②

NPRDC TR 86-27 SEPTEMBER 1986

A SHORTEST-PATH METHOD FOR ESTIMATING PERMANENT CHANGE OF STATION (PCS) COSTS

NAVY PERSONNEL RESEARCH AND DEVELOPMENT CENTER SAN DIEGO, CALIFORNIA

NAVY PERSONNEL RESEARCH
AND
DEVELOPMENT CENTER
San Diego, California 92152

86 10 13 319

390772 pim

NPRDC TR 86-27 September 1986

A SHORTEST-PATH METHOD FOR ESTIMATING PERMANENT CHANGE OF STATION (PCS) COSTS

Geoffrey L. Zimmerman

Reviewed by
Joe Silverman

Approved by
Martin F. Wiskoff

Released by
B. E. Bacon
Captain, U.S. Navy
Commanding Officer

Navy Personnel Research and Development Center
San Diego, California 92152-6800

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

AD-A173629

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S) NPRDC TR 86-27	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION Navy Personnel Research and Development Center; 6b OFFICE SYMBOL (If applicable) Code 61	7a. NAME OF MONITORING ORGANIZATION
6c ADDRESS (City, State, and ZIP Code) San Diego, CA 92152-6800	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Naval Military Personnel Command; 8b. OFFICE SYMBOL (If applicable) NMPC-46	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (City, State, and ZIP Code) Navy Annex, Washington, DC 20370	10 SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO; PROJECT NO; TASK NO; WORK UNIT ACCESSION NO WR75128

11 TITLE (Include Security Classification)
A SHORTEST-PATH METHOD FOR ESTIMATING PERMANENT CHANGE OF STATION (PCS) COSTS

12 PERSONAL AUTHOR(S)
Zimmerman, Geoffrey L.

13a. TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Technical Report	FROM 85 Mar TO 85 Oct	1986 September	19

16 SUPPLEMENTARY NOTATION

17 COSATI CODES

FIELD	GROUP	SUB-GROUP
12	02	

18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Algorithm, arcs, duty stations, mileage, network, nodes, official distance table, Permanent Change of Station (PCS), PCS Step I cost, shortest-path, spanning-tree method

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

A Permanent Change of Station Cost Generation Model (PCSMOD) has recently improved the efficiency and accuracy of computing individual PCS cost estimates, and the Commander, Naval Military Personnel Command (NMPC), has requested that the list of duty stations covered by PCSMOD be expanded. The network flow methodology used by PCSMOD, however, requires too much computer time to add a large number of duty stations. This report documents the development of a modification to PCSMOD in which a shortest-path algorithm calculates road mileages. This modification saves 729 hours of computer time.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT [✓ UNCLASSIFIED/UNLIMITED] ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a NAME OF RESPONSIBLE INDIVIDUAL Zimmerman, Geoffrey L.	22b TELEPHONE (Include Area Code) (619) 225-2371; 22c OFFICE SYMBOL Code 61

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

FOREWORD

The Permanent Change of Station Cost-Generation Model recently accepted for implementation by the Naval Military Personnel Command (NMPC) will automate the cost estimates for permanent change of station moves by Navy personnel and is likely to result in significant savings (see NPRDC TR 84-52). Recently, the Commander, NMPC, requested that the number of duty stations in the preliminary model be increased from 506 to 1096 before the operational implementation. Because this created a threefold increase in the number of distance calculations required, a new algorithm that required less computer time had to be devised.

This report describes the new algorithm for the "shortest-path" problem. It saves 729 hours of computer time over the old method and allows the 30 percent increase in accuracy that was created by adding more duty stations. Research was conducted as part of work unit WR75128 (Permanent Change of Station Costing).

B. E. BACON
Captain, U.S. Navy
Commanding Officer

JAMES W. TWEEDDALE
Technical Director

SUMMARY

Problem

A Permanent Change of Station Cost-Generation Model (PCSMOD) was developed by Navy Personnel Research and Development Center (NAVPERSRANDCEN) in 1984 to improve the efficiency and accuracy of computing individual permanent change of station (PCS) cost estimates. Recently, the Naval Military Personnel Command (NMPC) tasked NAVPERSRANDCEN to expand the number of duty stations in the model from 506 to 1,096. The old network flow methodology used for computing road mileages required too much computer time to add such a large number of duty stations. A more efficient methodology was needed.

Objective

The objective of this effort was to apply a more efficient methodology to expand the number of duty stations in PCSMOD and to document the computational efficiencies gained.

Approach

A new methodology using a shortest-path algorithm was developed and incorporated into PCSMOD. A computer-time comparison between the old and new algorithms was made to estimate the total amount of computer time saved.

Results and Discussion

The shortest-path algorithm saved 729 central processing unit (CPU) hours of computer time. The accuracy of PCSMOD mileage calculations also improved approximately 30 percent as a result of expanding the network of duty stations. The new methodology increases both the efficiency of computing PCS distances and the flexibility of the model to incorporate additional duty stations in the future.

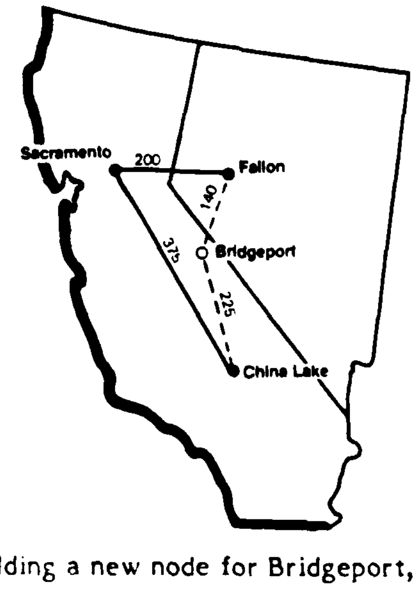

Figure 5 (see p. 11). Adding a new node for Bridgeport, California to the network.

CONTENTS

LIST OF FIGURES

INTRODUCTION

Background

The annual budget for permanent change of station (PCS) moves of Navy personnel currently exceeds $600 million. Navy policy makers are under continuing pressure to control moving costs and to constrain the growth of the PCS budget. Restricting the annual number of moves is the least desirable method of controlling the PCS budget because it conflicts with other important planning policies such as sea-shore rotation and fleet readiness. New methods to more efficiently execute the PCS budget are needed.

More efficient and accurate computation of individual PCS cost estimates is one way to improve PCS budget execution. The process of estimating the STEP I[1] cost of a PCS move is currently performed manually. This process can be divided into two parts. First, the shortest travel distance between the old and new duty stations, including any travel for temporary duty, is calculated using official distance tables and, when necessary, road atlases. Second, this distance estimate, as well as information specific to the individual mover (rank, number of dependents, etc.), is applied to the PCS STEP I cost tables to derive a STEP I cost estimate for the move. This manual process is very time consuming. In addition, when road maps are used to calculate the shortest driving distance between two points, errors in these mileage figures can occur. If this manual process were automated, much time would be saved and many errors eliminated.

In response to this need, the Permanent Change of Station Cost-Generation Model (PCSMOD) was developed (Wong, Jerardo, & Nakada, 1984) and is being implemented. PCSMOD has two components: (1) STEP I costing and (2) mileage calculation. The STEP I costing component consists of software designed to read cost tables and convert the mileage and personnel data into cost estimates. The mileage calculation component is imbedded in the model as a network in which the nodes are duty stations and the arcs are the road mileages between neighboring nodes. The software used to solve the network problem in the first version of the model was NETFLO, a computer algorithm written by Kennington and Helgason (1980). For a detailed description of the formulation of this model, see Wong, Jerado, and Nakada (1984).

The first PCSMOD included 506 major duty stations in the Continental United States (CONUS).[2] NETFLO was used to generate a distance table which contains the shortest routes between all pairs of these 506 nodes. For operational implementation, the Naval Military Personnel Command (NMPC) tasked the Navy Personnel Research and Development Center (NAVPERSRANDCEN) to expand the number of duty stations in the model from 506 to 1,096. Therefore, the network had to be modified to include these additional nodes.

[1] A STEP I cost represents the initial cost estimate while a STEP II cost is the final estimate prior to the move.

[2] Although PCSMOD covers world-wide moves, only CONUS travel mileage is required for calculating STEP I costs.

Problem

The original use of NETFLO to create the existing distance table was justified by time constraints and the need for rapid operational implementation. PCSMOD was a side product of the main research effort, with the request for this model not coming until the later stages of the larger study. Since the NETFLO software package was already available at NAVPERSRANDCEN, we were able to generate the existing distance table without first researching other software packages. However, expanding the number of duty stations covered from 506 to 1,096 increased the scope of the problem to such a degree that using NETFLO was no longer feasible. We estimated that NETFLO would require about 735 central processing unit (CPU) hours on NAVPERSRANDCEN's main frame computer (IBM 4341/M12) to expand the mileage table. An alternative approach had to be developed.

One reason for the large demand on computer time is that the number of from/to duty station combinations increases exponentially with the number of duty stations. A matrix of from/to combinations for all 1,096 duty stations contains 600,060 unique entries. The existing distance table, which covered 506 duty stations, contained only 127,863 unique entries.[3] Hence, doubling the number of duty stations quadrupled the number of from/to combinations. Even subtracting the existing entries left 472,197 entries that still needed to be calculated. This represented more than a threefold increase over the number of distances that had already been calculated.

A second reason for the large demand on computer time was that as the network is expanded, NETFLO must search over an increasing number of arcs (from/to combinations) to find the shortest distances between duty stations. As a result, the CPU time required for each mileage calculation is increased. In summary, expanding the network to include additional duty stations increases both the time and frequency of mileage calculations.

Objective

The objective of this effort was to investigate and apply a more efficient methodology to expand the number of duty stations in PCSMOD and to document the computational efficiencies gained.

[3]First, we eliminate the diagonal elements of the distance table, since the distance from node to itself is zero. Then, we eliminate one half of the off-diagonal elements since all distances are symmetric, i.e., d(i,j) = d(j,i). Hence,

$$((1096)^2 - 1096)/2 = 600,060$$

$$((506)^2 - 506)/2 = 127,863$$

APPROACH

Calculating road mileages between duty stations is a special form of network problem called the shortest-path problem. Several algorithms have been developed for solving the shortest-path problem. Algorithm characteristics vary according to the characteristics of the network to which they are applied, as summarized in Table 1. Arc lengths are not generally assumed to be symmetric (i.e., the distance from node i to node j may be different from the distance in the opposite direction; Pollack & Wiebenson, 1960). In this particular application, however, this generalization is unnecessary. Some algorithms, such as the Decomposition algorithm developed by Hu (1968), allow for negative arc lengths. This situation can arise when arc numbers represent costs and designated arcs are considered profitable (Dreyfus, 1969). This feature is also unnecessary for our purposes since negative distances are undefined.

Table 1

Shortest-path Algorithm Characteristics

Characteristic	Applicability to PCSMOD Distance Calculation	Comments
Assumes symmetric arc lengths	Yes	Distance from i to j equals distance from j to i.
Allows negative arc lengths	No	Negative distances are undefined.
Assumes the network has high density	No	Duty station network density is approximately 1 percent.
Assumes the network is sparse	Yes	Duty station network density is approxi- 1 percent.
Uses Spanning-Tree method	Yes	Desirable when updating the network.
Uses Matrix method	No	Requires too much computer storage.
Capable of retrieving optimal path	No	Unnecessary but useful for error checking.
Searches for second best route	No	Does not apply to PCSMOD distance problem.

Another important network characteristic is density. A network is said to be dense if the number of arcs is close to the square of N, where N is the number of nodes. A network is sparse if the percentage of arcs is small (Denardo & Fox, 1979). In the test version of PCSMOD, there are a total of 3,036 arcs which together represent only about 1.2 percent of the maximum possible number of arcs. A percentage of such small magnitude indicates a sparse network. In addition, the network defined in the updated version of the model will also be sparse since the number of additional arcs represents a similar percentage of the number of additional nodes squared.

Shortest-path algorithms also differ in the nature of the solution that is generated. "Spanning-Tree" methods search for the minimum distance from a given node (the root of the tree) to all other nodes in the network. "Matrix" algorithms compute the shortest path between every possible pair of nodes. In this context, the choice of method depends not only on the type of solution desired but also on considerations of computing efficiency. For instance, a characteristic problem of Matrix methods is that they require large amounts of computer memory (Pollack & Wiebenson, 1960). In addition, an advantage of Spanning-Tree methods is that adding new duty stations to the network in the future does not require recomputing the entire distance table. Only the spanning tree for each new node, representing one row of the distance table, needs to be calculated. For these reasons, only Spanning-Tree methods are considered here.

After calculating the minimum distance, a certain class of Spanning-Tree algorithms allows the user to retrieve the optimal path (i.e., the set of nodes that must be traversed while following the shortest route between the starting and ending nodes). Another feature is the ability to identify the second (or third, etc.) shortest route. For instance, an algorithm developed by Hoffman and Pavley (1959) computes the distance for a path from node i to node m that coincides with the shortest path up to the j^{th} node, then deviates to some specified node k, and finally proceeds to the terminal node m using the minimum distance from k to m (Dreyfus, 1969). Note that this method differs from the shortest-path method in that the path from i to k used in Hoffman and Pavley's method will not in general coincide with the shortest path from i to k.

Dijkstra's Algorithm

The method chosen for updating the distance table is an algorithm developed by Dijkstra (1959). This algorithm, which we will henceforth refer to as TREESORT, is a Spanning-Tree algorithm and therefore computes the shortest path from any given node to all remaining nodes in the network. TREESORT is designed for networks with nonnegative arc lengths. The version of TREESORT that we use is capable of retrieving the optimal paths once the shortest distances have been calculated. However, since we are only calculating the entries in a distance table, we do not make use of this feature.[4] Finally, TREESORT does not possess any special features such as the node-specific requirements discussed earlier. TREESORT is considered the most computationally efficient in this class of algorithms (Dreyfus, 1969).

In order to provide even a rudimentary explanation of the TREESORT algorithm, it is necessary to define some terms (for a more detailed discussion of Dijkstra's algorithm, see Dijkstra, 1959; Denardo & Fox, 1979; Dreyfus, 1969). These terms are described in Table 2. TREESORT uses a "label-setting" scheme in which every node is assigned a label.

[4]Although this feature is not needed for updating the distance table, it proves to be very useful during the process of identifying and correcting errors in the network once the distance table has been calculated.

Table 2

Definitions

Term	Meaning
1. Origin	The root of the spanning tree.
2. Destination	Any node in the network other than the origin.
3. Distance	The sum of the arc lengths connecting any two nodes. A distance may or may not be the shortest-path distance.
4. Adjacent	An adjacent node is one that can be reached without passing any other node. An adjacent distance is the "non-stop" distance between two adjacent nodes.
5. Label	A value associated with each node containing the distance between it and the origin.
6. Current node	The reference point in the network from which the algorithm is currently operating. At the outset, the current node is the origin.
7. Predecessor	When traveling the route defined by the shortest path from the origin to some destination j, the predecessor to node j is the last node encountered before reaching j.

Once a node has been considered, its label will contain a distance between it and the origin. If this distance is known to be the shortest-path distance between the origin and that destination, the label is permanent. Otherwise, it is temporary.

TREESORT can be divided into two parts, the Initialization component and the Recursive component. These are described below:

Initialization Component

1. Assign every node in the network a label and a predecessor, which are both initially set to zero.

2. Choose one node as the origin and define it as the current node. Make its label permanent.

3. Determine the distance from the origin to every adjacent destination. For each of these adjacent nodes, store this distance as its label and store the origin as its predecessor.

4. Choose the adjacent destination with smallest label. Make its label permanent. This distance is the shortest-path distance from the origin to that destination. This destination now becomes the new current node.

Recursive Component

1. Determine the distance from the current node to all nodes which are adjacent to the current node. Add each of these distances to the shortest-path distance last determined (current node's label). Compare each sum to the current labels of each of these destinations. If the sum is less than the current node's label (or if that destination has not yet been considered), replace that destination's label with the summed distance and store the current node as that destination's predecessor.

2. Of all these adjacent destinations, choose the one with the smallest label. Make this label permanent. This stored distance is the shortest-path distance from the origin to this destination. This new destination now becomes the current node.

Stopping Rule

If all labels are permanent, stop. If at least one label is still temporary, return to step 1 (Recursive Component).

Example

This process is illustrated in the following example. Consider the network in Figure 1. It has four nodes and five bidirectional arcs. The arcs are labeled with their respective distances in miles. That is, d(1,2) = 2 miles, d(1,3) = 4 miles, etc. In addition, there is both a square and a triangle next to each node. For a given node, the square represents its label and the triangle represents its predecessor. Notice that both of these have been set to zero for each node. Also notice that there is an asterisk, which signifies a permanent label, next to the label for node 1. We will choose node 1 as the origin. The arrow identifies node 1 as the current node.

We have already performed steps 1 and 2 of the algorithm. Figure 2 represents the network after we have completed steps 3 and 4. Notice that the shortest distance from node 1 to node 2 is 2 miles and that node 2's label is now permanent. Also, node 2's predecessor is node 1. Node 2 is now the current node. The label for node 3 has a value of 4 but this is not necessarily the shortest distance from node 1 to node 3 because its label is not yet permanent.

In Figure 3, we see the network after steps 5 and 6. Node 3's label is now permanent and its predecessor is node 2 (i.e., the shortest distance from the origin to node 3 is 3 miles via node 2). The algorithm then returns to step 5. Eventually, all labels will be permanent and the process will have been completed. The final result is shown in Figure 4.

This process constitutes one run of the TREESORT algorithm and gives us the shortest-path distances between node 1 and all other nodes. The procedure is then repeated for each remaining node as an origin. When this is done, the distance table consisting of all from/to combinations is complete.

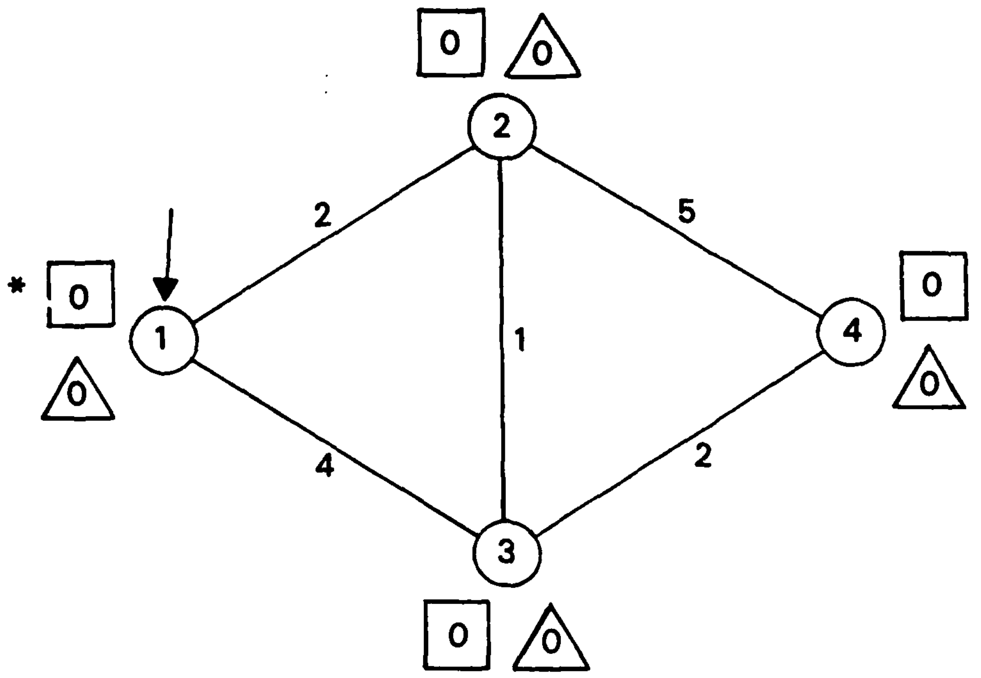

Figure 1. Initialization of example network.

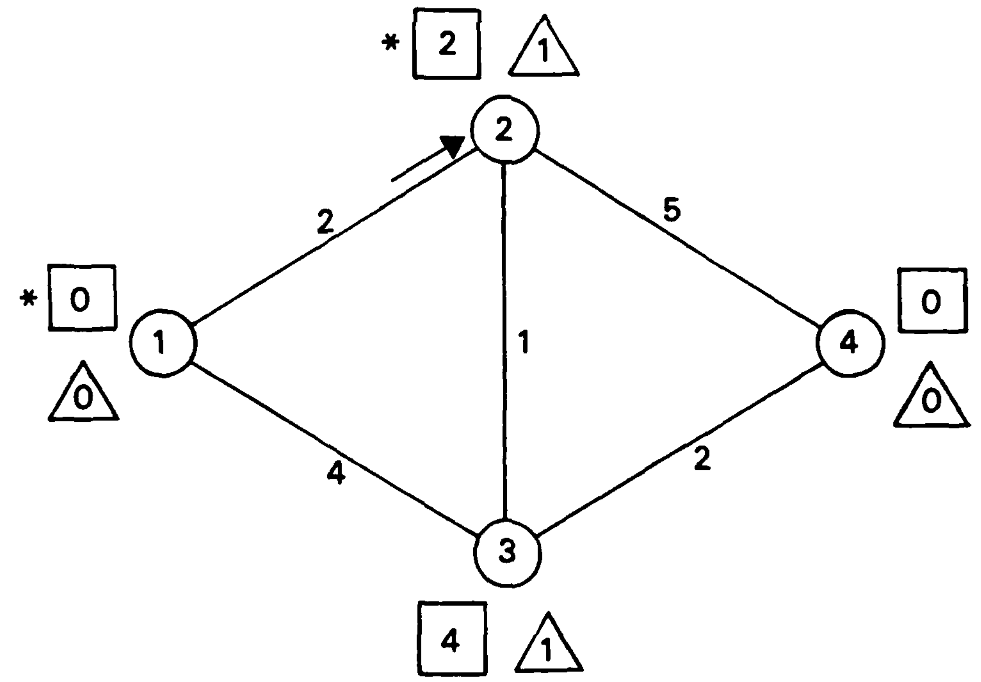

Figure 2. Example network after steps 3 and 4.

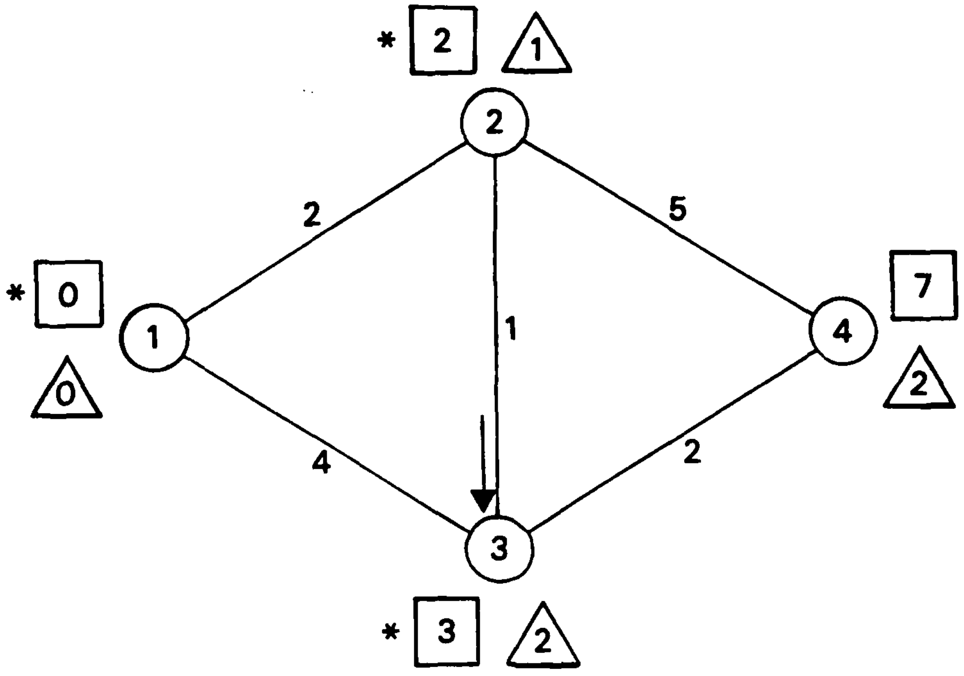

Figure 3. Example network after steps 5 and 6.

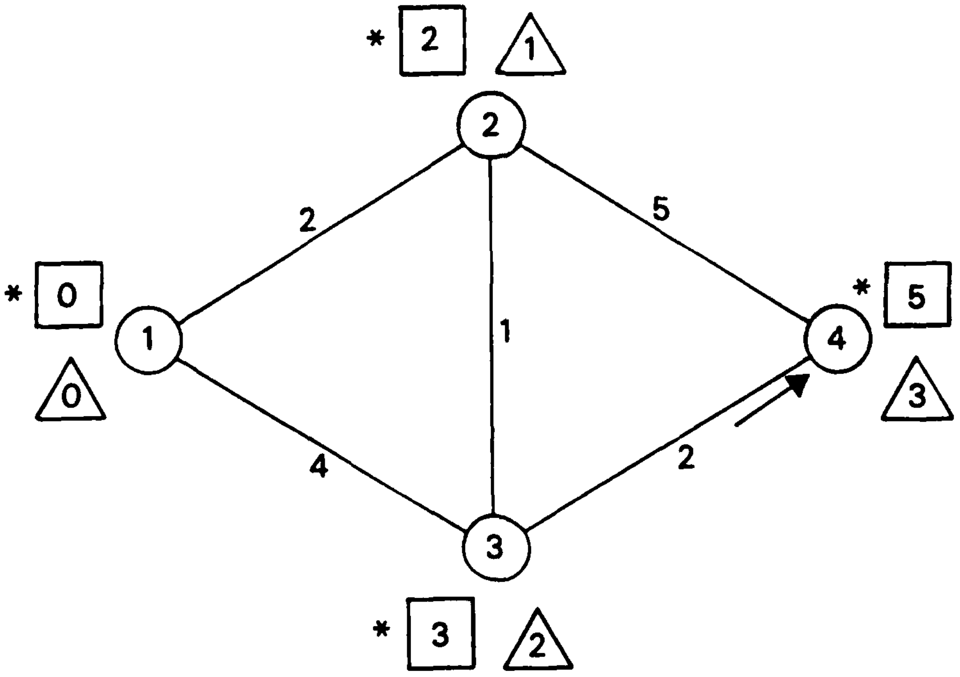

Figure 4. Example network after completion of algorithm.

RESULTS

The PCSMOD network (506 nodes, 3,036 arcs) was expanded to include the remaining CONUS duty stations. Of the 590 new nodes to be added, 45 could not be located on the road atlases. The remaining 545 new nodes, as well as a small number of intermediate nodes, were incorporated via 2,268 directional arcs. Since all distances are symmetric, we simply duplicate these mileages for the arc lengths in the opposite direction (i.e., once the distance from node i to node j is obtained, the distance from j to i is automatically known). The final network has 1,067 nodes and 7,572 arcs.

TREESORT

When using TREESORT to process this network, the elements of the distance matrix are filled one row at a time; each run of the algorithm results in the 1,066 distances d(i,j), where i represents the origin node and j represents all other. The shortest paths between node 1 and nodes 2 through 1,067 were calculated using TREESORT, requiring 19.4 seconds of CPU time. This process was repeated for each node in the network, requiring a total of 20,676 seconds of CPU time. This means that TREESORT required approximately 5.75 CPU hours to calculate the 1,137,422 (= 1066 x 1067) non-zero entries in the distance table.

Since the distance table is symmetric, twice the necessary number of computations were performed. TREESORT is designed for nonsymmetric arc lengths and cannot be easily adapted to the symmetric case. NETFLO, however, can make use of this feature because it is designed to calculate a single entry in the distance table, rather than an entire row, for each run of the model.

NETFLO

For comparison purposes, we needed to estimate the CPU time required by NETFLO to compute the new distance table. We did not use NETFLO to recalculate the entire distance table because this would have taken a very large amount of CPU time, the very problem we were trying to avoid. As an initial step, the distance from San Diego, California to Los Angeles, California (120 miles) was computed using NETFLO. This required 5.3 CPU seconds. The question then arose as to how sensitive this figure is to the distance between the chosen nodes. Therefore, NETFLO was used to calculate the shortest path between San Diego and Bangor, Maine. This distance was 3,229 miles and the elapsed time was 5.6 CPU seconds.

To test this hypothesis in a more rigorous fashion, we selected one duty station from each state in CONUS. San Diego was then chosen as the origin node. When NETFLO was used to calculate the shortest paths between the origin and each of these 49 state nodes (Alaska and Hawaii are excluded and Washington, DC is included), the total time required was 296.2 CPU seconds. It should be made clear here that the network was not reduced to one with only 49 nodes. It was left intact with all 1,067 nodes and 7,572 arcs. We simply performed 49 iterations of NETFLO on the entire network for various distances. In any case, these results indicate that NETFLO requires an average of 6.0 CPU seconds for each distance calculated.

The difference between the old and the new algorithms with respect to time requirements could now be estimated. NETFLO required an average of 6 CPU seconds to calculate a single entry in the distance table. The updated version of the network has 1,067 nodes implying 568,711 (=1067^2 - 1067)/2) unique entries in the distance table.

Therefore, if NETFLO were used to re-estimate the entire distance table, the total time required would be approximately 948 CPU hours. This figure contrasts with the 5.75 CPU hours that TREESORT required to accomplish the same task. Of course, if we do not replace the distances that had already been calculated in the original version of the model, the total time required by NETFLO could be reduced. The original distance table had 127,863 (=(506^2 -605)/2) unique entries. This leaves 440,848 new distances to be calculated, which would take a total of 735 CPU hours.

DISCUSSION

TREESORT is a special purpose algorithm designed to solve shortest-path problems. NETFLO is a general purpose algorithm designed to handle complex problems of pure network optimization. In a pure network model the flow along a specified arc can take any value. It is not restricted to being either 0 or 1, which is implicit in a shortest-path problem. In addition, networks will generally have multiple demand and/or supply nodes, both of which can vary from node to node. In a shortest-path problem, this is irrelevant because we are simply interested in minimizing the sum of arc lengths rather than the sum of arc lengths that are each multiplied by an associated flow.

As was previously mentioned, NETFLO was used in the developmental stages of PCSMOD because the widely accepted NETFLO software was already available at NAVPERSRANDCEN. The validity of formulating the PCS costing problem as a network model could be tested without first developing special purpose software to solve the shortest-path problem. Since the initial network was relatively small, computing efficiency was not an important issue. Only when NMPC requested model expansion to include additional duty stations was it necessary to research and apply an efficient shortest-path algorithm.

The network used initially in PCSMOD, upon which the existing entries of the distance table are based, was incomplete; more "direct" paths between any two given nodes may have appeared as a result of expanding the network with additional nodes and the arcs connecting them. This scenario is depicted in Figure 5, where Bridgeport is the new node added to the network. In this case, the pre-update distance between Fallon and China Lake is 575 miles, while the post-update distance is 365 miles.

To pursue this issue further, both versions of the distance table (pre- and post-update) were compared with the Army/Navy/Air Force Official Table of Distances (1981). The Official Distance Table (ODT) covers 878 locations in CONUS, Alaska, Hawaii, Canada, Canal Zone, Central America, Mexico, and Puerto Rico. However, the ODT does not contain every possible combination of origin and destination, reporting only the more commonly used distances. Of the 506 CONUS duty stations included in the original PCSMOD network, 367 were matched to locations in the ODT list. We extracted all non-zero entries in the ODT involving these 367 locations, yielding 63,956 distances with which entries in the pre- and post-update distance tables could be compared. Average absolute value percentage errors for the pre- and post-update distance tables were 3.04 percent and 2.07 percent, respectively. Therefore, the accuracy of the PCSMOD distance table improved approximately 30 percent when the ODT was taken as the benchmark.

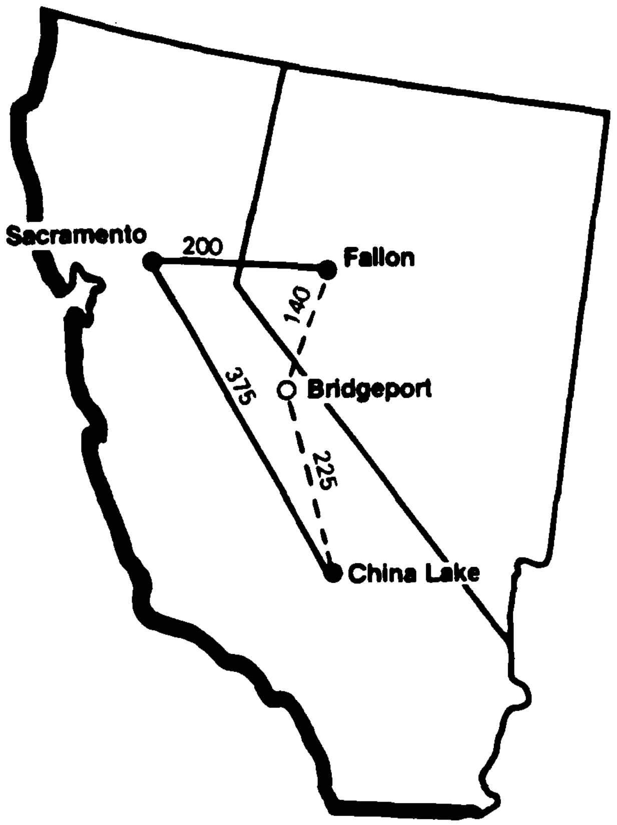

Figure 5. Adding a new node for Bridgeport, California to the network.

REFERENCES

Denardo, E. V., & Fox, B. L. (1979). Shortest-route methods: 1. Reaching, pruning, and buckets. Operations Research Quarterly, 27, 161-186.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphics. Numerische Mathematik, 1, 269-271.

Dreyfus, S. E. (1969). An appraisal of some shortest-path algorithms. Operations Research Quarterly, 17, 395-412.

Hoffman, W., & Pavley, R. (1959). A method for the solution of the Nth best path problem. Journal of the Association for Computing Machinery, 506-514.

Hu, T. C. (1968). A decomposition algorithm for the shortest paths in a network. Operations Research Quarterly, 16, 91-102.

Kennington, J. L., & Helgason, R. V. (1980). Algorithms for network programming. New York: Wiley.

Pollack, M., & Wiebenson, W. (1960). Solutions of the shortest-route problem--A review. Operations Research Quarterly, 8, 224-230.

Secretaries of the Army, Navy, and Air Force. (December 1981). Official table of distances--Continental United States, Alaska, Hawaii, Canada, Canal Zone, Central America, and Puerto Rico. Washington, DC: Departments of the Army, Navy, and Air Force.

Wong, D. C., Jerardo, A., & Nakada, M. (1984). Permanent change of station cost generation model (PCSMOD) (NPRDC Tech. Rep. 84-52). San Diego: Navy Personnel Research and Development Center.

DISTRIBUTION LIST

Chief of Naval Operations (OP-01B7) (2), (OP-987H)
Commander, Naval Military Personnel Command (NMPC-47)
Chief of Naval Research (Code 125), (Code 1111)
Commander, Army Research Institute for the Behavioral and Social Sciences, Alexandria (PERI-POT-I)
Commander, Air Force Human Resources Laboratory, Brooks Air Force Base (AFHRL/MO Manpower and Personnel Division), (TSRL/Technical Library FL 2870)
Superintendent, Naval Postgraduate School
Director of Research, U.S. Naval Academy
Institute for Defense Analyses, Science and Technology Division
Center for Naval Analyses
Canadian Forces Personnel, Applied Research Unit, Canada
DPSRSC, National Defense Headquarters, Canada
Ministry of Defense, Senior Psychologist, England (Naval)
D. Dennison, Army Personnel Research Establishment, Personnel Psychological Division, England (2)
Science 3 (RAF), Lacon House, England
1 Psychological Research Unit, NBH 3-44, Australia
Directorate of Psychology--AF, Department of Defense, Australia (Air for CE)
Navy Psychology, Australia (2)
Defense Psychology Unit, Defense HQ, New Zealand (2)
Dr. Regan
Defense Technical Information Center (DDAC) (2)