AN ALGORITHM FOR FINDING SHORTEST ROUTES FROM ALL SOURCE NODES TO A GIVEN DESTINATION IN GENERAL NETWORKS*

By JIN Y. YEN (*University of California, Berkeley*)

Summary. This paper presents an algorithm for finding all shortest routes from all nodes to a given destination in N-node general networks (in which the distances of arcs can be negative). If no negative loop exists, the algorithm requires $\frac{1}{2}M(N - 1)(N - 2)$, $1 < M \leqq N - 1$, additions and comparisons. The existence of a negative loop, should one exist, is detected after $\frac{1}{2}N(N - 1)(N - 2)$ additions and comparisons.

1. Introduction. All shortest routes from all nodes to a given destination in N-node networks can be solved in two different fashions. When the distances of arcs are all positive the problem can easily be solved by an efficient algorithm proposed by Dijkstra [4] which requires roughly $2N^2$ additions and comparisons. But when the distances of some arcs are negative (yet no negative loop exists) the problem becomes harder and can only be solved by the relatively inefficient and sometimes complicated algorithms proposed by Ford [8], Moore [11], Bellman [1], Floyd [7], Murchland [12], Dantzig [3], Dantzig, Blattner and Rao [2], Hu [9], Dreyfus [5], and others—in which as many as N^3 additions and comparisons can be necessary. (For the algorithm of [2] the upper bound on calculations may be a little under N^3 but well over $\frac{1}{2}N^3$. See Dreyfus [6] for reference.)

The purpose of this paper is to present an algorithm for solving the problem in which the distances of some arcs are negative. The advantages of the new algorithm are that, if no negative loop exists, it requires at most $\frac{1}{2}M(N - 1)(N - 2)$, $1 < M \leqq N - 1$, additions and comparisons; and, if there exists a negative loop, it is detected in $\frac{1}{2}N(N - 1)(N - 2)$ additions and comparisons.

Remark. Note that when the distances of arcs are all positive, Dijkstra's algorithm is superior to the new algorithm in most cases. Therefore, the new algorithm, in general, should be applied only when the distances of some arcs in the network are negative.

2. Notation. In an N-node network, let

(i), $i = 1, 2, \cdots, N$, be the nodes where (N) is the destination,

(i, j), $i \neq j$, be the directed arcs from (i) to (j),

(i)–(j)–$\cdots$–(N), $i \neq j \neq N$, be the route from (i) to (N) passing through $(j), \cdots$,

$d_{i,j} \gtreqless 0$, $i \neq j$, be the distances of (i, j). If (i, j) exists, $d_{i,j}$ is a finite number; otherwise, $d_{i,j}$ is considered equal to infinity,

$D = [d_{i,j}]$ be the distance matrix of the network,

$f_i^{(k)}$, $i = 1, 2, \cdots, N$, be the lengths of tentative shortest routes from (i) to (N) on the kth iteration,

f_i, $i = 1, 2, \cdots, N$, be the lengths of shortest routes from (i) to (N).

3. Functional equations. By the principle of dynamic programming, f_i, $i = 1, 2, \cdots, N - 1$, are the set of values that satisfy the following system of equations: $f_i = \min_{j \neq i} [f_j + d_{i,j}]$.

Our algorithm computes these values by means of the following iterative procedure:

* Received July 21, 1968; revised version received December 4, 1968.

$$f_i^{(0)} = d_{i,N}, \qquad i = 1, 2, \cdots, N, \tag{1.1}$$

$$f_i^{(2k-1)} = \min_{N \geqq j > i} [f_j^{(2k-1)} + d_{i,j}, f_i^{(2k-2)}],$$
$$i = N-1, N-2, \cdots, 1, \quad f_N^{(2k-1)} = f_N^{(2k-2)}, \tag{1.2}$$

$$f_i^{(2k)} = \min_{1 \leqq j < i} [f_j^{(2k)} + d_{i,j}, f_i^{(2k-1)}], \quad i = 2, 3, \cdots, N, \quad f_1^{(2k)} = f_1^{(2k-1)}, \tag{1.3}$$

for $k = 1, 2, \cdots$.

The iterative procedure of the algorithm is to be terminated when $f_i^{(2k)} = f_i^{(2k-1)}$, or $f_i^{(2k+1)} = f_i^{(2k)}$, $i = 1, 2, \cdots, N$. Then $f_i^{(2k)}$, or $f_i^{(2k+1)}$, $i = 1, 2, \cdots, N-1$, are the lengths of shortest routes from (i), $i = 1, 2, \cdots, N-1$, to (N).

If by the Nth iteration convergence has not occurred, a negative loop must exist and the problem has no solution.

4. Proof. Suppose the shortest path from (i) to (N) passes through P nodes, $0 \leqq P \leqq N-2$, enroute. Let it be (i)–(N_1)–(N_2)–$\cdots$–(N_P)–(N), where (N_i), $i = 1, 2, \cdots, P$, are distinct nodes of the original network. The path can be divided into M homogeneous blocks, $1 \leqq M \leqq N-1$, in which the numbers naming the nodes in each block either form a strictly increasing or decreasing sequence. We can depict the situation as follows:

$$\left|\begin{matrix}(N) > (N_1) > (N_2) > \cdots > (N_{r_1}) \\ \text{1st } \textit{homogeneous block} \text{ of size } r_1\end{matrix}\right| < \left.\begin{matrix}(N_{r_1+1}) < (N_{r_1+2}) < \cdots < (N_{r_2}) \\ \text{2nd } \textit{homogeneous block} \text{ of size } r_2\end{matrix}\right| >$$
$$\cdots \left|\begin{matrix}(N_{r_{M-1}+1}) \cdots (i) \\ M\text{th } \textit{homogeneous block} \text{ of size } r_M\end{matrix}\right| \tag{2.0}$$

For example, the path (1)–(3)–(5)–(4)–(2)–(6)–(7) consists of 3 *homogeneous blocks* $|(7) > (6) > (2)| < (4) < (5) \,|\, > (3) > (1)|$.

In the 1st *homogeneous block*, since $(N) > (N_1) > (N_2) > \cdots > (N_{r_1})$, the unique and optimal lengths from (N_1), (N_2), $\cdots$, (N_{r_1}) to (N)—as shown by Parikh [13] and others—are as follows:

$$f_i = \min_{N \geqq j > i} [f_j + d_{i,j}] = \min_{N > j > i} [f_j + d_{i,j}, d_{i,N}], \qquad i = N_1, N_2, \cdots, N_{r_1}, \tag{2.1}$$

which is contained in (1.2) for $k = 1$ as follows:

$$f_i^{(1)} = \min_{N \geq j > i} [f_j^{(1)} + d_{i,j}, f_i^{(0)}], \qquad i = N-1, N-2, \cdots, 1, \qquad f_N^{(1)} = f_N^{(0)}.$$

Therefore, the unique and optimal lengths from (N_1), (N_2), $\cdots$, (N_{r_1}) to (N) are determined in Iteration 1.

In the 2nd *homogeneous block*, since (*a*) f_i, $i = N_1, N_2, \cdots, N_{r_1}$, are already determined in Iteration 1, and (*b*) $(N_{r_1}) < (N_{r_1+1}) < (N_{r_1+2}) < \cdots < (N_{r_2})$, the unique and optimal lengths from (N_{r_1+1}), (N_{r_1+2}), $\cdots$, (N_{r_2}) to (N) are as follows:

$$f_i = \min_{\substack{j \in N_1, N_2, \cdots, N_{r_1}, i \\ 1 \leqq j < i}} [f_j + d_{i,j}, f_i], \qquad i = N_{r_1} + 1, N_{r_1} + 2, \cdots, N_{r_2} \tag{2.2}$$

which is contained in (1.3) for $k = 1$ as follows:

$$f_i^{(2)} = \min_{1 \leqq j < i} [f_j^{(2)} + d_{i,j}, f_i^{(1)}], \qquad i = 2, 3, \cdots, N, \quad f_1^{(2)} = f_1^{(1)}.$$

Therefore, the unique and optimal lengths from $(N_{r_1} + 1), (N_{r_1} + 2), \cdots, (N_{r_2})$ to (N) are determined in Iteration 2.

As shown by (2.0), since M, the number of *homogeneous blocks*, is bounded by $N - 1$, the successive approximation of the shortest lengths from (i), $i = 1, 2, \cdots, N - 1$, to (N) terminates in no more than $N - 1$ iterations.

Therefore, the iterative procedure of the algorithm converges in finite steps and the solutions thus obtained is unique and optimal.

REMARK. Note that the necessary and sufficient condition for convergence is that no negative loop exist in the network. If the algorithm does not converge in N iterations (i.e., $f_i^{(N-1)} \neq f_i^{(N)}$, $i = 1, 2, \cdots, N$) it is because there exists at least a shortest route from some (i) to (N) that has more than $N - 1$ arcs. In other words, the network contains at least a negative loop. In such case, no solution can be defined.

5. Illustration. Let a 5-node general network have the following D matrix.

$$D = \begin{array}{c|ccccc|} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & 1 & 3 & 0 & 2 \\ 2 & 2 & 0 & 4 & 2 & 2 \\ 3 & 1 & 0 & 0 & 1 & -1 \\ 4 & 2 & -1 & 3 & 0 & 3 \\ 5 & 0 & 1 & 4 & 1 & 0 \\ \hline \end{array}$$

Then the solution is to be obtained by applying the algorithm as follows:

ITERATION 0. Obtain $f_i^{(0)}$, $i = 1, 2, \cdots, 5$, as follows:

$$f_1^{(0)} = d_{1,5} = 2,$$
$$f_2^{(0)} = d_{2,5} = 2,$$
$$f_3^{(0)} = d_{3,5} = -1,$$
$$f_4^{(0)} = d_{4,5} = 3,$$
$$f_5^{(0)} = d_{5,5} = 0.$$

ITERATION 1. Starting from $i = 5$ toward $i = 1$ compute $f_i^{(1)}$, $i = 5, 4, \cdots, 1$, as follows:

$$f_5^{(1)} = f_5^{(0)} = 0,$$
$$f_4^{(1)} = f_4^{(0)} = 3,$$
$$f_3^{(1)} = \min [f_4^{(1)} + d_{3,4}, f_3^{(0)}] = \min (3 + 1, -1) = -1,$$
$$f_2^{(1)} = \min [f_4^{(1)} + d_{2,4}, f_3^{(1)} + d_{2,3}, f_2^{(0)}] = \min (3 + 2, -1 + 4, 2) = 2,$$
$$f_1^{(1)} = \min [f_4^{(1)} + d_{1,4}, f_3^{(1)} + d_{1,3}, f_2^{(1)} + d_{1,2}, f_1^{(0)}]$$
$$= \min (3 + 0, -1 + 3, 2 + 1, 2) = 2.$$

Note that only the upper triangle elements enter into the computation in Iteration 1.

ITERATION 2. Starting from $i = 1$ toward $i = 5$ compute $f_i^{(2)}$, $i = 1, 2, \cdots, 5$, as follows:

$$f_1^{(2)} = f_1^{(1)} = 2,$$

$$f_2^{(2)} = \min [f_1^{(2)} + d_{2,1}, f_2^{(1)}] = \min (2 + 2, 2) = 2,$$

$$f_3^{(2)} = \min [f_1^{(2)} + d_{3,1}, f_2^{(2)} + d_{3,2}, f_3^{(1)}] = \min (2 + 1, 2 + 0, -1) = -1,$$

$$f_4^{(2)} = \min [f_1^{(2)} + d_{4,1}, f_2^{(2)} + d_{4,2}, f_3^{(2)} + d_{4,3}, f_4^{(1)}]$$
$$= \min (2 + 2, 2 - 1, -1 + 3, 3) = 1.$$

$$f_5^{(2)} = 0.$$

Note that only the lower triangle elements enter into the computation in Iteration 2.

ITERATION 3. Similar to Iteration 1, $f_i^{(3)}$, $i = 5, 4, \cdots, 1$, are obtained as follows: $f_5^{(3)} = 0$, $f_4^{(3)} = 1$, $f_3^{(3)} = -1$, $f_2^{(3)} = 2$, $f_1^{(3)} = 1$.

ITERATION 4. Similar to Iteration 2, $f_i^{(4)}$, $i = 1, 2, \cdots, 5$, are obtained as follows: $f_1^{(4)} = 1$, $f_2^{(4)} = 2$, $f_3^{(4)} = -1$, $f_4^{(4)} = 1$, $f_5^{(4)} = 0$.

Note that $f_i^{(4)} = f_i^{(3)}$, $i = 1, 2, \cdots, 5$. This implies that f_i, $i = 1, 2, \cdots, 5$, are determined in Iteration 3. Therefore, the algorithm terminates. Then the lengths of shortest routes from (i), $i = 1, 2, 3, 4$, to (5) are respectively equal to $1, 2, -1, 1$.

The shortest routes can be obtained from the optimal policy table recorded in the iterative process. They are as follows:

Shortest route from (1) to (5): (1)–(4)–(2)–(5).
Shortest route from (2) to (5): (2)–(5).
Shortest route from (3) to (5): (3)–(5).
Shortest route from (4) to (5): (4)–(2)–(5).

Acknowledgment. The author is very grateful to Professors Stuart E. Dreyfus and Jack D. Rogers for their reading of the manuscript and instructions.

REFERENCES

1. R. E. Bellman, *On a routing problem*, Quart. Appl. Math. **16**, 87–90 (1958)
2. G. B. Dantzig, W. O. Blattner and M. R. Rao, *All shortest routes from a fixed origin in a graph*, Technical Report No. 66–2, Operations Research House, Stanford University, Nov. 1966
3. G. B. Dantzig, *All shortest routes in a graph*, Technical Report No. 66–3, Operations Research House, Stanford University, Nov. 1966
4. E. W. Dijkstra, *A Note on two problems in connexion with graphs*, Numer. Math. **1**, 269–271 (1959)
5. S. E. Dreyfus, *An appraisal of some shortest path algorithms*, Memorandum RM-5433-PR, RAND Corp., Oct. 1967
6. ———, *An appraisal of some shortest path algorithms (revised)*, Memorandum RM-5433-1-PR, RAND Corp., Sept. 1968
7. R. W. Floyd, *Algorithm 97, shortest path*, Comm. ACM (6) **5**, 345 (1962)
8. L. R. Ford, Jr., *Network flow theory*, RAND Corp., P-923, Aug. 1956
9. T. C. Hu, *Revised matrix algorithms for shortest paths*, SIAM J. Appl. Math. (1) **15**, 207–218 (1967)
10. T. C. Hu, *A decomposition algorithm for shortest paths in a network*, Operations Res. (1) **16**, 91–102 (1968)
11. E. F. Moore, *The shortest path through a maze*, Proc. Internat. Sympos. on the Theory of Switching, II, April, 1957
12. J. D. Murchland, *A new method for finding all elementary paths in a complete directed graph*, London School of Economics, Report LSE-TNT-22, 1965

13. A. C. Parikh, *Some theorems and algorithms for finding optimal paths over graphs with engineering applications*, Ph.D. thesis, Purdue University, 1960
14. J. Y. Yen, *Some algorithms for finding the shortest routes through the general networks*, Paper presented at the SIAM 2nd Internat. Conf. on Computation Methods on Optimization Problems, San Remo, Italy, Sept. 1968
15. ———, *A matrix algorithm for solving all shortest routes from a fixed origin in the non-negative networks*, Paper presented at the TIMS 15th Internat. Meeting, Cleveland, Ohio, Sept. 1968