MEMORANDUM
RM-5539-PR
APRIL 1968

AN INTERDICTION MODEL FOR SPARSELY TRAVELED NETWORKS

Richard D. Wollmer

PREPARED FOR:
UNITED STATES AIR FORCE PROJECT RAND

The RAND Corporation
SANTA MONICA • CALIFORNIA

MEMORANDUM
RM-5539-PR
APRIL 1968

AN INTERDICTION MODEL FOR SPARSELY TRAVELED NETWORKS

Richard D. Wollmer

This research is supported by the United States Air Force under Project RAND — Contract No. F44620-67-C-0015 — monitored by the Directorate of Operational Requirements and Development Plans, Deputy Chief of Staff, Research and Development, Hq USAF. RAND Memoranda are subject to critical review procedures at the research department and corporate levels. Views and conclusions expressed herein are nevertheless the primary responsibility of the author, and should not be interpreted as representing the official opinion or policy of the United States Air Force or of The RAND Corporation.

The RAND Corporation
[illegible]

PREFACE

This Memorandum presents an algorithm for choosing locations to place air or ground forces in order to prevent an opposing force from proceeding through a transportation or supply network. It is a part of continuing RAND research on the effectiveness of interdiction and is particularly applicable to infiltration and counterinsurgency.

The model has been programmed for use on RAND's computer, and should also be useful to other organizations, including agencies interested in targeting strikes against lightly traveled line-of-communications networks.

SUMMARY

This Memorandum presents an algorithm for determining where to place forces in order to maximize the probability of preventing an opposing force from proceeding from one particular node in a network to another.

The usual gaming assumptions are invoked in this model; namely, that the strategy for placing forces is known to the opponent and that he will choose a path through the network which, based on this knowledge, maximizes his probability of successful traverse. As given quantities, the model requires a list of the arcs and nodes of the network, the number of forces available to stop the opposing force, and the probabilities for stopping the opposition at the arcs and nodes as functions of the number of forces placed there. From this data, the model calculates the probabilities for placing the force at the arcs and nodes when one force is available, and the expected numbers of forces to place at the arcs and nodes when multiple forces are available.

A computer program for the model has been written in Fortran IV. Though originally intended for the IBM 7044, it may be adapted to fit other computers. The program presently handles problems with up to 300 arcs, 150 nodes, and 25 forces. By changing the dimension statements, these quantities can be increased for larger computers, or their proportions can be modified for computers of the same size.

ACKNOWLEDGMENTS

The computer program contained in Appendix A was written by Steven Glaseman with the assistance of Richard Clasen. The author is extremely indebted to both of these men, and to Eugene Durbin, who originally suggested this area of research.

CONTENTS

PREFACE iii

SUMMARY v

ACKNOWLEDGMENTS vii

Section
I. INTRODUCTION 1

II. INTERCEPTOR'S PROBLEM 3

III. AN INCREMENTAL APPROACH FOR ONE INTERCEPTING FORCE 7

IV. MULTIPLE INTERCEPTING FORCES 11

Appendix
A. COMPUTER PROGRAM (FLOG) 17
B. EXAMPLE 27
C. CONVEXITY ASSUMPTION ON MULTIPLE FORCES 37
D. FINDING MAXIMUM PATH VALUE 39

REFERENCES 41

I. INTRODUCTION

For many years RAND has been interested in problems of interdiction and infiltration. Interdiction has been dealt with extensively from the standpoint of reducing the throughput capacity of a supply network. The most recent study owes most of its theoretical development to Wollmer (6), operating models to Durbin (1), and operational studies to J. W. Higgins. This approach is inadequate, however, in applying interdiction to infiltration and counterinsurgency. The reason is that requirements are so small under those circumstances that air and ground bombardment could never hope to reduce a network's throughput capacity below the level necessary to meet minimum requirements. For these situations, interdiction must therefore be approached from the standpoint of direct assault.

In the situation depicted here, an infiltrator attempts to proceed from one point to another in a network. An interceptor, who may possess one or more forces, attempts to stop him by placing forces along arcs or nodes that he expects the infiltrator to travel. His problem is to place his forces so as to minimize the infiltrator's probability of successful traverse. The infiltrator's problem is of course to select a path that will maximize this probability. Both problems can be represented by a zero-sum two-person game. As the next section will show, however, the game matrix is very large and difficult to generate. By using an incremental approach, a solution may be obtained much more easily for the interceptor.

This report presents a model based on such an approach. As inputs, it requires a list of the arcs and nodes of the network, the

number of forces available to stop the opposing force, and the probabilities for stopping the opposition at the arcs and nodes as functions of the number of forces placed there. From this information, the model calculates the expected number of forces to place at each arc and each node. For the case where only one intercepting force is available, the expected values are probabilities of force placement; for more than one force, expectations are to be interpreted in the most obvious way. Specifically, if the expected number of forces to place at a particular location is 4¼, one would place four forces there always and a fifth force one-fourth of the time.

The solutions obtained by this model are always optimal in the gaming sense (i.e., the infiltrator's best chance of successful traverse is as small as possible) if the interceptor has one force available, and are optimal or nearly optimal if he has more than one.

II. INTERCEPTOR'S PROBLEM

The network is characterized by sets of elements called arcs and nodes. Nodes are points or junctions, and arcs are line segments joining nodes. An arc that joins node i to node j is designated by the symbol (i,j). The node from which the infiltrator starts is called the source; the one he attempts to reach is called the sink.

It is assumed that the infiltrator and interceptor are each composed of a single force. Later on this restriction will be relaxed for the interceptor. As is customary in game theory, it is assumed that the interceptor's strategy will become known to the infiltrator. However, this does not necessarily mean that the infiltrator will know the location of the force. Specifically, if the interceptor decides to adopt a mixed strategy such as placing his force at node a with probability $\frac{1}{2}$ and at node b with probability $\frac{1}{2}$, the infiltrator will know the probabilities but will not know when the force will appear at a and when it will appear at b. The infiltrator will react by traveling only on source-to-sink paths that maximize his probability of successful traverse. The interceptor tries to choose a location for his force that will minimize that probability. He bases his choice on the topology of the network and upon the vulnerability of the arcs and nodes.

Note that mixed strategies are often necessary. To see this consider a network in which the source and sink are relatively invulnerable to attack and any other particular arc or node can be bypassed. In this situation, placing a force at any arc or node other than the source or sink with probability one would be ineffective since the

infiltrator would then choose a path that did not include the arc or node where the force was placed. Since placing the force at the source or sink would also be ineffective, no pure strategy would be desirable.

To develop a good mixed strategy, it is necessary to define quantities measuring the effectiveness of placing a force at the various arcs and nodes, and quantities expressing the interceptor's strategy. These are as follows:

$p(i)$ = probability that the infiltrator will be stopped at node i, given that he attempts to cross it and the interceptor chooses to place his force there

$p(i,j)$ = probability that the infiltrator will be stopped at arc (i,j) given that he attempts to cross it and the interceptor chooses to place his force there

$\pi(i)$ = probability that the interceptor places his force at node i

$\pi(i,j)$ = probability that the interceptor places his force at arc (i,j)

The quantities $p(i)$ and $p(i,j)$ may themselves include many elements and factors. For example, suppose it is determined that if node i is on the infiltrator's chosen path and the interceptor chooses to place his force there, there is a probability of 0.8 that the interceptor will arrive there before the infiltrator. Suppose further that if both were at node i, the interceptor's probability of detecting the infiltrator is 0.9, and if the two engaged in direct combat the probability of the interceptor's winning is 0.7. Then $p(i)$ would be the product of these three factors--0.8, 0.9, and 0.7--or .504.

The probability that the infiltrator can successfully cross a particular node, a_i, is $1 - \pi(a_i)p(a_i)$, and that of crossing a particular arc, (a_i,a_j) is $1 - \pi(a_i,a_j)p(a_i,a_j)$. Assuming these probabilities are independent* and the infiltrator attempts to reach the sink by the path $a_1, \ldots, a_n$, his probability of successful traverse is

$$(1) \quad K = \prod_{i=i}^{n} [1 - \pi(a_i) \times p(a_i)] \times \prod_{i=1}^{n} [1 - \pi(a_i,a_{i+1}) \times p(a_i,a_{i+1})].$$

This quantity, K, will be referred to as the value of the path $a_1, \ldots, a_n$. Given the p(i), p(i,j), $\pi(i)$, and $\pi(i,j)$, the infiltrator's problem is to find a source-sink path of maximum value. The interceptor's problem, given the p(i) and p(i,j), is to choose $\pi(i) \geq 0$ and $\pi(i,j) \geq 0$ such that the maximum value of all source-sink paths is minimized, subject to the constraint $\Sigma \pi(i) + \Sigma \pi(i,j) \leq 1$.

The problem of finding the $\pi(i)$ and $\pi(i,j)$ is really one in game theory. It can be represented by a game matrix and solved by linear programming. The linear program solution would include an optimal strategy for the infiltrator as well as the interceptor. The value of the game would be equal to both the infiltrator's maximum guaranteed probability of successful traverse, and one minus the interceptor's guaranteed probability of stopping the infiltrator. However, the game matrix would require a column for every possible source-sink path and a row for each of the interceptor's pure strategies. (The number of these pure strategies becomes exceedingly large when the interceptor is allowed multiple forces.) Hence, this formulation is impractical.

---

*While this assumption is not strictly true, it will be shown later that this leads to no inaccuracies for the one-intercepting-force case.

The incremental approach to be presented avoids these problems of enumeration and the resulting large number of variables for the linear program. Furthermore, it yields an optimal strategy when the interceptor has but one force, and an approximately optimal strategy when he has more than one force.

III. AN INCREMENTAL APPROACH FOR ONE INTERCEPTING FORCE

This section solves the interceptor's problem by a marginal analysis or steepest-ascent approach when the interceptor has but one force.

Specifically, the approach is as follows. Initially, all $\pi(i)$ and $\pi(i,j)$ are assigned values of zero. Then they are increased by small amounts $\Delta\pi(i) \geq 0$ and $\Delta\pi(i,j) \geq 0$. The proportions for the $\Delta\pi(i)$ and $\Delta\pi(i,j)$ are such that the additional force allocation, $\Sigma\, \Delta\pi(i) + \Sigma\, \Delta\pi(i,j)$, divided by the decrease in maximum path value, is minimized as these two quantities tend toward zero. This is equivalent to maximizing the decrease in maximum path value per unit of additional force allocation. The $\pi(i)$ and $\pi(i,j)$ are then increased again in the same manner until they sum up to one.

For a specific path, $a_1, \ldots, a_n$, increasing $\pi(a_i)$ by $\Delta\pi(a_i)$ replaces the factor $[1 - \pi(a_i)p(a_i)]$ in Eq. (1) by $[1 - (\pi(a_i) + \Delta\pi(a_i))p(a_i)]$ and reduces the value of the path from K to $K - \Delta K$, where

$$K - \Delta K = K\left\{\frac{1 - [\pi(a_i) + \Delta\pi(a_i)] \times [p(a_i)]}{1 - \pi(a_i) \times p(a_i)}\right\}.$$

Solving for the quotient of the additional force allocation and the decrease in path value, the following expression is obtained:

$$\frac{\Delta\pi(a_i)}{\Delta K} = \frac{1}{K}\left[\frac{1}{p(a_i)} - \pi(a_i)\right].$$

If force allocation were increased at an arc instead of a node, the same argument would yield an expression for the additional force

allocation divided by decrease in path value identical to the preceding one, with arc probabilities substituted for node probabilities. Specifically, for arc (a_i, a_{i+1}), this would be

$$\frac{\Delta\pi(a_i, a_{i+1})}{\Delta K} = \frac{1}{K}\left[\frac{1}{p(a_i, a_{i+1})} - \pi(a_i, a_{i+1})\right].$$

Note that in order to obtain any decrease in maximum path value, the $\Delta\pi(i)$ and $\Delta\pi(i,j)$ must be strictly greater than zero along a subset, C, of the arcs and nodes that intersects all paths of maximum value. An efficient allocation would require that all members of C remain on maximum value paths, for otherwise, lessening the force increase at an arc or node not on a maximum value path by a small amount would decrease $\Sigma\Delta\pi(i) + \Sigma\Delta\pi(i,j)$, while not affecting the maximum path value. Then, of course, this small amount of force could be redistributed among arcs and nodes on maximum value paths to get a strict improvement. Thus, the $\Delta\pi(i)$ and $\Delta\pi(i,j)$ assigned to the chosen C must be such that the ΔK's are equal for each of its members. In other words, they must be proportional to

(2) a) $\frac{1}{p(i)} - \pi(i)$

b) $\frac{1}{p(i,j)} - \pi(i,j)$

The constant $\frac{1}{K}$, where K is the maximum path value, was dropped in Expression (2), of course. The total increase in force allocation per unit decrease in the value of a maximum probability path would be:

$$(3) \qquad \frac{1}{K}\sum_{i\in C}\left[\frac{1}{p(i)} - \pi(i)\right] + \frac{1}{K}\sum_{(i,j)\in C}\left[\frac{1}{p(i,j)} - \pi(i,j)\right].$$

Thus, the problem is reduced to finding a subset of arcs and nodes intersecting all maximum probability paths that minimizes Expression (2).

Initially, all paths have value one since all $\pi(i)$ and $\pi(i,j)$ are zero. Thus C is required to be a set blocking all source-sink paths. If the nodes and arcs are assigned capacities equal to the quantities of Expression (2a) and (2b), the problem of finding C is reduced to one of finding a minimum cut.* However, this is equal to the value of the maximum source-sink flow and can be found by the maximum flow algorithm of Ref. 3.

Note that once additional force is allocated to the minimum cut, C, the capacities of all arcs and nodes in C decrease by the force increase while the capacities of all other arcs and nodes remain the same. Thus, the value of C will be reduced by $\Sigma\Delta\pi(i) + \Sigma\Delta\pi(i,j)$ while all other cuts will decrease by amounts that do not exceed this. Hence, C remains minimum and the entire unit of force may be allocated to C.

The $\pi(i)$ and $\pi(i,j)$ may therefore be solved for as follows:

1. Assign all nodes capacities of $1/p(i)$ and all arcs capacities of $1/p(i,j)$ (since all $\pi(i)$ and $\pi(i,j)$ are zero).
2. Maximize flow from source to sink.
3. Let C be the minimum cut set and V its value. Set $\pi(i) = p(i)/V$ for $i \in C$ and $\pi(i,j) = p(i,j)/V$ for $(i,j) \in C$. Set

*Essentially, a cut set is a set of arcs and nodes blocking all source-sink paths. Its value is the sum of the capacities of its arcs and nodes, and the minimum value of all cuts is equal to the maximum value of the source-sink flow.

all other $\pi(i)$ and $\pi(i,j)$ equal to zero.

Note that since only one cut is obtained, each path of maximum value has force allocated to only one of its arcs or nodes, and both its value and traverse probability are equal to the probability associated with that particular arc or node. Hence, the independence assumption does not lead to inaccuracies for the one-intercepting-force case. Finally, the maximum path value is $1 - \frac{1}{V}$, where V is the value of C found in step 3.

IV. MULTIPLE INTERCEPTING FORCES

The incremental procedure of the last section may be modified to handle the more general case in which there is more than one intercepting force. The quantities measuring the effectiveness of placing forces at the various arcs and nodes and those defining the interceptor's strategy must be redefined as follows:

$p(i)_k$ = probability that the infiltrator will be stopped at node i, given that he attempts to cross it and the interceptor has placed k of his forces there;

$p(i,j)_k$ = probability that the infiltrator will be stopped at arc (i,j), given that he attempts to cross it and the interceptor places k of his forces there;

$\pi(i)$ = expected number of forces the interceptor places at node i;

$\pi(i,j)$ = expected number of forces the interceptor places at arc (i,j),

where $p(i)_0$ and $p(i,j)_0$ are both identically zero. Also let

$$\tilde{\pi}(i) = \pi(i) - [\pi(i)]$$

$$\tilde{\pi}(i,j) = \pi(i,j) - [\pi(i,j)]$$

Note that $\tilde{\pi}(i)$ and $\tilde{\pi}(i,j)$ are merely the fractional parts of $\pi(i)$ and $\pi(i,j)$. It will be assumed that

$$p(i)_{k+1} - p(i)_k \leq p(i)_k - p(i)_{k-1}$$

$$p(i,j)_{k+1} - p(i,j)_k \leq p(i,j)_k - p(i,j)_{k-1}, \quad k \geq 1.$$

Mathematically, this is essentially a convexity assumption on the value of forces at the arcs and nodes. Physically, it expresses the fact

that multiple forces may result in overkill. This assumption assures that for a given value of $\pi(i)$, the probability associated with node i is minimized by allocating $[\pi(i)] + 1$ forces at i with probability $\tilde{\pi}(i)$, and $[\pi(i)]$ forces with probability $1 - \tilde{\pi}(i)$. In other words, if $\pi(i) = 3\frac{1}{2}$, $\pi(i)$ would be realized by assigning three forces at i half the time and four half the time, as opposed to such a policy as allocating two forces half the time and five half the time. A similar result holds for the arcs.* Thus, only policies of this type need be considered and the $\pi(i)$ and $\pi(i,j)$ completely define the interceptor's strategy.

As in Eq. (1) of Sec. II, the value of a path is still the product of the probabilities associated with its nodes and arcs; however, the probability associated with node i is now

$$\tilde{\pi}(i)(1 - p(i)_{[\pi(i)]+1}) + (1 - \tilde{\pi}(i))(1 - p(i)_{[\pi(i)]})$$

or

$$1 - p(i)_{[\pi(i)]} - \tilde{\pi}(i)(p(i)_{[\pi(i)]+1} - p(i)_{[\pi(i)]})$$

and that with arc (i,j) is

$$1 - p(i,j)_{[\pi(i,j)]} - \tilde{\pi}(i,j)(p(i,j)_{[\pi(i,j)]+1} - p(i,j)_{[\pi(i,j)]}).$$

If $a_1, \ldots, a_n$ is a path of value K, and $\pi(a_i)$ is increased by $\Delta\pi(a_i)$, reducing the path value to $K - \Delta K$, the expressions for K and $K - \Delta K$, as before, differ only in the factor for node a_i. Thus, for sufficiently small $\Delta\pi(a_i)$, the expression for the quotient of the force increase and path decrease is

*See Appendix C.

$$\frac{\Delta\pi(a_i)}{\Delta K} = \frac{1}{K}\left\{\frac{1 - p(a_i)_{[\pi(a_i)]}}{p(a_i)_{[\pi(a_i)]+1} - p(a_i)_{[\pi(a_i)]}} - \tilde{\pi}(a_i)\right\},$$

and decreasing path value by increasing force at an arc instead of a node yields the analogous expression,

$$\frac{\Delta\pi(a_i,a_j)}{\Delta K} = \frac{1}{K}\left\{\frac{1 - p(a_i,a_j)_{[\pi(a_i,a_j)]}}{p(a_i,a_j)_{[\pi(a_i,a_j)]+1} - p(a_i,a_j)_{[\pi(a_i,a_j)]}} - \pi(a_i,a_j)\right\}.$$

As in Sec. III, force must be allocated along all arcs of some cut set, and the problem is reduced to finding a minimum cut set, which in turn reduces to one of finding a maximum source-sink flow. The node and arc capacities are now

(4) a) $$c(i) = \frac{1 - p(i)_{[\pi(i)]}}{p(i)_{[\pi(i)]+1} - p(i)_{[\pi(i)]}} - \tilde{\pi}(i)$$

b) $$c(i,j) = \frac{1 - p(i,j)_{[\pi(i,j)]}}{p(i,j)_{[\pi(i,j)]+1} - p(i,j)_{[\pi(i,j)]}} - \tilde{\pi}(i,j).$$

Note that the $c(i)$ and $c(i,j)$ are no longer decreasing functions of $\pi(i)$ and $\pi(i,j)$ except over intervals whose endpoints possess the same integer part. Specifically, $c(i)$ is a decreasing function of $\pi(i)$ for $0 \leq \pi(i) < 1$, $1 \leq \pi(i) < 2$, etc. However, $c(i)$ increases at the points $\pi(i) = 1, 2$, etc. Thus, when a minimum cut set is found and additional force is allocated along its arcs and nodes, that cut remains minimum provided the forces on these arcs and nodes do not reach or exceed their next highest integer values. This requires that the incremental approach be modified as follows.

Starting with all $\pi(i)$ and $\pi(i,j)$ equal to zero, capacities are assigned to the arcs and nodes of the network which are equal to the quantities defined by Expression (3). Flow is maximized from source to sink and the $\pi(i)$ and $\pi(i,j)$ corresponding to the arcs and nodes of the minimum cut set are increased by amounts $\Delta\pi(i)$ and $\Delta\pi(i,j)$ which are proportional to their capacities. The constant of proportionality, M, is the smallest possible such constant that will either increase some $\pi(i)$ or $\pi(i,j)$ to the next highest integer or will increase $\Sigma\pi(i) + \Sigma\pi(i,j)$ to n. If the latter does not happen, the new values of $\pi(i)$ and $\pi(i,j)$ are used to calculate new capacities and flow is maximized again to obtain a new cut. The process is repeated until $\Sigma\pi(i) + \Sigma\pi(i,j) = n$.

Specifically, the algorithm for the general case of n intercepting forces is as follows:

1. Set all $\pi(i) = 0$ and all $\pi(i,j) = 0$.
2. Assign the arcs and nodes of the network capacities of c(i) and c(i,j) respectively, where c(i) and c(i,j) are as defined by Eq. (4).
3. Maximize flow from source to sink and let C be the arcs and nodes of the minimum cut set found.
4. Compute $M_1 = \min_{i \epsilon C} \dfrac{\{1 - \tilde{\pi}(i)\}}{c(i)}$

$$M_2 = \min_{(i,j)\epsilon C} \frac{\{1 - \tilde{\pi}(i,j)\}}{c(i,j)}$$

$$M_3 = \frac{n - \Sigma\pi(i) - \Sigma\pi(i,j)}{\sum_{i\epsilon C} c(i) + \sum_{(i,j)\epsilon C} c(i,j)}$$

5. Let $M = \min (M_1, M_2, M_3)$. Set $\Delta\pi(i) = Mc(i)$ if $i \epsilon C$ and $\Delta\pi(i,j) = Mc(i,j)$ if $(i,j)\epsilon C$. Set all other $\Delta\pi(i)$ and $\Delta\pi(i,j)$ equal to zero.

6. Set $\pi(i) = \pi(i) + \Delta\pi(i)$ and $\pi(i,j) = \pi(i,j) + \Delta\pi(i,j)$ for all i and (i,j). If $\Sigma\,\pi(i) + \Sigma\,\pi(i,j) < n$, go back to step 2. Otherwise terminate.

In step 4, M_1 is that constant of proportionality which would increase some $\pi(i)$ to an integer, M_2 the one which would increase some $\pi(i,j)$ to an integer, and M_3 the one which would increase $\Sigma\,\pi(i) + \Sigma\,\pi(i,j)$ to n.

At termination the $\pi(i)$ and $\pi(i,j)$ represent the expected number of forces to place at node i and arc (i,j), respectively. The policy will be to place $[\pi(i)] + 1$ forces at node i with probability $\tilde{\pi}(i)$ and $[\pi(i)]$ forces with probability $1 - \tilde{\pi}(i)$. The value of the resulting maximum probability path can be calculated by the algorithm of Appendix D.

If the iterations yield a sequence of but one minimum cut set (i.e., if steps 2 through 6 are performed only once), the solutions found are optimal, as in the case of one intercepting force, and also the independence assumption on the arc and node probabilities leads to no inaccuracies in maximum path value. However, if the procedure yields a sequence of several cuts, neither optimality nor independence can be guaranteed. Nevertheless, the nature of the strategy indicates that independence can be violated only by the fractional parts of the $\pi(i)$ and $\pi(i,j)$. The maximum path value can be kept track of during the course of the algorithm. Specifically, if K is the maximum path value at the beginning of an iteration, then the maximum path value at the end of the iteration is $K(1 - M)$, where M is that found in Step 5.* Of course, $K = 1$ at the beginning of the first iteration.

*See Appendix D.

Appendix A

COMPUTER PROGRAM (FLOG)*

A computer program for the model was written in Fortran IV for use on the IBM 7044. It can easily be adapted for use on other high speed computers. The main inputs are the total number of forces available to the interceptor, and, for each arc and node, his probability of stopping an infiltrator attempting to cross it as a function of the number of forces placed there. The outputs are the expected number of forces to place at the arcs and nodes and the infiltrator's maximum probability of successful traverse.

The program presently handles problems with up to 300 arcs, 150 nodes, and 25 forces. These capabilities can be modified to meet individual needs by changing the dimension statements.

In changing the dimension statements, note that the subscripts of the N1, N2, X, CA, PA, and DPA arrays, and the first subscript of the U array, are all equal to the maximum number of arcs. The subscripts of the NL1, NL2, Y, CN, PN, and DPN arrays, and the first subscript of the V array, are all equal to the maximum number of nodes. The maximum number of forces is equal to the second subscripts of the U and V arrays.

---

*The program of this section was written by Steve Glaseman and Richard Clasen.

FLOG

DATA SUBMISSION INSTRUCTIONS

I. Card 1:

Col.	Data
1-3	Total number of nodes.
4-6	Total number of arcs.
7-9	Number of source node.
10-12	Number of sink node.
13-15	Total number of forces involved.
16	Output flag. 0 = print output only at end of problem.
	1 = print output after each iteration.
17	Input flag. 0 = no data cases following,
	1 = case following present case.

II. Group 1:

Punch the number of the first node of each arc as follows:

a. Start in col. 1.
b. Three columns per entry.
c. Ten entries per card.

III. Group 2:

Punch the number of the last node of each arc in the same format as group 1.

IV. Group 3:

Punch the arc input probabilities as follows:

a. Start in col. 1.
b. Six columns per entry.
c. Ten entries per card.

As a guide, consider the following: with eight forces and ten arcs, group 3 would consist of eight cards, each with ten entries. The first card would contain arc 1 with 1 through 8 forces, and arc 2 with 1 and 2 forces, etc.

V. Group 4:

Punch the node input probabilities in the same format as group 3.

VI. Notes

1. All card entries are right-justified.
2. Data deck must adhere to group order in above instructions.

Each node must be identified by a number not to exceed three digits. Output may be requested at the end of each iteration or maximum-flow problem, or may be requested only at the conclusion of the entire problem. Multiple sets of data may be run. If this is done, a 0 must be placed in column 17 of the first card of the last case, and a 1 in column 17 of the first card of all other cases.

The output format and the program follow.

OUTPUT FORMAT

SOURCE	NET FLOW	SINK	MAX PATH VALUE
xxx	xx.xxxxx	xxx	x.xxxxx

NODE	FLOW	CUT SET	FORCES
xxx	x.xxxxx	x	x.xxxxx

ARC	FLOW	CUT SET	FORCES
xxx	x.xxxxx	x	x.xxxxx

A one in the cut set column indicates that that particular node or arc is in the minimum cut set and hence will have its number of forces increased. A zero indicates it is not in the cut set. The other headings are self-explanatory.

```
$IBFTC FLOG
C
C       TO EXAMINE GENERAL ATTACKER - EVADER NETWORK.
C       INPUTS = ATTACKER'S ARC AND NODE FORCE PLACEMENT PROB.
C
        DIMENSION N1(300), N2(300), U(300,25), V(150,25), X(300),
       1Y(150), CA(300), NL1(150), NL2(150), OBJ(1), NCS(150), ACS(50,50),
       2PN(150), CN(150), PA(300), DPN(150), DPA(300)
        INTEGER   ACS
C
C       INITIALIZATION.
C
   18   DO 19 I = 1,50
        Y(I) = 0.
        NL1(I) = 0
        NL2(I) = 0
        NCS(I) = 0
        DPN(I) = 0.
        DO 20 J = 1,50
   20   ACS(I,J) = 0
   19   CONTINUE
        DO 21 I = 1,100
        PA(I) = 0.
        PN(I) = 0.
        CA(I) = 0.
   21   CN(I) = 0.
        DO 22 I = 1,150
        X(I) = 0.
        DPA(I) = 0.
   22   CONTINUE
        PV = 1.
C
C       READ INPUT PARAMETERS.
C
        READ 1000, M,N,ISRC,ISNK,IF,IFLAG,JFLAG
        READ 2000, (N1(I), I = 1,N)
        READ 2000, (N2(I), I = 1,N)
        READ 3000, ((U(I,J),J = 1,IF),I = 1,N)
        READ 3000, ((V(I,J),J = 1,IF),I = 1,M)
        F = FLOAT(IF)
C
C       M      = NUMBER OF NODES.
C       N      = NUMBER OF ARCS.
C       ISRC   = SOURCE NODE.
C       ISNK   = SINK NODE.
C       IF     = NUMBER OF FORCES INVOLVED.
C       IFLAG = PRINT FLAG.  0 = END PRINT ONLY, 1 = RUNNING ACCOUNT.
C       JFLAG = INPUT FLAG. 0 = 1 DATA SET, 1 = MULTIPLE DATA SETS.
C       N1(I) = ID OF FIRST NODE OF ARC I.
C       N2(I) = ID OF LAST NODE OF ARC I.
C       U(I,J)= PROBAB.  OF ARCS
C       V(I,J)= PROBAB.  OF NODES
C
C       CHANGE ARC AND NODE PROB. INTO CAPACITIES.
C
        DO 1  J = 1,N
    1   CA(J) = 1./U(J,1)
        DO 2  J = 1,M
    2   CN(J) = 1./V(J,1)
C
```

```
C       SEND COLUMN TO ALGORITHM.
C
  100 DO 101 I = 1,M
  101 NCS(I) = 0
      DO 102 I = 1,N
      N1I = N1(I)
      N2I = N2(I)
  102 ACS(N1I,N2I) = 0
      CALL FLOWMX(M,N,ISRC,ISNK,N1,N2,CA,X,CN,Y,NL1,NL2,OBJ)
C
C       UPON RETURN, FIND ARCS AND NODES IN CUT SET.
C
      DO 3 I = 1,M
    3 IF(NL1(I).EQ.0.AND.NL2(I).NE.0) NCS(I) = 1
      DO 4 I = 1,N
      N1I = N1(I)
      N2I = N2(I)
      IF(NL2(N1I).EQ.0.AND.NL1(N2I).NE.0) ACS(N1I,N2I) = 1
    4 IF(NL2(N2I).EQ.0.AND.NL1(N1I).NE.0) ACS(N1I,N2I) = 1
C
C       CALCULATE MOST LIMITING ARCS AND NODES IN CUT SET.
C
      SM2 = 9999.
      DO 5 I = 1,N
      N1I = N1(I)
      N2I = N2(I)
      IF(ACS(N1I,N2I).NE.1) GO TO 5
C
      AM2=((FLOAT(IFIX(PA(I)+.00002))+1.)-PA(I))/CA(I)
      IF(AM2.GE.SM2) GO TO 5
      SM2 = AM2
    5 CONTINUE
C
C       SM2 IS NOW MOST LIMITING ARC IN CUT SET.
C
      SM1 = 9999.
      DO 6 I = 1,M
      IF(NCS(I).NE.1) GO TO 6
      AM1=((FLOAT(IFIX(PN(I)+.00002))+1.)-PN(I))/CN(I)
      IF(AM1.GE.SM1) GO TO 6
      SM1 = AM1
    6 CONTINUE
C
C       SM1 IS NOW MOST LIMITING NODE IN CUT SET.
C
      SUMNP = 0.
      SUMNC = 0.
      DO 7 I = 1,M
      SUMNP = SUMNP + PN(I)
      IF (NCS(I).NE.1) GO TO 7
      SUMNC = SUMNC + CN(I)
    7 CONTINUE
      SUMAP = 0.
      SUMAC = 0.
      DO 8 I = 1,N
      SUMAP = SUMAP + PA(I)
      N1I = N1(I)
      N2I = N2(I)
      IF(ACS(N1I,N2I).NE.1) GO TO 8
      SUMAC = SUMAC + CA(I)
```

```
    8 CONTINUE
C
      SM3 =          ( 1 - SUMNP - SUMAP) /(SUMNC + SUMAC)
C
C     FIND MOST LIMITING M OF ALL M'S.
C
      ALIM = AMIN1(SM1,SM2,SM3)
C
C     CALCULATE MAXIMUM PATH VALUE.
C
      PV = PV * (1. - ALIM)
C
C     INCREMENT DELTA-PI ARRAYS.
C
      DO 9  I = 1,M
      IF(NCS(I).NE.1) GO TO 9
      PN(I) = PN(I) + (CN(I) * ALIM)
    9 CONTINUE
      DO 10 I = 1,N
      N1I = N1(I)
      N2I = N2(I)
      IF(ACS(N1I,N2I).NE.1) GO TO 10
      PA(I) = PA(I) + (CA(I) * ALIM)
   10 CONTINUE
C
C     TEST FOR DONE. IF NOT, COMPUTE NEW CAPACITIES. PRINT STATE.
C
      SPNI = 0.
      DO 201  I = 1,M
  201 SPNI = SPNI + PN(I)
      SPAI = 0.
      DO 202  I = 1,N
  202 SPAI = SPAI + PA(I)
      TEST = SPNI + SPAI + .0002
      IF(TEST.GE.F.OR.IFLAG.EQ.1) GO TO 13
  200 DO 11 I = 1,M
      IF(NCS(I).NE.1) GO TO 11
      J = IFIX(PN(I) + .0002)
      IF(J.EQ.0) GO TO 300
C
C
      CN(I) = ((1.-V(I,J)) / (V(I,J+1)-V(I,J))) - PN(I)+FLOAT(J)
      GO TO 11
  300 CN(I) = CN(I) - PN(I)
C
   11 CONTINUE
      DO 12  I = 1,N
      N1I = N1(I)
      N2I = N2(I)
      IF(ACS(N1I,N2I).NE.1) GO TO 12
      J = IFIX(PA(I) + .0002)
      IF(J.EQ.0) GO TO 400
C
      CA(I) = ((1.-U(I,J)) / (U(I,J+1)-U(I,J))) - PA(I)+FLOAT(J)
      GO TO 12
  400 CA(I) = CA(I) - PA(I)
C
   12 CONTINUE
      GO TO 100
C
```

```
C
   13 WRITE(6,4000)
      WRITE(6,4001) ISRC, OBJ, ISNK, PV
      WRITE(6,4002)
      DO 14 I = 1,M
   14 WRITE(6,4004) I, Y(I), ACS(I), PN(I)
      WRITE(6,4005)
      DO 15  I = 1,M
      N1I = N1(I)
      N2I = N2(I)
      IF(X(I).LT.0.) GO TO 16
      WRITE(6,4003) N1I, N2I, X(I), ACS(N1I,N2I), PA(I)
   15 CONTINUE
      GO TO 17
   16 XI = X(I)
      AVX= ABS(XI)
      WRITE(6,4003) N2I, N1I, AVX,  ACS(N1I,N2I), PA(I)
      GO TO 15
   17 IF(TEST.LT.F) GO TO 200
C
C     DONE, GET OFF MACHINE.
C
      WRITE(6,5000)
      IF(JFLAG.EQ.1) GO TO 18
      CALL EXIT
C
C     FORMATS.
C
 1000 FORMAT(4I3,I3,2I1)
 2000 FORMAT(10I3)
 3000 FORMAT(10F6.4)
 4000 FORMAT(1H1,53X,6HSOURCE,3X,8HNET FLOW,3X,4HSINK,25X,15HMAX. PATH V
     1ALUE)
 4001 FORMAT(1HO,54X,I3,3X,F12.5,3X,I3,25X,F12.5//)
 4002 FORMAT(1H ,42X, 4HNODE,10X,4HFLOW,10X,7HCUT SET, 7X,6HFORCES//)
 4003 FORMAT(1H ,40X, I3,1H,,I3,3X,F12.5,11X,I2,3X,F12.5)
 4004 FORMAT(1H ,42X, I3,5X,F12.5,11X,I2,3X,F12.5)
 4005 FORMAT(1HO,// 44X, 3HARC,10X,4HFLOW,10X,7HCUT SET, 7X,6HFORCES//)
 5000 FORMAT (1HO, 62X, 14HEND OF PROBLEM)
      END
$IBFTC FLOWMX
       SUBROUTINE FLOWMX(NODES,ARCS,SRC,SNK,I,J,HI,FLOW,NC,NF,NA,NB,OBJ)
       INTEGER NODES,ARCS,SNK,SRC,I(2000),J(2000),HI(2000),FLOW(2000)
       INTEGER NC(1000),NF(1000),NA(1000),NB(1000),OBJ
C
      ACHG(K) = FLOAT(K)/E1
      ACHG(X) = E1 *AMOD(X,E2)
      E1 = 2.**10
      E2 = 2.**24
      EQUIVALENCE(I,P)
      DO 1 JI= 1,ARCS
      L = HI(JI)
    1 HI(JI)= ACHG(P)
      DO 2 II= 1,NODES
      L = NC(II)
    2 NC(II) = ACHG(P)
      CALL FLOWM(NODES,ARCS,SRC,SNK,I,J,HI,FLOW,NC,NF,NA,NB,OBJ)
      DO 3 JI= 1,ARCS
      P = ACHG(HI(JI))
      HI(JI)= L
```

```
      P = BCHG(FLOW(JI))
    3 FLOW(JI)= L
      DO 4 II= 1,NODES
      P = BCHG(NF(II))
      NF(II)= L
      P= BCHG(NC(II))
    4 NC(II)= L
      P = BCHG(OBJ)
      OBJ = L
      RETURN
      END
$IDFIC FLOWM
       SUBROUTINE FLOWM (NODES,ARCS,SRC,SNK,I,J,HI,FLOW,NC,NF,NA,NB,OBJ)
       INTEGER NODES,ARCS,SNK,SRC,I(2000),J(2000),HI(2000),FLOW(2000)
       INTEGER NC(1000),NF(1000),NA(1000),NB(1000),OBJ
C
C
C     DEFINITION OF CALLING SEQUENCE
C
C
C    NAME    USE
C
C    NODES   NUMBER OF NODES
C    ARCS    NUMBER OF ARCS
C    SRC     SOURCE NODE
C    SNK     SINK NODE
C    I       LIST OF FIRST NODES
C    J       LIST OF SECOND NODES
C    HI      UPPER BOUNDS FOR ARCS
C    FLOW    AMOUNT OF FLOW IN ARCS
C    NC      NODE CAPACITY  (INPUT)
C    NF      NODE FLOW
C    NA      SOURCE NODE LABELS
C    NB      SINK NODE LABELS
C    OBJ     OBJECTIVE VALUE
C
C   BEGIN
       INTEGER A,AA,N,N1,N2,GTARCS,INC,LABEL
        LOGICAL TYPE
       GTARCS = ARCS +1
       DO 10 A =1,ARCS
         FLOW(A) = 0
         IF (I(A).LE.0.OR.I(A).GT.NODES) GO TO 999
         IF (J(A).LE.0.OR.J(A).GT.NODES) GO TO 999
         IF (I(A).EQ.J(A)) GO TO 999
         HI(A) = IABS(HI(A))
   10 CONTINUE
       DO 20 N = 1,NODES
         NF(N) = 0
         NC(N) = IABS(NC(N))
   20 CONTINUE
       OBJ = 0
C ZERO NODE LABELS
  100 DO 200 N =1,NODES
         NA(N) =0
         NB(N) =0
  200 CONTINUE
C LABEL SOURCE NODE
       NB(SRC) = 1
       IF (NC(SRC).EQ.NF(SRC)) RETURN
```

```
      NA(SRC) = GTARCS
C LABEL
  210 LABEL = 0
      DO 250 A=1,ARCS
        N1 = I(A)
        IF (N1.LT.0) GO TO 250
        N2 = J(A)
        IF (FLOW(A).LT.0) GO TO 225
        IF (NA(N1).NE.0) GO TO 220
        IF (NB(N2).EQ.0) GO TO 250
        IF (FLOW(A).EQ.0) GO TO 224
        NA(N1) = -A
        IF (NB(N1).NE.0) GO TO 240
        NB(N1) = -GTARCS
        GO TO 240
  220   IF (NB(N2).NE.0.OR.FLOW(A).EQ.HI(A)) GO TO 245
        NB(N2) = A
        IF (NF(N2).EQ.NC(N2)) GO TO 240
        NA(N2) = GTARCS
        GO TO 240
  224   IF (NA(N2).EQ.0) GO TO 250
        GO TO 226
  225   IF (NA(N2).EQ.0) GO TO 230
  226   IF (NB(N1).NE.0.OR.FLOW(A).EQ.(-HI(A))) GO TO 245
        NB(N1) = -A
        IF (NF(N1).EQ.NC(N1)) GO TO 240
        NA(N1) = +GTARCS
        GO TO 240
  230   IF (NB(N1).EQ.0) GO TO 250
        NA(N2) = +A
        IF (NB(N2).NE.0) GO TO 240
        NB(N2) = -GTARCS
  240   LABEL = 1
        IF (NA(SNK).NE.0) GO TO 260
  245   I(A) = -N1
  250 CONTINUE
C   GO BACK AND LABEL MORE IF SOME NODE WAS LABELED ON LAST LOOP
      IF (LABEL.NE.0) GO TO 210
C  RESTORE POSITIVE SIGNS TO FIRST NODE LIST
  260 DO 270 A = 1,ARCS
        I(A) = IABS(I(A))
  270 CONTINUE
C   IF NOTHING LABELED ON LAST LOOP, DONE
      IF (LABEL.EQ.0) RETURN
C  BREAKTHRU,  FIND THE INCREMENT
  300 INC = NC(SRC)-NF(SRC)
C   FOLLOW PATH BACK FROM SINK
      N = SNK
      TYPE =.TRUE.
  310 AA= NA(N)
      IF (.NOT.TYPE) AA= NB(N)
      A = IABS(AA)
      IF (A.GT.ARCS) GO TO 320
      IF (AA.LT.0) GO TO 315
      N2 = I(A)
      IF (FLOW(A).LT.0)  GO TO 316
      INC = MIN0(INC,HI(A)-FLOW(A))
      GO TO 318
  315 N2 = J(A)
      IF (FLOW(A).LE.0) GO TO 317
```

```
      INC = MIN0(INC,IABS(FLOW(A)))
      GO TO 318
  317 INC = MIN0(INC,FLOW(A)+HI(A))
  318 N = N2
      GO TO 340
  320 IF (TYPE) GO TO 325
      INC = MIN0(INC,FI(N))
      GO TO 340
  325 INC = MIN0(INC,NC(N)-NF(N))
  340 TYPE = .NOT.TYPE
      IF (N.NE.SRC) GO TO 310
C  INCREMENT ARCS
      N = SNK
      TYPE =.TRUE.
      OBJ = OBJ + INC
  350 AA = NC(N)
      IF (TYPE) AA = SA(N)
      A  = IABS(AA)
      IF (A.GT.ARCS) GO TO 365
      IF (AA.LT.0) GO TO 355
      FLOW(A) = FLOW(A) + INC
      N = I(A)
      GO TO 370
  355 FLOW(A) = FLOW(A) - INC
      N = J(A)
      GO TO 370
  365 IF (TYPE) GO TO 368
      NF(N) = NF(J) - INC
      GO TO 370
  368 NF(N) = NF(N) + INC
  370 TYPE =.NOT.TYPE
      IF (N.NE.SRC.OR.TYPE) GO TO 350
C  FLOW INCREMENTED, RETURN TO LABELING
      GO TO 100
  999 PRINT 998,A,I(A),J(A)
  998 FORMAT(48H ARCS INCORRECTLY SETUP IN NETWORK ROUTINE, ARC I6,3H =(
     X16,1H,I6,1H))
      RETURN
      END
```

Appendix B

EXAMPLE

For illustrative purposes, the algorithm is used to find the optimal placement of forces for the network of Fig. 1, with output printed after each iteration. Ten forces are available, and all arcs and nodes have identical probabilities associated with them. Thus, for each node, i, and each arc (i,j), we have:

$p(i)_1 = .6666$	$p(i,j)_1 = .6666$
$p(i)_2 = .8332$	$p(i,j)_2 = .8332$
$p(i)_3 = .8748$	$p(i,j)_3 = .8748$
$p(i)_4 = .8852$	$p(i,j)_4 = .8852$
$p(i)_5 = .8878$	$p(i,j)_5 = .8878$
$p(i)_6 = .8884$	$p(i,j)_6 = .8884$
$p(i)_7 = .8886$	$p(i,j)_7 = .8886$
$p(i)_8 = .8887$	$p(i,j)_8 = .8887$
$p(i)_9 = .8888$	$p(i,j)_9 = .8888$
$p(i)_{10} = .8889$	$p(i,j)_{10} = .8889$

The input deck is given in Fig. 2 and the output deck or results follow.

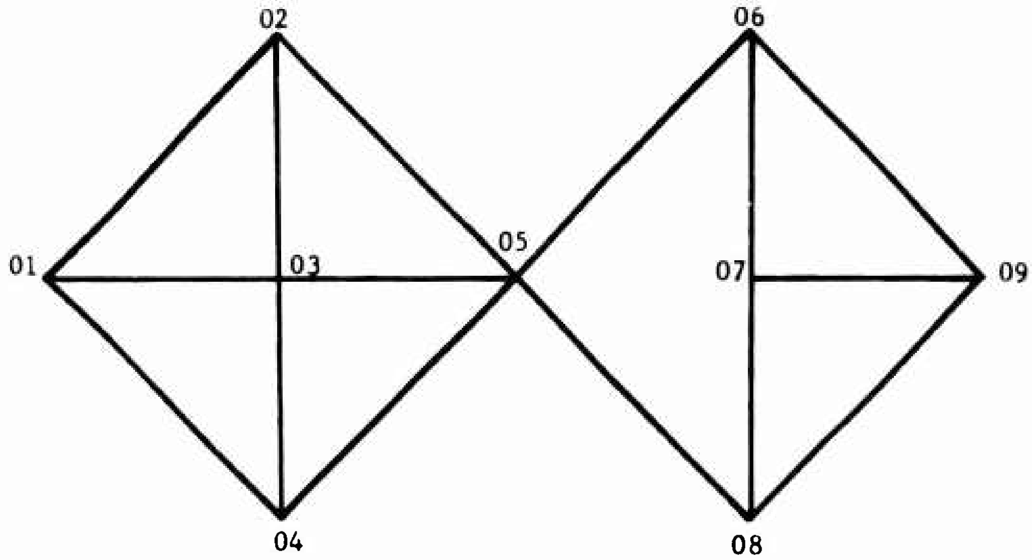

Fig. 1--Sample problem network

```
0090150010090101
001001001002002003003004005005
006006007007008
002003004003005004005005006008
007009008009009
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
0.66660.83320.87480.88520.88780.88840.88860.88870.88880.8889
```

Fig. 2--Input deck

OUTPUT DECK

SOURCE	NET FLOW	SINK	MAX. PATH VALUE
1	1.50000	9	0. 3333

NODE	FLOW	CUT SET	FORCES
1	1.50000	1	1.00000
2	1.50000	0	0.00000
3	0.00000	0	0.00000
4	0.00000	0	0.00000
5	1.50000	0	0.00000
6	1.50000	0	0.00000
7	0.00000	0	0.00000
8	0.00000	0	0.00000
9	1.50000	0	0.00000

ARC	FLOW	CUT SET	FORCES
1, 2	1.50000	0	0.00000
1, 3	0.00000	0	0.00000
1, 4	0.00000	0	0.00000
2, 3	0.00000	0	0.00000
2, 5	1.50000	0	0.00000
3, 4	0.00000	0	0.00000
3, 5	0.00000	0	0.00000
4, 5	0.00000	0	0.00000
5, 6	1.50000	0	0.00000
5, 8	0.00000	0	0.00000
6, 7	0.00000	0	0.00000
6, 9	1.50000	0	0.00000
7, 8	0.00000	0	0.00000
7, 9	0.00000	0	0.00000
8, 9	0.00000	0	0.00000

SOURCE	NET FLOW	SINK	MAX. PATH VALUE
1	1.50000	9	0.11111

ARC	FLOW	CUT SET	FORCES
1	1.50000	0	1.00000
2	1.50000	0	0.00000
3	0.00000	0	0.00000
4	0.00000	0	0.00000
5	1.50000	1	1.00000
6	1.50000	0	0.00000
7	0.00000	0	0.00000
8	0.00000	0	0.00000
9	1.50000	0	0.00000

ARC	FLOW	CUT SET	FORCES
1, 2	1.50000	0	0.00000
1, 3	0.00000	0	0.00000
1, 4	0.00000	0	0.00000
2, 3	0.00000	0	0.00000
2, 5	1.50000	0	0.00000
3, 4	0.00000	0	0.00000
3, 5	0.00000	0	0.00000
4, 6	0.00000	0	0.00000
5, 6	1.50000	0	0.00000
5, 8	0.00000	0	0.00000
6, 7	0.00000	0	0.00000
7, 9	1.50000	0	0.00000
7, 8	0.00000	0	0.00000
7, 9	0.00000	0	0.00000
8, 9	0.00000	0	0.00000

SOURCE	NET FLOW	SINK	MAX. PATH VALUE
1	1.50000	9	0.03704

NODE	FLOW	CUT SET	FORCES
1	1.50000	0	1.00000
2	1.50000	0	0.00000
3	0.00000	0	0.00000
4	0.00000	0	0.00000
5	1.50000	0	1.00000
6	1.50000	0	0.00000
7	0.00000	0	0.00000
8	0.00000	0	0.00000
9	1.50000	1	1.00000

ARC	FLOW	CUT SET	FORCES
1, 2	1.50000	0	0.00000
1, 3	0.00000	0	0.00000
1, 4	0.00000	0	0.00000
2, 3	0.00000	0	0.00000
2, 5	1.50000	0	0.00000
3, 4	0.00000	0	0.00000
3, 5	0.00000	0	0.00000
4, 5	0.00000	0	0.00000
5, 6	1.50000	0	0.00000
5, 8	0.00000	0	0.00000
6, 7	0.00000	0	0.00000
6, 9	1.50000	0	0.00000
7, 8	0.00000	0	0.00000
7, 9	0.00000	0	0.00000
8, 9	0.00000	0	0.00000

SOURCE	NET FLOW	SINK	MAX. PATH VALUE
1	2.00098	9	0.01853

NODE	FLOW	CUT SET	FORCES
1	2.00098	1	2.00000
2	1.50000	0	0.00000
3	0.50098	0	0.00000
4	0.00000	0	0.00000
5	2.00098	0	1.00000
6	1.50000	0	0.00000
7	0.50098	0	0.00000
8	0.50098	0	0.00000
9	2.00098	0	1.00000

ARC	FLOW	CUT SET	FORCES
1, 2	1.50000	0	0.00000
1, 3	0.50098	0	0.00000
1, 4	0.00000	0	0.00000
2, 3	0.00000	0	0.00000
2, 5	1.50000	0	0.00000
3, 4	0.00000	0	0.00000
3, 5	0.50098	0	0.00000
4, 5	0.00000	0	0.00000
5, 6	1.50000	0	0.00000
5, 8	0.50098	0	0.00000
6, 7	0.00000	0	0.00000
6, 9	1.50000	0	0.00000
8, 7	0.50098	0	0.00000
7, 9	0.50098	0	0.00000
8, 9	0.00000	0	0.00000

SOURCE	NET FLOW	SINK	MAX. PATH VALUE
1	2.00098	9	0.00927

NODE	FLOW	CUT SET	FORCES
1	2.00098	0	2.00000
2	1.50000	0	0.00000
3	0.50098	0	0.00000
4	0.00000	0	0.00000
5	2.00098	1	2.00000
6	1.50000	0	0.00000
7	0.50098	0	0.00000
8	0.50098	0	0.00000
9	2.00098	0	1.00000

ARC	FLOW	CUT SET	FORCES
1, 2	1.50000	0	0.00000
1, 3	0.50098	0	0.00000
1, 4	0.00000	0	0.00000
2, 3	0.00000	0	0.00000
2, 5	1.50000	0	0.00000
3, 4	0.00000	0	0.00000
3, 5	0.50098	0	0.00000
4, 5	0.00000	0	0.00000
5, 6	1.50000	0	0.00000
5, 8	0.50098	0	0.00000
6, 7	0.00000	0	0.00000
6, 9	1.50000	0	0.00000
8, 7	0.50098	0	0.00000
7, 9	0.50098	0	0.00000
8, 9	0.00000	0	0.00000

SOURCE	NET FLOW	SINK	MAX. PATH VALUE
1	2.00098	9	0.00464

NODE	FLOW	CUT SET	FORCES
1	2.00098	0	2.00000
2	1.50000	0	0.00000
3	0.50098	0	0.00000
4	0.00000	0	0.00000
5	2.00098	0	2.00000
6	1.50000	0	0.00000
7	0.50098	0	0.00000
8	0.50098	0	0.00000
9	2.00098	1	2.00000

ARC	FLOW	CUT SET	FORCES
1, 2	1.50000	0	0.00000
1, 3	0.50098	0	0.00000
1, 4	0.00000	0	0.00000
2, 3	0.00000	0	0.00000
2, 5	1.50000	0	0.00000
3, 4	0.00000	0	0.00000
3, 7	0.50098	0	0.00000
4, 5	0.00000	0	0.00000
5, 6	1.50000	0	0.00000
5, 8	0.50098	0	0.00000
6, 7	0.00000	0	0.00000
6, 9	1.50000	0	0.00000
7, 8	0.50098	0	0.00000
7, 9	0.50098	0	0.00000
8, 9	0.00000	0	0.00000

SOURCE	NET FLOW	SINK	MAX. PATH VALUE
1	3.00000	9	0.00155

NODE	FLOW	CUT SET	FORCES
1	3.00000	0	2.00000
2	1.50000	0	0.00000
3	1.50000	0	0.00000
4	0.00000	0	0.00000
5	3.00000	0	2.00000
6	1.50000	0	0.00000
7	1.50000	0	0.00000
8	1.50000	0	0.00000
9	3.00000	0	2.00000

ARC	FLOW	CUT SET	FORCES
1, 2	1.50000	0	0.00000
1, 3	1.50000	0	0.00000
1, 4	0.00000	0	0.00000
2, 3	0.00000	0	0.00000
2, 5	1.50000	0	0.00000
3, 4	0.00000	0	0.00000
3, 5	1.50000	0	0.00000
4, 9	0.00000	0	0.00000
5, 6	1.50000	1	1.00000
5, 8	1.50000	1	1.00000
6, 7	0.00000	0	0.00000
6, 9	1.50000	0	0.00000
8, 7	1.50000	0	0.00000
7, 9	1.50000	0	0.00000
8, 9	0.00000	0	0.00000

SOURCE	NET FLOW	SINK	MAX. PATH VALUE
1	3.00000	9	0.00052

NODE	FLOW	CUT SET	FORCES
1	3.00000	0	2.00000
2	1.50000	0	0.00000
3	1.50000	0	0.00000
4	0.00000	0	0.00000
5	3.00000	0	2.00000
6	1.50000	1	1.00000
7	1.50000	0	0.00000
8	1.50000	1	1.00000
9	3.00000	0	2.00000

ARC	FLOW	CUT SET	FORCES
1, 2	1.50000	0	0.00000
1, 3	1.50000	0	0.00000
1, 4	0.00000	0	0.00000
2, 3	0.00000	0	0.00000
2, 5	1.50000	0	0.00000
3, 4	0.00000	0	0.00000
3, 5	1.50000	0	0.00000
4, 5	0.00000	0	0.00000
5, 6	1.50000	0	1.00000
5, 8	1.50000	0	1.00000
6, 7	0.00000	0	0.00000
6, 9	1.50000	0	0.00000
8, 7	1.50000	0	0.00000
7, 9	1.50000	0	0.00000
9, 9	0.00000	0	0.00000

Appendix C

CONVEXITY ASSUMPTION ON MULTIPLE FORCES

It was mentioned earlier that the assumption $p(i)_{k+1} - p(i)_k \leq p(i)_k - p(i)_{k-1}$ assures that allocating $[\pi(i)] + 1$ forces with probability $\tilde{\pi}(i)$ and $[\pi(i)]$ forces with probability $1 - \tilde{\pi}(i)$ minimizes the probability that the infiltrator can successfully cross node i for given $\pi(i)$. This appendix will verify that statement.

Let $P\{k,i\}$ be the probability that k forces are placed at node i. Then, $\Sigma kP\{k,i\} = \pi(i)$ and the probability associated with node i is $1 - \Sigma P\{k,i\}p(i)_k$. Minimizing this probability is equivalent to maximizing $\Sigma P\{k,i\}p(i)_k$. The following theorem shows that for given $\pi(i)$, this occurs when the conditions above are satisfied.

THEOREM. The quantity (i)$\Sigma P\{k,i\}p(i)_k$ is maximized, subject to the constraint (ii)$\Sigma kP\{k,i\} = \pi(i)$ when $P\{[\pi(i)] + 1,i\} = \tilde{\pi}(i)$, $P\{[\pi(i)],i\} = 1 - \tilde{\pi}(i)$, and all other $P\{k,i\} = 0$.

PROOF. Let $h = \max\{k/P\{k,i\} > 0\}$, $\ell = \min\{k/P\{k,i\} > 0\}$, and $d = h - \ell$. Let $\bar{P}\{k,i\} = P\{k,i\}$ be a set of values of smallest d which maximizes (i) subject to (ii). Suppose the theorem is false. Then $h - \ell = d \geq 2$. Let $m = \min\{\bar{P}\{h,i\}, \bar{P}\{\ell,i\}\}$. Set $P\{h,i\} = \bar{P}\{h,i\} - m$ and $P\{h - 1,i\} = \bar{P}\{h - 1,i\} + m$. Then set $P\{\ell,i\} = \bar{P}\{\ell,i\} - m$ and increase $P\{\ell + 1,i\}$ by m. (If $h - \ell = 2$, then $P\{\ell + 1 = \bar{P}\{\ell + 1\} + 2m$). Either $P\{h,i\}$ or $P\{\ell,i\}$ is now zero, decreasing d by at least one unit. The increase in (i) is equal to $m(-p(i)_\ell + p(i)_{\ell+1} - p(i)_h + p(i)_{h-1})$. Since (ii) is still satisfied, this is strictly negative from our choice of $\bar{P}\{k,i\}$. However, since $h > \ell + 1$,

$$p(i)_h - p(i)_{h-1} \leq p(i)_{\ell+1} - p(i)_\ell$$

$$- p(i)_\ell + p(i)_{\ell+1} - p(i)_h + p(i)_{h-1} \geq 0$$

for a contradiction. QED.

Appendix D

FINDING MAXIMUM PATH VALUE

This appendix briefly explains the procedure for keeping track of the maximum path value during the algorithm.

Let K be the maximum path value at the beginning of the algorithm and let V be the value of C, the minimum cut set. Substituting the generalized capacities (Expression (4a-b)) for the ones (Expression (2a-b)) that appear in Expression (3), one obtains:

$$\frac{\Sigma \Delta\pi(i) + \Sigma \Delta\pi(i,j)}{\Delta K} = \frac{1}{K} V .$$

But $\Sigma \Delta\pi(i) + \Sigma \Delta\pi(i,j) = MV$, where M is as defined in step 5 of the multiple intercepting force algorithm. Thus

$$\frac{MV}{\Delta K} = \frac{1}{K} V$$

$$\text{or } K - \Delta K = K(1 - M).$$

REFERENCES

1. Durbin, E. P., An Interdiction Model of Highway Transportation, The RAND Corporation, RM-4945-PR, May 1966.

2. Ford, L. R., Jr., and D. R. Fulkerson, Flows In Networks, The RAND Corporation, R-375, December 20, 1960.

3. Wollmer, R. D., Maximizing Flow Through A Network With Node and Arc Capacities, The RAND Corporation, RM-5383-PR, July 1967.

4. _____, "Removing Arcs From A Network," J. Op. Res. Soc. Am., 12, 934-940, (1964).

5. _____, "Stochastic Sensitivity Analysis of Maximum Flow and Shortest Route Networks," Management Science, To be published.

6. _____, Some Meth s for Determining the Most Vital Link in a Railway Network, The RAND Corporation, RM-3321-ISA, April 1963.

DOCUMENT CONTROL DATA

1. ORIGINATING ACTIVITY	2a. REPORT SECURITY CLASSIFICATION
THE RAND CORPORATION	UNCLASSIFIED
	2b. GROUP

3. REPORT TITLE

AN INTERDICTION MODEL FOR SPARSELY TRAVELED NETWORKS

4. AUTHOR(S) (Last name, first name, initial)

Wollmer, Richard D.

5. REPORT DATE	6a. TOTAL No. OF PAGES	6b. No. OF REFS.
April 1968	50	6

7. CONTRACT OR GRANT No.	8. ORIGINATOR'S REPORT No.
F44620-67-C-0045	RM-5539-PR

9a. AVAILABILITY/LIMITATION NOTICES	9b. SPONSORING AGENCY
DDC-1	United States Air Force Project Rand

10. ABSTRACT

An algorithm and FORTRAN IV computer program for choosing locations at which to place assault forces to prevent infiltrators from proceeding through a transportation or supply network. Interdiction is by direct assault rather than by reducing the throughput capacity of a supply network. The model assumes that the strategy for placing forces is known to the infiltrator and that he will choose a path through the network that maximizes his probability of successful traverse. Inputs to the model are a list of the arcs and nodes of the network, the number of forces available to stop the infiltrator, and the probabilities for stopping him at the arcs and nodes as functions of the number of forces placed there. The model calculates the optimal placement when one force is available, and the optimal or nearly optimal placement of forces when multiple forces are available. The computer program, written to implement the model on RAND's IBM 7044, can readily be adapted to other high-speed computers. It now handles problems with up to 300 arcs, 150 nodes, and 25 forces, and can easily be modified by changing the dimension statement.

11. KEY WORDS

Network theory
Interdiction
Infiltration
Counterinsurgency and insurgency
Computer programs