIMINATED. THE INDIRECT LINK ONLY GOES THROUGH
98      C ONE INTERMEDIATE RSC.
99      C
100           DO 139 L=1,NR1
101           L1=L+1
102           DO 138 M=L1,NR
103           IY=ITEST(I,J,L,M)
104           IF (IY .EQ. 1) GO TO 810   @NEXT STEP NOT TO BE TESTED
105           IN=NTEST(ORINET,L,M)
106           IF (IN .EQ. 1) GO TO 138
107     810   CONTINUE
108           DO 137  N=1,NR
109           IF (L .EQ. N) GO TO 137
110           IY=ITEST(I,J,L,N)
111           IF(IY .EQ. 1) GOTO 137
112           IN=NTEST(ORINET,L,N)
113           IF (IN .EQ. 0) GO TO 137
```

```
      IY=ITEST(I,J,N,M)
      IF(IY .EQ. 1) GOTO 137
      IN=NTEST(ORINET,M,N)
      IF (IN .EQ. 1) GO TO 138
137   CONTINUE
      GO TO 788
138   CONTINUE
139   CONTINUE
      SDIVTI=0
      SDIVTJ=0
      CALL TRFDIV(IFLOP)
      IF (IFLOP .EQ. 1) GO TO 201
      CALL MINAD(IIAD,MINCST)
      GO TO 202
201   CONTINUE
      CALL NETWKC(MINCST)
202   CONTINUE
      ISAV=ORICST-MINCST
      IF (MAXSAV .GE. ISAV) GO TO 788
      MAXSAV=ISAV
      IFLIP=IFLOP  @IFLIP=GLOBAL INDICATOR
      IF (IFLOP .EQ. 1) GO TO 204
      IIMAX=IIAD
      IMAX=I
      JMAX=J
      DO 666 NN=1,N3
      ILINAD(NN)= LINADI(NN)  @CHANGE OF LINE REQ.
      JLINAD(NN)= LINADJ(NN)
666   CONTINUE
204   CONTINUE
      DO 331  K1=1,NR
      DO 331  K2=1,NR
      TRRM(K1,K2)=TRR(K1,K2)
331   CONTINUE
788   CONTINUE
777   CONTINUE
      IF (MAXSAV .LE. 0) GO TO 9999
      CALL NETUP(IFLIP,IIMAX,IMAX,JMAX)
      ITALLY=ITALLY+1
      GO TO 999
9999  CONTINUE
109   FORMAT(1X,' THIS NETWORK HAS BEEN UPDATED FOR ',I5,' TIMES',//)
      WRITE(6,109) ITALLY
      DO 81 I=1,N3
      DO 81 J=1,2
      NETSUM(I,J)=0
81    CONTINUE
      DO 91 I=1,NR1
      K=I+1
      DO 92 J=K,NR
      DO 93 K1=1,N3
      LINEQ(K1)=ORINET(I,J,K1)
93    CONTINUE
      II=NRSC(I)
      JJ=NRSC(J)
      CALL ICOSTJ(LINEQ,II,JJ,LNKCHW,LNKCLN)
      DO 94 KK=1,N3
```

```
171             NETSUM(KK,1)=NETSUM(KK,1)+LNKCLN(KK,1)
172             NETSUM(KK,2)=NETSUM(KK,2)+LNKCLN(KK,2)
173             DO 95 KL=1,N4
174             NETSUM(KK,1)=NETSUM(KK,1) + LNKCHW(KK,KL,1)
175             NETSUM(KK,2)=NETSUM(KK,2) + LNKCHW(KK,KL,2)
176       95    CONTINUE
177       94    CONTINUE
178       92    CONTINUE
179       91    CONTINUE
180             ORICS1=0
181             ORICS2=0
182             CALL OUTPRT(2)
183             RETURN
184             SUBROUTINE OUTPRT(N)
185       C           ********************
186       C
187       C PRINT OUT INTERREGIONAL NETWORK CONFIGURATION AND ITS COSTS
188       C
189       C           ********************
190             DO 110 I=1,N3
191             ORICS1=ORICS1+NETSUM(I,1)
192             ORICS2=ORICS2+NETSUM(I,2)
193       110   CONTINUE
194             NTURN=NR/10+1
195             DO 2001 L=1,NTURN
196             LL=(NTURN-1)*10 + 1
197             LU=NTURN*10
198             IF(LU .GT. NR) LU=NR
199             IF(N .EQ. 2) GOTO 2100
200             WRITE(IWT,2002) (J,J=LL,LU)
201       2002  FORMAT('1',//,10X,'INITIAL INTERREGIONAL NETWORK CONFIURATION',
202            * ///,20X,10(5X,I3,2X))
203             GOTO 2101
204       2100   CONTINUE
205             WRITE(IWT,2102) (J,J=LL,LU)
206       2102  FORMAT('1',10X,'FINAL OPTIMAL INTERREGIONAL NETWORK CONFIUGRATION',
207            * ///,20X,10(5X,I3,2X))
208       2101  CONTINUE
209             DO 2003 I=1,NR
210             WRITE(IWT,2004) I
211       2004  FORMAT('  REGION',/,I4)
212             DO 2108 M=1,N3
213             WRITE(IWT,2008)LINAME(M),(ORINET(I,K,M),K=LL,LU)
214       2108  CONTINUE
215       2008  FORMAT((4X,A6,10X,10(5X,I3,2X)))
216             WRITE(IWT,2201) (RHOF2(I,J),J=LL,LU)
217       2201  FORMAT(4X,'LINE UTILIZATION',10(4X,F4.3,2X))
218       2003  CONTINUE
219       2001  CONTINUE
220             WRITE(IWT,2006)
221       2006  FORMAT(//,17X,'INST. COST',7X,'RECUR. COST',9X,'SUBTOTAL')
222             DO 2005 K=1,N3
223             ISUM=NETSUM(K,1)+NETSUM(K,2)
224             WRITE(IWT,2007) K,LINAME(K),(NETSUM(K,I),I=1,2), ISUM
225       2007  FORMAT(//,I5,3X,A6,5X,I6,11X,I6,10X,I8)
226       2005  CONTINUE
227             ORICST=ORICS1+ORICS2
```

```
228           WRITE(IWT,2009) ORICS1,ORICS2,OPICST
229     2009  FORMAT(//,9X,'TOTAL',4X,I7,10X,I7,10X,I8)
230           RETURN
231           FUNCTION ITEST(I,J,K,L)
232     C           ********************
233     C
234     C TEST EQUIVALENCE BETWEEN SUBSET(I,J) AND SUBSET(K,L)
235     C
236     C           ********************
237           ITEST=0
238           IF (I .EQ. K .AND. J .EQ. L) ITEST=1
239           IF (I .EQ. L .AND. J .EQ. K) ITEST=1
240           RETURN
241           FUNCTION NTEST(NET,I,J)
242     C           ********************
243     C
244     C TEST DIRECT LINE CONNECTIVITY BETWEEN I AND J.
245     C
246     C           ********************
247           DIMENSION NET(MW,MW,NP3)
248           DO 103  I1=1,N3
249           IF (NET(I,J,I1) .GT. 0) GO TO 108
250     103   CONTINUE
251           NTEST=0       @ NO CONNECTION
252           RETURN
253     108   NTEST=1
254           RETURN        @ YES, THERE IS A CONNECTION
255           SUBROUTINE TRFDIV(IFLOP)
256     C           ********************
257     C
258     C DIVERT TRAFFIC BETWEEN I AND J THROUGH OTHER RSCS.
259     C
260     C           ********************
261     C
262     C IT RETURNS WITH IFLOP=1 WHEN SUCCESSFUL ; OTHERWISE IFLOP=0.
263     C IT ALSO CREATES TEMPORARY MATRICES TRR AND NETCNF.
264     C
265           SDIVTI=0. @TOTAL TRAFFIC DIVERTED (I TO J)
266           SDIVTJ=0. @TOTAL TRAFFIC DIVERTED (J TO I)
267           DO 205  K=1,NR
268           DIVTRI(K)=0.  @ TRAFFIC DIVERTED THRU REGION K (I TO J)
269           DIVTRJ(K)=0.  @ TRAFFIC DIVERTED THRU REGION K (J TO I)
270     205   CONTINUE
271           DO 220 II=1,NR
272           IF (II.EQ.I.OR.II.EQ.J) GO TO 220
273           IC1=NTEST(ORINET,I,II)
274           IC2=NTEST(ORINET,II,J)
275           IF(IC1.EQ.0 .OR. IC2.EQ.0) GOTO 220
276     C
277     C     DIVERT I TO J TRAFFIC THRU II
278     C
279           DIVTRI(II)=0.
280           DIVTRJ(II)=0.
281           CALL LINTRF(I,II,A)
282           DELTR= A-TR(I,II)
283           IF (DELTR.LE.0.0) GO TO 160
284     C
```

```
285             DIVTR=DELTR
286             CALL LINTRF(II,J,B)
287             DELTR= B-TR(II,J)
288             IF (DELTR.LE.0.0) GO TO 160
289             DIVTRI(II)= AMIN1(DELTR,DIVTR)
290             IF ((DIVTRI(II)+ SDIVTI).GT.TR(I,J))GO TO 140
291             SDIVTI= SDIVTI+ DIVTRI(II)
292             GO TO 160
293       C
294       140   DIVTRI(II)= TR(I,J)- SDIVTI
295             SDIVTI= TR(I,J)
296       C
297       C     DIVERT J TO I TRAFFIC THRU II
298       C
299       160   CONTINUE
300             CALL LINTRF(J,II,A)
301             DELTR= A-TR(J,II)
302             IF (DELTR.LE.0.0) GO TO 220
303       C
304             DIVTR=DELTR
305             CALL LINTRF(II,I,B)
306             DELTR= B-TR(II,I)
307             IF (DELTR.LE.0.0) GO TO 220
308             DIVTRJ(II)= AMIN1(DELTR,DIVTR)
309             IF ((DIVTRJ(II)+ SDIVTJ) .GT. TR(J,I)) GO TO 180
310             SDIVTJ= SDIVTJ+ DIVTRJ(II)
311             GO TO 200
312       C
313       180   DIVTRJ(II)= TR(J,I)- SDIVTJ
314             SDIVTJ= TR(J,I)
315       C
316       200   CONTINUE
317             IF ((SDIVTI .EQ. TR(I,J)).AND. (SDIVTJ .EQ.TR(J,I)))GO TO 340
318       220   CONTINUE
319             IFLOP=0
320             GO TO 360
321       340   IFLOP=1
322       360   CONTINUE
323       C
324       C CREATE A NEW TRAFFIC MATRIX WHICH ELIMINATES THE TRAFFIC BETWEEN
325       C NODES I AND J AND A TEMPORARY NETWORK NETCNF FOR THE PURPOSE
326       C OF COST EVALUATION
327       C
328             DO 191 K1=1,NR
329             DO 191 K2=1,NR
330             TRR(K1,K2)=TR(K1,K2)
331             DO 191 K3=1,N3
332             NETCNF(K1,K2,K3)=ORINFT(K1,K2,K3)
333       191   CONTINUE
334             DO 190 KI=1,N3
335             NETCNF (I,J,KI)=0
336             NETCNF (J,I,KI)=0
337             NETCNF (I,I,KI)=0
338             NETCNF (J,J,KI)=0
339       190   CONTINUE
340             TRR(I,J)=0
341             TRR(J,I)=0
```

```
342             DO 380 IK=1,NR
343             IF (I .EQ. IK .OR. J .EQ. IK) GO TO 380
344             TRR(I,IK)= TR(I,IK)+ DIVTRI(IK)
345             TRR(IK,J)= TR(IK,J)+ DIVTRI(IK)
346             TRR(J,IK)= TR(J,IK)+ DIVTRJ(IK)
347             TRR(IK,I)= TR(IK,I)+ DIVTRJ(IK)
348             ATRMAX=AMAX1(TRR(I,IK),TRR(IK,I))
349             BTRMAX=AMAX1(TRR(J,IK),TRR(IK,J))
350             IDR1=NRSC(I)
351             IDR2=NRSC(IK)
352             CALL LINNUM(ATRMAX,0.,LINEQ,LINUP,0,RHO)
353             RHOF2(I,IK)=RHO
354             RHOF2(IK,I)=RHO
355             IDR1=NRSC(J)
356             CALL LINNUM(BTRMAX,0.,LINEQU,LINUP,0,RHO)
357             RHOF2(J,IK)=RHO
358             RHOF2(IK,J)=RHO
359             DO 430 NN=1,N3
360             NETCNF(I,IK,NN)=LINEQ(NN)
361             NETCNF(IK,I,NN)=LINEQ(NN)
362             NETCNF(IK,J,NN)=LINEQU(NN)
363             NETCNF(J,IK,NN)=LINEQU(NN)
364     430     CONTINUE
365     380     CONTINUE
366             RETURN
367             SUBROUTINE LINTRF(I,J,A)
368     C               ********************
369     C
370     C CONVERT LINES INTO TRAFFIC CAPACITIES BETWEEN NODES I AND J.
371     C
372     C               ********************
373             A=0
374             DO 100 IR=1,N3
375             A=A+ ORINET(I,J,IR)*LINCAP(IR)*UTILIZ(IR)
376     100     CONTINUE
377             RETURN
378             SUBROUTINE NETWKC(SUMCST)
379     C               ********************
380     C FIND TOTAL INTERREGIONAL NETWORK COST,SUMCST,BASED ON SPECIFIC
381     C CONFIGURATION NETCNF
382     C               ********************
383             INTEGER SUMCST
384             SUMCST=0
385             DO 420 IR=1,NR1
386             IR1=IR+1
387             DO 400 IK=IR1,NR
388             IC=NTEST(NETCNF,IR,IK)
389             IF (IC.EQ.0)  GO TO 400
390             II=NRSC(IR)
391             JJ=NRSC(IK)
392             DO 150  III=1,N3
393             LINEQU(III)= NETCNF(IR,IK,III)
394     150     CONTINUE
395             CALL TCOST(LINEQU,II,JJ,LNKCHW,LNKCLN)
396             DO 501 J1=1,N3
397             DO 510 J2=1,2
398             DO 520 J3=1,N4
```

```
399             SUMCST=SUMCST+LNKCHW(J1,J3,J2)
400       520   CONTINUE
401             SUMCST=SUMCST+LNKCLN(J1,J2)
402       510   CONTINUE
403       501   CONTINUE
404       400   CONTINUE
405       420   CONTINUE
406             RETURN
407             SUBROUTINE MINAD(IIAD,MINCST)
408       C           ********************
409       C
410       C CAPACITY INCREASE IS REQUIRED WHEN IFLOP=0. AND THE CAPACITY AT
411       C MAXIMUM COST SAVINGS.
412       C
413       C           ********************
414             DIMENSION LINDI(NP3),LINDJ(NP3)
415             MINCST=0
416             RTRFI=TR(I,J)-SDIVTI @REMAINING TRAFFIC FROM I TO J
417             RTRFJ=TR(J,I)-SDIVTJ @REMAINING TRAFFIC FROM J TO I
418             DO 500 II=1,NR
419             IF(II.EQ.I.OR.II.EQ.J) GO TO 500
420             IF(TRR(I,II) .EQ.0. .OR. TRR(II,J) .EQ.0.) GO TO 500
421       C
422       C     DETERMINE DELTA COST FOR INCREASED CAPACITY IN ALTERNATE ROUTES
423       C     LINK (I,II)
424       C
425             AIII=TRR(I,II) + RTRFI
426             AJII= TRR(II,I) +RTRFJ
427             AM= AMAX1(AIII,AJII)
428             IDR1=NRSC(I)
429             IDR2=NRSC(II)
430             CALL LINNUM(AM,0.,LINEQA,LINUP,0,RHO)
431             DO 151 NN=1,N3
432             LINDI(NN)= LINEQA(NN)- NETCNF(I,II,NN)
433             NETCNF(I,II,NN)= LINEQA(NN)
434             NETCNF(II,I,NN)= LINEQA(NN)
435       151   CONTINUE
436       C
437       C     LINK(II,J)
438       C
439             BIII=TRR(II,J)+RTRFI
440             BJII=TRR(J,II)+RTRFJ
441             BM= AMAX1(BIII,BJII)
442             IDR1=NRSC(J)
443             CALL LINNUM(BM,0.,LINEQB,LINUP,0,RHO)
444             DO 111  NN=1,N3
445             LINDJ(NN)= LINEQB(NN)- NETCNF(J,II,NN)
446             NETCNF(J,II,NN)= LINEQB(NN)
447             NETCNF(II,J,NN)= LINEQB(NN)
448       111   CONTINUE
449             CALL NETWKC(SUMCST)
450             IF (SUMCST.GT.MINCST) GO TO 120
451             DO 207 NN=1,N3
452             LINADI(NN)= LINDI(NN)
453             LINADJ(NN)= LINDJ(NN)
454       207   CONTINUE
455             IIAD =II
```

```
456             MINCST =SUMCST
457       120   CONTINUE
458       C
459       C RESET TO INITIAL NETWORK CONFIGRUATION FOR NEXT TRY
460       C
461             DO 250 NN= 1,N3
462             NETCNF(I,II,NN)= NETCNF(I,II,NN)- LINDI(NN)
463             NETCNF(II,I,NN)= NETCNF(II,I,NN)- LINDI(NN)
464             NETCNF(J,II,NN)= NETCNF(J,II,NN)- LINDJ(NN)
465             NETCNF(II,J,NN)= NETCNF(II,J,NN)- LINDJ(NN)
466       250   CONTINUE
467       500   CONTINUE
468             TRR(I,IIAD)=TRR(I,IIAD)+RTREI
469             TRR(IIAD,J)=TRR(IIAD,J)+RTREI
470             TRR(J,IIAD)=TRR(J,IIAD)+RTREJ
471             TRR(IIAD,I)=TRR(IIAD,I)+RTREJ
472             RETURN
473             SUBROUTINE NETUP(IFLIP,IIAD,I,J)
474       C           ********************
475       C
476       C UPDATE THE INTERREGIONAL NETWORK WHEN THERE IS SOME SAVINGS
477       C
478       C           ********************
479             IF (IFLIP.EQ.1) GO TO 700
480       C
481       C     UPDATE THE NETWORK TRAFFIC MATRIX AND
482       C     UPDATE THE OPTIMAL INTERREGIONAL NETWORK
483       C
484             DO 99  NN=1,N3
485             ORINET (I,IIAD,NN)=ORINET(I,IIAD,NN)+ TLINAD(NN)
486             ORINET (IIAD,I,NN)=ORINET(IIAD,I,NN)+ TLINAD(NN)
487             ORINET (J,IIAD,NN)=ORINET(J,IIAD,NN)+ JLINAD(NN)
488             ORINET (IIAD,J,NN)=ORINET(IIAD,J,NN)+ JLINAD(NN)
489       99    CONTINUE
490       700   CONTINUE
491             DO 701 NN=1,N3
492             ORINET(I,J,NN)=0
493             ORINET(J,I,NN)=0
494       701   CONTINUE
495       C
496       C     RESET TRAFFIC MATRIX TR(NR,NR)
497       C
498             DO 900 IR= 1,NR
499             DO 910 IK= 1,NR
500             TR(IR,IK)= TRRM(IR,IK)
501       910   CONTINUE
502       900   CONTINUE
503       C
504       C UPDATE TOTAL COST FOR OVERALL NETWORK
505       C
506             ORICST=ORICST-MAXSAV
507       C
508       C UPDATE NLINK MATRIX
509       C
510             NLINK(I)=NLINK(I)-1
511             NLINK(J)=NLINK(J)-1
512             RETURN
513             END
```

```
@PRT STACOM.ICOSTJ/0777
```

```
51928*STACOM(1).ICOSTJ/0777
      1            SUBROUTINE ICOSTJ(LINEQU,I,J,LNKCHW,LNKCLN)
      2      C           ********************
      3      C
      4      C CALCULATE INSTALLATION ANNUAL RECURRING COSTS NEEDED FOR
      5      C COMMUNICATION LINK BETWEEN NODES I AND J.  LNKCHW= OTHERS
      6      C LNKCLN= LINES. I AND J ARE GLOBAL INDICE FOR SYSTEM TERMINATIONS
      7      C UNDER CONSIDERATION, LINEQU= LINE CONFIGURATION BETWEEN I AND J
      8      C
      9      C           ********************
     10            PARAMETER NP1=130,NP2=1,NP3=4,NP4=3,NPC=350
     11            PARAMETER NP6=(NPC*NPC/2-NPC+1)/4+1
     12            PARAMETER NP7=4
     13            DIMENSION LINEQU(NP3), LNKCHW(NP3,NP4,2), LNKCLN(NP3,2)
     14            COMMON/LINCHR/LINMIX(NP3),LINCAP(NP3),UTILIZ(NP3)
     15           *   /CONST/N1,N2,N3,N4,N7,NCITY
     16           *   /RCOST/AINSTC(NP2,NP3,NP4,3,2,2),RECRC(NP2,NP3,NP4,3,2,2),
     17           * ANSTLN(NP2,NP3,3,2,2), RECRLN(NP2,NP3,3,2,16),IDUPLX(NP3)
     18           *   /INF/IRATEJ(NP2,NP2),IRAND(NPC,2),IFLAG(NP2,NP3)
     19           *   /ADD/ IADD(NP1),KCHG,KADD @TERMINALS WITH SAME V=H
     20           *   /REF/IREF(NPC),TRAFD(NP1,2,NP7),DSTNCE(NP6),MAPADR(NP1)
     21      C
     22      C INITIALIZATION
     23      C
     24            II=MAPADR(I)
     25            JJ=MAPADR(J)
     26            IADDTN=IADD(J)
     27            DO 100 NL=1,N3
     28            DO 100 NM=1,2
     29            LNKCLN(NL,NM)=0
     30            DO 100 NK=1,N4
     31            LNKCHW(NL,NK,NM)=0
     32      100   CONTINUE
     33            KRATEI= IRAND(II,1)   @ RATE STRUCTURE TYPE FOR NODE I
     34            KRATEJ= IRAND(JJ,1)   @ RATE STRUCTURE TYPE FOR NODE J
     35            KDENSI= IRAND(II,2)   @ TRAFFIC DENSITY TYPE FOR NODE I
     36            KDENSJ= IRAND(JJ,2)   @ TRAFFIC DENSITY TYPE FOR NODE J
     37            KDNSTY= KDENSI+KDENSJ @ ACTUAL DENSITY(2=H-H,1=H-L AND 0=L-L)
     38            KK=KDNSTY+1  @3=H-H, 2=H-L AND 1=L-L
     39            DST  = DIST(I,J)  @ DISTANCE BETWEEN NODES I AND J
     40            IDST=DST
     41            ITIP=1       @PRIME COST FOR H/W UNIT
     42            IF(DST .LE. 0.5) ITIP=2     @DISCOUNT COST FOR ADDITIONAL UNIT
     43            KR    = IRATEJ(KRATEI,KRATEJ) @ ACTUAL RATE STRUCTURE TO BE USED
     44            DO 1  IL= 1,N3
     45            IDPX= IDUPLX(IL)       @ DUPLEXING MODE 1=H AND 2=F
     46            NDV = LINEQU(IL)      @ NUMBER OF LINES REQUIRED
     47            NDV1=NDV*IADDTN*KADD
     48      C
     49      C CALCULATE COSTS FOR NON-LINE TYPE CHARGES
     50      C
     51            IF (NDV.EQ.0) GO TO 1 @ NO LINES ARE REQUIRED
     52            DO 2 IV=1,N4   @HIGH DENSITY RATE
     53      C
     54      C INSTALLATION COSTS FOR NON-LINE TYPE CHARGES
     55      C
     56            LNKCHW (IL,IV,1)= AINSTC(KR,IL,IV,KK,IDPX,ITIP)*KCHG*NDV
```

```
     1       + AINSTC(KR,IL,IV,KK,IDPX,2)*NDV1
C
C ANNUAL RECURRING COSTS FOR NON-LINE TYPE CHARGES
C
      LNKCHW (IL,IV,2)=( RECRC(KR,IL,IV,KK,IDPX,ITIP)*KCHG*NDV
     1       + RECRC(KR,IL,IV,KK,IDPX,2)*NDV1)*12.
2     CONTINUE
C
C     CALCULATE LINE COSTS
C
      LIN= IFLAG(KR,IL)      @ LINEAR IF 1 AND NONLINEAR OTHERWISE
C
C     ANNUAL LINE INSTALLATION COST
C
      AN=1.
      LNKCLN(IL,1)= ANSTLN(KR,IL,KK,IDPX,2)*AN*NDV
      IF (LIN.NE.1) GO TO 41
C
C     LINEAR LINE RECURRING COST FUNCTION
C
      RN=DST/RECRLN(KR,IL,KK,IDPX,1)
      LNKCLN(IL,2)=  RECRLN(KR,IL,KK,IDPX,2)*RN*NDV*12.
      GO TO 32
41    CONTINUE
C
C     NONLINEAR LINE RECURRING FUNCTION
C
      DO 10 NON=1,8
        NON2=2*NON
        NON1=NON2-1
        COST=RECRLN(KR,IL,KK,IDPX,NON2)
        DT=RECRLN(KR,IL,KK,IDPX,NON1)
        IF (DST.GT.DT) GO TO 51
        LNKCLN(IL,2)= COST*DST*NDV*12+LNKCLN(IL,2)
        GO TO 32
51      CONTINUE
        LNKCLN (IL,2)= COST*DT*NDV*12+LNKCLN(IL,2)
        DST=DST-DT
10    CONTINUE
32    CONTINUE
1     CONTINUE
      KCHG=1
      KADD=1
      RETURN
      END
@PRT STACOM.RHOFUN/0777
```

```
51928*STACOM(1),RHOFUN/0777
      1              SUBROUTINE RHOFUN(T1,T2,LINEQU,LNLMT,RHOLIN,RHO)
      2       C             ********************
      3       C
      4       C CALCULATE LINE UTILIZATION
      5       C        T1= LINE TO SWITCHER TRAFFIC
      6       C        T2= SWITCHER TO LINE TRAFFIC
      7       C     LNLMT= HIGHEST LINE TYPE
      8       C   LINEQU= LINE CONFIGURATION
      9       C
     10       C             ********************
     11               PARAMETER NP3=4
     12               COMMON/LINCHR/ LINMIX(NP3),LINCAP(NP3),UTILIZ(NP3)
     13              *    /CONST/ N1,N2,N3,N4,N7,NCITY
     14              *    /SUM/ASUM(4),RSUM
     15              *    /XMT/ TIMXMT(7,NP3),WAIT(6)
     16              *    /MSLA/ AMSL(7)
     17               DIMENSION LINEQU(1),RHOLIN(1)
     18               RHO=0.
     19               CAP=0.
     20               DO 8 N=1,N3
     21               CAP=CAP+LINEQU(N)*LINCAP(N)
     22       8       CONTINUE
     23               CN=LINCAP(LNLMT)/CAP  @NORMALIZATION FACTOR
     24               XSAC1=CN*T2*ASUM(3)/(RSUM*AMSL(5)*8.) @OUTPUT WITH PRIO 1
     25               XSAC2=0.
     26               IF(AMSL(6) .EQ. 0.) GOTO 1201
     27               XSAC2=CN*T2*ASUM(4)/(RSUM*AMSL(6)*8.) @OUTPUT WITH PRIO 2
     28       1201    CONTINUE
     29               XSAC3=CN*T1/(AMSL(4)*8.) @INPUT TRFFIC IN TRANS
     30               RHOLIN(1)=XSAC1*TIMXMT(5,LNLMT)
     31               RHOLIN(2)=XSAC2*TIMXMT(6,LNLMT)
     32               RHOLIN(3)=XSAC3*TIMXMT(4,LNLMT)
     33               RHO=RHOLIN(1)+RHOLIN(2)+RHOLIN(3)
     34               RETURN
     35               END

@PRT STACOM.LINNUM/0777
```

```
51928*STACOM(1),LINNUM/0777
      1            SUBROUTINE LINNUM(T1,T2,LINEQU,LNLMT,JFLAG,RHO)
      2      C           ********************
      3      C
      4      C FIND LINE CONFIGURATION BASED ON THE GIVEN TRAFFIC AND
      5      C APPLICABLE LINE TYPE
      6      C       JFLAG= 1 FOR MULTIDROP LINE CASE
      7      C          T1= LINE TO SWITCHER TRAFFIC
      8      C          T2= SWITCHER TO LINE TRAFFIC
      9      C
     10      C           ********************
     11            PARAMETER NP3=4
     12            COMMON/LINCHR/ LINMIX(NP3),LINCAP(NP3),UTILIZ(NP3)
     13           *   /CONST/ N1,N2,N3,N4,N7,NCITY
     14           *   /MSLA/ AMSL(7)
     15            INTEGER TRAF
     16            DIMENSION LINEQU(1),RHOLIN(3)
     17            TRAF=T1+T2
     18            DO 1 I=1,N3
     19            LINEQU (I)=0
     20      1     CONTINUE
     21            LNLMT=0
     22            CALL REFER
     23            LNLMTU=LNLMT
     24      C
     25      C SET UP INITIAL LINE CONFIGURATION
     26      C
     27            IF(JFLAG .EQ. 1) GOTO 10
     28      3     CONTINUE
     29            LINEQU(LNLMT)=LINEQU(LNLMT)+1
     30            LCAP=LINCAP(LNLMT)*UTILIZ(LNLMT)
     31            IF(TRAF.LT.LCAP) GOTO 7
     32            TRAF=TRAF-LCAP
     33            CALL REFER
     34            GOTO 3
     35      10    CONTINUE
     36            LINEQU(LNLMT)=TRAF/(LINCAP(LNLMT)*UTILIZ(LNLMT))+1
     37      7     CONTINUE
     38      70    CONTINUE
     39            CALL RHOFUN(T1,T2,LINEQU,LNLMTU,RHOLIN,RHO)
     40            IF(RHO .LT. UTILIZ(LNLMT)) GOTO 150
     41            IF(LNLMTU.NE.N3) GOTO 72
     42            IF(JFLAG.NE.1) GOTO 73
     43            LINEQU(N3)=LINEQU(N3)+1  Q NEEDED TO BE MODIFIED
     44            GOTO 70
     45      73    CONTINUE
     46            DO 2 N=1,N3
     47              IF(LINEQU(N) .NE. 0) GOTO 20
     48      2     CONTINUE
     49      20    CONTINUE
     50            NL=N
     51            IF(NL.EQ.N3) GOTO 74
     52            LINEQU(NL)=0
     53      22    CONTINUE
     54            NL=NL+1
     55            IF(LINMIX(NL).EQ.0) GOTO 22
     56            LINEQU(NL)=LINEQU(NL)+1
```

```
57              GOTO 70
58        74    CONTINUE
59              LINEQU(1)=1
60              GOTO 70
61        72    CONTINUE
62              LINEQU(LNLMTU)=0
63              LNLMTU=LNLMTU+1
64              IF(LINMIX(LNLMTU).EQ.0) GOTO 72
65              LINEQU(LNLMTU)=1
66              GOTO 70
67        150   CONTINUE
68              LNLMT=LNLMTU
69              RETURN
70              SUBROUTINE REFER
71        C
72        C FIND THE UPPER LIMIT OF LINE TYPE ALLOWED
73        C
74              DO 14 NN=1,N3
75                LTRAF=TRAF/UTILIZ(NN)+0.5
76                IF(LINMIX(NN) .EQ.0) GOTO 14
77                LNLMT=NN
78                IF(LINCAP(NN) .GT. LTRAF) GOTO 15
79        14    CONTINUE
80        15    CONTINUE
81              RETURN
82              END
```

```
QPRT STACOM.PACK/0777
```

```
51928*STACOM(1).PACK/0777
     1              COMPILER (FLD=ABS)
     2              SUBROUTINE PACK(I,K,L,IA)
     3        C           ********************
     4        C
     5        C RETRIEVE/STORE DATA FROM/INTO ARRAY IA
     6        C L=1 FOR STORING AND L=2 FOR RETRIEVAL
     7        C K= DISTANCE DATA CONCERNED
     8        C
     9        C           ********************
    10              DIMENSION IA(1)
    11              IQ=(I-1)/4 @THE WORD LOCATION
    12              IR=I-IQ*4   @THE QUARTER CONCERNED
    13              IQ=IQ+1
    14              IS=(IR-1)*9
    15              IF(L.EQ.1) GOTO 10
    16        C
    17        C RETRIEVE IT(9 BITS) BEGINNING AT IS-TH BIT OF THE IQ-TH WORD
    18        C
    19              K=FLD(IS,9,IA(IQ))
    20              RETURN
    21        C
    22        C STORE IT(9 BITS) BEGINNING AT IS-TH BIT OF THE IQ-TH WORD
    23        C
    24        10    CONTINUE
    25              FLD(IS,9,IA(IQ))=K
    26              RETURN
    27              END

@PRT STACOM.DIST/0777
```

```
51928*STACOM(0),DIST/0777
     1              FUNCTION DIST(I,J)
     2       C             ********************
     3       C
     4       C FIND DISTANCE BETWEEN I AND J
     5       C
     6       C             ********************
     7              PARAMETER NP1=130,NPC=360
     8              PARAMETER NP6=(NPC*NPC/2-NPC+1)/4+1
     9              PARAMETER NP7=4
    10              COMMON /REF/IREF(NPC),TRAFD(NP1,2,NP7),
    11             *            DSTNCE(NP6),MAPADR(NP1)
    12              INTEGER DSTNCE
    13              DIST=0.
    14              IF(I.EQ.J) RETURN
    15              II=MAPADR(I)  @ACTUAL CITY INDEX
    16              JJ=MAPADR(J)
    17              IF(II.EQ.JJ) RETURN
    18              IJL=LINK(II,JJ)
    19              CALL PACK(IJL,IDIST,2,DSTNCE)
    20              DIST=IDIST
    21              IF(IDIST .NE. 511) RETURN
    22              DIST=RECOVR(IJL)
    23              RETURN
    24              END

@PRT L.LINK/0777
```

```
51928*STACOM(0).LINK/0777
      1              FUNCTION LINK(J,K)
      2        C           ********************
      3        C
      4        C FIND THE RELATIVE LOCATION FOR (J,K) COMBINATION
      5        C WHICH IS THEN USED FOR FINDING DISTANCE BETWEEN SYSTEM
      6        C TERMINATIONS J AND K
      7        C
      8        C           ********************
      9              PARAMETER  NP1=130,NPC=360
     10              PARAMETER NP6=(NPC*NPC/2-NPC+1)/4+1
     11              PARAMETER NP7=4,NP3=4
     12              COMMON /CONST/ N1,N2,N3,N4,N7,NCITY
     13             1  /REF/IREF(NPC),TRAFD(NP1,2,NP7),DSTNCE(NP6),MAPADR(NP1)
     14              INTEGER DSTNCE
     15              LINK=0
     16              IF(J.GT.NCITY.OR.K.GT.NCITY.OR.K.EQ.J) RETURN
     17              JJ=J
     18              KK=K
     19              IF(J.LT.K) GOTO 1
     20              JJ=K
     21              KK=J
     22        1     CONTINUE
     23              LINK=(JJ-1)*NCITY + KK - IREF(JJ)
     24              RETURN
     25              END

@PRT L.RECOVR/0777
```

```
51928*STACOM(0),RECOVR/0777
     1              FUNCTION RECOVR(I)
     2       C             ********************
     3       C
     4       C RETRIEVE OVERFLOW DISTANCE DATA FROM IVRD
     5       C
     6       C             ********************
     7              PARAMETER NPC=360,NPO=10*NPC
     8              COMMON /OVER/ IVRD(NPO,2),IOVER1
     9              DO 10 N=1,IOVER1
    10              IF(I .EQ. IVRD(N,1)) GOTO 20
    11       10     CONTINUE
    12              WRITE(6,99) I
    13       99     FORMAT(1X,' NO OVERFLOW DATA HAS BEEN FOUND ',
    14            *     'FOR LOCAL INDEX',2I6)
    15              STOP
    16       20     CONTINUE
    17              RECOVR=IVRD(N,2)
    18              RETURN
    19              END

@PRT L.PLOTPT/0777
```

```
51928*STACOM(0),PLOTPT/0777
      1             SUBROUTINE PLOTPT(L1,L3)
      2       C           ********************
      3       C
      4       C SUBROUTINE FOR MOVING CALCOMP PEN WITH OR WITHOUT PEN DOWN
      5       C   L3=1 FOR MOVING WITH PEN DOWN
      6       C      =2 FOR MOVING WITH PEN UP
      7       C      =3 FOR CLOSING THE PLOTTING
      8       C
      9       C           ********************
     10             PARAMETER NPC=360
     11             COMMON/VH/IVERT(NPC),IHORZN(NPC)
     12             DIMENSION IBUF(1000)
     13             DATA IP/0/  @FLAG FOR PLOTS CALL
     14             DATA X/1.2566/
     15             IF(IP .NE. 0) GOTO 50
     16             CALL PLOTS
     17       50    CONTINUE
     18             IF(L3 .EQ. 3) GOTO 100   @PLOTTING IS TO BE CLOSED
     19             AV=IVERT(L1)
     20             AH=IHORZN(L1)
     21             BV=AV*COS(X)+AH*SIN(X)
     22             BH=AV*SIN(X)-AH*COS(X)
     23             BH=(8300.-BH)/301.
     24             BV=(BV-5500)/301.
     25             IF(L3 .EQ. 2) GOTO 80
     26             CALL SYMBOL(BH,BV,0.025,4,0.,-2)  @PEN IS DOWN
     27       80    CONTINUE
     28             CALL PLOT(BH,BV,3)  @PEN IS UP
     29             IP=1    @PLOTS CALL IS NOT NEEDED ANY MORE
     30             RETURN
     31       100   CONTINUE
     32             CALL PLOT(10.,0.,999)
     33             RETURN
     34             END

@PRT L.RSPNSE/0777
```

```
51928*STACOM(0).RSPNSE/0777
      1            SUBROUTINE RSPNSE(T1,T2,LINTYP,M,IOK)
      2      C     ******************
      3      C
      4      C CALCULATE MEAN RESPONSE TIME FOR THE PROPOSED MULTIDROP LINE
      5      C
      6      C WAIT(6)=ITEMIZED DELAYS DUE TO
      7      C     1=WAIT FOR POLLING    2=WAIT FOR I/O       3=INPUT XMT TIME
      8      C     4=CPU TURNAROUND      5=OUTPUT QUEUE WAIT 6=OUTPUT XMT TIME
      9      C
     10            PARAMETER NP3=4
     11            COMMON/RESP/ RHOLIN(6),RSPTIM
     12           1   /XMT/ TIMXMT(7,NP3), WAIT(6)
     13           2   /BOUND/ NTERMS,TIMREQ,MPROC,MPLOT
     14           3   /CONST/N1,N2,N3,N4,N7,NCITY
     15            DIMENSION LDUMMY(NP3)
     16            DO 10 N=1,N3
     17              LDUMMY(N)=0
     18      10    CONTINUE
     19            LDUMMY(LINTYP)=1
     20            IOK=0
     21      C
     22            CALL RHOFUN(T1,T2,LDUMMY,LINTYP,RHOLIN,P=0)
     23            RHOLIN(4)=1.-RHO
     24            IF(RHOLIN(4) .LE. 0.) RETURN
     25            WAIT(1)=(TIMXMT(1,LINTYP)+TIMXMT(2,LINTYP))*(M-1)/2
     26            WAIT(2)=(1.-RHOLIN(4))*TIMXMT(7,LINTYP)/(1-RHOLIN(1)-
     27           1        RHOLIN(2))*RHOLIN(4))
     28            WAIT(3)=TIMXMT(3,LINTYP)
     29            WAIT(5)=(1.-RHOLIN(4))*TIMXMT(7,LINTYP)/(1-RHOLIN(1))
     30            WAIT(6)=TIMXMT(5,LINTYP)
     31            RSPTIM=0.
     32            DO 11 J=1,6
     33              RSPTIM=WAIT(J)+RSPTIM
     34      11    CONTINUE
     35            IF(RSPTIM .GT. TIMREQ) RETURN
     36            IOK=1     RESPONSE TIME CRITERION IS SATISFIED
     37            RETURN
     38            END

QPRT L.INPUT/0777
```

APPENDIX B

GLOSSARY

AT&T	American Telephone and Telegraph Company
@RUN, etc.	Control statements under EXEC 8 system (of the UNIVAC computer system)
BPS	Bits per second
CalComp	CALifornia COMputer Products
Central Link	The direct link between a computer and a remote terminal
Centroid	The geographical center of a set of system terminations
Communication Network	A network with several terminals connected by a set of communication channels
Communication Protocol	The system used for performing interfacing (hand-shaking) between a computer and a remote terminal
CPU	Central Processing Unit
Data Base	A collection of cross-referenced set of files which allows systematic data filing and retrieval by a digital computer
D Bank	Storage area for data under EXEC-8 system of the UNIVAC Computer System.
Drop	A chargeable item associated with each terminal on a multidrop line
EXEC-8	UNIVAC 1100 series executive system
FORTRAN	FORmula TRANslator
FORTRAN V	A FORTRAN type of high level language which is only applicable in UNIVAC computers
I Bank	Storage area for program instructions under EXEC 8 system of the UNIVAC Computer System
ID	IDentification
Line Utilization	The ratio of traffic on a line to the line capacity

MPL	Multischedule Private Line, one of the interstate tariffs used by AT&T
Multidrop Line	A communication line which has more than one terminal and is connected to a data processing system
Multidrop Network	A communication network where one or more lines are multidrop lines
PUNCH$, etc.	System designated file name for punch card output, etc.
Regional Network	A network which connects all terminals in a given region
Regional Switching Center (RSC)	A regional data processing center which is used to provide the message switching capability for all terminals in the region
STACOM	STAte Criminal Justice COMmunication Project
Star Network	A communication network where each system termination is directly connected to the central data processing system
SUP-Time	A run time estimate by the EXEC-8 accounting subsystem which accounts for the amount of time spent by a run on usage of CPU, I/O processing and execution of system control statements and executive requests
System Termination	A logical node in the communication system under the STACOM program, which consists of one or more physical terminals
TELPAK	A specific tariff for a telecommunication network
Terminal	A device that allows users of a data processing system to gain access to that system in a more convenient manner than the input/ output devices local to that system
Terminal Response Time	The duration from the time a user initiates a request for network service at the terminal to the time he receives a complete response
Tree	A graph which has a root node without any predecessors and other nodes have unique predecessors

UNIVAC	UNIVersal Automatic Computer, a computer trade name by Sperry Rand Corporation
Vertical Horizontal (V-H) Coordinates	A pair of numbers which are designated by AT&T for cities and used for the purpose of calculating distance between any two cities

☆ U. S. GOVERNMENT PRINTING OFFICE : 1978 260-992/2159

END