ED 169 949 IR 007 239

AUTHOR Cooper, George E.; And Others
TITLE MOVANAID: An Interactive Aid for Analysis of Movement Capabilities.
INSTITUTION Vector Research, Inc., Ann Arbor, Mich.
SPONS AGENCY Army Research Inst. for the Behavioral and Social Sciences, Arlington, Va.
REPORT NO ARI-TP-305
PUB DATE Aug 78
CONTRACT DAHC-29-74-C-003
NOTE 75p.; Some figures may not reproduce well due to small type size and/or occasional broken type face

EDRS PRICE MF01/PC03 Plus Postage.
DESCRIPTORS *Computer Programs; Decision Making; *Information Processing; Information Systems; Information Utilization; *Military Science; Transportation

ABSTRACT
A computer-drive interactive aid for movement analysis, called MOVANAID, has been developed to be of assistance in the performance of certain Army intelligence processing tasks in a tactical environment. It can compute fastest travel times and paths through road networks for military units of various types, as well as fastest times in which simultaneous maneuvers of certain types can be completed. Military situations in which the aid might be useful are discussed, and the manner in which users interact with the aid is described. Details about the analytic basis upon which the aid has been constructed and a discussion of the method used for computer implementation are also given. Possible extensions of the capabilities of the aid are also discussed. (Author/JVP)

ED169949

Technical Paper 305

MOVANAID: AN INTERACTIVE AID FOR ANALYSIS OF MOVEMENT CAPABILITIES

George E. Cooper and Michael H. Moore
Vector Research, Inc.

and

Stanley M. Halpin

BATTLEFIELD INFORMATION SYSTEMS TECHNICAL AREA

Submitted as complete and
technically accurate, by:
Edgar M. Johnson
Technical Area Chief

Approved By:

Milton H. Birnbaum, Acting Director
ORGANIZATIONS AND SYSTEMS
RESEARCH LABORATORY

Joseph [illegible]
TECHNICAL DIRECTOR ([illegible]TE)

U.S. ARMY RESEARCH INSTITUTE FOR THE BEHAVIORAL AND SOCIAL SCIENCES
5001 Eisenhower Avenue, Alexandria, Virginia 22333

Office, Deputy Chief of Staff for Personnel
Department of the Army

August 1978

Army Project Number
2Q16210A754

Intelligence Systems

IR007239

2

FOREWORD

The Battlefield Information Systems Technical Area is concerned with the demands of the future battlefield for increased man-machine complexity to acquire, transmit, process, disseminate, and utilize information. Research is focused on the interface problems and interactions within command and control centers and is concerned with such areas as topographic products and procedures, tactical symbology, user-oriented systems, information management, staff operations and procedures, and sensor systems integration and utilization.

One area of special interest is the development of aids to support command staff information processing and decisionmaking. The introduction of tactical data systems creates the opportunity for a large increment in operational effectiveness through exploitation of their potential to assist in the acquisition, analysis, correlation, integration, and utilization of data. To achieve this potential current manual procedures must be carefully analyzed to identify critical tasks capable of crediting computer support, but not amenable to rapid manual processing. Aids developed in this context must be experimentally verified and validated in field settings.

The present publication discusses the results of a preliminary analysis matching computer capabilities to intelligence processing and the development of a prototype analysis aid, MOVANAID. This aid demonstrates the capability of automation to assist the intelligence analyst in the evaluation of enemy mobility capabilities.

Research in the area of problem solving and decision support is conducted as an in-house effort augmented by contracts with organizations that have unique capabilities and facilities for research in this area. The present effort was conducted by personnel from Vector Research, Inc. under contract DAHC[illegible]-C-[illegible]3 with the in-house collaboration of Stanley M. Halpin under the program direction of Robert S. Andrews. This effort was responsive to requirements of Army Project 2Q[illegible]20A754 and to special requirements of the Combined Arms Combat Development Activity, Fort Leavenworth, Kansas.

[illegible]

JOSEPH ZEIDNER
Technical Director (Designate)

3/4/5

MOVANAID: AN INTERACTIVE AID FOR ANALYSIS OF MOVEMENT CAPABILITIES

BRIEF

Requirement:

To develop an on-line analytic aid [illegible]r use in the processing of tactical intelligence data, and to implement such an aid in the ARI Training and Information Systems Facility (TISF).

Procedure:

[illegible] analysis of current doctrine for tactical intelligence processing and a parallel examination of available analytic procedures in the mathema[illegible]al and operations research literature led to the tentative identi[illegible]ation of several potentially applicable aids. An evaluation of these aids resulted in the selection of an aid for analysis of movement ca[illegible]ilities for furth[illegible] development and implementation.

Produc[illegible]

A[illegible] [illegible]-line interactive [illegible], MOVANAID, has been implemented in the ARI TI[illegible] This aid provides an intelligence analyst with the capability for a [illegible] examination of expected travel times and route selections for ene[illegible] units, thus facil[illegible]ting the analysis of enemy capabilities.

Utiliza[illegible]:

MOVANAID serves as a pro[illegible]type for a general class of analytic aids, and as such provides th[illegible] basis for further development and evaluation. The specific solution to analysis of mobility questions, as well as the more general form of interactive analytic aids that are represented by MOVANAID, has immediate impact on developing intelligence processing systems. The laboratory implementation of MOVANAID provides a test bed fo[illegible] examining general principles of man-machine systems and specific fact[illegible]rs influencing analysts' use of computer-based aids.

MOVANAID: AN INTERACTIVE AID FOR ANALYSIS OF MOVEMENT CAPABILITIES

CONTENTS

MOVANAID: AN INTERACTIVE AID FOR ANALYSIS OF MOVEMENT CAPABILITIES

BACKGROUND AND OBJECTIVE

Recent and continuing technological advances in data collection and information exchange in tactical settings have had a considerable impact on the timeliness, quantity, and quality of information that the commander has available for planning and executing his mission. The advantage of access to massive quantities of data, however, may be quickly obviated by the problems associated with effective use of the data. It is clear that some form of computer-based storage and retrieval system will be necessary if Army tactical operating units are to capitalize on the advantages provided by extensive information; the Army has a variety of tactical data systems under development that are intended to meet this need.

Computer-based information storage and retrieval systems are typically "input/output bound"; that is, the central processing unit of the computer spends only a fraction of the time in productive operation, and spends the bulk of the time waiting for input or output devices to perform their respective tasks. ARI hopes to develop computer programs and computer-supported procedures that use this "free time," thereby maximizing the computer use and providing valuable assistance to tactical staffs. The Intelligence Systems Program at ARI is exploring ways of using the developing tactical data systems to support information processing and other aspects of the intelligence production cycle. The current project focuses on the analyses conducted in the process of transforming raw information to processed intelligence. The objectives of this project are as follows:

1. To further develop the concept of analytic aids for intelligence processing and to suggest specific aids which might be of some value;

2. To select one such analytic aid and to develop the necessary algorithms and procedures (or to adapt existing algorithms and procedures) to support the analytic portion of the aid; and

3. To implement this aid in the ARI Training and Information Systems Facility (TISF).

The aid selected for further development was MOVANAID, an on-line interactive Movement Analysis Aid. This aid incorporates principles and algorithms for network analysis developed over a period of several years within the operations research community. The concept of MOVANAID and the details of its implementation are discussed below, following a description of the process which led to the selection of MOVANAID.

Concept Development

The development of concepts for computer-based analytic aids was approached from two directions: (1) an examination of the relevant Army field manuals, instructional materials from the Army Intelligence School, and other doctrinal literature related to the activities and duties of Division G2 (Intelligence) staff personnel (coupled with extensive discussions with experienced Intelligence officers); and (2) an examination of techniques, algorithms, and concepts available in the operations research literature. Many possible analytic aids were suggested and evaluated for their potential for development within the constraints of the present project. An "ideal" aid was defined as one that would (1) be nontrivial (i.e., perform operations faster and/or better than could be done by an unaided analyst); (2) have a reasonable chance of having the development phase completed within the resource limitations of the project; (3) avoid dealing with special categories of intelligence data or products that might require some level of classification for the project; (4) be applicable within the context of division-level mid-intensity conventional warfare; (5) focus on intelligence processing rather than on collection or dissemination; and (6) involve mathematical, computational, and/or operations research approaches to information processing. Several potential aids were analyzed in the process of selecting an aid for development. This report is limited to a discussion of MOVANAID, the analytic aid selected for development and implementation in the ARI laboratory facility.

Three distinguishable levels of development are possible for a tool such as MOVANAID. The preliminary version, developed for demonstration and testing, will be different in detail and perhaps in character from a laboratory development version structured for test and evaluation. The laboratory version, in turn, will differ from any final version which may be implemented within an Army tactical data system. Changes in the aid will be made from one level of development to the next to take into account experience with the aid to that point. This paper describes the second stage or laboratory version of the aid. The computer program described in Appendix A[1] is the preliminary version from which the current program was developed; the computational logic remains the same, although details of the interaction with the aid differ considerably in the more recent version.

[1]Appendixes A and B have been retained in the master file at ARI, Alexandria, Va., and are not included in this paper.

Military Movement Analysis Problem

One of the continuing responsibilities of Division G2 in a tactical situation is the preparation of periodic intelligence estimates[2] that include a discussion of significant enemy capabilities and probable courses of action. Estimates of enemy mobility are frequently necessary for intelligence and other estimates. It is plain, for example, that if an enemy is not physically capable of maneuvering to a location by a given time, he will not have the capability to attack that point at that time. Similarly, the enemy's capability to reinforce front-line units by a given time depends strongly on the speed of movement of the reinforcing units and the distance to be covered.

For an example involving the determination of probable enemy courses of action, suppose it is known or suspected that an enemy attack is imminent and the G2 therefore wishes to identify the avenue or avenues of approach that the enemy is likely to select. It may be possible for the G2 to eliminate some of the alternative avenues on the basis of mobility considerations alone, because the enemy would be unlikely to select an avenue along which his movement would be too slow to meet the requirements of the tactical situation.[3]

A thorough movement analysis requires extensive computation; indeed, it is doubtful that such an analysis could be conducted by intelligence processors within a time consistent with the needs of the dynamic tactical environment without using a computer-based aid. Thus, MOVANAID has the potential not only for reducing the burden on the G2 staff but also for improving the intelligence product.

MOVANAID: A USER'S VIEW

SIMTOS

The ARI test facility (TISF--Training and Information Systems Facility) consists of a large laboratory space containing various relocatable computer interface devices and other equipment. One portion of the TSIF is currently arranged to accommodate SIMTOS, a Simulated Tactical Operations System. A SIMTOS G2 player is charged with the preparation of various estimates of enemy capabilities within the context of a division level planning exercise; the scenario involves

[2] Intelligence estimates are discussed in detail in Department of the Army Field Manual 30-5 (Combat Intelligence, Chapter 6 and Appendix J) and in Department of the Army Field Manual 101-5 (Staff Officer's Field Manual: Staff Organization and Procedures, Appendix B).

[3] In many cases, mobility factors are a primary consideration in the enemy's choice of avenue of approach (see Department of the Army Field Manual 30-102 (Handbook on Aggressor, Chapter 5).

preparation for a likely Aggressor attack against the 20th (U.S.) Mechanized Division positions in southeastern Germany. The SIMTOS player is provided with appropriate maps and overlays and is connected to a computerized data base via a CRT screen and keyboard and a teletypewriter. Although MOVANAID as described here has not been integrated into SIMTOS, the development of the aid was directed toward use within SIMTOS. In particular, the road network (Figure 1) is from the SIMTOS area, and all examples of its use are based on questions facing the SIMTOS player. The approximate location of the SIMTOS area is indicated on Figure 1 by appropriate UTM grid coordinates.

MOVANAID Capabilities

Users may call on MOVANAID to help them answer two basic questions about enemy movement capabilities:

1. How soon and by what path can a unit of type x, now located in location y, maneuver to destination z?

2. How soon and by what paths can units of types $x_1,\ldots,x_n$, now located in locations $y_1,\ldots,y_n$, complete a simultaneous maneuver to destinations $z_1,\ldots,z_n$?

Variations on each of these questions are possible, but MOVANAID is not presently equipped to treat them all. For questions of the second type, it is understood that it is not required of any unit i, $1 \le i \le n$, that the i^{th} unit travel to the i^{th} destination z; rather, a unit may maneuver through the network to any destination so long as each unit travels to some destination and each destination is the recipient of some unit. Such a question, therefore, asks for the assignment of units to destinations for which the largest travel time is a minimum. Note also that simultaneous maneuvers by several units could require two or more units to occupy the same arc or node at the same time, a situation that might lead to maneuvering delays in practice. Such conflicts are neither identified nor resolved by MOVANAID in its present form. A way to expand the capabilities of the aid so that it can take account of such conflicts is suggested in a later section.

The present MOVANAID is computer driven, and the interface between user and aid is on-line and interactive. Users interact with the aid in a hands-on fashion; information is transmitted from aid to subject via a cathode ray tube (CRT), and from user to aid via a keyboard. Information is transmitted in the form of written text; graphic displays are not used in this version of MOVANAID. A user can input only a small fraction of the data required by the aid to answer any eligible question. The bulk of the data (including all information about the network of roads to be considered by MOVANAID) has been made available to the aid in advance of any use by subjects. It is expected that any operational use of MOVANAID would necessarily involve a similar preparation of network information prior to use. (See Appendix B for a

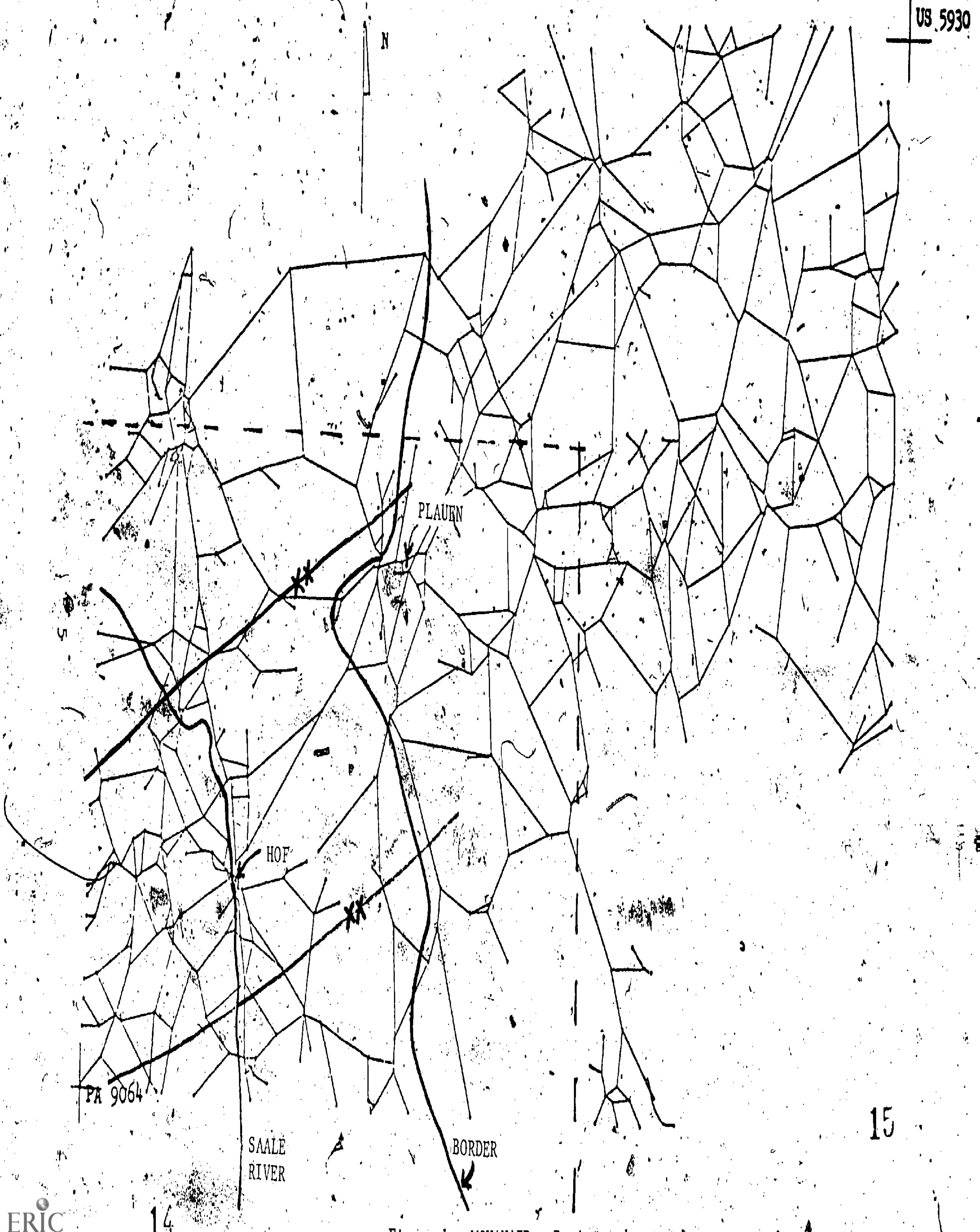


Figure 1. MOVANAID: Basic road network.

discussion of the procedures used for network generation in the SIMTOS area of operation.)

To make inputs to (and interpret outputs from) MOVANAID, users must be familiar with certain conventions that have been adopted for identifying nodes and classifying arcs of the SIMTOS road network. For purposes of quickly and easily identifying nodes of the network, a coordinate scheme resembling the usual Cartesian system has been adopted.

First, a convenient origin of coordinates is selected for the area of interest, thus establishing (imaginary) horizontal and vertical reference axes. Next, the area of operations is divided into square subareas (which we call "sectors") of side-length 10 kilometers. Next, each sector is subdivided into 100 squares (called "subsectors") of side-length 1 kilometer. Finally, nodes of the network are numbered with a five-digit number. The first two digits and the second two digits of a node number indicate, respectively, the horizontal and vertical coordinates of the lower left corner of the subsector within which the node lies.

It should be noted that in the SIMTOS network, the tens digit of these coordinates denotes the number of sectors rather than tens of kilometers, as might be supposed. This peculiarity is caused by an anomaly in the SIMTOS map in which all sectors having a lower left corner 20 kilometers to the right of the origin are not square and are less than 10 kilometers wide.

The last digit of the five-digit node number is always taken to be "0" unless two or more nodes of the network lie in the same subsector; in the latter case, the digits "1",...,"9" are also used. For example, if two nodes of the network are found to lie in a subsector whose coordinates are (18,54), one of the nodes would be numbered 18540 and the other would be numbered 18541. It is immaterial which node receives which number. A large overlay of the road network, such as Figure 1, showing the numbers as they exist in the MOVANAID data base is made available to users.

In computing travel times, the aid uses internally stored information about each arc of the road network (e.g., Figure 1) as follows:

1. Length

2. Type of route classified as one of five types:

   (a) major highway (e.g., autobahn),

   (b) main all-weather road (road with hard surface and two or more lanes),

   (c) other all-weather roads (hard surface and two lanes or less),

(d) artificial route (traveled in zero time, used by the computational routines. See Appendix B.),

(e) cross-country route.

3. Speed capability, classified as:

(a) highway speeds (traveled at speeds typical of nonurban areas),

(b) city speeds (traveled at speeds typical of urban areas),

(c) cross-country speeds, classified as one of three types:

(1) speeds typical of easy cross-country routes,

(2) speeds typical of cross-country routes of medium difficulty,

(3) speeds typical of difficult cross-country routes.

Although the above network definition is part of the permanent data input to the aid prior to use, the user may amend any of it on a temporary basis by entering new data on an interactive basis with the aid.

The remaining information necessary for the computation of unit movement times resides on permanent disk storage in a "speed table." This table describes the average speed of a unit of a given type (foot, tracked vehicles, or trucks) for given conditions (night, day, wet, dry) and for each of the nine route conditions (three highway types with urban or open country speeds and three cross-country speeds). This table is a simple extension of the table given in FM 30-102 (chapter 22) for Aggressor movement times. There is no provision for a user to change the speed table except by revising the permanent disk storage before use.

Interaction with MOVANAID

A diagram representing the user's interaction with MOVANAID is shown in Figure 2. The general format of the interaction is as follows: Any given CRT display contains information and/or a list of options in the upper half of the screen, with additional instructions and options on the lower half. The user selects an option, enters information, or otherwise responds to instructions by entering the appropriate data from the CRT keyboard.

The user first sees a brief series of introductory displays (Figure 3), which are supplemented by an oral briefing and instruction on the use of the system. After progressing through these displays, the user comes to the "Analysis Options" display (Figure 4). Here the user has six options from which to choose: decide to modify the map

MOVANAID INTERACTIVE DISPLAYS

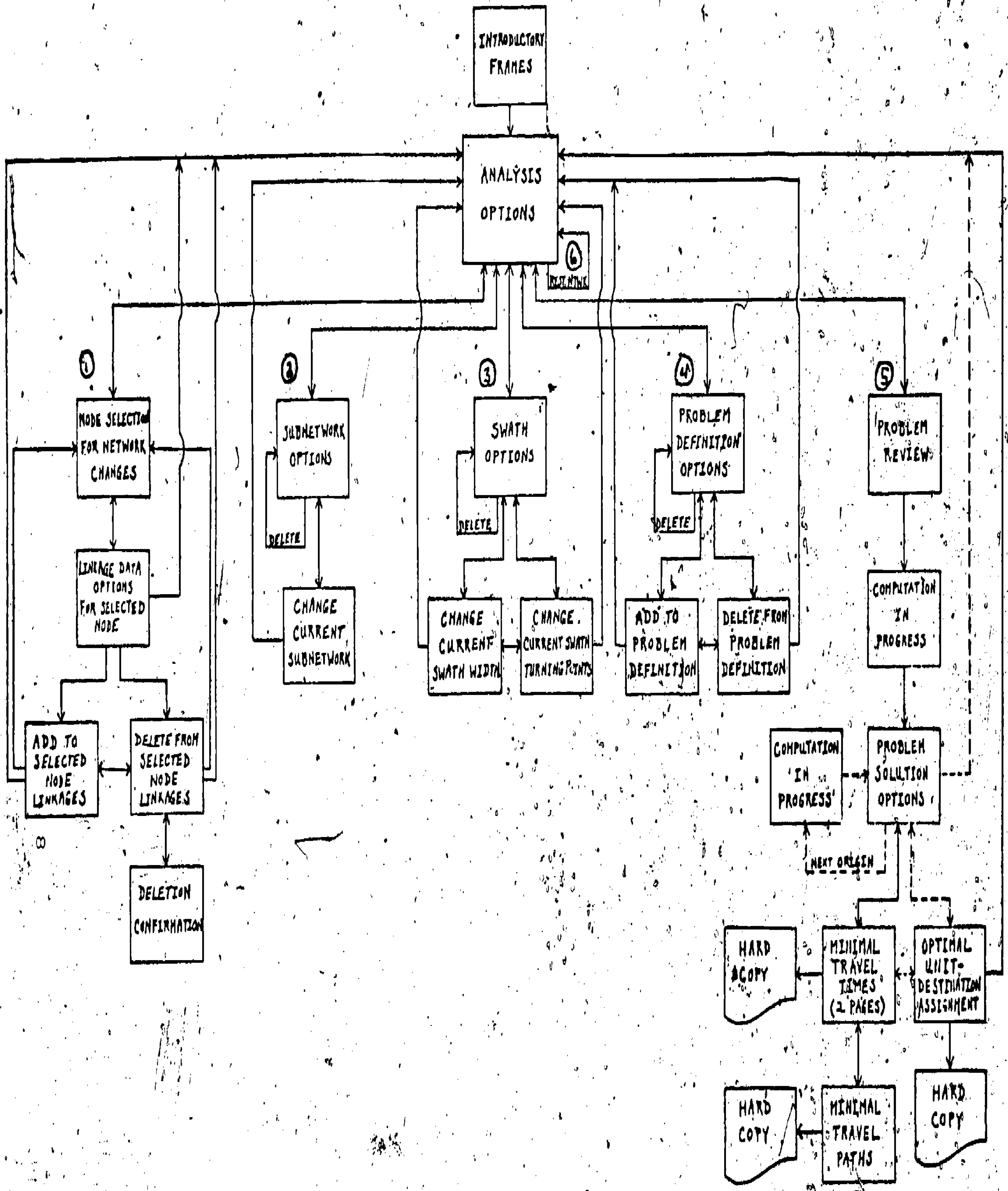
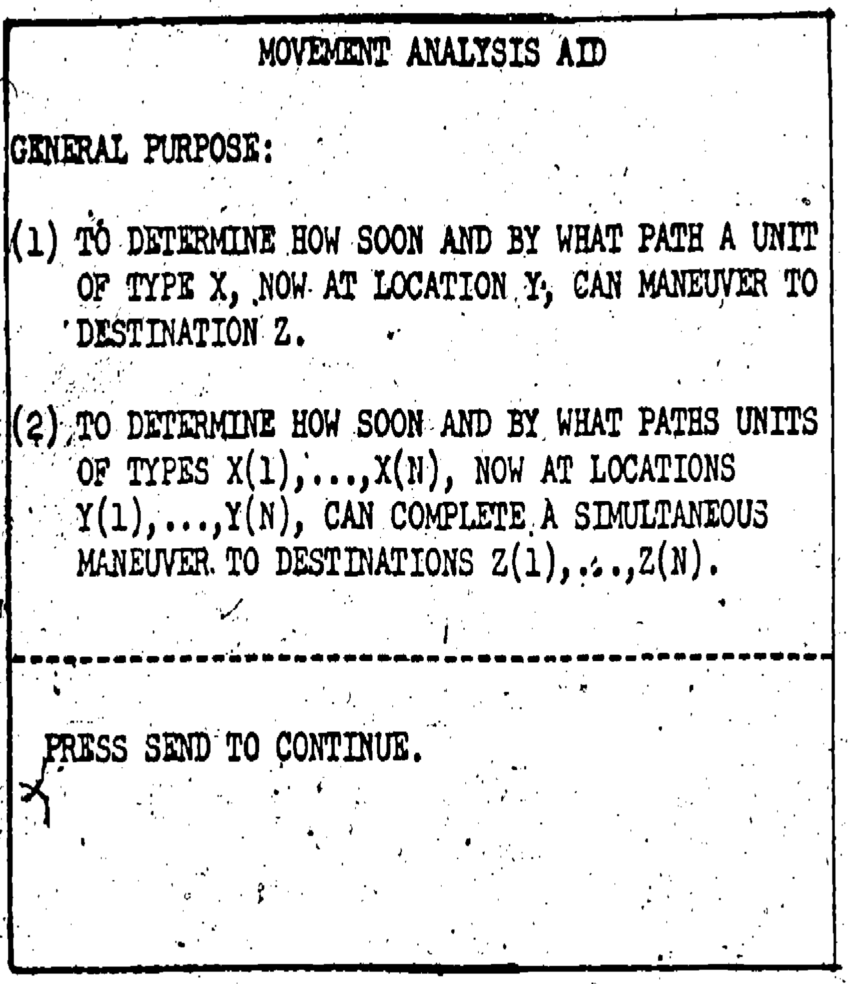

INTRODUCTORY FRAMES
ANALYSIS OPTIONS
6
RESET NTWK
1
NODE SELECTION FOR NETWORK CHANGES
LINKAGE DATA OPTIONS FOR SELECTED NODE
ADD TO SELECTED NODE LINKAGES
DELETE FROM SELECTED NODE LINKAGES
DELETION CONFIRMATION
2
SUBNETWORK OPTIONS
DELETE
CHANGE CURRENT SUBNETWORK
3
SWATH OPTIONS
DELETE
CHANGE CURRENT SWATH WIDTH
CHANGE CURRENT SWATH TURNING POINTS
4
PROBLEM DEFINITION OPTIONS
DELETE
ADD TO PROBLEM DEFINITION
DELETE FROM PROBLEM DEFINITION
5
PROBLEM REVIEW
COMPUTATION IN PROGRESS
COMPUTATION 'IN PROGRESS'
PROBLEM SOLUTION OPTIONS
NEXT ORIGIN
HARD COPY
MINIMAL TRAVEL TIMES (2 PAGES)
OPTIMAL UNIT-DESTINATION ASSIGNMENT
HARD COPY
MINIMAL TRAVEL PATHS
HARD COPY

Figure 2. Flow diagram of user interaction with MOVANAID.



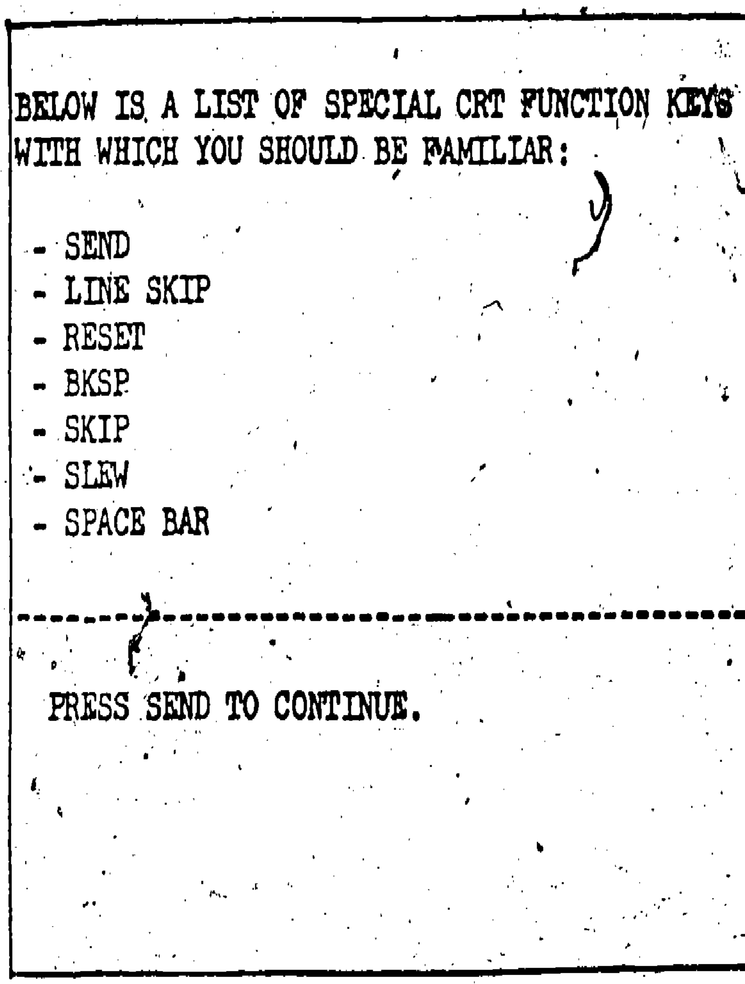
MOVEMENT ANALYSIS AID

GENERAL PURPOSE:

(1) TO DETERMINE HOW SOON AND BY WHAT PATH A UNIT OF TYPE X, NOW AT LOCATION Y, CAN MANEUVER TO DESTINATION Z.

(2) TO DETERMINE HOW SOON AND BY WHAT PATHS UNITS OF TYPES X(1),...,X(N), NOW AT LOCATIONS Y(1),...,Y(N), CAN COMPLETE A SIMULTANEOUS MANEUVER TO DESTINATIONS Z(1),...,Z(N).

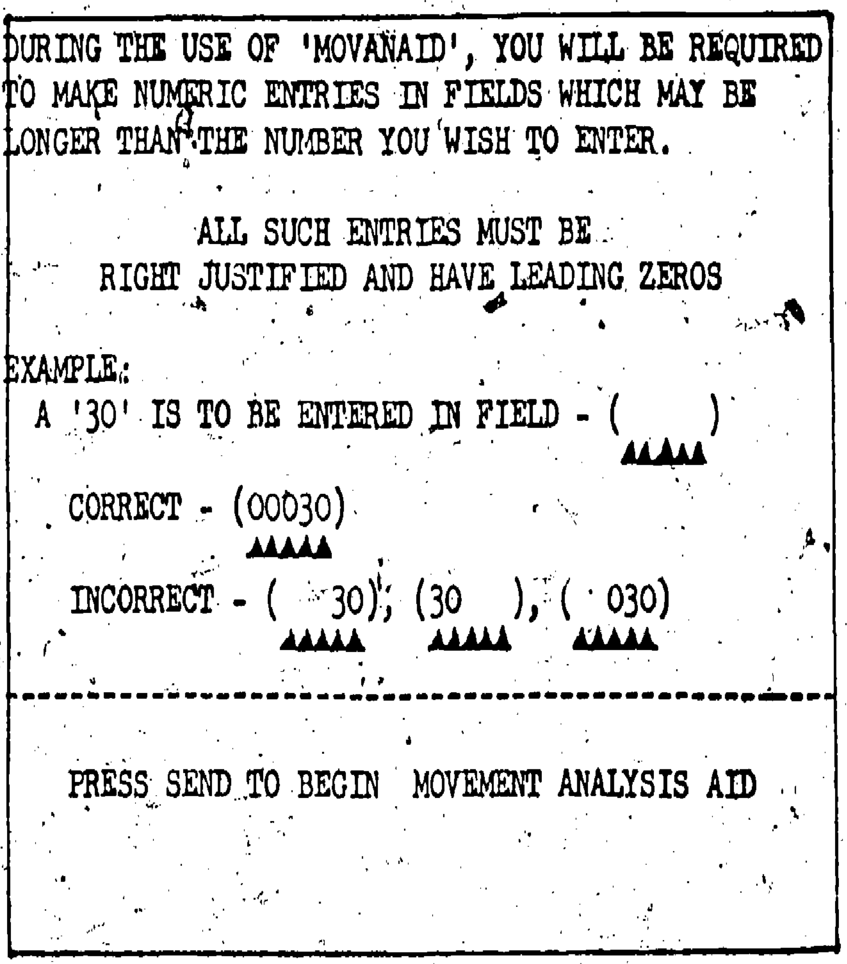
PRESS SEND TO CONTINUE.



BELOW IS A LIST OF SPECIAL CRT FUNCTION KEYS WITH WHICH YOU SHOULD BE FAMILIAR:

- SEND
- LINE SKIP
- RESET
- BKSP
- SKIP
- SLEW
- SPACE BAR

PRESS SEND TO CONTINUE.



DURING THE USE OF 'MOVANAID', YOU WILL BE REQUIRED TO MAKE NUMERIC ENTRIES IN FIELDS WHICH MAY BE LONGER THAN THE NUMBER YOU WISH TO ENTER.

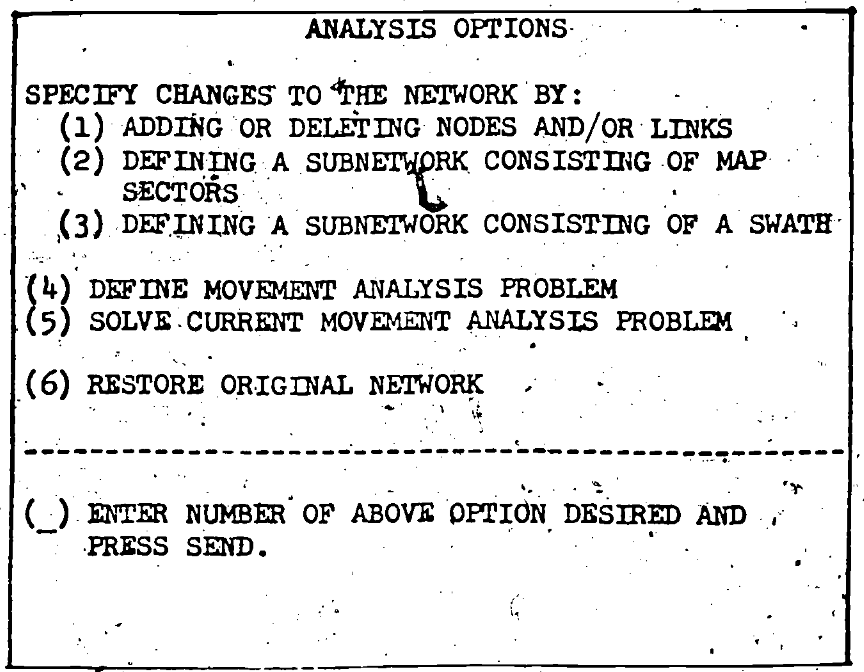
ALL SUCH ENTRIES MUST BE
RIGHT JUSTIFIED AND HAVE LEADING ZEROS

EXAMPLE:

```
  A '30' IS TO BE ENTERED IN FIELD - (     )
                                      ^^^^^
     CORRECT - (00030)
                ^^^^^
     INCORRECT - (   30), (30   ), (  030)
                  ^^^^^    ^^^^^    ^^^^^
```

PRESS SEND TO BEGIN MOVEMENT ANALYSIS AID

Figure 3. MOVANAID introductory displays.



ANALYSIS OPTIONS

SPECIFY CHANGES TO THE NETWORK BY:
(1) ADDING OR DELETING NODES AND/OR LINKS
(2) DEFINING A SUBNETWORK CONSISTING OF MAP SECTORS
(3) DEFINING A SUBNETWORK CONSISTING OF A SWATH

(4) DEFINE MOVEMENT ANALYSIS PROBLEM
(5) SOLVE CURRENT MOVEMENT ANALYSIS PROBLEM

(6) RESTORE ORIGINAL NETWORK

(_) ENTER NUMBER OF ABOVE OPTION DESIRED AND PRESS SEND.

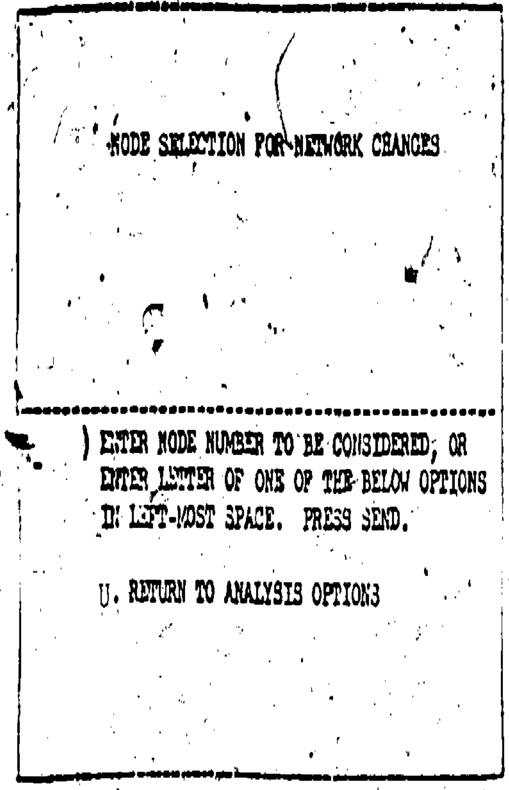
Figure 4: Display for analysis options.

network in one of several ways, define a problem to be solved, request a problem solution, or restore the network to its original state (see Figures 2 and 4). As shown in Figure 4, the choice of one of these options involves selecting the appropriate number from the "menu," entering that number in the indicated spot on the screen, and signaling the computer by pressing the "Send" key on the keyboard.

To temporarily add or delete nodes or arcs, the user would choose Option 1 on the Analysis Options display. MOVANAID then displays a request [illegible] the number of the node in the network that will be affected by the ch[illegible] Figure 5a). After receiving this information, MOVANAID displays [illegible] current linkage data for that node (Figure 5b), i.e., the numbers [illegible] other nodes connected to that "primary" node, the road-type of [illegible]necting link, and the distance in kilometers. The user specifies [illegible]her to add or delete nodes (or return to earlier option displays [illegible]d the appropriate screen is then displayed (Figures 5c and 5d). [illegible] the user is adding information, the new display (Figure 5c) provides a space to define the data for a new link from the "primary" node to another node. When the user signals the computer by pressing the "Send" key on the keyboard, MOVANAID reads in the data, checks for formatting and other clerical errors, and displays the newly modified "current linkage data" for the same primary node. The user may continue to add or delete information or return to earlier option displays.

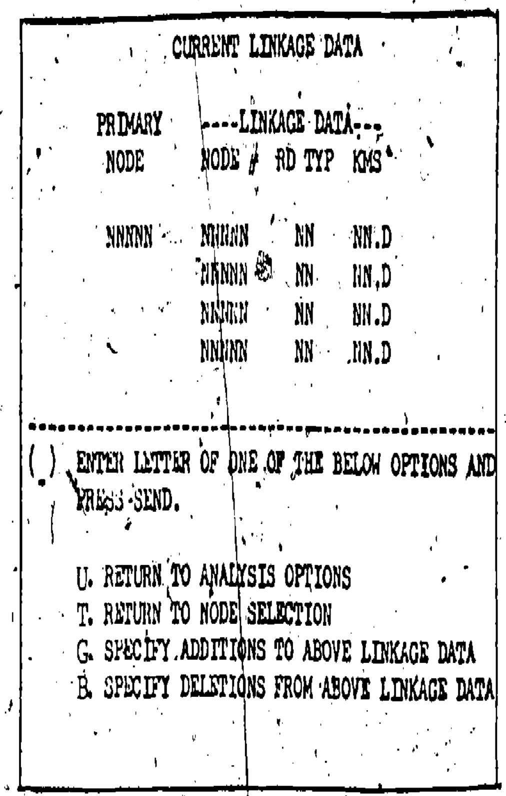
When the user selects the deletion option, the current linkage data for the primary node again appears. The user is able selectively to eliminate a link from the primary node to one of the other nodes, or to completely eliminate a node from the network. An error in the addition of linkage data would be relatively simple for the user to correct; it would only be necessary to delete any link that was mistakenly added. However, restoring a link or node that was mistakenly deleted would require the user to remember or recreate all related linkage data. It is always possible to restore the network to its original form by choosing 6 on the Analysis Options display. However, using this option to correct an inadvertent deletion would result in the elimination of all other changes the user might have made while working on a particular problem. Therefore a user request for deletion of a link or node is followed by a display which describes the requested change and asks the user to verify that this is desired (Figure 5e).

The length of time consumed in the computation of a minimum path or minimax assignment problem is a function of the number of nodes in the network to be considered. If a user is interested in reducing the time for MOVANAID to operate, or is interested in considering a restricted area of the network for reasons related to the tactical problem, then the user may choose one of two "subnetwork" options on the Analysis Options display; MOVANAID will then consider only that subnetwork during problem solution. The first of these options allows the user to define a subnetwork in terms of "map sectors." Choosing this option leads to a display of the current map-sector subnetwork (if one



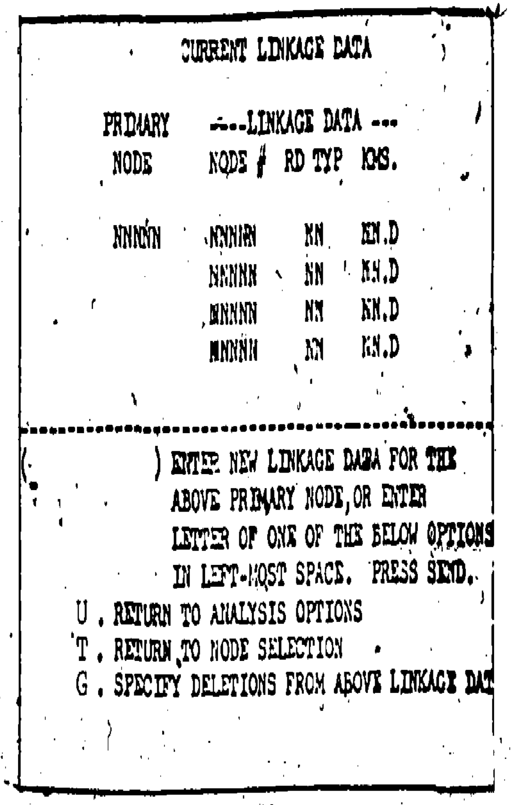
```
NODE SELECTION FOR NETWORK CHANGES

-----------------------------------------------
( ) ENTER NODE NUMBER TO BE CONSIDERED, OR
    ENTER LETTER OF ONE OF THE BELOW OPTIONS
    IN LEFT-MOST SPACE.  PRESS SEND.

    U. RETURN TO ANALYSIS OPTIONS
```

5(a)



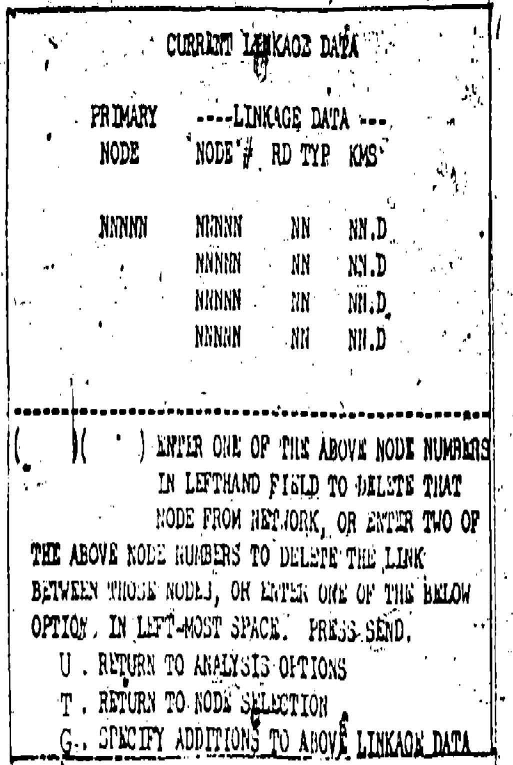
```
              CURRENT LINKAGE DATA

   PRIMARY    ----LINKAGE DATA----
    NODE      NODE #  RD TYP  KMS

   NNNNN      NNNNN     NN    NN.D
              NNNNN     NN    NN.D
              NNNNN     NN    NN.D
              NNNNN     NN    NN.D

-----------------------------------------------
( ) ENTER LETTER OF ONE OF THE BELOW OPTIONS AND
    PRESS SEND.

    U. RETURN TO ANALYSIS OPTIONS
    T. RETURN TO NODE SELECTION
    G. SPECIFY ADDITIONS TO ABOVE LINKAGE DATA
    B. SPECIFY DELETIONS FROM ABOVE LINKAGE DATA
```

5(b)



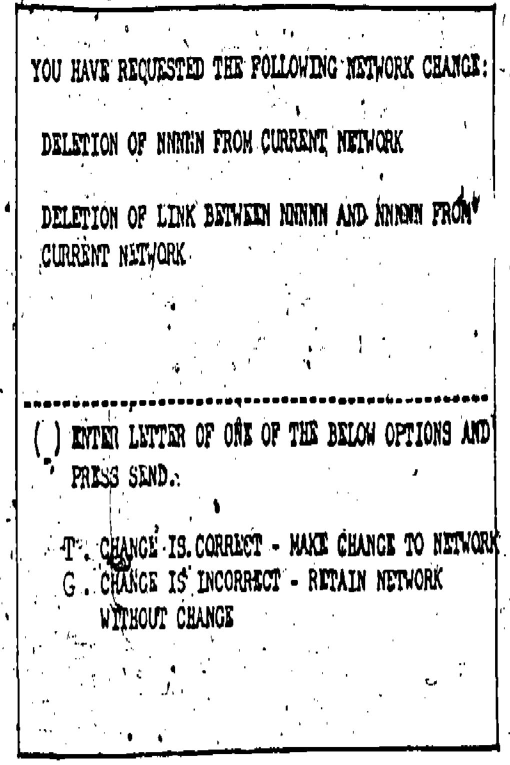
```
              CURRENT LINKAGE DATA

   PRIMARY    ----LINKAGE DATA ---
    NODE      NODE #  RD TYP  KMS.

   NNNNN      NNNNN     NN    NN.D
              NNNNN     NN    NN.D
              NNNNN     NN    NN.D
              NNNNN     NN    NN.D

-----------------------------------------------
(           ) ENTER NEW LINKAGE DATA FOR THE
              ABOVE PRIMARY NODE, OR ENTER
              LETTER OF ONE OF THE BELOW OPTIONS
              IN LEFT-MOST SPACE.  PRESS SEND.
    U . RETURN TO ANALYSIS OPTIONS
    T . RETURN TO NODE SELECTION
    G . SPECIFY DELETIONS FROM ABOVE LINKAGE DAT
```

5(c)



```
              CURRENT LINKAGE DATA

   PRIMARY    ----LINKAGE DATA ---
    NODE      NODE #  RD TYP  KMS

   NNNNN      NNNNN     NN    NN.D
              NNNNN     NN    NN.D
              NNNNN     NN    NN.D
              NNNNN     NN    NN.D

-----------------------------------------------
(   )(   ) ENTER ONE OF THE ABOVE NODE NUMBERS
           IN LEFTHAND FIELD TO DELETE THAT
           NODE FROM NETWORK, OR ENTER TWO OF
THE ABOVE NODE NUMBERS TO DELETE THE LINK
BETWEEN THOSE NODES, OR ENTER ONE OF THE BELOW
OPTIONS IN LEFT-MOST SPACE.  PRESS SEND.
   U . RETURN TO ANALYSIS OPTIONS
   T . RETURN TO NODE SELECTION
   G . SPECIFY ADDITIONS TO ABOVE LINKAGE DATA
```

5(d)



```
YOU HAVE REQUESTED THE FOLLOWING NETWORK CHANGE:

 DELETION OF NNNNN FROM CURRENT NETWORK

 DELETION OF LINK BETWEEN NNNNN AND NNNNN FROM
 CURRENT NETWORK

-----------------------------------------------
( ) ENTER LETTER OF ONE OF THE BELOW OPTIONS AND
    PRESS SEND.

    T . CHANGE IS CORRECT - MAKE CHANGE TO NETWORK
    G . CHANGE IS INCORRECT - RETAIN NETWORK
        WITHOUT CHANGE
```

5(e)

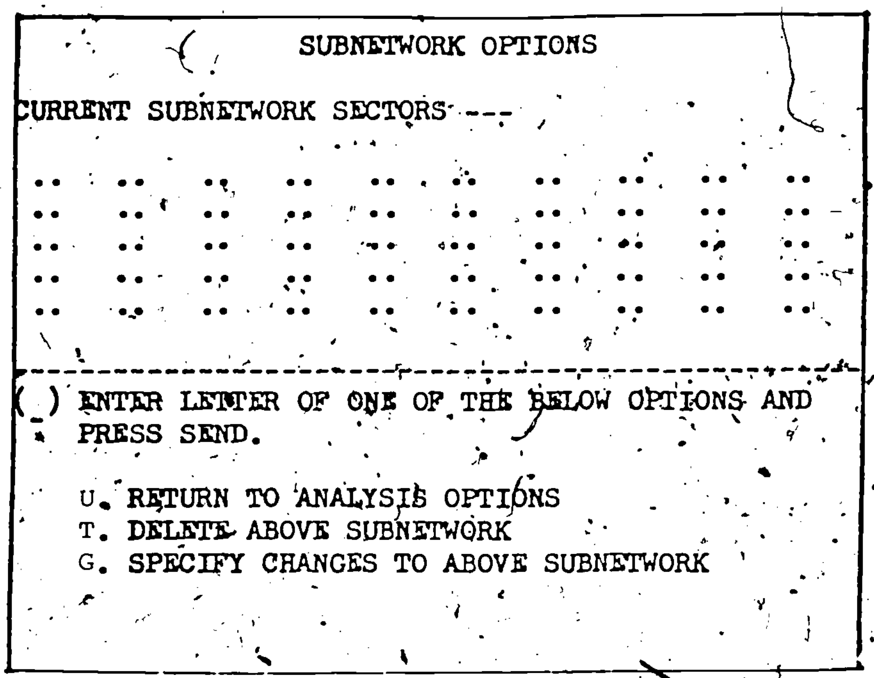
Figure 5. Network change displays.

exists) and allows the user to delete that subnetwork, to specify changes to the subnetwork, or to return to the analysis options (Figure 6a). If the user chooses to change the subnetwork, MOVANAID again displays the current subnetwork information and requests the user to add or delete a map sector from that subnetwork, or to return to an earlier options display (Figure 6b). Choosing to add or delete a map sector results in a new presentation of this display with the newly revised list.

The second subnetwork option a user may select from the Analysis Options is referred to as a "swath" option. If there is no subnetwork currently defined in terms of map sectors and if the user chooses the swath option, MOVANAID presents a display of the current swath (if any exists) and provides the user with options to delete the swath, make changes to the swath, or return to the Analysis Options (Figure 7a). A swath is defined in terms of its width and up to ten turning points. Depending on the choice indicated by the user, MOVANAID presents a display for swath width definition (or redefinition) or for turning point addition or deletion (Figures 7b, 7c).

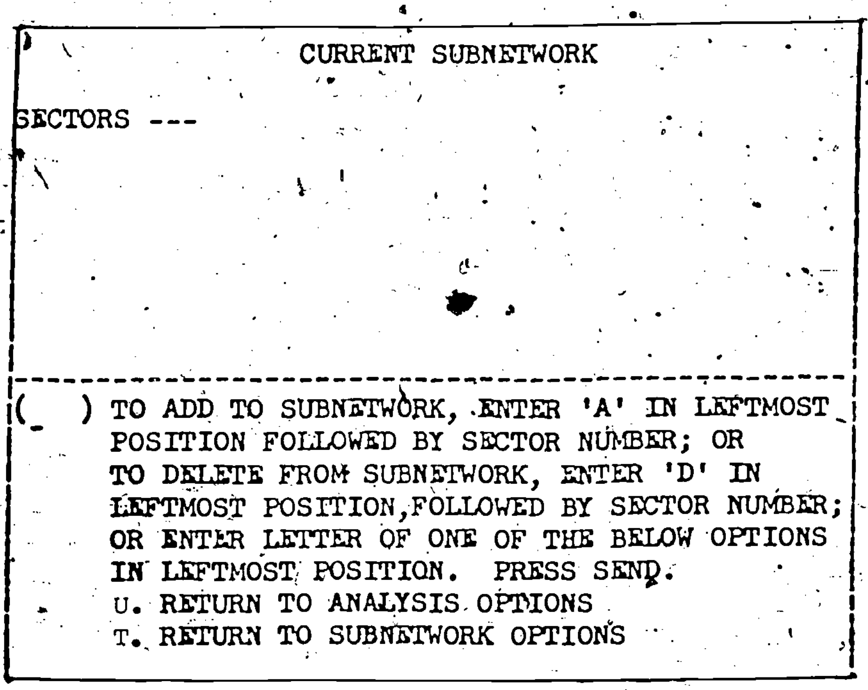
When ready to define the movement problem to be solved, the user chooses the option from the Analysis Options, and MOVANAID presents a display showing the current problem definition (if any exists) and providing additional options for problem definition (Figure 8a). The user may choose to delete the current definition entirely, or to add or delete single lines of data. A problem is specified by indicating a unit starting position, unit destination, unit type, and the conditions of movement for up to ten origin/destination pairs. The starting point and destination must be nodes in the current network; the "unit type" and "movement conditions" are codes referring to data available to the user as hard-copy tables (see Tables 1 and 2). If the user enters more than one line of data through successive uses of the addition option (Figure 8b), then MOVANAID will solve not only for the minimum route from each origin to each destination, but may also solve the problem of the optimal assignment of units to destinations.

When the user has modified the network and defined the problem satisfactorily, the user may request MOVANAID to provide the solution by choosing Option 5 from the Analysis Options. A display is presented giving a review of any actions taken (e.g., the number of network deletions) and of the problem definition; the user has the option to continue to the solution or to return to modify some aspect of the problem. If the user chooses to continue, MOVANAID begins the computations for the problem solution, and a display informs the user that computation is in progress (Figure 9a). Once this step is taken, the user cannot return to Analysis Options until all solutions have been obtained.



```
                    SUBNETWORK OPTIONS
CURRENT SUBNETWORK SECTORS ---

..  ..  ..  ..  ..  ..  ..  ..  ..  ..
..  ..  ..  ..  ..  ..  ..  ..  ..  ..
..  ..  ..  ..  ..  ..  ..  ..  ..  ..
..  ..  ..  ..  ..  ..  ..  ..  ..  ..
..  ..  ..  ..  ..  ..  ..  ..  ..  ..

-------------------------------------------------
( ) ENTER LETTER OF ONE OF THE BELOW OPTIONS AND
    PRESS SEND.

    U. RETURN TO ANALYSIS OPTIONS
    T. DELETE ABOVE SUBNETWORK
    G. SPECIFY CHANGES TO ABOVE SUBNETWORK
```

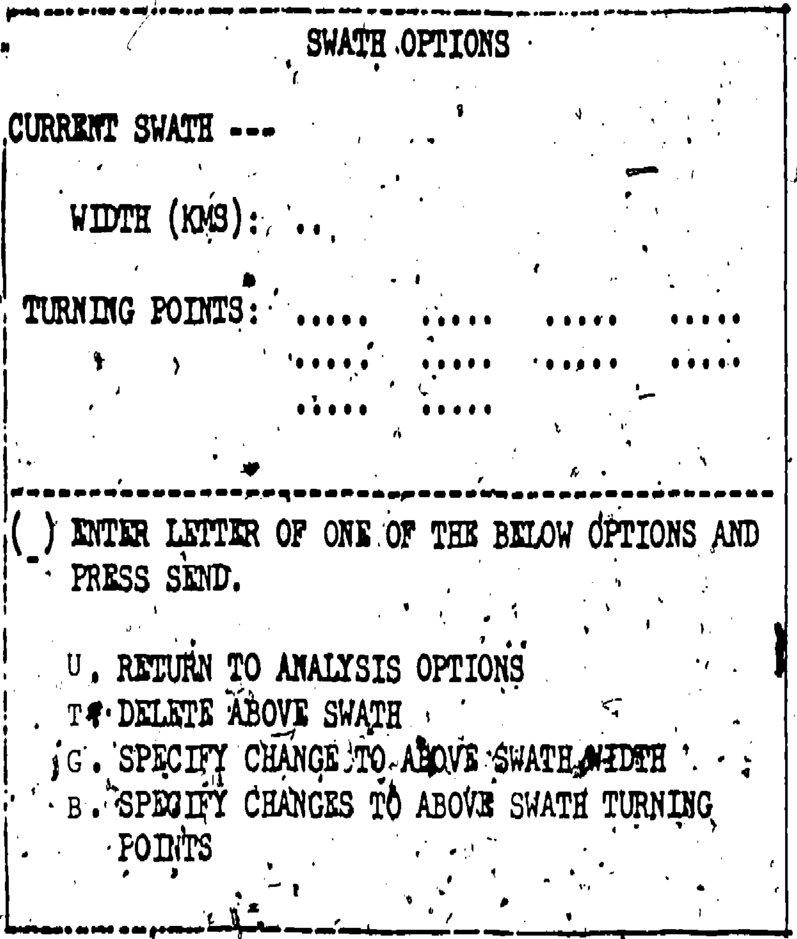
6(a)



```
                  CURRENT SUBNETWORK

SECTORS ---




-------------------------------------------------
(   ) TO ADD TO SUBNETWORK, ENTER 'A' IN LEFTMOST
      POSITION FOLLOWED BY SECTOR NUMBER; OR
      TO DELETE FROM SUBNETWORK, ENTER 'D' IN
      LEFTMOST POSITION,FOLLOWED BY SECTOR NUMBER;
      OR ENTER LETTER OF ONE OF THE BELOW OPTIONS
      IN LEFTMOST POSITION.  PRESS SEND.
      U. RETURN TO ANALYSIS OPTIONS
      T. RETURN TO SUBNETWORK OPTIONS
```

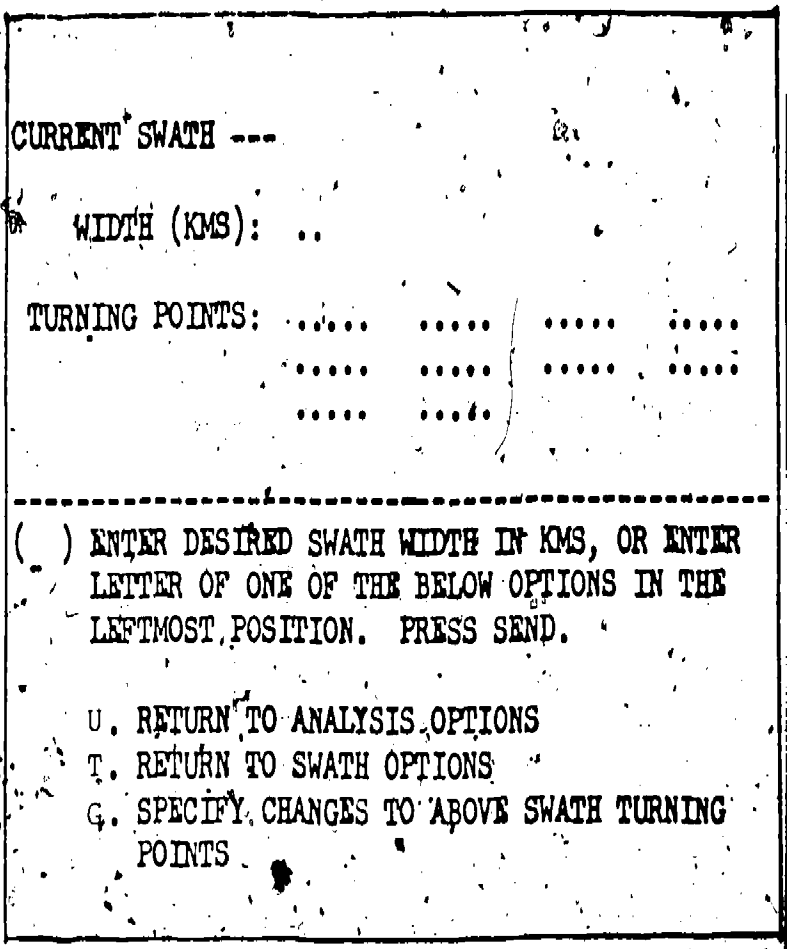
6(b)

Figure 6. Subnetwork options displays.



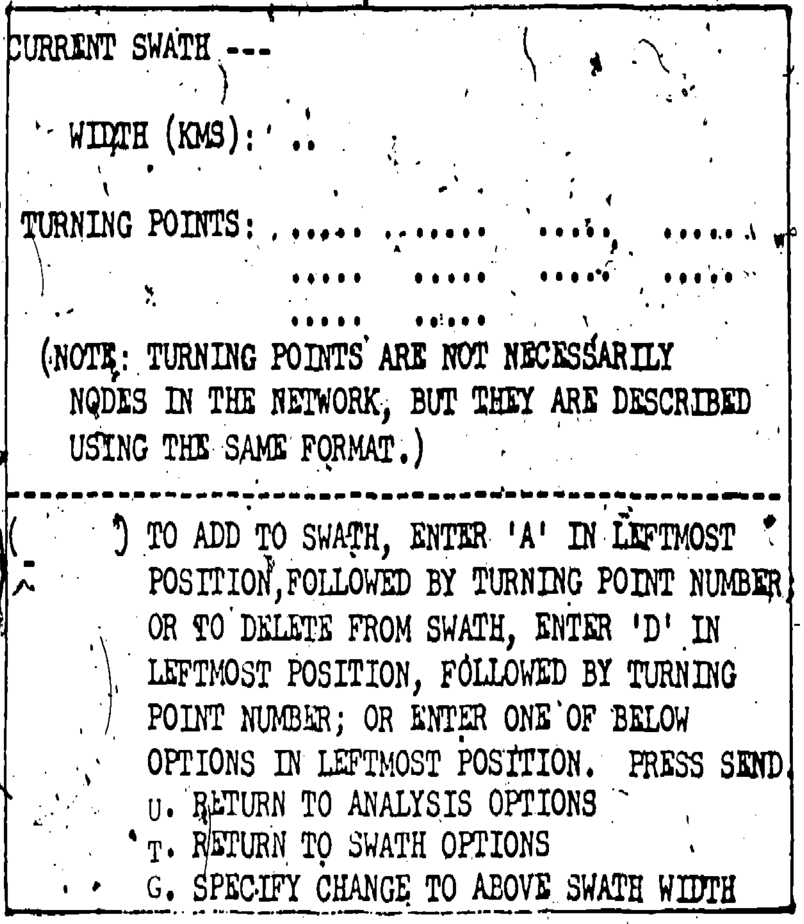
```
                    SWATH OPTIONS

CURRENT SWATH ---

   WIDTH (KMS): ..

 TURNING POINTS:  .....   .....   .....   .....
                  .....   .....   .....   .....
                  .....   .....

------------------------------------------------
( ) ENTER LETTER OF ONE OF THE BELOW OPTIONS AND
    PRESS SEND.

    U. RETURN TO ANALYSIS OPTIONS
    T. DELETE ABOVE SWATH
    G. SPECIFY CHANGE TO ABOVE SWATH WIDTH
    B. SPECIFY CHANGES TO ABOVE SWATH TURNING
       POINTS
```

7(a)



```
CURRENT SWATH ---

   WIDTH (KMS): ..

 TURNING POINTS: .....   .....   .....   .....
                 .....   .....   .....   .....
                 .....   .....

-------------------------------------------------
( ) ENTER DESIRED SWATH WIDTH IN KMS, OR ENTER
    LETTER OF ONE OF THE BELOW OPTIONS IN THE
    LEFTMOST POSITION.  PRESS SEND.

    U. RETURN TO ANALYSIS OPTIONS
    T. RETURN TO SWATH OPTIONS
    G. SPECIFY CHANGES TO ABOVE SWATH TURNING
       POINTS
```

7(b)



```
CURRENT SWATH ---

   WIDTH (KMS): ..

 TURNING POINTS: .....  .....  .....  .....
                 .....  .....  .....  .....
                 .....  .....
  (NOTE: TURNING POINTS ARE NOT NECESSARILY
   NODES IN THE NETWORK, BUT THEY ARE DESCRIBED
   USING THE SAME FORMAT.)
-------------------------------------------------
(      ) TO ADD TO SWATH, ENTER 'A' IN LEFTMOST
         POSITION, FOLLOWED BY TURNING POINT NUMBER;
         OR TO DELETE FROM SWATH, ENTER 'D' IN
         LEFTMOST POSITION, FOLLOWED BY TURNING
         POINT NUMBER; OR ENTER ONE OF BELOW
         OPTIONS IN LEFTMOST POSITION.  PRESS SEND.
         U. RETURN TO ANALYSIS OPTIONS
         T. RETURN TO SWATH OPTIONS
         G. SPECIFY CHANGE TO ABOVE SWATH WIDTH
```

7(c)

7. Swath options displays.

Table 1

Unit Types

MOVANAID unit code	Unit identification	Number of vehicles	Column length (open) 32 KPH	Column length (open) 12 KPH	Closing time (32 KPH)
01	Mtz Rfl Div	2334	265.2	157.0	6 hr 54 min
02	Tank Div	2028	232.8	139.8	6 hr 4 min
03	Mtz Rgt	369	42.9	26.0	1 hr 7 min
04	Mdm Tk Rgt	250	30.3	19.3	47 min
05	Mtz Bn	67	6.6	3.6	10 min
06	Mdm Tk Bn	51	5.9	3.0	9 min
07	Mtz Div HHC	35	3.3	1.8	5 min
08	Div Arty	427	44.4	23.7	1 hr 9 min
09	Recon Bn	72	7.4	4.0	12 min
10	Engr Bn	114	11.7	6.3	18 min
11	Sig Bn	38	3.5	1.9	5 min
12	Cml Co	14	1.2	.7	2 min
13	Intel Co	17	1.3	.8	2 min
14	Div Svc Elem	304	31.6	16.9	49 min
15	Amph Tk Plc	5	.2	.1	1 min
16	Regt Arty	59	5.7	3.1	9 min

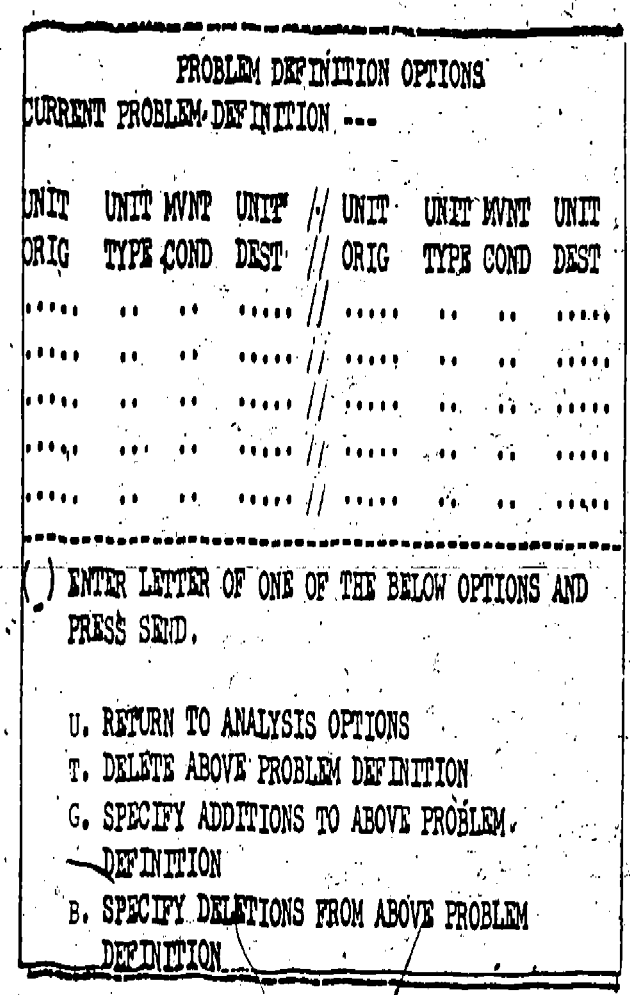
Note: Column length computations are based on the following assumptions:

1. Average vehicle length = 5 meters.
2. Vehicle distance, column gap as stated in FM 30-102 (Revised).
3. Divisions move in 16 battalions separated by 2 km.
4. Rgts. move in 4 battalions separated by 2 km.
5. Travel is in an "open column" mode.
6. At cross-country speeds, Bn's still maintain 2 km gap.

Table 2

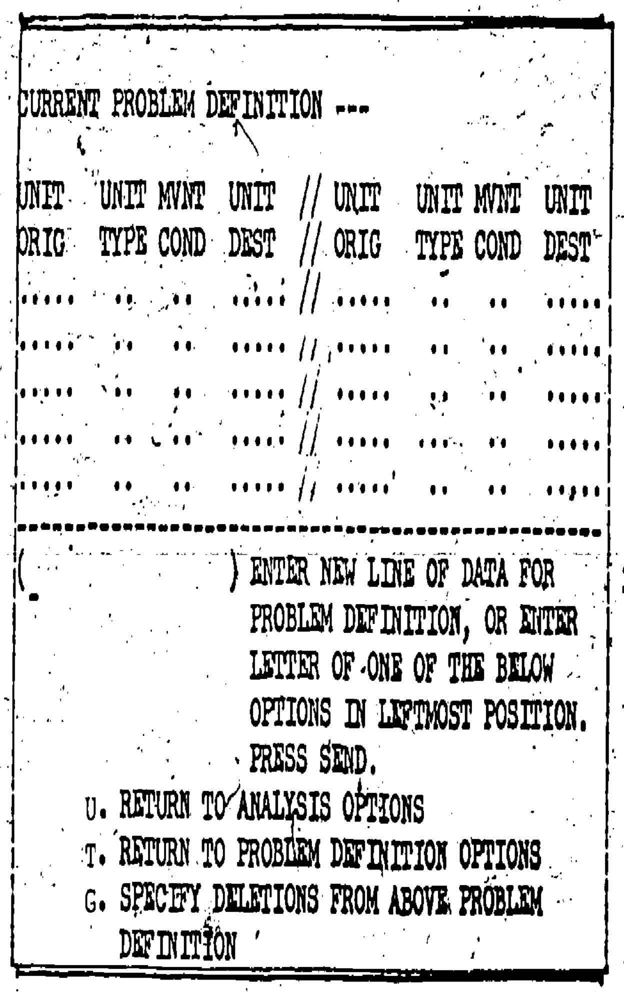
Travel Speeds Assumed for MOVANAID Demonstration (KMPH)

	Movement condition											
	Foot march				Tracked vehicles				Trucks			
	Day		Night		Day		Night		Day		Night	
	Dry	Wet	Dry	Wet	Dry	Wet	Dry	Wet	Dry	Wet	Dry	Wet
Type of Arc	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
Road type 1												
11. Country	4.0	3.0	3.0	2.0	30.	30.	15.	15.	35.	35.	28.	25.
12. City	4.0	3.0	3.0	2.0	30.	30.	15.	15.	35.	35.	28.	25.
Road type 2												
21. Country	4.0	3.0	3.0	2.0	30.	30.	15.	15.	32.	28.	24.	20.
22. City	4.0	3.0	3.0	2.0	15.	15.	7.5	7.5	16.	14.	12.	10.
Road type 3												
31. Country	4.0	3.0	3.0	2.0	25.	25.	13.	13.	25.	20.	18.	17.
32. City	4.0	3.0	3.0	2.0	12.5	12.5	6.5	6.5	12.5	10.	9.	8.5
40. Artificial	999.9	999.9	999.9	999.9	999.9	999.9	999.9	999.9	999.9	999.9	999.9	999.9
51. Easy cross-country	2.0	1.0	1.0	0.5	20.	20.	10.	10.	15.	12.	10.	8.
52. Medium cross-country	1.0	0.6	0.5	0.3	15.	15.	8.	8.	12.	9.	8.	6.
53. Difficult cross-country	0.6	0.3	0.1	0.1	10.	10.	5.	5.	9.	5.	6.	4.



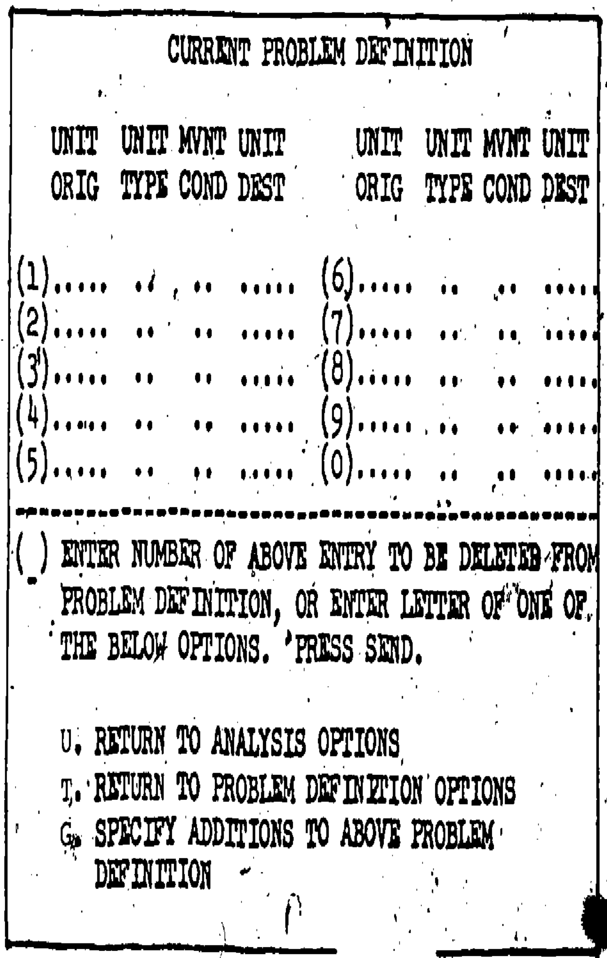
```
          PROBLEM DEFINITION OPTIONS
CURRENT PROBLEM DEFINITION ---

UNIT  UNIT MVNT UNIT  // UNIT  UNIT MVNT UNIT
ORIG  TYPE COND DEST  // ORIG  TYPE COND DEST
.....  ..   ..  ..... // .....  ..   ..  .....
.....  ..   ..  ..... // .....  ..   ..  .....
.....  ..   ..  ..... // .....  ..   ..  .....
.....  ..   ..  ..... // .....  ..   ..  .....
.....  ..   ..  ..... // .....  ..   ..  .....
---------------------------------------------
( ) ENTER LETTER OF ONE OF THE BELOW OPTIONS AND
    PRESS SEND.

    U. RETURN TO ANALYSIS OPTIONS
    T. DELETE ABOVE PROBLEM DEFINITION
    G. SPECIFY ADDITIONS TO ABOVE PROBLEM
       DEFINITION
    B. SPECIFY DELETIONS FROM ABOVE PROBLEM
       DEFINITION
```

8(a)



```
CURRENT PROBLEM DEFINITION ---

UNIT  UNIT MVNT UNIT  // UNIT  UNIT MVNT UNIT
ORIG  TYPE COND DEST  // ORIG  TYPE COND DEST
.....  ..   ..  ..... // .....  ..   ..  .....
.....  ..   ..  ..... // .....  ..   ..  .....
.....  ..   ..  ..... // .....  ..   ..  .....
.....  ..   ..  ..... // .....  ..   ..  .....
.....  ..   ..  ..... // .....  ..   ..  .....
---------------------------------------------
(                  ) ENTER NEW LINE OF DATA FOR
                     PROBLEM DEFINITION, OR ENTER
                     LETTER OF ONE OF THE BELOW
                     OPTIONS IN LEFTMOST POSITION.
                     PRESS SEND.
    U. RETURN TO ANALYSIS OPTIONS
    T. RETURN TO PROBLEM DEFINITION OPTIONS
    G. SPECIFY DELETIONS FROM ABOVE PROBLEM
       DEFINITION
```

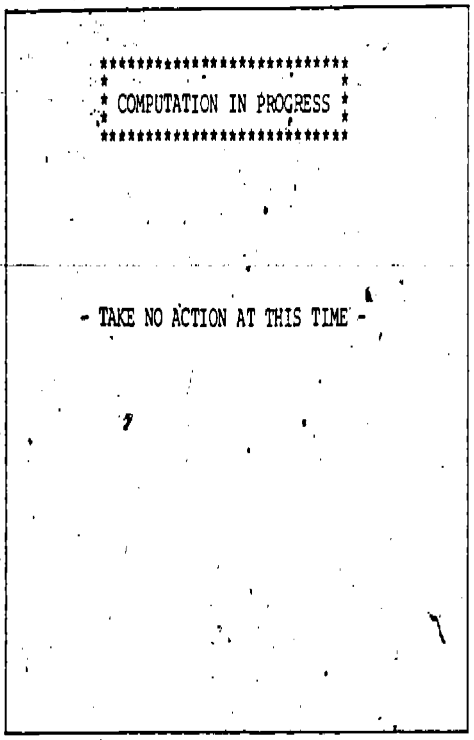
8(b)



```
          CURRENT PROBLEM DEFINITION

   UNIT  UNIT MVNT UNIT      UNIT  UNIT MVNT UNIT
   ORIG  TYPE COND DEST      ORIG  TYPE COND DEST

(1).....  ..   ..  .....  (6).....  ..   ..  .....
(2).....  ..   ..  .....  (7).....  ..   ..  .....
(3).....  ..   ..  .....  (8).....  ..   ..  .....
(4).....  ..   ..  .....  (9).....  ..   ..  .....
(5).....  ..   ..  .....  (0).....  ..   ..  .....
--------------------------------------------------
( ) ENTER NUMBER OF ABOVE ENTRY TO BE DELETED FROM
    PROBLEM DEFINITION, OR ENTER LETTER OF ONE OF
    THE BELOW OPTIONS. PRESS SEND.

    U. RETURN TO ANALYSIS OPTIONS
    T. RETURN TO PROBLEM DEFINITION OPTIONS
    G. SPECIFY ADDITIONS TO ABOVE PROBLEM
       DEFINITION
```

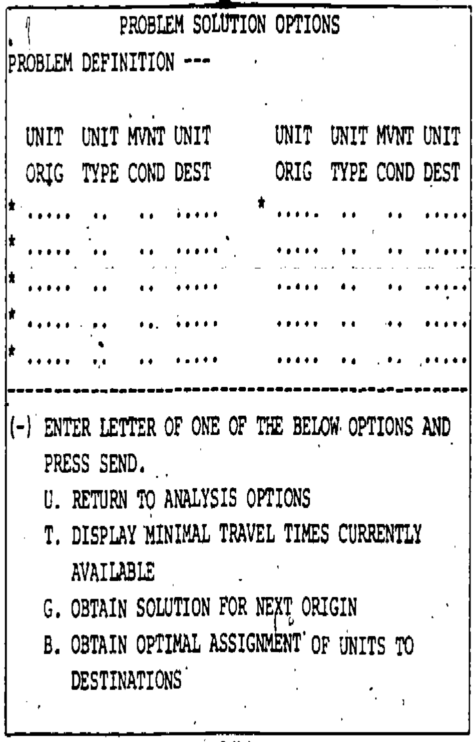
8(c)

8. Problem definition plan.



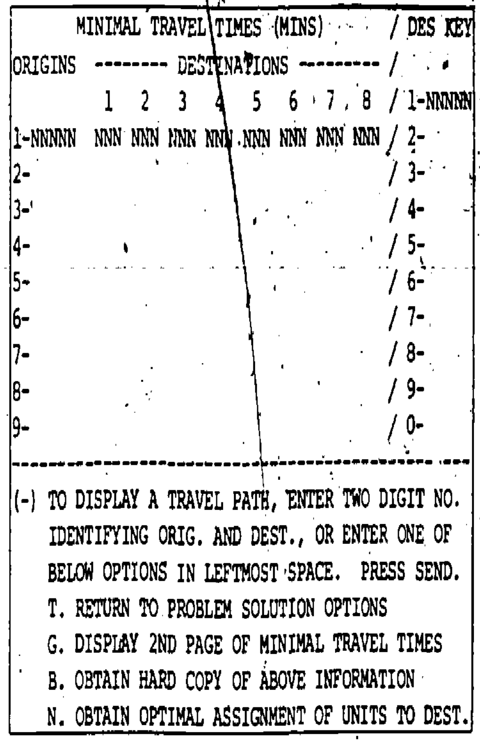
```
**************************
* COMPUTATION IN PROGRESS *
**************************



   - TAKE NO ACTION AT THIS TIME -
```

9(a)



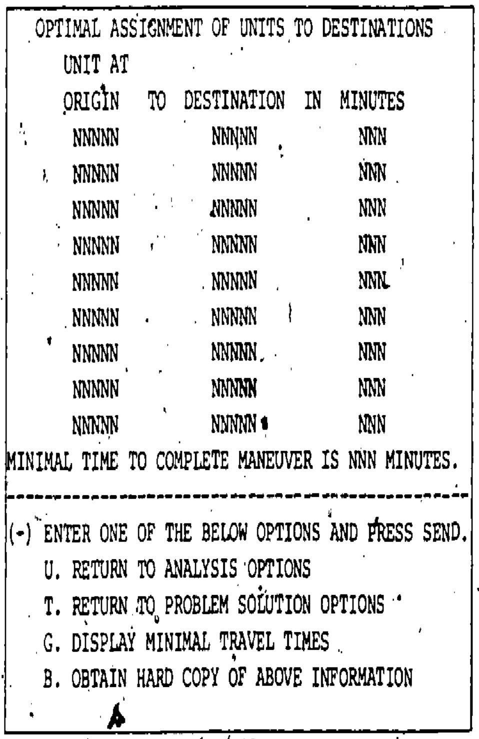
```
              PROBLEM SOLUTION OPTIONS
PROBLEM DEFINITION ---


  UNIT  UNIT MVNT UNIT       UNIT  UNIT MVNT UNIT
  ORIG  TYPE COND DEST       ORIG  TYPE COND DEST
* .....  ..   ..  .....    * .....  ..   ..  .....
* .....  ..   ..  .....      .....  ..   ..  .....
* .....  ..   ..  .....      .....  ..   ..  .....
* .....  ..   ..  .....      .....  ..   ..  .....
* .....  ..   ..  .....      .....  ..   ..  .....
--------------------------------------------------
(-) ENTER LETTER OF ONE OF THE BELOW OPTIONS AND
    PRESS SEND.
    U. RETURN TO ANALYSIS OPTIONS
    T. DISPLAY MINIMAL TRAVEL TIMES CURRENTLY
       AVAILABLE
    G. OBTAIN SOLUTION FOR NEXT ORIGIN
    B. OBTAIN OPTIMAL ASSIGNMENT OF UNITS TO
       DESTINATIONS
```

9(b)



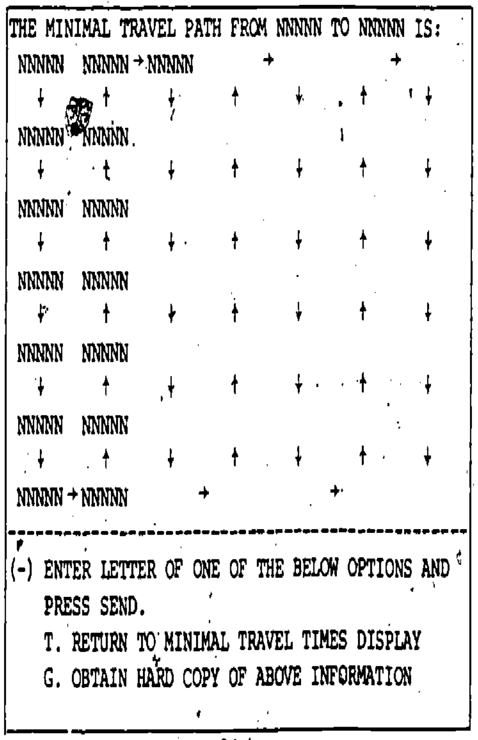
```
       MINIMAL TRAVEL TIMES (MINS)           / DES KEY
ORIGINS  -------- DESTINATIONS ---------     /
          1   2   3   4   5   6   7   8      / 1-NNNNN
1-NNNNN  NNN NNN NNN NNN NNN NNN NNN NNN     / 2-
2-                                           / 3-
3-                                           / 4-
4-                                           / 5-
5-                                           / 6-
6-                                           / 7-
7-                                           / 8-
8-                                           / 9-
9-                                           / 0-
--------------------------------------------------
(-) TO DISPLAY A TRAVEL PATH, ENTER TWO DIGIT NO.
    IDENTIFYING ORIG. AND DEST., OR ENTER ONE OF
    BELOW OPTIONS IN LEFTMOST SPACE.  PRESS SEND.
    T. RETURN TO PROBLEM SOLUTION OPTIONS
    G. DISPLAY 2ND PAGE OF MINIMAL TRAVEL TIMES
    B. OBTAIN HARD COPY OF ABOVE INFORMATION
    N. OBTAIN OPTIMAL ASSIGNMENT OF UNITS TO DEST.
```

9(c)



```
 OPTIMAL ASSIGNMENT OF UNITS TO DESTINATIONS
    UNIT AT
    ORIGIN   TO  DESTINATION  IN  MINUTES
     NNNNN        NNNNN            NNN
     NNNNN        NNNNN            NNN
     NNNNN        NNNNN            NNN
     NNNNN        NNNNN            NNN
     NNNNN        NNNNN            NNN
     NNNNN        NNNNN            NNN
     NNNNN        NNNNN            NNN
     NNNNN        NNNNN            NNN
     NNNNN        NNNNN            NNN
MINIMAL TIME TO COMPLETE MANEUVER IS NNN MINUTES.
-------------------------------------------------
(-) ENTER ONE OF THE BELOW OPTIONS AND PRESS SEND.
    U. RETURN TO ANALYSIS OPTIONS
    T. RETURN TO PROBLEM SOLUTION OPTIONS
    G. DISPLAY MINIMAL TRAVEL TIMES
    B. OBTAIN HARD COPY OF ABOVE INFORMATION
```

9(d)



```
THE MINIMAL TRAVEL PATH FROM NNNNN TO NNNNN IS:
 NNNNN  NNNNN→NNNNN        →          →
   ↓     ↑      ↓     ↑     ↓    ↑     ↓
 NNNNN  NNNNN
   ↓     ↑      ↓     ↑     ↓    ↑     ↓
 NNNNN  NNNNN
   ↓     ↑      ↓     ↑     ↓    ↑     ↓
 NNNNN  NNNNN
   ↓     ↑      ↓     ↑     ↓    ↑     ↓
 NNNNN  NNNNN
   ↓     ↑      ↓     ↑     ↓    ↑     ↓
 NNNNN  NNNNN
   ↓     ↑      ↓     ↑     ↓    ↑     ↓
 NNNNN→NNNNN     →         →
--------------------------------------------------
(-) ENTER LETTER OF ONE OF THE BELOW OPTIONS AND
    PRESS SEND.
    T. RETURN TO MINIMAL TRAVEL TIMES DISPLAY
    G. OBTAIN HARD COPY OF ABOVE INFORMATION
```

9(e)

Figure 9. Problem solution display.

MOVANAID solves the defined problem one "line" at a time. Thus, the minimal paths from the first unit origin to all unit destinations are obtained first. When the solution is available, a display shows the problem definition with an asterisk beside the first line to indicate that that portion of the problem has been solved (Figure 9b). The user has the option of displaying the minimal travel times MOVANAID has computed, continuing to the next portion of the solution, or, if there was only one origin/destination pair in the problem, returning to Analysis Options. If the user continues to the next line of a problem with multiple origin/destination pairs, he will again have these options when the next set of computations is completed. In addition, if MOVANAID has completed the last line of a problem with more than one line, then the user can request the additional computation of the optimal assignment of units to destinations. Note that "illegal" options (e.g., a return to Analysis Options as a request for optimum assignment when there are portions of the problem still to be shown) are eliminated from the analyst's display.

When the user requests the movement times obtained by MOVANAID, they are displayed in a matrix format (see Figure 9c). From this display, the user has the option to obtain a hard copy of the information (typed on a computer-driven typewriter at the user's work station), to see the "second page" of information (relevant only when there are more than eight origin/destination pairs), to obtain the optimal assignment of units (if all lines of the problem have been solved), or to return to the problem-solution-options display. When the minimal path is requested, MOVANAID will display all nodes (up to 49) on the quickest path from the specified origin to the specified destination (Figure 9d); a typed hard copy can also be produced. When the user requests the optimal assignment solution, the origin/destination pairs are displayed that minimize the total time for a simultaneous maneuver of each of the units to one of the specified destinations (Figure 9e).

<u>Examples of MOVANAID Use</u>

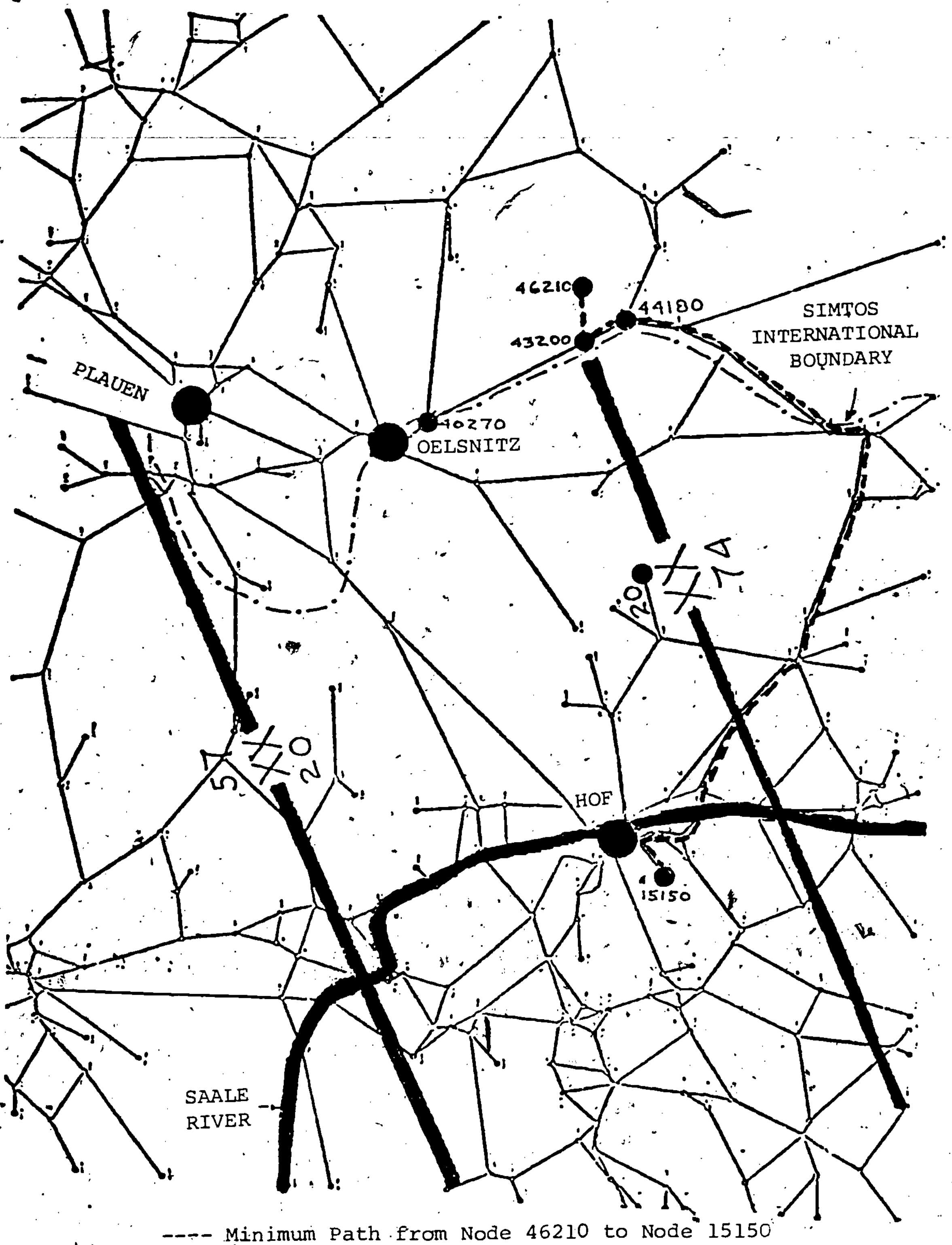
This section contains a series of examples, derived in part from the SIMTOS scenario, that demonstrate the material just discussed. These examples also show how the user could prepare a problem for solution by MOVANAID. The sequence of examples assumes that the user has progressed through the introductory material, has the Analysis Options displayed, and has made no changes to the network. The results shown are identical in content and format to those produced on the CRT. Note that the description of analyst/MOVANAID interactions in the examples below will not detail each individual step of the process; e.g., returning to Analysis Options from some point might actually require a sequence of 2 or 3 inputs but may be described as though it were one step.

The 20th (U.S.) Division G2 section has just received a message giving the probable locations of four Aggressor tank units. One of these is the 34th Medium Tank Regiment, located in the vicinity of UTM (Universal Transverse Mercator) map coordinate, UR0682. The G2 wishes to know the minimum time for this unit to reach the vicinity of Hill 614 (QA0476), a key terrain feature. Consulting the 1:50000 scale map with network overlay, the analyst notes that no node exists at the unit location. The analyst determines the grid location of the tank regiment and labels it 46210 following the method discussed above. The analyst notes the need for a link from this node to a nearby main road, and plans to create node 43200 at the point of intersection. The analyst judges that the arc between the two new nodes (dashed line on Figure 10) is passable at approximately the speed of an easy cross-country route, and measures the distance on the map using a simple map-measuring device.

Example 1. To make the necessary changes in the network, the analyst first chooses Option 1 from the Analysis Options (Figure 4) and then indicates interest in node 46210. MOVANAID responds with an error message, informing the analyst that 46210 is not currently in the network. The analyst notes on the map that the new node, 43200, will fall on a link between 44180 and 40270, and enters 44180 as the node to be considered. MOVANAID displays the linkage information for 44180 (Figure 11). The analyst enters a line of data linking 44180 to 43200, and deletes the link 44180 ↔ 40270. The analyst then returns to node selection, indicates 43200 as the node to be considered, and enters a line of data connecting 43200 with 40270. Note that these last two steps, the deletion of 44180 ↔ 40270 and the addition of 43200 ↔ 40270, are necessary if the analyst wants to place the new node 43200 specifically on the link 44180 ↔ 40270. The analyst must carefully plan any additions or deletions to insure that the MOVANAID network and any solutions based on that network are consistent with the analyst's mental picture of that network.

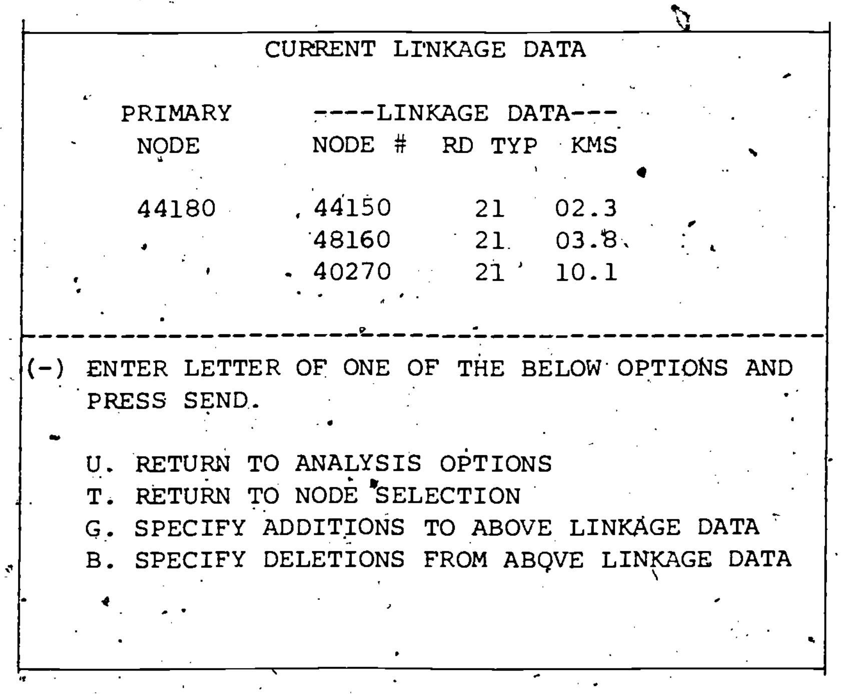
After replacing 44180 ↔ 40270 with 44180 ↔ 43200 ↔ 40270, the analyst indicates interest in working with 43200. The linkage data are displayed and the analyst can proceed to define the 43200 ↔ 46210 link. With the network satisfactorily modified, and with no interest in restricting the analysis to a subnetwork, the analyst returns to Analysis Options and selects Option 4, Problem Definition. Obtaining the Additions display, the analyst enters a line of data: 46210, 04, 05, 15150 (specifying a unit type 04 to move under conditions 05 from 46210 to 15150). Since the analyst is interested in only the one unit, the analyst then returns to Analysis Options and chooses Problem Solution. After reviewing the problem definition, the analyst initiates the computations.

After the computation period, MOVANAID signals the availability of solution output. The analyst requests a display of minimal travel time (Figure 12a) and of the nodes on this minimal path (Figure 12b). Tracing the solution on a map (Figure 10), the analyst is somewhat



---- Minimum Path from Node 46210 to Node 15150

Figure 10. Illustration of example 1.



```
                   CURRENT LINKAGE DATA

   PRIMARY       ----LINKAGE DATA---
    NODE         NODE #  RD TYP  KMS

    44180        44150      21   02.3
                 48160      21   03.8
                 40270      21   10.1

-----------------------------------------------------
(-) ENTER LETTER OF ONE OF THE BELOW OPTIONS AND
    PRESS SEND.

    U. RETURN TO ANALYSIS OPTIONS
    T. RETURN TO NODE SELECTION
    G. SPECIFY ADDITIONS TO ABOVE LINKAGE DATA
    B. SPECIFY DELETIONS FROM ABOVE LINKAGE DATA
```

Figure 11. Linkage data for node 44180.

```
      MINIMAL TRAVEL TIMES (MINS)             / DES KEY
ORIGINS  -------- DESTINATIONS ---------      /
          1   2   3   4   5   6   7   8       / 1-15150
1-46210  153                                  / 2-
2-                                            / 3-
3-                                            / 4-
4-                                            / 5-
5-                                            / 6-
6-                                            / 7-
7-                                            / 8-
8-                                            / 9-
9-                                            / 0-
----------------------------------------------------
(11) TO DISPLAY A TRAVEL PATH, ENTER TWO DIGIT NO.
     IDENTIFYING ORIG. AND DEST., OR ENTER ONE OF
     BELOW OPTIONS IN LEFTMOST SPACE.  PRESS SEND.
     T. RETURN TO PROBLEM SOLUTION OPTIONS
     G. DISPLAY 2ND PAGE OF MINIMAL TRAVEL TIMES
     B. OBTAIN HARD COPY OF ABOVE INFORMATION
     N. OBTAIN OPTIMAL ASSIGNMENT OF UNITS TO DEST
```

12(a)

```
THE MINIMAL TRAVEL PATH FROM 46210 TO 15150 IS:
46210   18140 → 17150         →               →
  ↓       ↑       ↓         ↑       ↓       ↑      ↓
43200   19140   17160
  ↓       ↑       ↓         ↑       ↓       ↑      ↓
44180   23130   17170
  ↓       ↑       ↓         ↑       ↓       ↑      ↓
44150   26101   17172
  ↓       ↑       ↓         ↑       ↓       ↑      ↓
43130   26102   15150
  ↓       ↑       ↓         ↑       ↓       ↑      ↓
39090   30100
  ↓       ↑                 ↑       ↓       ↑      ↓
39071 → 36080           →               →
----------------------------------------------------
(-) ENTER LETTER OF ONE OF THE BELOW OPTIONS AND
    PRESS SEND.
    T. RETURN TO MINIMAL TRAVEL TIMES DISPLAY
    G. OBTAIN HARD COPY OF ABOVE INFORMATION
```

12(b)

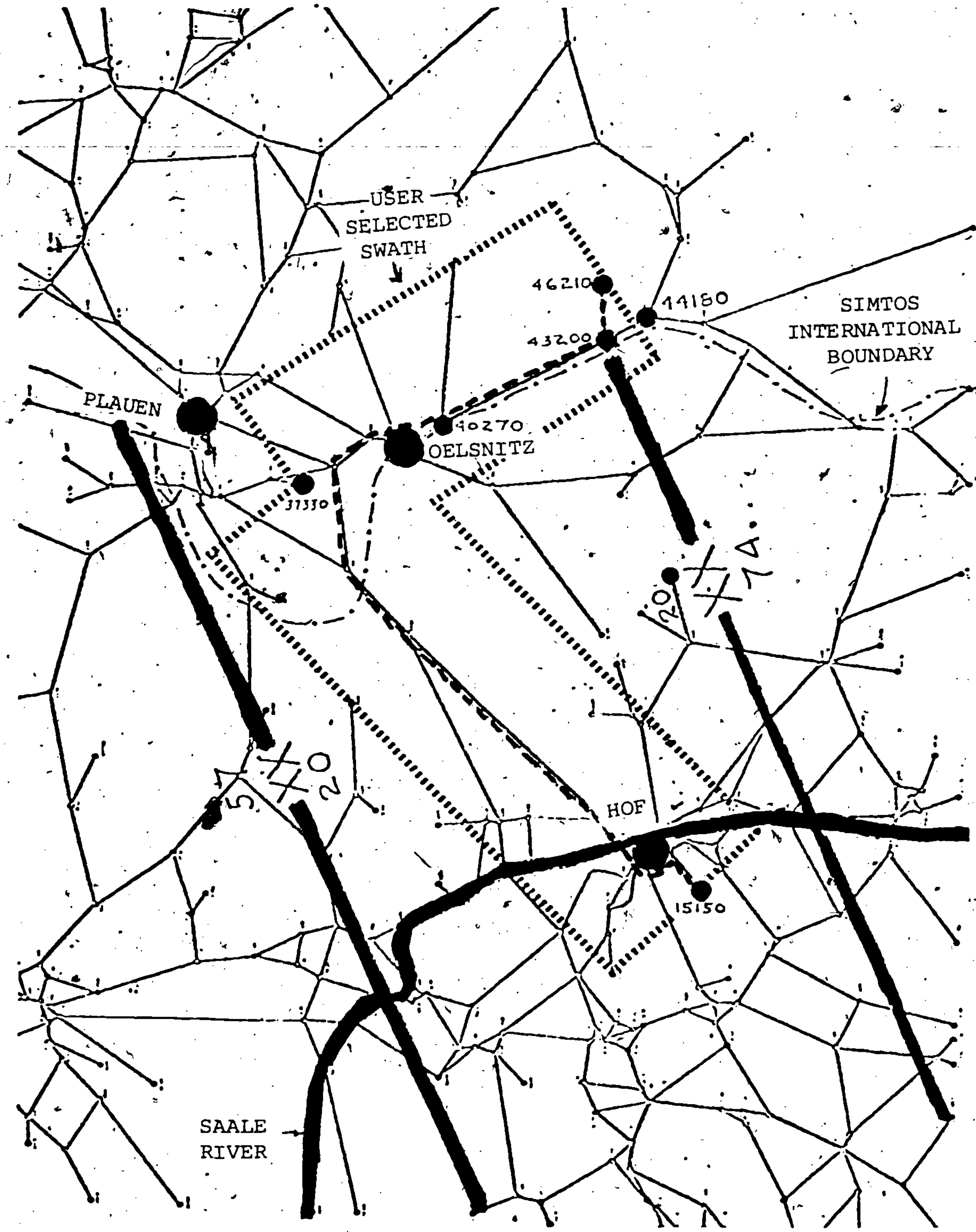
Figure 12. Example 1 solution displays.

dissatisfied with this solution because it would require the Aggressor unit to engage in a "crossing of lines" maneuver that would lead to some delays not anticipated by the MOVANAID algorithms; therefore, the analyst returns to the Analysis Options to reformulate the problem.

Example 2. Looking at the map, the analyst sees what appears to be a reasonable route for the Aggressor unit: a move to the north on the Aggressor side of the border as far as the OELSNITZ region, then a swing toward HOF on one of several roads. By defining an appropriate subnetwork, the analyst can restrict MOVANAID's attention to that route and find the relevant minimal travel time. The analyst chooses the Swath option and defines a swath 10 kilometers wide with turning points 46210, 37330, and 15150 (Figure 13). The network additions and deletions made for example 1 are still appropriate to the new problem, as is the problem definition. Thus, the analyst now requests a problem solution and initiates computation. MOVANAID again notifies the user when computation is complete for the first (and only) line of this problem; the user first looks at the minimal travel time and then at the associated minimal path (Figures 14a and 14b). The analyst notes that the time for this route is only slightly longer than for the previous route and that the new route makes sense given the tactical situation, requests a "hard copy" of the nodes on this route for use in plotting it on the map, and then returns to Analysis Options.

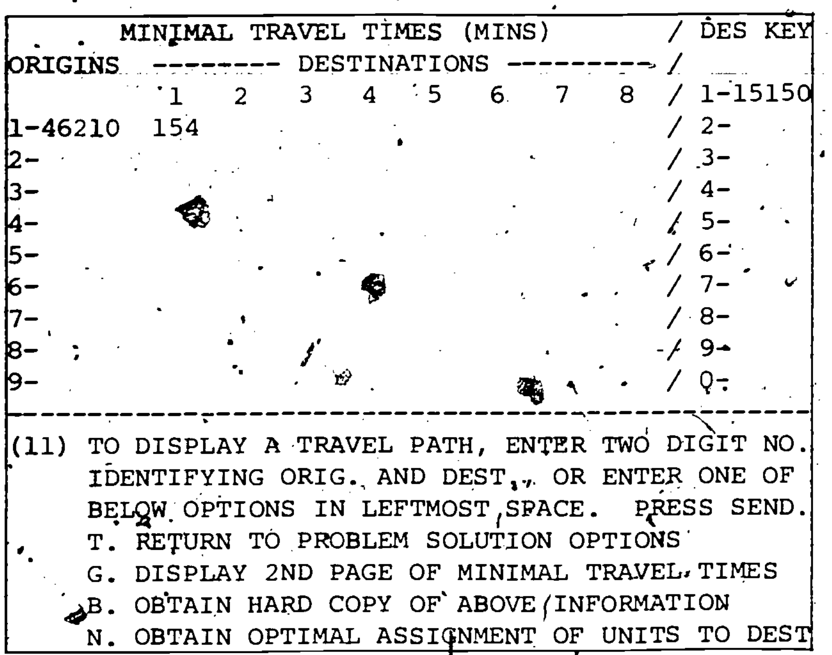
Example 3. In the previous examples, MOVANAID has had its attention restricted to the original network as modified to connect the origin of the unit of interest into the network. The basic network consists only of hard-surface all-weather roads; the solutions produced have not considered the possibility of cross-country travel. The analyst has noted, however, that there appears to be a reasonable cross-country route that might reduce the time from 46210 to 15150. The analyst starts final treatment of the problem by adding a link to the network, 43200 ↔ 32170, and then requests a problem solution. However, the analyst notes that there is still a subnetwork defined by a swath, and returns to delete the swath. The analyst then calls for a solution and initiates computation. In this case, the minimal travel time is 143 minutes (the shorter distance compensating for the slower cross-country travel times for a portion of the route) on the path displayed in Figure 15.

At this point, the analyst knows of three routes from the current location of the unit of interest to the anticipated destination. All routes would take slightly under 2-1/2 hours to travel, given the assumptions about average speed in the speed table and including 47 minutes closing time for the unit when arriving at the destination (see Table 1). No clear-cut advantage can be seen for any of the routes on the basis of travel time alone, and the analyst will be required to consider other aspects of the tactical situation in discussing the probable enemy course of action. However, MOVANAID has provided him with a good estimate of the movement time for the unit in question, and has allowed him to explore several plausible alternative routes in



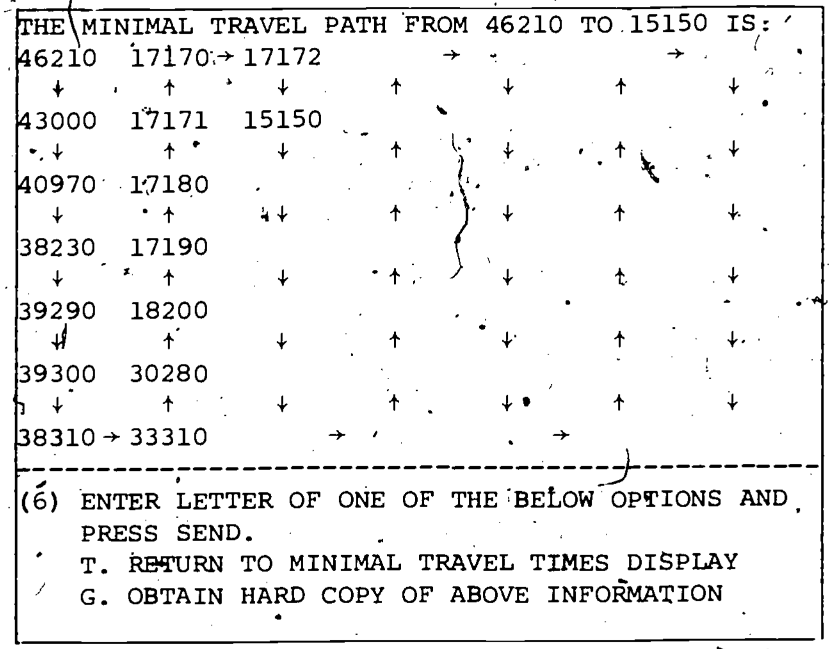
---- Minimum Path from Node 46210 to Node 15150

Figure 13. Illustration of example 2.



```
    MINIMAL TRAVEL TIMES (MINS)                / DES KEY
ORIGINS  -------- DESTINATIONS ---------->   /
          1    2    3    4    5    6    7    8   / 1-15150
1-46210  154                                     / 2-
2-                                               / 3-
3-                                               / 4-
4-                                               / 5-
5-                                               / 6-
6-                                               / 7-
7-                                               / 8-
8-                                               / 9-
9-                                               / 0-
-----------------------------------------------------------
(11) TO DISPLAY A TRAVEL PATH, ENTER TWO DIGIT NO.
     IDENTIFYING ORIG. AND DEST., OR ENTER ONE OF
     BELOW OPTIONS IN LEFTMOST SPACE.  PRESS SEND.
     T. RETURN TO PROBLEM SOLUTION OPTIONS
     G. DISPLAY 2ND PAGE OF MINIMAL TRAVEL TIMES
     B. OBTAIN HARD COPY OF ABOVE INFORMATION
     N. OBTAIN OPTIMAL ASSIGNMENT OF UNITS TO DEST
```

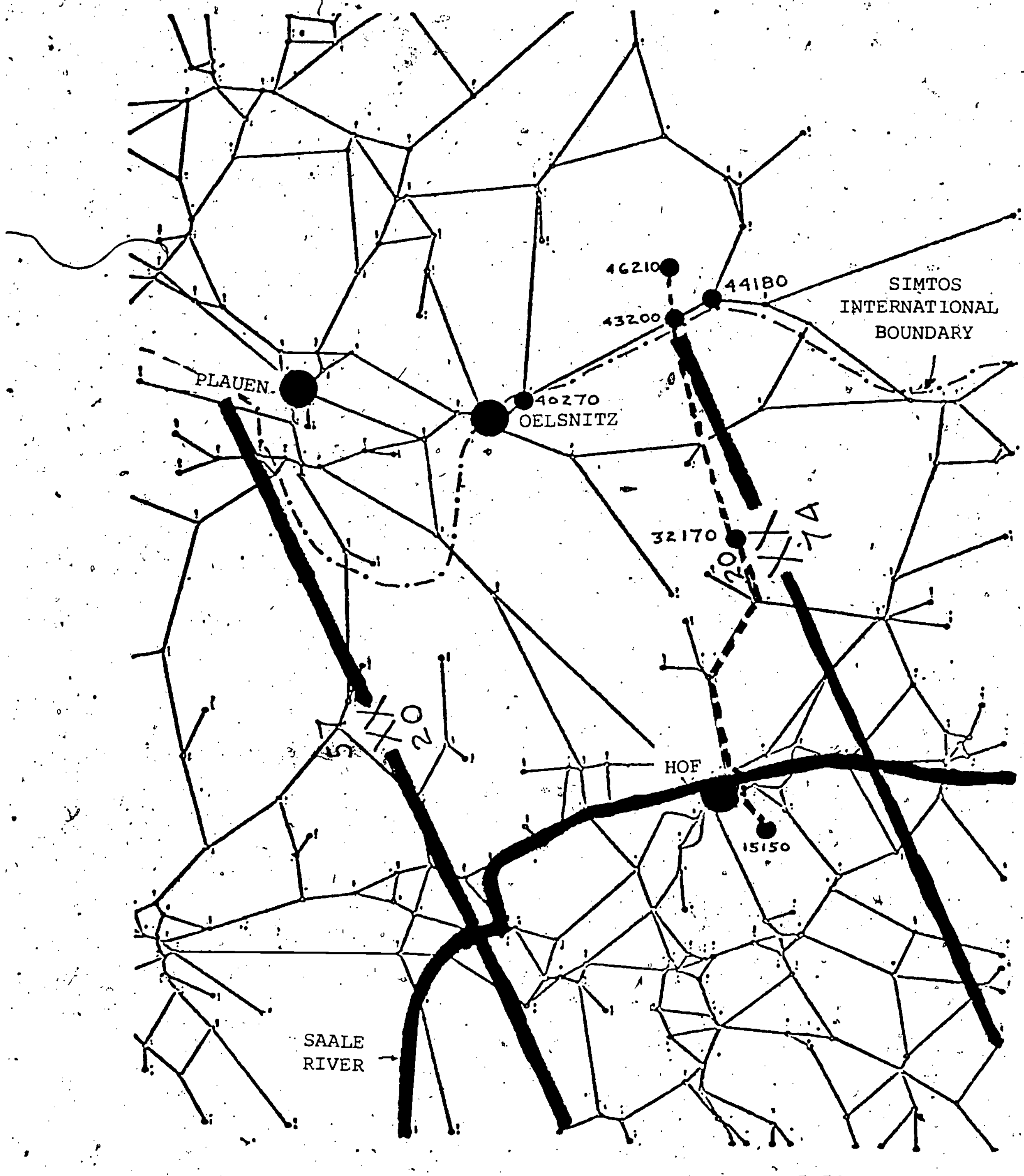
14(a)



```
THE MINIMAL TRAVEL PATH FROM 46210 TO 15150 IS:
46210    17170 → 17172        →              →
  ↓        ↑       ↓      ↑       ↓      ↑      ↓
43000    17171   15150
  ↓        ↑       ↓      ↑       ↓      ↑      ↓
40970    17180
  ↓        ↑       ↓      ↑       ↓      ↑      ↓
38230    17190
  ↓        ↑       ↓      ↑       ↓      ↑      ↓
39290    18200
  ↓        ↑       ↓      ↑       ↓      ↑      ↓
39300    30280
  ↓        ↑       ↓      ↑       ↓      ↑      ↓
38310 →  33310         →             →
-----------------------------------------------------------
(6) ENTER LETTER OF ONE OF THE BELOW OPTIONS AND
    PRESS SEND.
    T. RETURN TO MINIMAL TRAVEL TIMES DISPLAY
    G. OBTAIN HARD COPY OF ABOVE INFORMATION
```

14(b)

Figure 14. Example 2 solution displays.



---- Minimum Path from Node 46210 to Node 15150

Figure 15. Illustration of example 3.

a relatively short time. A great deal of the value of MOVANAID depends on its careful use by an analyst who is able to define a problem in such a way as to obtain solutions consistent with the realities of the tactical situation. While MOVANAID may designate one route as "optimal," the analyst should compare that solution with the analyst's own map reading and, if necessary, create new links and/or subnetworks to obtain a new MOVANAID solution similar to the analyst's own subjective analysis. At that point, the analyst may determine which set of assumptions and results seems most reasonable. The examples above, involving three different attempts to obtain a meaningful solution to a simple problem, should be considered typical of the approach a successful user will take.

CONSTRUCTION OF MOVANAID

Analytic Basis

The movement analysis aid MOVANAID has been discussed in the previous section in "black-box" terms, as seen by the user, and the manner in which users interact with the aid has been described. (We do not discuss in this section the computer implementation of MOVANAID, or give detail about the data upon which the aid operates; these and similar matters will be discussed later, in the section entitled The Construction of MOVANAID: Computer Implementation, and Appendix B, respectively.) There are two basic functions which MOVANAID performs in generating answers to questions posed by users: the computation of the fastest paths from origins to destinations, and the computation of unit assignments to destinations. These computations are performed in MOVANAID by algorithms developed elsewhere.

Computation of Fastest Paths

Several algorithms for finding fastest paths in a network have been discussed in the literature; each requires, as data, travel times for the individual arcs of the network. One such algorithm, which we call "MINPATH," has been included in MOVANAID for fastest-path computations.

Although there are reasonable grounds for attributing the MINPATH algorithm to Dantzig (see Dantzig (1960) and also Dantzig (1963, pp. 363-366)), it must be admitted that the origin of the algorithm is now obscure. There are several similar fastest path algorithms which predate Dantzig's (see, for example, Bellman (1958) and Moore (1957)). Still another algorithm, that of Minty (1957), is particularly interesting. In Minty's algorithm, one first builds a string model of the network, with arc lengths proportional to travel times; then, grasping the network in one hand at the node of destination, one simply stretches the network.

The computation of individual arc travel times is straightforward; one simply uses road type, traveling unit type, weather, and time of day information to find the appropriate rate of travel for each arc in the speed table (Table 2), and divides the arc length by this rate to determine the travel time. Similar remarks apply to the computation of the time necessary for the collapse of a column of maneuvering units which travel in columnar formation. (In its present configuration, the MINPATH portion of MOVANAID adds a column closing time (this being computed on the basis of the user-specified type of maneuvering unit) to the fastest path time to obtain the minimum time within which the last element of a maneuvering column can reach the destination by traveling through the network.) Thus, it suffices for present purposes to assume that travel times for arcs, together with unit closing times, have already been determined.

To illustrate the MINPATH algorithm, we will consider the network N of 11 nodes (A through K) and 19 arcs shown in Figure 16. The number appearing next to each arc in the figure is the travel time for that arc. Let us suppose that node A has been specified by a user as being the node of origin, and that the user wants to know the fastest travel time and associated path from node A to some node of destination, say node K.

The first step is to form the matrix T of arc travel times for the network as shown in Figure 17. The rows and columns of T are labeled with node labels A through K, and the elements of T are the travel times between nodes corresponding to the rows and columns in which the elements appear. Arc travel times between nodes of the network that are not connected by a single arc are taken to be infinity (∞). Included in the figure are path information columns and a travel time row. These will be used in the illustration of the algorithm.

The convention is that no node is self-connected; thus the travel time between any node and itself is always ∞. We could equally well adopt the perhaps more natural convention that each node is self-connected with associated travel time zero. It turns out, however, that one would gain nothing from this, and the number of computations required by the algorithm would be slightly increased. Any ordering of row and/or column labels is permissible; however, if the order in which the rows are labeled is the same as the order for the columns, then T will have the convenient property of being symmetrical with ∞ appearing everywhere on the diagonal. Finally, although the network N is undirected (meaning that travel is permitted in either direction along any arc), it is convenient for our discussion to regard the node labels heading the columns of T as being points where a traveler may be at some stage of a journey, and the row headings as being places where he may go next.

44

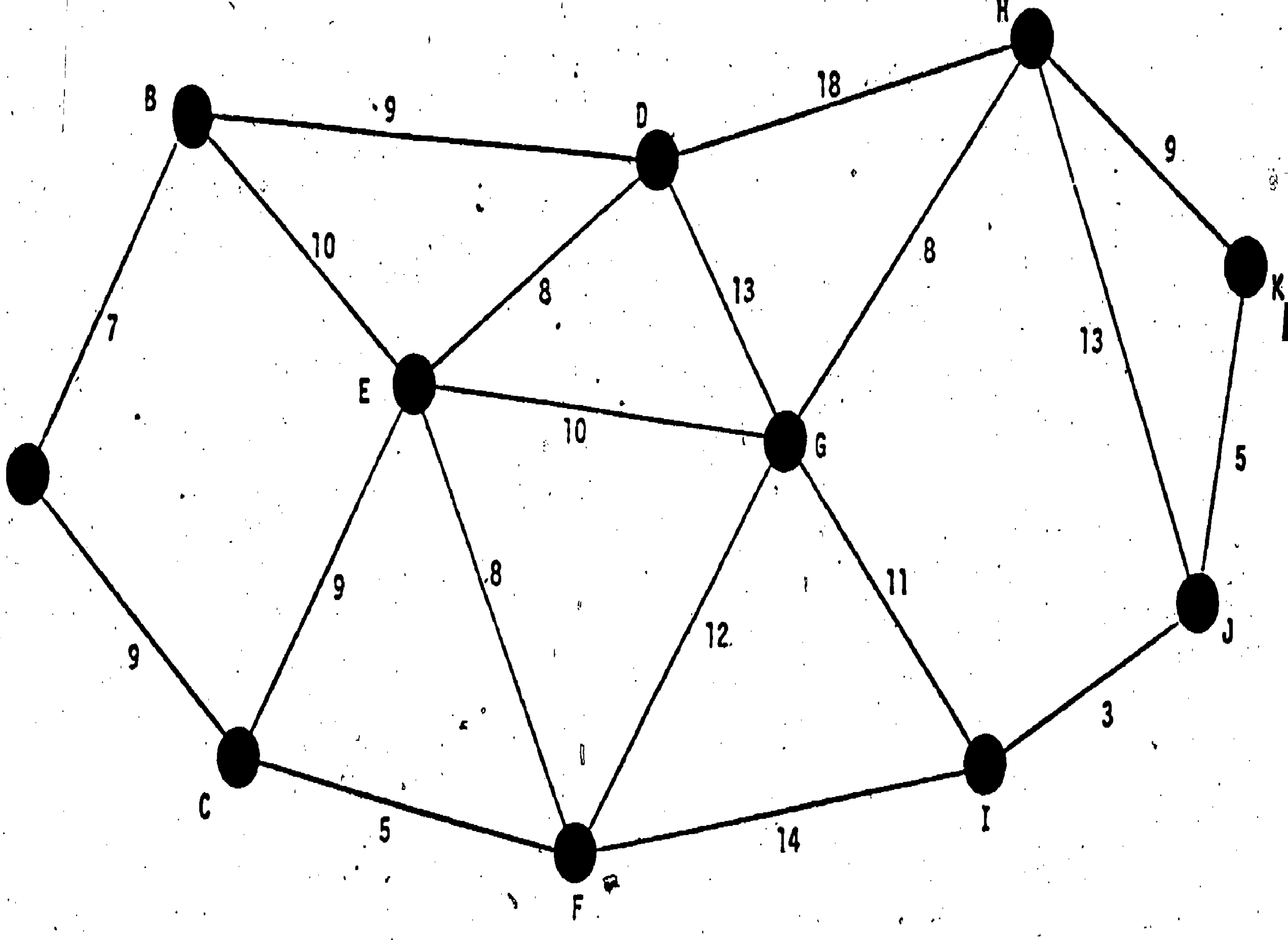


Figure 16. A simple network.

path information columns

travel time row											
	A	B	C	D	E	F	G	H	I	J	K
A	∞	7	9	∞	∞	∞	∞	∞	∞	∞	∞
B	7	∞	∞	9	10	∞	∞	∞	∞	∞	∞
C	9	∞	∞	∞	9	5	∞	∞	∞	∞	∞
D	∞	9	∞	∞	8	∞	13	18	∞	∞	∞
E	∞	10	9	8	∞	8	10	∞	∞	∞	∞
F	∞	∞	5	∞	8	∞	12	∞	14	∞	∞
G	∞	∞	∞	13	10	12	∞	8	11	∞	∞
H	∞	∞	∞	18	∞	∞	8	∞	∞	13	9
I	∞	∞	∞	∞	∞	14	11	∞	∞	3	∞
J	∞	∞	∞	∞	∞	∞	∞	13	3	∞	5
K	∞	∞	∞	∞	∞	∞	∞	9	∞	5	∞

Figure 17. Travel time matrix for the network of Figure 12.

The MINPATH algorithm is based on the continuous expansion of a set of "active" nodes to which fastest travel times are known. When this set includes all nodes, the algorithm terminates. The algorithm is initiated by performing the following operations on the relevant matrix of travel times:

1. Initialize the set of active nodes by indicating that the node of origin is active. The rows of active nodes are eliminated from further calculations, since fastest travel times to these nodes are already known. A circle around the column label and a line through the row are convenient notations for an active node in illustrations.

2. Establish a reference mark for measuring time (we take the mark to be zero) by writing "0" in the time row directly above the node of origin.

3. Enter the label of the node of origin in the path information column associated with that node.

The result of performing these operations on the travel time matrix of Figure 17 is shown in Figure 18. In operation 1, the active nodes of N are those which have already been visited by the traveler on one of several journeys which the algorithm will ultimately construct. Initially, of course, the traveler has "traveled" only to the node of origin, and this node therefore comprises the entire initial set of active nodes. The significance of operation 1 lies in the fact that, by removing the row of T corresponding to the node of origin from further consideration, we are actually eliminating all possibility of the traveler's ever returning to that node once he leaves it. Indeed, it is clear that the fastest path from any node to any other node can never involve the traveler's return to a node already visited; otherwise, the traveler would expend travel time without making progress. With respect to operation 2, we interpret the "0" which has been written in the time row above the node of origin to be the shortest possible time in which the traveler can reach the node of origin, and operation 3 merely indicates the corresponding fastest path.

Following initiation, the MINPATH algorithm proceeds iteratively according to the steps listed below (see Figure 19 for MINPATH flowchart):

Step 1. In each column of T corresponding to an active node, find the shortest travel time from the node to any nonactive node (that is, to any node whose row has not been eliminated); for each active node, keep track of the nonactive node so determined.

	A										
	0										
	A	B	C	D	E	F	G	H	I	J	K
~~A~~	~~∞~~	~~7~~	~~9~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~
B	7	∞	∞	9	10	∞	∞	∞	∞	∞	∞
C	9	∞	∞	∞	9	5	∞	∞	∞	∞	∞
D	∞	9	∞	∞	8	∞	13	18	∞	∞	∞
E	∞	10	9	8	∞	8	10	∞	∞	∞	∞
F	∞	∞	5	∞	8	∞	12	∞	14	∞	∞
G	∞	∞	∞	13	10	12	∞	8	11	∞	∞
H	∞	∞	∞	18	∞	∞	8	∞	∞	13	9
I	∞	∞	∞	∞	∞	14	11	∞	∞	3	∞
J	∞	∞	∞	∞	∞	∞	∞	13	3	∞	5
K	∞	∞	∞	∞	∞	∞	∞	9	∞	5	∞

Figure 18. Travel time matrix for the network of Figure 12 after initialization for the fastest path algorithm.

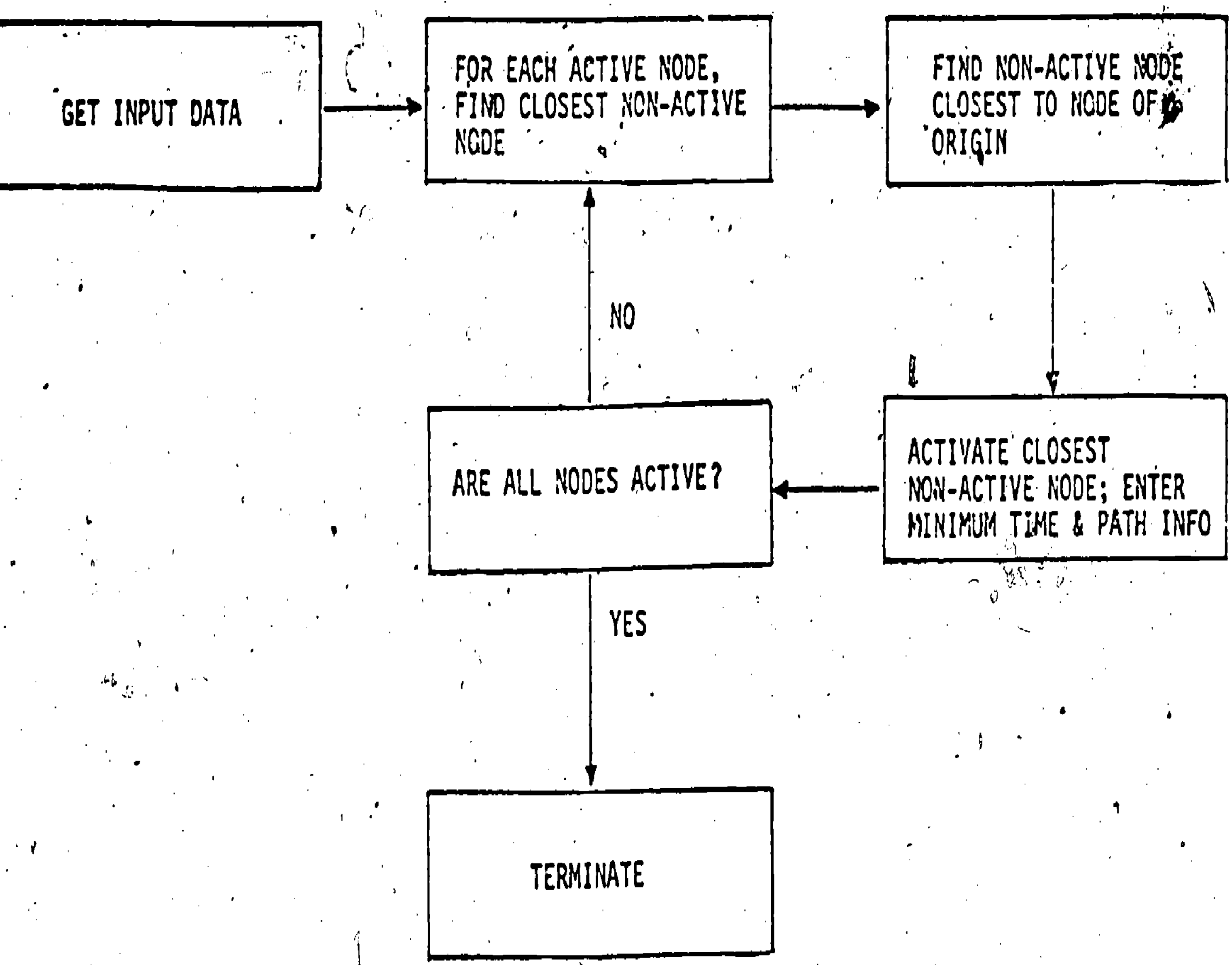

GET INPUT DATA
FOR EACH ACTIVE NODE, FIND CLOSEST NON-ACTIVE NODE
FIND NON-ACTIVE NODE CLOSEST TO NODE OF ORIGIN
ACTIVATE CLOSEST NON-ACTIVE NODE; ENTER MINIMUM TIME & PATH INFO
ARE ALL NODES ACTIVE?
NO
YES
TERMINATE

Figure 19. A flowchart for the MINPATH algorithm.

Step 2. Form the sum of each of the shortest travel times identified in step 1 with the time recorded in the time row above each corresponding active node, and identify the smallest of these sums together with the active node P and nonactive node Q which are involved in the minimum sum.

Step 3. Designate node Q as being active, enter the minimum sum identified in step 2 in the time row immediately above the new active node Q, and copy (in the path column belonging to node Q) the path in the path column belonging to node P, appending to this the symbol of the new active node Q.

Step 4. If all nodes of N are active, terminate; otherwise, go back to step 1.

A tie encountered in the performance of step 1 may result in alternative fastest paths to the nonactive nodes involved in the tie. If information of this type is desired, one must keep track of all the nonactive nodes so tied. Ties encountered in the performance of step 3 are of no consequence and may be ignored by making an arbitrary choice of the minimum sum; the ignored cases will reappear in succeeding iterations and can be disposed of one at a time. Note also that, if we count each performance of steps 1 through 4 as being one iteration of the algorithm, there are precisely n - 1 iterations until termination for an N node network.

We shall now go through the first few iterations of the algorithm beginning with the already initialized travel time matrix of Figure 18.

For step 1, we observe that only node A is active; we therefore look only in column A. We find that the fastest time from node A to a nonactive node is 7, the nonactive node involved here being node B. There are no ties.

For step 2, we form the sum of the time 7 just obtained and time 0 indicated in the time row in column A, obtaining the result 7. Trivially, we search for the minimum sum and, there being only one to consider, we determine that the minimum sum is 7, active node A and nonactive node B being the source of the minimum sum.

For step 3, we designate node B as being active (by drawing a circle around the column heading "B" as shown in Figure 17). We also enter the previously identified minimum sum of 7 in column B of the time row, transfer the path information "A" contained in the path column associated with the new active node B, and append to this the symbol "B" of the new active node.

For step 4, row B is crossed out, thereby eliminating it from further consideration and, since nonactive nodes still remain, we return to step 1. This completes the first iteration of the algorithm; the travel time matrix and associated labels should now appear as shown in Figure 20.

We may conclude at this stage that the fastest travel time through the network from the origin, node A, to node B is 7, and that the associated fastest path is the direct one: A → B.

Proceeding now with step 1 of the second iteration, we observe that in Figure 20, nodes A and B are active. In column A, the fastest travel time from node A to any nonactive node is to node C in time 9, whereas in column B the fastest time is to node D in time 9. In step 2, we form the sums:

9 + 0 = 9 (active node A to nonactive node C)

9 + 7 = 16 (active node B to nonactive node D)

and observe that 9 is the smallest of these. For step 3, the nonactive node C involved in the minimum sum is activated; the minimum sum 9 is entered in column C of the time row, and path information "A" contained in the path column associated with the old active node A is transferred to the path column associated with the new active node C, the symbol "C" then being appended. In step 4, row C is crossed out, and we return again to step 1. This completes the second iteration; the extent of labeling of the travel time matrix at this stage is shown in Figure 21.

In the third iteration, we determine that active node C and nonactive node F produce the minimum sum of 14. Node F is therefore activated, and the various labels are entered in the prescribed way. At the end of the third iteration, the travel time matrix has the appearance shown in Figure 22.

In this example, the algorithm terminates with the activation of node K upon completion of the tenth iteration. The travel time matrix at termination is shown in Figure 23. From the figure, we read off the fastest travel times and paths from the node or origin, A, to the other nodes of the network:

To node B:	Time 7/path A → B
To node C:	Time 9/path A → C
To node D:	Time 16/path A → B → D
To node E:	Time 17/path A → B → E
To node F:	Time 14/path A → C → F
To node G:	Time 26/path A → C → F → G
To node H:	Time 34/path A → B → D → H or A → C → F → G → H
To node I:	Time 28/path A → C → F → I
To node J:	Time 31/path A → C → F → I → J
To node K:	Time 36/path A → C → F → I → J → K.

	A	AB									
	0	7									
	Ⓐ	Ⓑ	C	D	E	F	G	H	I	J	K
~~A~~	~~∞~~	~~7~~	~~9~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~
~~B~~	~~7~~	~~∞~~	~~∞~~	~~9~~	~~10~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~
C	9	∞	∞	∞	9	5	∞	∞	∞	∞	∞
D	∞	9	∞	∞	8	∞	13	18	∞	∞	∞
E	∞	10	9	8	∞	8	10	∞	∞	∞	∞
F	∞	∞	5	∞	8	∞	12	∞	14	∞	∞
G	∞	∞	∞	13	10	12	∞	8	11	∞	∞
H	∞	∞	∞	18	∞	∞	8	∞	∞	13	9
I	∞	∞	∞	∞	∞	14	11	∞	∞	3	∞
J	∞	∞	∞	∞	∞	∞	∞	13	3	∞	5
K	∞	∞	∞	∞	∞	∞	∞	9	∞	5	∞

Figure 20. Travel time matrix for the network of Figure 16 after one iteration of the MINPATH algorithm.

	A	AB	AC								
	0	7	16								
	A	B	C	D	E	F	G	H	I	J	K
~~A~~	~~∞~~	~~7~~	~~9~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~
~~B~~	~~7~~	~~∞~~	~~∞~~	~~9~~	~~10~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~
~~C~~	~~9~~	~~∞~~	~~∞~~	~~∞~~	~~9~~	~~5~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~
D	∞	9	∞	∞	8	∞	13	18	∞	∞	∞
E	∞	10	9	8	∞	8	10	∞	∞	∞	∞
F	∞	∞	5	∞	8	∞	12	∞	14	∞	∞
G	∞	∞	∞	13	10	12	∞	8	11	∞	∞
H	∞	∞	∞	18	∞	∞	8	∞	∞	13	9
I	∞	∞	∞	∞	∞	14	11	∞	∞	3	∞
J	∞	∞	∞	∞	∞	∞	∞	13	3	∞	5
K	∞	∞	∞	∞	∞	∞	∞	9	∞	5	∞

Figure 21. Travel time matrix for the network of Figure 12 after two iterations of the MINPATH algorithm.

	A	AB	AC			ACF					
	0	7	9			4					
	A	B	C	D	E	F	G	H	I	J	K
~~A~~	~~∞~~	~~7~~	~~9~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~
~~B~~	~~7~~	~~∞~~	~~∞~~	~~9~~	~~10~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~
~~C~~	~~9~~	~~∞~~	~~∞~~	~~∞~~	~~9~~	~~5~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~	~~∞~~
D	∞	9	∞	∞	8	∞	13	18	∞	∞	∞
E	∞	10	9	8	∞	8	10	∞	∞	∞	∞
~~F~~	~~∞~~	~~∞~~	~~5~~	~~∞~~	~~8~~	~~∞~~	~~12~~	~~∞~~	~~14~~	~~∞~~	~~∞~~
G	∞	∞	∞	13	10	12	∞	8	11	∞	∞
H	∞	∞	∞	18	∞	∞	8	∞	∞	13	9
I	∞	∞	∞	∞	∞	14	11	∞	∞	3	∞
J	∞	∞	∞	∞	∞	∞	∞	13	3	∞	5
K	∞	∞	∞	∞	∞	∞	∞	9	∞	5	∞

Figure 22. Travel time matrix for the network of Figure 12 after three iterations of the MINPATH algorithm.

	A	AB	AC	ABD	ABE	ACF	ACFG	ABDH ACFGH	ACFI	ACFIJ	ACFIJK
	0	7	9	16	17	14	26	34	28	31	36
	A	B	C	D	E	F	G	H	I	J	K
A	∞	7	9	∞	∞	∞	∞	∞	∞	∞	∞
B	7	∞	∞	9	10	∞	∞	∞	∞	∞	∞
C	9	∞	∞	∞	9	5	∞	∞	∞	∞	∞
D	∞	9	∞	∞	8	∞	13	18	∞	∞	∞
E	∞	10	9	8	∞	8	10	∞	∞	∞	∞
F	∞	∞	5	∞	8	∞	12	∞	14	∞	∞
G	∞	∞	∞	13	10	12	∞	8	11	∞	∞
H	∞	∞	∞	18	∞	∞	8	∞	∞	13	9
I	∞	∞	∞	∞	∞	14	11	∞	∞	3	∞
J	∞	∞	∞	∞	∞	∞	∞	13	3	∞	5
K	∞	∞	∞	∞	∞	∞	∞	9	∞	5	∞

Figure 23. Travel time matrix for the network of Figure 12 after termination of the MINPATH algorithm.

Note that there are two fastest paths from A to H in this example; this result derives from a tie occurring in step 1 of the ninth iteration.

We conclude our discussion about the algorithm MINPATH by pointing out that we have not formally proved in this report that the algorithm produces the shortest travel times and paths in any network. However, a formal proof closely follows the logic of the steps of the algorithm, and is available elsewhere [see Dantzig (1963, pp. 363-366) or Ford and Fulkerson (1962, pp. 130-134)].

Optimal Assignment of Units to Destinations

We now turn to a description of the method whereby MOVANAID computes solutions to assignment problems. Consider a situation where there are n travelers $U_1,...,U_n$ at origins $O_1,...,O_n$ and n destinations $D_1,...,D_n$, one for each traveler. (More general types of assignment problems involving, for example, more than one traveler to be assigned to certain destinations, are possible and of interest. Some of these more general problems are discussed below in the section entitled "Extension of MOVANAID." As presently structured, MOVANAID computes solutions only to assignment problems of the type indicated here.) Suppose that the minimum time necessary for each traveler to reach each destination is known, and that we wish to assign travelers to destinations in such a way that the largest travel time required by any traveler to reach his assigned destination is a minimum. These times are computed by the MINPATH portion of MOVANAID. It is assumed that there are no delays due to interference among simultaneously maneuvering travelers.

It is convenient to organize the information for the problem in a "travel time matrix" A whose rows correspond to the travelers and whose columns correspond to the destinations; the ij^th element of the matrix is the time required for traveler U_i to reach destination D_j along the fastest path. A travel time matrix for a hypothetical assignment problem involving six travelers and six destinations is shown in Figure 24.

It is possible in principle to determine which of the assignments minimizes the maximum travel time by simply examining all possible assignments; a method of this type is called a "solution by enumeration." However, applying this method to a problem of even moderate size requires the examination of an enormous number of assignments--so much so that solution by enumeration must be regarded as being unattractive. (A problem of size 10, for example, would require the examination of $10! = 3628800$ assignments.) A more efficient approach to solving assignment problems is available and has been used for MOVANAID.

An iterative algorithm (which we call "MINASSIGN") for neatly and rapidly solving assignment problems of our type is described by Gross in Ford and Fulkerson (1962). Gross's algorithm draws heavily on earlier work by Konig (1950) and Egervary (1931). Briefly, the Konig-Egervary theorem shows that the assignment problem is equivalent to a problem in

	D_1	D_2	D_3	D_4	D_5	D_6
U_1	4	3	5	1	4	2
U_2	1	3	2	2	5	4
U_3	6	7	1	2	2	3
U_4	8	3	5	6	1	4
U_5	1	2	1	7	6	2
U_6	2	1	6	5	5	2

Figure 24. Travel time matrix for a hypothetical assignment problem of size six.

network flows; Gross's algorithm uses a "labeling" technique, normally used to optimize flows in networks, to solve the assignment problem. The general MINASSIGN technique is illustrated in Figure 25. A full account of these matters is given by Ford and Fulkerson (1962).

For an assignment problem of size n, the MINASSIGN algorithm manipulates an n x n "working matrix" W, each of whose cells is in one of three states: a cell may be "admissible" and contain the symbol "1"; a cell may be admissible and empty; or a cell may be inadmissible. The algorithm, which is complicated at first glance but whose logic is more clear after an example, proceeds according to the following steps:

Step 1. Choose any feasible assignment by entering the symbol "1" in the appropriate rows and columns of W. An assignment is called "feasible" if each traveler is assigned to a destination, and each destination is allocated to a traveler; a "1" appearing in the ijth cell of W is interpreted to mean that traveler U_i is assigned to destination D_j. Note that, in a feasible assignment, one 1 appears in each row and column of W.

Step 2. Determine the largest travel time T involved in the present assignment.

Step 3. Designate all cells of W for which the corresponding travel time is $\geq$ T as being inadmissible, and delete all 1's which occur in inadmissible cells.

Step 4. Identify all rows of W which do not contain a 1 as being "initial rows"; identify all columns of W which do not contain a 1 as being "breakthrough columns."

Step 5. If, for any $1 \leq i, j \leq n$,

(a) Row i is an initial row or is labeled, and

(b) Column j is unlabeled, and

(c) Cell W_{ij} is admissible,

label column j with the number "i." If, after accomplishing (a) through (c) above for all $1 \leq i \leq n$, no column has been newly labeled, conclude that the present assignment is optimal and terminate; otherwise:

If a breakthrough column has been labeled, go to step 7;

If a breakthrough column has not been labeled, go to step 6.

58.

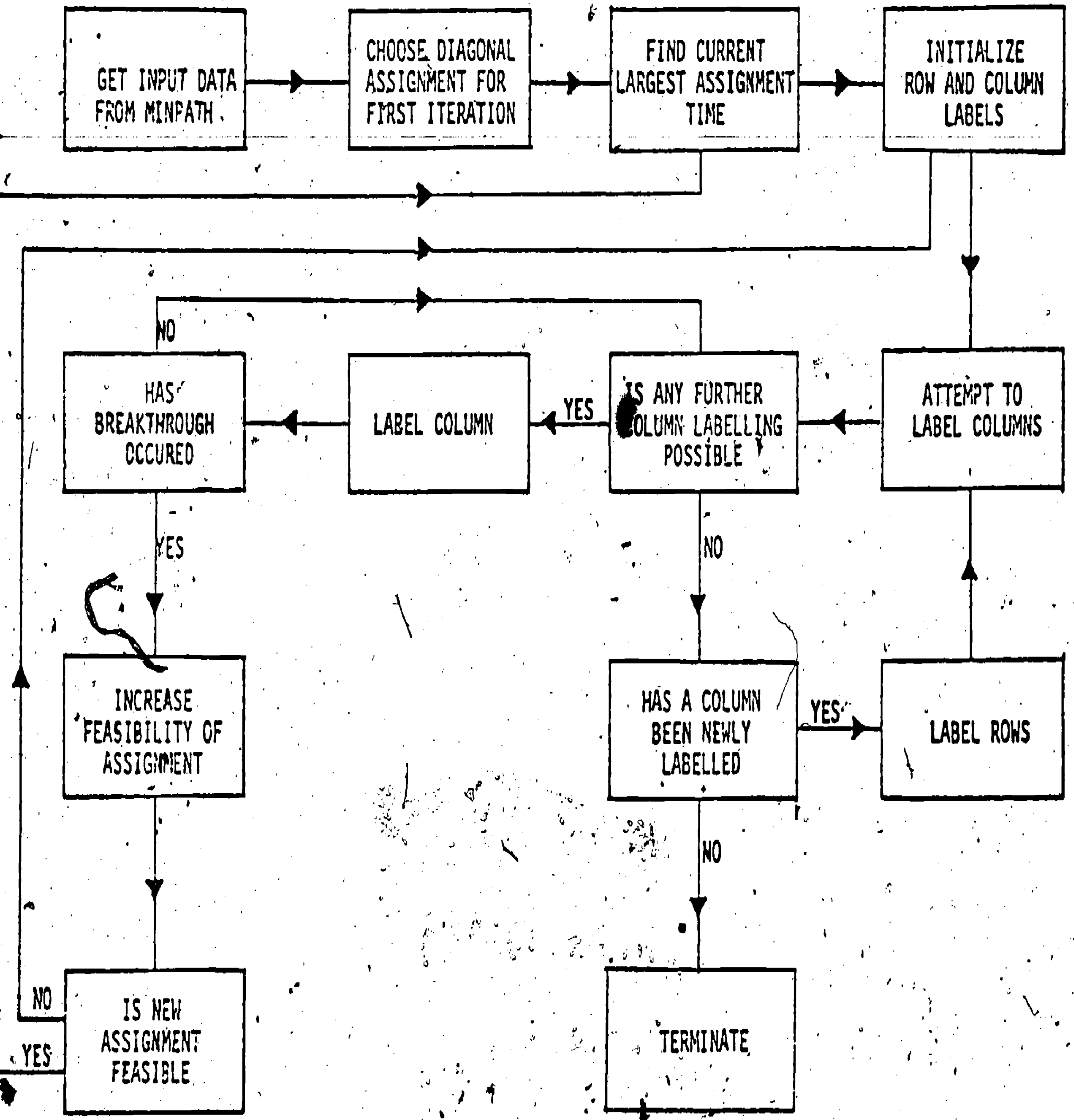

GET INPUT DATA FROM MINPATH
CHOOSE DIAGONAL ASSIGNMENT FOR FIRST ITERATION
FIND CURRENT LARGEST ASSIGNMENT TIME
INITIALIZE ROW AND COLUMN LABELS
ATTEMPT TO LABEL COLUMNS
IS ANY FURTHER COLUMN LABELLING POSSIBLE
YES
LABEL COLUMN
HAS BREAKTHROUGH OCCURED
NO
YES
INCREASE FEASIBILITY OF ASSIGNMENT
IS NEW ASSIGNMENT FEASIBLE
NO
YES
NO
HAS A COLUMN BEEN NEWLY LABELLED
YES
LABEL ROWS
NO
TERMINATE

Figure 25. Flowchart for the assignment algorithm MINASSIGN.

Step 6. If, for any $1 \leq i,j \leq n$,

(a) Column j is labeled, and

(b) Row i is unlabeled, and

(c) Cell W_{ij} contains a 1,

label row i with the number "j." After completion of (a) through (c) above for all $1 \leq j \leq n$, return to step 5.

Step 7. When a breakthrough column receives a label "i," do the following:

(a) Enter a "1" in the ith row of the breakthrough column.

(b) If the ith row is not an initial row, remove the "1" appearing in the column of the ith row as specified by the label "j." on that row, and go to (c); if the ith row is an initial row, go to (d).

(c) Enter a "1" in the row corresponding to the label "1" of the jth column just located in (b) above; if this row is not an initial row, return to (b); otherwise go to (d).

(d) Discard all labels and designations of initial rows and breakthrough columns; if the number of 1's appearing in W is less than n, go back to step 4; otherwise, go back to step 2.

Some comments about the steps will clarify the process. First, although any feasible assignment will suffice for step 1, the diagonal assignment wherein traveler U_i is assigned to destination D_i for $1 \leq i \leq n$ is natural and easily implemented by machine in practice. For hand computations, on the other hand, it is often possible to reduce the number of computations required for termination of the algorithm by making as "good" an initial feasible assignment as one can readily find.

Referral to the travel time matrix A is necessary for step 2. It is convenient to regard each passage through step 2 as being the beginning of an iteration of the algorithm.

Note that the performance of step 3 always results in fewer than n 1's remaining in admissible cells of W; some or all of the cells involved in the present assignment are made inadmissible in step 3. Thus, we might refer to the allocation of admissible 1's that still remain in W after the performance of step 3, if any, as being a "partial feasible assignment." In step 4, any convenient symbol, such as "√" or "*," will suffice in hand computations to identify initial rows and breakthrough columns.

Steps 5 and 6 are the "labeling steps"; in step 5, columns are labeled with row numbers; in step 6, rows with column numbers. Not all rows nor all columns need receive labels. Note that in step 5 it is necessary to keep track of whether or not it has been possible to label any previously unlabeled column with the row number of any initial or labeled row, whereas no corresponding procedure is required in step 6. Note also that, in step 5, all possible column labeling should be accomplished. However, if a breakthrough column receives a label, one may dispense with the rest of step 5, if any remains, and go directly to step 7 before forming any conclusions about optimality or going to another step. Similarly, in step 6, all possible row labeling should be accomplished before reverting to column labeling in step 5.

One reaches step 7 if and only if a breakthrough column receives a label. When breakthrough occurs, the various row and column labels contain the information necessary to reassign travelers to destinations in such a way that the "partialness" of the partial feasible assignment (created in step 3) is increased by one. If, on the one hand, an increase of one in the partial assignment is sufficient to change the partial feasible assignment into a full feasible assignment (involving all the travelers and destinations), one will have found a new (full) feasible assignment that is strictly better than the one in step 2. Thus, in this case, one will wish to discover whether there is a still better full feasible assignment by repeating the entire process beginning with step 2. If, on the other hand, the increase of one in the partial feasible assignment does not result in a full feasible assignment, one does not yet know whether a new strictly better full feasible assignment exists, and another repetition of the labeling process, beginning with step 4, is necessary.

We will now illustrate the behavior of the MINASSIGN algorithm by applying it to the assignment problem whose travel time matrix is shown in Figure 24. Note, incidentally, that a rather good assignment has been hidden in the matrix of Figure 21 as follows:

$U_1 \rightarrow D_4$ in time 1

$U_2 \rightarrow D_1$ in time 1

$U_3 \rightarrow D_3$ in time 1

$U_4 \rightarrow D_5$ in time 1

$U_5 \rightarrow D_2$ in time 2

$U_6 \rightarrow D_6$ in time 2.

Thus, the algorithm must yield an assignment at least as good as this one.

We could, in step 1, try to reduce the number of computations required by choosing as good an assignment as we can readily find, as suggested above. However, for purposes of illustration, we choose the initial assignment to be the diagonal one, which we indicate by circling the appropriate elements of a copy of the travel time matrix at the beginning of iteration 1 as shown in Figure 26.

For step 2, observe that the largest travel time involved in the present assignment is 6. It occurs as a result of assigning traveler U_4 to destination D_4.

In step 3, all cells of the working matrix which correspond to times ≤ 6 in the travel time matrix are made inadmissible by crossing them out. At this stage, the working matrix is illustrated by the first tableau for iteration 1 shown in Figure 26 except for the row and column labels which also appear--these are entered in later steps.

In step 4, we observe that rows four and five of the first tableau do not contain a 1; we therefore designate these rows as being initial rows by affixing to each the symbol "*." Similarly, columns four and five do not contain a 1; hence these are designated as breakthrough columns, again through the use of the symbol "*."

In step 5, observe that only rows four and five are either labeled or initial rows (both are initial rows); hence only these rows may be used to label columns. We begin with row four. It is not possible to label column one with a "4" since, although column one is unlabeled, cell w_{41} is inadmissible. It is possible, however, to label column two with a "4," column two being unlabeled and cell w_{42} being admissible; thus, column two receives the label "4." Noting that column two is not a breakthrough column, we continue with the labeling process and observe that column three should also be labeled with a "4." Column four (one of the breakthrough columns) is, unfortunately, ineligible to be labeled with a "4" since, although column four is unlabeled, cell w_{44} is inadmissible. For column five, however, we find that not only is column five unlabeled but also that cell w_{45} is admissible; column five therefore receives the label "4." Since column five is one of the breakthrough columns, we may dispense with further column labeling (see the comments on step 5 above) and go directly to step 7 to construct the second tableau for iteration 1. The final form for the first tableau for iteration 1 is shown in Figure 26.

Observe, in step 7, that one of the breakthrough columns of the first tableau (column five) is labeled with the label "4." Hence, according to 7a, we enter a 1 in the fourth row of that column (as is shown in the second tableau for iteration 1 in Figure 26) and, since the fourth row is one of the initial rows, we skip ahead to 7d. In 7d, we observe that the number of 1's which now appear in the second tableau is five, and since $5 < 6$, we return to step 4 to continue with iteration 1. The appearance of the working matrix at this stage is

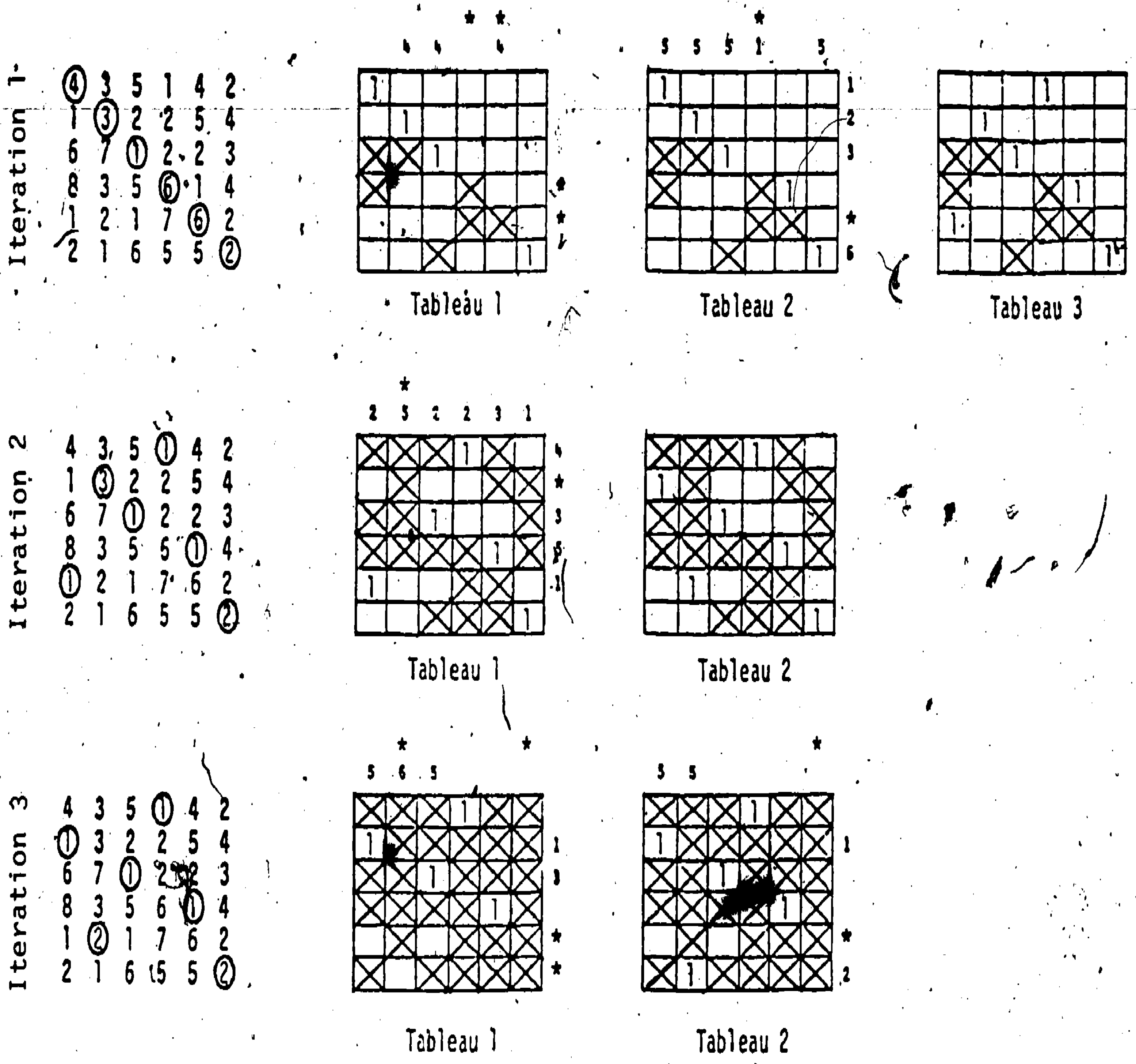


Figure 26. Operation of the MINASSIGN algorithm for the assignment problem of Figure 12.

shown in the second tableau for iteration 1 in Figure 26 (except for the row and column symbols and labels).

Proceeding with step 4 on the second tableau, row five does not contain a 1 and hence is designated an initial row. Similarly, column four is designated a breakthrough column.

In step 5, columns one, two, three, and six receive the label "5." Since none of these columns is a breakthrough column, and since further labeling of columns is not possible at this point, we are directed to step 6.

In step 6, we note that columns one, two, three, and six are the only labeled columns; hence, these are the only columns which may be used to label rows. In column one, a 1 appears in row one; hence row one is labeled "1." After similarly labeling rows two, three, and six, respectively, with labels "2," "3," and "6," we revert to column labeling in step 5.

In step 5, we observe that rows one, two, three, and six have been newly labeled, so that further column labeling may now be possible. With row one, we observe that although cell w_{11} is admissible, column one has already been labeled with the label "5"; we do not change the label on column one but rather skip ahead to column two and then to column three, both of which have also already been labeled. Column four, however, is eligible to receive the label "1" and, column four being a breakthrough column, we may omit further column labeling as before and go directly to step 7. The final appearance of the second tableau for iteration 1 is shown in Figure 26.

Observe, in step 7, that the breakthrough column (column four) has the label "1." We therefore insert a 1 in row one of column four. Row one is not an initial row, and we therefore look for a label on row one and find that the label is "1"; this is the number of the column of row one where we will find a 1. After removing the 1 so located, we are directed to insert a 1 in the row of column one indicated by the label on column one, here "5." Because row five is an initial row, the movement of 1's is complete. Further, there now being six 1's in the working matrix for iteration 1, we conclude that iteration 1 is also complete and that there is a better assignment than the one which was used at the beginning of the iteration (in our case, the diagonal assignment). This assignment, which is the initial assignment for iteration 2, is indicated by the pattern of 1's which appears in the third and final tableau for iteration 1 shown in Figure 26.

Iteration 2, for which just two tableaux are required, is accomplished in a similar fashion. After completion of iteration 2, we find that the new assignment created at the end of iteration 1 (which is shown in Figure 26 by circling appropriate elements of the time matrix at the beginning of iteration 2) is still not optimal, and we obtain a better assignment in the pattern of 1's which appears in the final tableau of iteration 2; the latter is the initial assignment for iteration 3.

Iteration 3 proceeds as before until we reach the point of labeling rows and columns in the second tableau. All possible rows and columns are then labeled, yet no breakthrough column receives a label. We conclude at that point (step 5), therefore, that the initial assignment for iteration 3 is optimal, and we terminate. The optimal assignment is as follows:

$U_1 \rightarrow D_4$ in time 1

$U_2 \rightarrow D_1$ in time 1

$U_3 \rightarrow D_3$ in time 1

$U_4 \rightarrow D_5$ in time 1

$U_5 \rightarrow D_2$ in time 2

$U_6 \rightarrow D_6$ in time 2.

This is the assignment mentioned earlier.

Optimal assignments need not be unique. For the example problem, a different assignment as good as the one identified by the algorithm is as follows:

$U_1 \rightarrow D_4$ in time 1

$U_2 \rightarrow D_1$ in time 1

$U_3 \rightarrow D_3$ in time 1

$U_4 \rightarrow D_5$ in time 1

$U_5 \rightarrow D_6$ in time 2

$U_6 \rightarrow D_2$ in time 2.

The choice of initial assignment for iteration 1, together with certain other factors, determines the optimal assignment to which the algorithm will converge.

Computer Implementation

In the previous section, we have described in some detail the theoretical basis for MOVANAID. In the translation of such theoretical developments into practice, it is not uncommon to encounter one or more practical difficulties arising from the environment in which the theory is being applied. In the present section, we discuss the most significant such practical consideration which arose in the implementation of MOVANAID--namely, a constraint on computer storage space. We also describe the effect of this constraint on the structure and operation of the aid.

MOVANAID was constructed for use in the Training and Information Systems Facility (TISF) at ARI. The TISF consists of a CDC model 3300 computer connected to laboratory equipment, notably CDC 210 CRT display terminals and IBM selectric typewriters used for the SIMTOS experiments. Of the total amount of core storage available in the CDC 3300, only about half is usable for the programs which control laboratory experiments. If the aid were to be used in conjunction with SIMTOS (or some other experiment with its own computer support requirements), only a small portion of this memory space would be available at the time a subject might call upon the aid for his experimental tasks. Thus, the necessary strategy in this case was to be conservative with computer storage space.

Two techniques were used to conserve storage: data packing and the use of storage techniques designed for sparse matrices for network data. By data packing, we mean simply that several items of data in decimal integer form are compressed for storage in a single 24-bit word of computer memory. Binary packing of data can be used to increase speed of computation. However, packing of decimal data has compensating benefits, such as readability of the data base. Each link attached to any given node is described by one data word in which is coded the node number of the attached node, the length of the link, and the "type" of link (see Appendix B for a detailed description of link type); this coding saves about 8n words of storage (where n is the number of nodes).

In the theoretical discussion of MINPATH, linkage information was presented in an n x n matrix (where n is the number of nodes). The use of a square array of data to store this information is clearly wasteful of storage in a network in which nodes are connected to only a few others, i.e., a sparse matrix. Instead, we assume a constant four links per node and store the information in a linear array. This procedure causes no loss of generality, because nodes with more than four links are easily accommodated by the use of artificial nodes as described in Appendix B. The number four was chosen because it adequately reflects the SIMTOS network. Other numbers of links per node may be appropriate for other networks. At present, network data are stored such that all arcs are two-way connections, i.e., as an undirected network. Even more storage could be saved if only the one-way link information were stored, but greatly increase computation time. Thus, the network linkages are stored in approximately 4n words of memory rather than the n^2 needed for the square array.

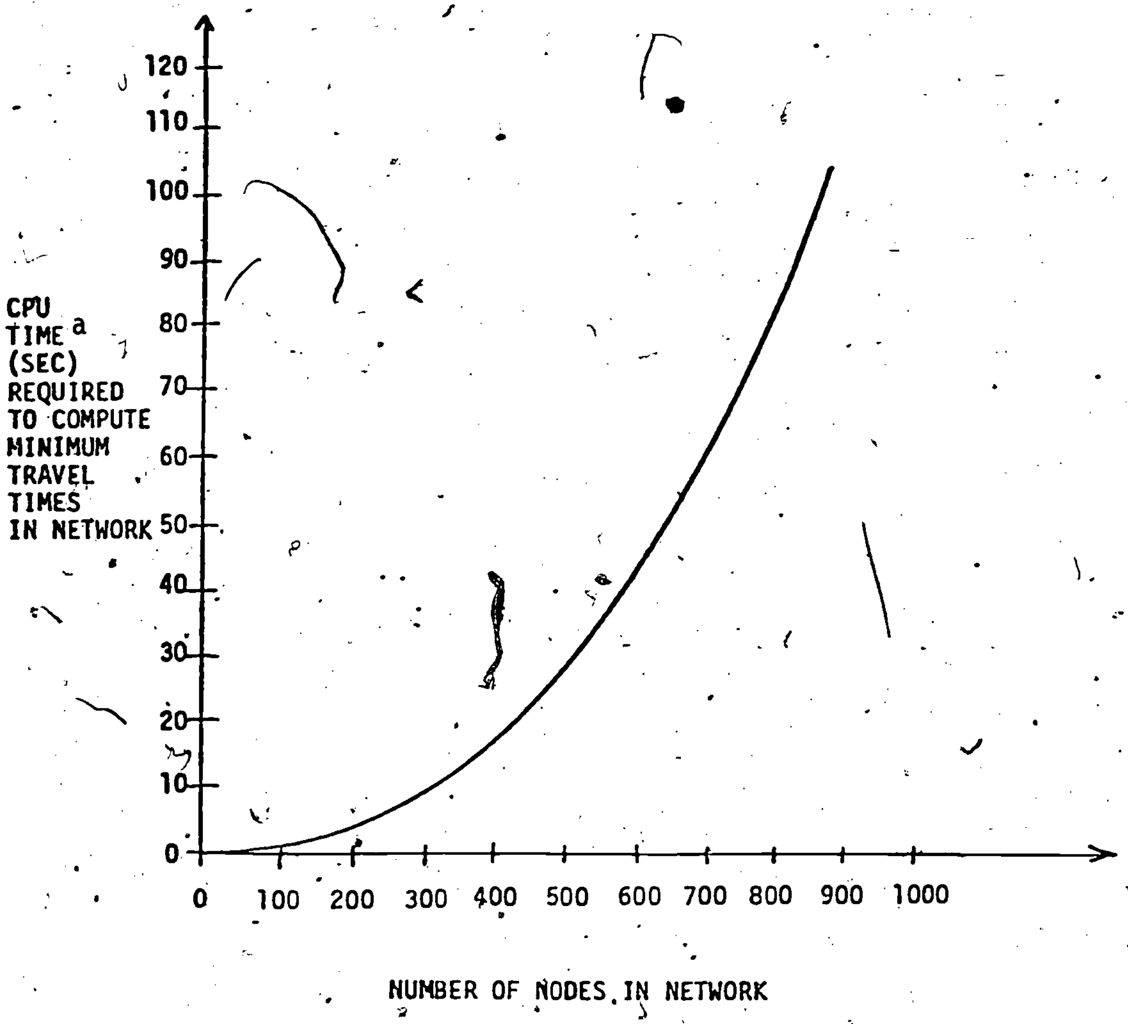
The use of these storage conservation measures is not without drawbacks. They affect the operation of the aid in two significant ways. First, they effectively limit the size of the networks with which MOVANAID can be used because the data packing scheme chosen imposes a limit on the size of the data being packed. Other packing schemes could be used and would result in different constraints on the size of the data being packed. Because three decimal digits in the packed word are allowed for the node number and three for the arc length, there can be at most 999 sequentially numbered nodes, and arcs may be 99.9 kilometers in length (individual lengths are specified in tenths of kilometers).

The other difficulty with the storage conservation procedures is that they significantly increase the computer time required to solve a problem. The packed words must be unpacked by MOVANAID before the information can be processed. However, for networks of the size produced for SIMTOS (approximately 400 nodes), the tradeoff has been reasonable. To understand this, consider the graph of Figure 27. Although available resources have not permitted extensive computational experiments with the full-scale MOVANAID, preliminary experiments were conducted with the MINPATH routine alone in its elementary form.

These results, shown in Figure 27, are exclusive of input/output or any data preprocessing. They are based on a model implementation using unpacked data, as might be the case if storage space were unlimited. The data show that without the storage conservation measures, networks of the largest type that MOVANAID can treat could be solved in approximately 2 minutes, while networks of the current size of the data base could be handled in about 15 seconds. Experience (though limited) with MOVANAID has shown that small networks (less than 200 nodes) require only slightly greater solution times than those in Figure 27 show, and that networks of the size of SIMTOS require about twice as much time as Figure 27 shows. The latter case results in times of about 30 seconds rather than 15 for the SIMTOS network, but this is still within reason. The rate of increase in computation time apparently grows rapidly with network size, however, so that the tradeoff for storage space may not be as attractive for networks of the largest type that MOVANAID can treat. Experimental confirmation is needed.

EXTENSION OF MOVANAID

In this section we suggest some ideas for enlargement of the scope, for quality improvement of the information now provided to users by MOVANAID. Most remarks are directed toward possible directions for extension of the two basic modules of MOVANAID: MINPATH and MINASSIGN.



[a]Times were determined experimentally on an IBM 360/67 computer.

Figure 27. Minimum path computation times as a function of network size.

Generalizations for MINPATH

Multiple-shortest path information may occasionally be of interest to intelligence processors. Such information might be useful, for example, in a case where the friendly force is considering the interdiction of roads. As presently configured, however, the MINPATH module of MOVANAID does not disclose such information to users. Path information of this type could be produced (at the expense of an increase in computer storage requirements) by making minor changes in the present structure of MINPATH. Similarly, analysts might also wish to identify, in addition to fastest times and paths, the second and third fastest times and paths, and so on. Algorithms exist and could be included in MINPATH to produce information of these types, if desired.

Another interesting direction for generalizing MINPATH would involve relaxing the assumption (heretofore implicit) that a military unit can move along a road en masse. The movement of military units along roads does not normally take place en masse for at least two reasons. First, the tactical situation may dictate that movement be accomplished in a columnar formation of gap-separated segments or "serials." Second, columnar movement along a road may be forced on a unit whose size is large compared to the capacity of the road. The remarks below address only the latter phenomenon.

Consider a network, each of whose arcs is associated, in addition to a travel time (which we interpret to be the time required for an "individual"--perhaps a foot soldier or a truck--to move from one end of the arc to the other), with an arc capacity[4] representing the maximum number of individuals who can enter the arc per unit time. A network of this type is shown in Figure 28; in the figure, the first number associated with an arc is the travel time for that arc, and the second number is the arc capacity. We take our problem to be the determination of the path which the unit should take in order to move from a prespecified node of origin to a given node of destination in as short a time as possible, subject to the constraint that unit movement rates should not exceed any of the arc capacities.

As an example of the sort of difficulty one may encounter in the generalized problem, assume that the arc capacities shown in Figure 28 represent the maximum number of tanks per hour that the arcs can accommodate, and suppose that we wish to know how soon, and by what path, a tank battalion at node "A" could move in its entirety to a destination at node "Z." Consider two paths joining node "A" to node "Z" as follows:

[4]Definite procedures for computing road capacities have already been established by the Army. Road capacities are computed as functions of surface type, surface and shoulder width, maximum curvature and gradient, amount of moisture, and certain other parameters. See Department of the Army Field Manual 55-15 (Transportation Reference Data, Chapter 2) for more details.

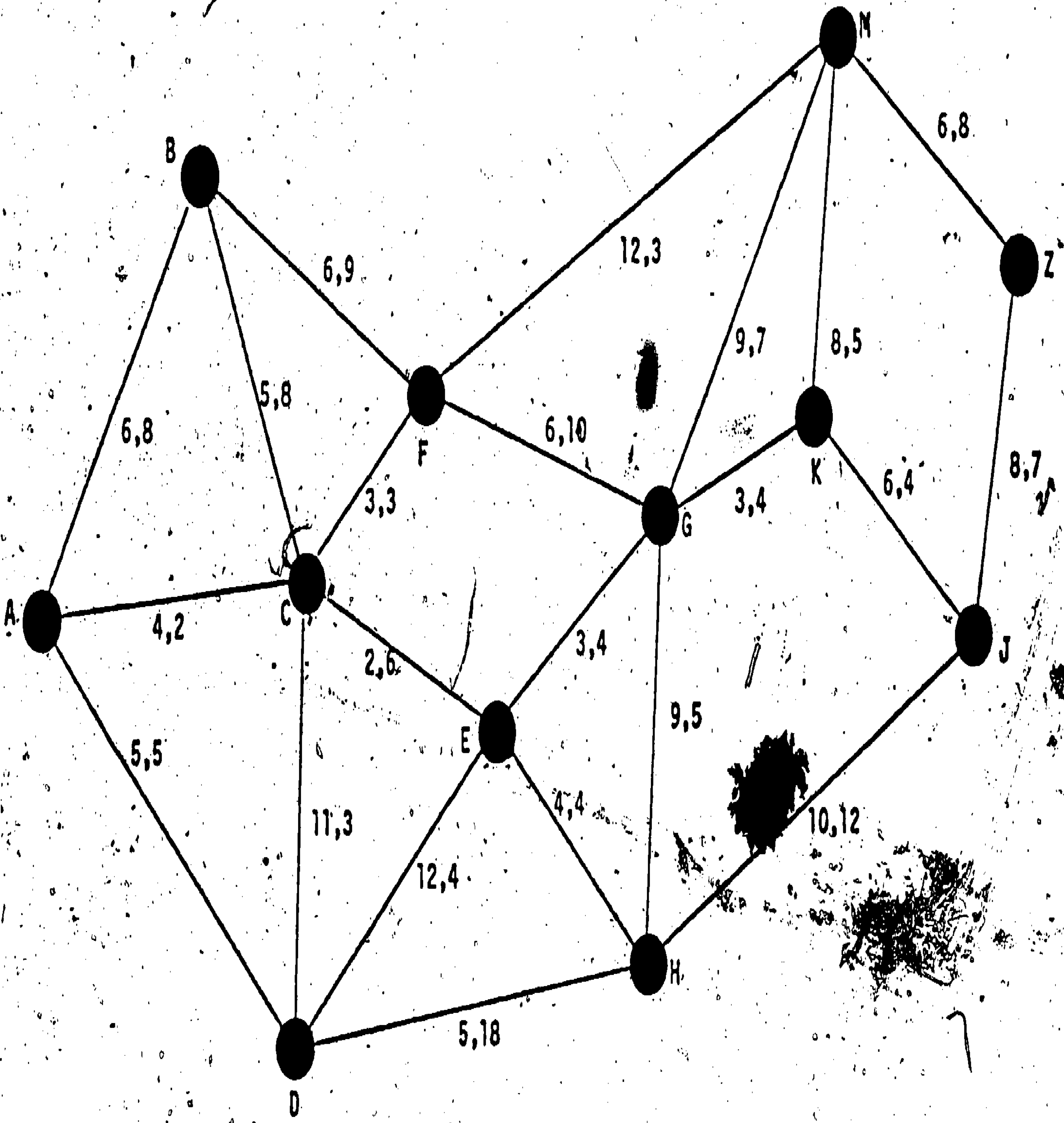


Figure 28. A network with arc capacities in addition to travel times.

$$P_1: A \to C \to E \to G \to K \to J \to Z$$

$$P_2: A \to B \to F \to G \to M \to Z.$$

It may be confirmed that the sum of the travel times along path P_1 is 26, whereas the corresponding sum for P_2 is 33. Thus, path P_1 would be preferable to P_2 if it were possible for the battalion to move en masse along each arc of P_1. However, if the size n of the battalion were larger than the smallest arc capacity of any of the arcs of P_1, it may happen that the unit could arrive at node "Z" sooner using P_2 (all of whose arcs have rather large capacities) than it could using P_1.

It may be shown that the generalized problem we wish to solve is the following: Determine

$$\min_{P} \left[\sum_{(X,Y)\varepsilon P} t(X,Y) + d(P) \right],$$

where

A is the node of origin

Z is the node of destination

P is a path joining A to Z

$t(X,Y) \geq 0$ is the travel time associated with arc (X,Y)

$c(X,Y) \geq 0$ is the capacity associated with arc (X,Y)

n is the number of individuals in the source at A

$$d(P) = \max\left(\frac{n}{\min_{(X,Y)\varepsilon P} c(X,Y)} - 1, 0\right).$$

The units in which the t(X,Y), c(X,Y) and n are measured must be compatible.

This problem, though perhaps formidable at first glance (due to the presence of the term d(P) inside the square brackets), may be solved by an algorithm similar to MINPATH.

Note, incidentally, that in the special case where all arcs of the network have the same capacity $c \leq n$, we have

$$\min_{P} \left[\sum_{(X,Y) \varepsilon P} t(X,Y) + d(P) \right]$$

$$= \frac{n}{c} - 1 + \min_{P} \sum_{(X,Y) \varepsilon P} t(X,Y) .$$

In this case, therefore, the optimization problem involved is identical to the one considered earlier: Find the path joining the origin to the destination for which the sum of the arc travel times is a minimum. Here, the additive constant $(n/c) - 1$ is the column closing time imposed on a maneuvering unit by roads whose capacity is inadequate to permit movement en masse.

Another potentially important generalization of the MINPATH algorithm arises from the fact that all of the preceding development and discussion has been directed toward the computation of fastest times and paths through a road network for one unit at a time. Nowhere have we admitted the possibility that two or more units, moving simultaneously through the network, might interfere with each other by occupying the same arc or node simultaneously. In reality, of course, delays might well ensue in such circumstances, and we therefore think it proper to consider enlarging the model to take into account this phenomenon. Such an enlargement might lead to changes, or even abandonment, of the MINASSIGN portion of MOVANAID.

It appears that the enlargement of the model to take into account interunit interference might be accomplished through the imposition of <u>node capacities</u> on some or all nodes of the network. A node capacity is, of course, an upper bound on the rate at which traffic, from whatever source, can pass through the node. A technique whereby networks with node capacities may be converted to equivalent but larger networks with only arc capacities is suggested by Ford and Fulkerson (1962, pp. 23-26).

<u>Generalizations for MINASSIGN</u>

There are two minor ways in which the MINASSIGN portion of MOVANAID might be refined. First, it may be desirable to arrange for the algorithm to disclose alternative optimum assignments when such exist. Second, it may be desirable to arrange for MINASSIGN to optimize assignments with respect not only to largest assignment time (as is now done), but also with respect to the remaining assignment times. Although these modifications would be relatively easy to implement, we feel that the added capability would not be nec sary:

The assignment model to which the present version of MINASSIGN is applicable assumes that there will always be the same number of travelers as destinations, each traveler being assigned to exactly one destination and each destination receiving precisely one traveler. A number of interesting variations on this theme are possible. Suppose, for example, that we generalize the model to allow the possibility of having more travelers than destinations while retaining (for the moment) the notion that each destination should be the recipient of precisely one traveler. The relevant objective would be, as before, the optimal assignment of one traveler to each destination and, in addition, the optimal selection of a subset of travelers to be so assigned. A military application of a model of this type would arise in cases where the enemy had available in the rear of the battle area a larger number of reinforcement units than there were front-line positions to be reinforced.

In another variation of the present assignment model, we might wish to require that some or all of the destinations be recipients of more than one traveler. An enlarged model of this type would be applicable, for example, in cases where it was thought that the enemy would like to reinforce front-line units with more than one reinforcement unit, with some front-line units perhaps more heavily reinforced than others.

It appears that these and other variations of the assignment model may all be treated by a general model involving m potential travelers and n potential destinations together with the freedom to require that destinations be assigned a prespecified number of travelers. The relevant objective, of course, would be the optimal selection and assignment of a subset of the travelers to a subset of the destinations, subject to side conditions.

In view of the fact that the generalized assignment model suggested above is substantially more general than the model to which the present MINASSIGN algorithm is applicable, it is perhaps surprising to find that MINASSIGN can, with relatively minor modifications, solve any of the problems which might arise in the more general model. One such modification would involve the use of artificial travelers and/or destinations. The revised algorithm would proceed in much the same fashion as before.

SUMMARY AND CONCLUSIONS

The initial purpose of this project was to develop and implement some form of computer-based analytic aid in the ARI laboratory. The implemented aid was expected to serve as a test bed for research on general principles of man-machine interaction. As the project moved from the general consideration of a number of aids to the specific development of MOVANAID, it became apparent that the process of developing interactive aids is a nontrivial task. Given a relatively simple intelligence processing task and readily available mathematical techniques, it required a large effort to implement a preliminary version of MOVANAID. This report provides a description of the results of that effort.

MOVANAID has value beyond its intended role as a research vehicle for studying man-machine interaction. It serves as a demonstration and a prototype of the large class of potential interactive analytic aids. It also provides a baseline for further development and eventual implementation of a movement analysis aid to support tactical intelligence analysis. The requirements for such aids and the form which they take will be clarified through an iterative process of test and revision.

REFERENCES

Bellman, Richard. On a Routing Problem. Quarterly of Applied Mathematics, 1958, 16, 87-90.

Dantzig, G. B. On the Shortest Route Through a Network. Management Science, 1960, 6, 187-190.

Dantzig, G. B. Linear Programming and Extensions. Princeton, N.J.: Princeton University Press, 1963.

Department of the Army. Combat Intelligence. Field Manual No. FM 30-5, 1971.

Department of the Army. Handbook on Aggressor. Field Manual No. FM 30-102, 1973.

Department of the Army. Staff Officer's Field Manual: Staff Organization and Procedures. Field Manual No. FM 101-5, 1972.

Department of the Army. Transportation Reference Data. Field Manual No. FM 55-15, 1963.

Egerváry, J. Matrixox Kombinatorikus Tulajonságairol. Mat. és Fiz., 1931, Lapok 38, 16-28; translation by H. W. Kuhn, On Combinatorial Properties of Matrices. George Washington University Logistics Papers, 1955, 11.

Ford, L. R., Jr., & Fulkerson, D. R. Flows in Networks. Princeton, N.J.: Princeton University Press, 1962.

Konig, D. Theorie der Endlichen und Unendlichen Graphen. New York: Chelsea, 1950.

Minty, G. J. A Comment on the Shortest Route Problem. Operations Research, 1957, 5, 724.

Moore, E. F. The Shortest Path Through a Maze. (Unpublished report). Bell Telephone Laboratories, 1957.