

27
10/30/79
24 7/7/15

SAND 79-1534
Unlimited Release

Optimal Improvement of Networks to Nuclear Safeguards Problems - An Alternate View

Peter J. Slater



Sandia Laboratories

Unlimited Release
JAN 77 - 1536
Printed August 1976

Optimal Improvement of Networks Related
to Nuclear Safeguards Problems -
An Alternate View

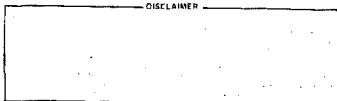
Peter J. Slater
Applied Mathematics Division (JH1)

4-62
1

Abstract

Networks have been used to model fixed-site security systems and to determine optimal travel routes of thieves or saboteurs. Considered here is the problem of upgrading the security by optimally investing in means to increase the travel time and/or detection probabilities over the arcs of the network. It is shown how to formulate a single linear programming problem to answer the following question. If the cost of increasing travel time is linear for each arc, what is the minimum amount of money one must invest to guarantee that there does not exist a path for thief or saboteur of length less than a fixed time T ?

DISCLAIMER



1. Introduction

When networks have been used to model fixed-site security systems, as in [1,2,4,5], one of the important problems is to determine optimal travel routes of thieves or saboteurs from a source point S (generally taken to be the outside of the modelled facility) to a target point or objective G (or to a collection of such points in the model). Travel routes can be considered to be optimal if they minimize travel time or detection probability, or some combination of these two factors. If, for example, one is interested in those routes providing an adversary with the minimum travel time M from S to G , one might further be interested in paths the adversary might traverse with a travel time close to M . If there is a time T (with $T > M$), and it is necessary that no path from S to G have a traversal time less than T , then one must increase the travel time over all paths of length less than T , and not just the original shortest paths. As in [6], it has been noted that "travel time" and "detection probability" problems are essentially equivalent. Here the problems will be discussed in the context of travel times.

Standard network terminology, as in [2], will be used. In particular, network N has a node set $V(N)$ and arc set $A(N)$, where each arc $a_i \in A(N)$ consists of an ordered pair (u_i, v_i) of nodes from $V(N)$. Node u_i is the tail of a_i , and v_i is its head. Associated with each arc a_i is the (non-negative) number ℓ_i which will represent its length or travel time. By the length, or "time length," of a (directed) path P is meant the sum of the lengths of the arcs in P . With the source point S a designated node, each node $n \in V(N)$ has a "reach time" r_n which equals the length of a shortest path from S to n . Of course, if $S = n$ then $r_n = 0$. As an example, the network in Figure 1 has the arc lengths indicated and the reach times tabulated.

and note that any S to t path corresponds to a path in N from f to some g_i whose length is exactly one less. To produce an optimal investment problem equivalent to the problem in N , increase T by one, and then one need only guarantee that none of these new (nonexistent) arcs will be designated for improvement. This can be accomplished by making the unit cost of improvement extremely large for each of these new arcs.

Jacobson's approach and method of solution can be summarized as follows. Assume a "fixed number of dollars" (that is, a budget B) and seek to optimally allocate this money over the arcs. As noted [6], each path from f to t in N imposes a constraint since its overall length must be sufficiently large. The total number of S to t paths can be extremely large, however, and one is interested only in increasing the lengths of those which are comparatively short. One can therefore adopt the following strategy. First, determine all of the shortest S to t paths in N , and solve a linear programming problem whose number of variables is one more than the number of arcs which appear in at least one of these shortest paths, and whose number of constraints is one more than the number of paths found. In brief, the solution determines how to optimally allocate the budget B over those edges to maximize the minimum length of this restricted set of S to t paths.

Since one may have increased the length of each of these paths sufficient to make them all longer than some other path in N , one must iterate the following step.

- I. Assuming B to be allocated over $A(N)$ as previously determined, and thus some of the arcs have been lengthened, find the new set of shortest S to t paths, and assume each of them has length L .
- II. Let P denote the collection of these paths along with all of the paths found previously.

- II. Solve a linear programming problem to determine how to optimally allocate the budget B over the arcs of the paths in P so as to maximize the minimum length of a path in P . The number of constraints will be one more than the size of P , and the number of variables will be one more than the number of arcs which appear in at least one path in P . Let L' be the minimum length of a path in P under this new allocation of B .
- III. If $L = L'$, or $L' - L$ is sufficiently small, then end Step 1. Otherwise the next step is commenced at I.

The comparative efficiency of Jacobsen's algorithm derives from the fact that not all of the d to a paths are considered, but rather only those which are sufficiently short. Even the number of shortest paths, however, can be extremely large in comparison with the size of the graph.

Assuming each arc in network N_2 in Figure 2 is directed from left to right and they have the same length, L , then there are 4,24 shortest d to a paths of length $4L$. The first linear program to be run in Jacobsen's algorithm would therefore involve 73 variables and 4,097 constraints. Subsequent linear programs in the algorithm would each be even larger.

The alternate method described in the next section will run in a single step and will not involve any computation of shortest paths (as in Part I of each step of Jacobsen's algorithm). A single (sparse) linear programming problem will be solved. The number of constraints will be one more than the number of arcs, and the number of variables will equal the number of arcs plus the number of vertices minus one. For network N_2 in Figure 2, one thus has a single problem with 96 variables and only 75 constraints.

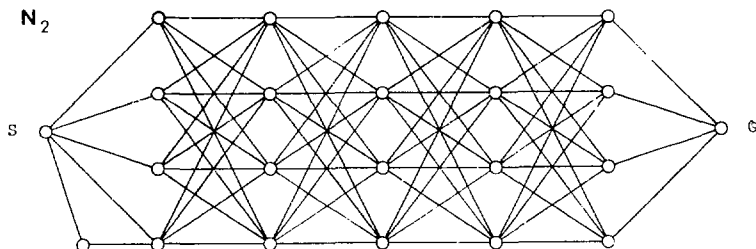


Figure 2. A network with 4,096 S to G shortest paths.

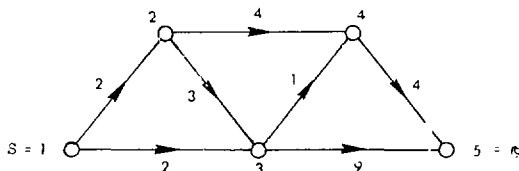
2. The Alternate View

The method previously described assumes a fixed budget B and provides a method for optimally allocating this money over the arcs of a network N so as to maximize the length of the shortest path in N from node S to node G . An alternate view is to assume a time T for which no S to G path in N will have length less than T . That is, for a chosen time value T one asks for the minimum value of B whose investment in increasing the lengths of the arcs of N can make the reach time of the object node G at least as large as T .

Consider, as an example, the network N_3 in Figure 3 with source $S = 1$ and object node $G = 5$. The shortest path from 1 to 5 has length seven (that is, $r_5 = 7$). The starting reach values are $r_1 = 0$, $r_2 = 2$, $r_3 = 2$, $r_4 = 3$ and $r_5 = 7$. How much, B , must one invest to increase r_5 to $T = 18$?

Since d_i represents the distance by which the i^{th} arc will be increased, and c_i represents the cost to increase d_i by one, the total cost is $c_1 d_1 + c_2 d_2 + c_3 d_3 + c_4 d_4 + c_5 d_5 + c_6 d_6 + c_7 d_7$, and this should be minimized subject to $r_5 \geq 18$.

N_3



arc	1	2	3	4	5	6	7
tail	1	1	2	2	3	3	4
head	2	3	3	4	4	5	5
length l_i	2	2	3	4	1	9	4
unit cost c_i	5	6	4	2	3	8	3
distance increased d_i	d_1	d_2	d_3	d_4	d_5	d_6	d_7

Figure 3. Network N_3 with desired reach time $T = 10$.

Let us consider the relationships among the final reach values r_1, r_2, r_3, r_4 and r_5 . One must have $r_1 = 0$ and $r_5 \geq 18$. Furthermore, for example, we can consider arc $a_7 = (4,5)$ whose travel time will be $l_7 + d_7 = 4 + d_7$. This implies that $r_5 \leq r_4 + 4 + d_7$. By considering arc $a_6 = (3,5)$ one sees that $r_5 \leq r_3 + 9 + d_6$. In general, for arc $a_i = (u_i, v_i)$ one will have $r_{v_i} \leq r_{u_i} + l_i + d_i$. The particular example being considered can thus be solved by the solution to the following linear program. (Note again that $r_1 = 0$.)

$$\begin{aligned}
 &\text{Minimize } 5d_1 + 6d_2 + 4d_3 + 2d_4 + 3d_5 + 8d_6 + 3d_7 \\
 &\quad + 2r_2 + 0r_3 + 0r_4 + 0r_5
 \end{aligned}$$

subject to

$$r_5 \approx 18$$

$$d_1 + 0 - r_2 \approx -2$$

$$d_2 + 0 - r_3 \approx -2$$

$$d_3 + r_2 - r_3 \approx -3$$

$$d_4 + r_2 - r_4 \approx -4$$

$$d_5 + r_3 - r_4 \approx -1$$

$$d_6 + r_3 - r_5 \approx -9$$

$$d_7 + r_4 - r_5 \approx -4$$

where

$$d_i \geq 0, \quad r_i \geq 0.$$

Sometimes it is more convenient to solve the dual of the linear programming problem. In general, for the dual problem the number of constraints will equal the number of arcs plus the number of vertices minus one, and the number of variables will be one more than the number of arcs. The equivalent dual maximization problem for the preceding problem is the following.

$$\text{Maximize} \quad 18x_1 - 2x_2 - 2x_3 - 3x_4 - 4x_5 - x_6 - 9x_7 - 4x_8$$

subject to

$$x_2 \leq 5$$

$$x_3 \leq 6$$

$$x_4 \leq 4$$

$$x_5 \leq 2$$

$$x_6 \leq 3$$

$$x_7 \leq 8$$

$$x_8 \leq 3$$

$$-x_2 + x_4 + x_5 \leq 0$$

$$-x_3 - x_4 + x_6 + x_7 \leq 0$$

$$-x_5 - x_6 + x_8 = 0$$

$$x_1 = x_7 = x_8 = 0$$

Where

$$x_i \geq 0$$

The solution has $Q = \{d_1, d_2, d_3, i_1, d_5, d_6, d_7, r_2, r_3, r_4, r_5\} = \{4, 7, 0, 0, 3, 0, 4, 6, 9, 10, 18\}$ and $P = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\} = \{11, 5, 0, 3, 0, 1, 8, 3\}$, and the value determined is $B = 74$. Thus one increases the lengths of arcs $(1, 2)$ and $(4, 5)$ by four and of $(1, 3)$ by seven. In this case every s to t path now has length 18.

In general, assume one has network N with node set $V(N) = \{1, 2, \dots, p\}$ and arc set $\{a_1, a_2, \dots, a_q\}$ with $a_i = (u_i, v_i)$. The original arc lengths, $\ell_1, \ell_2, \dots, \ell_q$, and the unit costs for each arc, c_1, c_2, \dots, c_q , are known, and T is chosen. The problem to be solved is the following (where $r_n = 0$ for source $S = n$, and k is the objective).

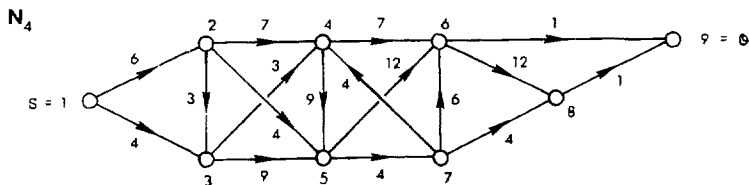
$$\begin{aligned} \text{Minimize} \quad & c_1 d_1 + c_2 d_2 + \dots + c_q d_q \\ & + Or_1 + \dots + Or_p \end{aligned}$$

$$\begin{aligned} \text{subject to} \quad & r_k \geq T \\ & d_1 + r_{u_1} - r_{v_1} \geq -\ell_1 \\ & d_2 + r_{u_2} - r_{v_2} \geq -\ell_2 \\ & \vdots \\ & d_q + r_{u_q} - r_{v_q} \geq -\ell_q \end{aligned}$$

$$\text{where} \quad d_i \geq 0, \quad r_i \geq 0$$

The sparseness of the problem is demonstrated by the fact that each constraint involves at most three variables.

As a final example, consider the network N_4 in Figure 4 which is taken from [6] and modified so that there is only a single objective node. The linear programming problem corresponding to N_4 is the following, where $r_1 = 0$ since $S = 1$.



Arc	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Tail	1	1	2	2	2	3	3	4	4	5	5	7	7	6	7	6	8
Head	2	3	3	4	5	4	5	5	6	6	7	4	6	8	8	9	9
Length l_i	6	4	3	7	4	3	9	9	7	12	4	4	6	12	4	1	1
Unit cost c_i	3	2	4	2	3	1	5	3	8	6	3	7	4	3	3	10,000	10,000
Distance increased d_i	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}	d_{13}	d_{14}	d_{15}	d_{16}	d_{17}

Node	1	2	3	4	5	6	7	8	9
Original reach time	0	6	4	7	10	14	14	18	15
Final reach time									44

Figure 4. Network N_4 with desired reach time $r_9 = T = 44$.

$$\begin{aligned}
 \text{Minimize} \quad & 3d_1 + 2d_2 + 4d_3 + 2d_4 + 3d_5 + d_6 + 5d_7 + 3d_8 + 3d_9 \\
 & + 6d_{10} + 3d_{11} + 7d_{12} + 4d_{13} + 3d_{14} + 3d_{15} + 10,000d_{16} \\
 & + 10,000d_{17} + 0(r_2 + \dots + r_9)
 \end{aligned}$$

subject to

$$\begin{aligned}
 r_9 & \geq 44 \\
 d_1 + 0 - r_2 & \geq -6 \\
 d_2 + 0 - r_3 & \geq -4 \\
 d_3 + r_2 - r_3 & \geq -3
 \end{aligned}$$

$$d_4 + r_2 - r_4 \geq -7$$

$$d_5 + r_2 - r_5 \geq -4$$

$$d_6 + r_3 - r_4 \geq -3$$

$$d_7 + r_3 - r_5 \geq -9$$

$$d_8 + r_4 - r_5 \geq -9$$

$$d_9 + r_4 - r_6 \geq -7$$

$$d_{10} + r_5 - r_6 \geq -10$$

$$d_{11} + r_5 - r_7 \geq -4$$

$$d_{12} + r_7 - r_4 \geq -4$$

$$d_{13} + r_7 - r_6 \geq -6$$

$$d_{14} + r_6 - r_5 \geq -17$$

$$d_{15} + r_7 - r_8 \geq -4$$

$$d_{16} + r_6 - r_7 \geq -1$$

$$d_{17} + r_8 - r_9 \geq -5$$

where

$$d_i \geq 0, \quad r_i \geq 0.$$

2. Observations

Obviously either approach can be used to obtain an (approximate) answer for the other question by running a sequence of problems. For example, assume one has a fixed budget B^* and wants to maximize the minimum S to G , travel time T^* in N by using the approach described in Section

2. One could run a sequence of problems with times T_1, T_2, \dots , and obtain answers B_1, B_2, \dots . If $B_1 < B^*$ one makes an appropriate choice of T_{i+1} with $T_{i+1} > T_i$; if $B_1 > B^*$ one makes an appropriate choice of T_{i+1} with $T_{i+1} < T_i$. Clearly one can approximate T^* as closely as one desires.

An advantage to using the approach described in Section 2 is that only a single (sparse) linear programming problem need be solved, the size of which will often be smaller than the problem in each step of the algorithm described in Section 1. Since one knows the number of vertices and the number of arcs, the size of the linear programming problem to be run is known. It will often be difficult or impossible to determine beforehand how large the linear programming problems in Jacobsen's algorithm will grow to be.

Even when the size of the network (number of nodes and number of arcs) is large, one may well be able to take advantage of the sparse nature of the proposed solution problem. Note that each constraint involves at most three variables, one d_i and two r_j 's.

Acknowledgments

I am grateful to Bernie L. Hulme (5642) and to Drayton D. Boozer (1758) for many interesting conversations concerning security and safeguards, and thanks are due to David R. Strip (4416) for first describing to me the problem discussed in this report.

References

- [1] H. A. Bennett, D. D. Boozer, L. D. Chapman, M. L. Daniel, D. E. G. B. Hulme and G. B. Varnado, "Safeguards System Effectiveness Modeling," Institute of Nuclear Materials Management, Vol. 5, No. 3, Fall 1976.
- [2] J. A. Bondy and U. S. R. Murty, Graph Theory with Applications, American Elsevier Publishing Co., Inc., New York, 1976.
- [3] B. L. Hulme, "Graph Theoretic Models of Theft Problems I. The Basic Theft Model," SAND75-0595, Sandia Laboratories, Albuquerque, Nov. 1976.
- [4] B. L. Hulme, "Pathfinding in Graph-Theoretic Sabotage Models I. Simultaneous Attack by Several Teams," SAND76-0314, Sandia Laboratories, Albuquerque, July 1976.
- [5] B. L. Hulme, "MINDPT: A Code for Minimizing Detection Probability up to a Given Time Away from a Sabotage Target," SAND77-0639, Sandia Laboratories, Albuquerque, December 1977.
- [6] J. E. Jacobsen, "Optimal Improvement of Graphs Related to Nuclear Safeguards Problems," SAND76-0435, Sandia Laboratories, Albuquerque, August 1977.

Distribution

Dr. A. J. Goldman (4)
Operations Research Division 712
National Bureau of Standards
Washington, D. C. 20036

Stephen Jacobsen
Westwood Systems Engineering, Inc.
1600 Westwood Blvd., Suite 203
Berkeley, CA 94604

Dr. J. Maurer
Mathematics Department
Fine Hall
Princeton University
Princeton, NJ 08540

444 E. E. Smith
444 V. E. Blake, Jr.
444 K. M. Plugge
470 E. D. Link
470 C. E. Olson
470 D. J. Rooser
470 J. L. Darby
5600 R. J. Dotry
5603 M. R. Scott
4410 D. J. McCloskey
4410 J. W. Hickman
4413 N. R. Ortiz
4414 G. B. Varnado
4414 R. E. Worrell
4416 L. D. Chapman
4416 H. A. Bennett
4416 D. Engi
4416 D. R. Strip
5600 D. B. Shuster
Attn: 5610 A. A. Lieber
5620 M. M. Newsom
5630 R. C. Maylew
5640 G. J. Simmons
5641 R. J. Thompson
5641 J. A. Davis
5641 C. A. Morgan
5641 I. J. Slater (10)
5642 L. F. Shampine
5642 B. L. Hulme
5642 K. L. Hiebert
5642 R. J. Hanson
8266 E. A. Aas
3141 T. L. Werner (5)
3151 W. L. Garner, DOE/TIC (Unlim. Release) (3)
3154-3 R. Campbell, For DOE/TIC (25)