Dynamic Programming Treatment of the Travelling Salesman Problem*

RICHARD BELLMAN

RAND Corporation, Santa Monica, California

Introduction

The well-known travelling salesman problem is the following: "A salesman is required to visit once and only once each of n different cities starting from a base city, and returning to this city. What path minimizes the total distance travelled by the salesman?"

The problem has been treated by a number of different people using a variety of techniques; cf. Dantzig, Fulkerson, Johnson [1], where a combination of ingenuity and linear programming is used, and Miller, Tucker and Zemlin [2], whose experiments using an all-integer program of Gomory did not produce results in cases with ten cities although some success was achieved in cases of simply four cities. The purpose of this note is to show that this problem can easily be formulated in dynamic programming terms [3], and resolved computationally for up to 17 cities. For larger numbers, the method presented below, combined with various simple manipulations, may be used to obtain quick approximate solutions. Results of this nature were independently obtained by M. Held and R. M. Karp, who are in the process of publishing some extensions and computational results.

Dynamic Programming Formulation

Consider the problem as a multistage decision problem. With no loss in generality, since the tour is a round trip, fix the origin at some city, say 0. Suppose that at a certain stage of an optimal tour starting at 0 one has reached a city i and there remain k cities $j_1, j_2, \cdots, j_k$ to be visited before returning to 0. Then it is clear that, the tour being optimal, the path from i through $j_1, j_2, \cdots, j_k$ in some order and then to 0 must be of minimum length; for, if not the entire tour could not be optimal, since its total length could be reduced by choosing a shorter path from i through $j_1, j_2, \cdots, j_k$ to 0.

Therefore, let us define

$$f(i; j_1, j_2, \cdots, j_k) \equiv \text{length of a path of minimum length from } i \text{ to 0 which passes once and only once through each of the remaining } k \text{ unvisited cities } j_1, j_2, \cdots, j_k. \quad (1)$$

Thus, if we obtain $f(0; j_1, j_2, \cdots, j_n)$, and a path which has this length, the problem has been solved.

* Received March, 1961; revised July, 1961.

Let us also define d_{ij} to be the distance between the ith and jth cities. Then as a consequence of the above remarks, we have that

$$f(i; j_1, j_2, \cdots, j_k) = \min_{1 \leqq m \leqq k} \{d_{ij_m} + f(i; j_1, j_2, \cdots, j_{m-1}, j_{m+1}, \cdots, j_k)\}. \quad (2)$$

This is an application of the general principle of optimality in the theory of dynamic programming [3].

The iterative procedure given by (2) is initiated through the use of the known function

$$f(i; j) = d_{ij} + d_{j0} \quad (3)$$

from which we obtain $f(i; j_1, j_2)$, which, in turn, through (2) yields $f(i; j_1, j_2)$, and so on until $f(0; j_1, j_2, \cdots, j_n)$ is obtained. The sequence of values of m which minimize the expression in the braces on the right-hand side of (2) gives a desired minimal path.

Computational Feasibility

The only problem to be faced in using the foregoing algorithm to obtain a solution to the travelling salesman problem for an arbitrarily large number of cities is the storage problem. We shall constantly talk in terms of "rapid access." Were a solution to particular problem urgently required, and if we were willing to spend enough time, we could greatly increase the size of the problem that can be solved with the foregoing method, by using auxiliary storage techniques.

To tabulate the function defined in (1), it is necessary only to take account of the totality of values $j_1, j_2, \cdots, j_k$, not of the order in which the cities are to be visited. Consequently, in storing the function $f(i; j_1, j_2, \cdots, j_k)$, we face the task of tabulating all ways of choosing k quantities from among $n-1$ quantities. This quantity is largest when k is the nearest integer to $(n-1)/2$. Take the case where $n-1$ is even. Since, by Stirling's formula,

$$\binom{2m}{m} = \frac{2m!}{(m!)^2} \cong \frac{2^{2m}}{\pi m}, \quad (5)$$

we have the following approximate values:

$$\begin{aligned} &11 \text{ cities} \quad (2m = 10): \quad \frac{2^{10}}{\sqrt{5\pi}} < 300, \\ &17 \text{ cities} \quad (2m = 16): \quad \frac{2^{16}}{\sqrt{8\pi}} < 12{,}000, \\ &21 \text{ cities} \quad (2m = 20): \quad \frac{2^{20}}{\sqrt{10\pi}} < 200{,}000. \end{aligned} \quad (6)$$

It follows that the case of 11 cities can be treated routinely, that 17 cities requires the largest of current fast memory computers, but that problems involving 21 cities are for a few years at least beyond our reach. One can improve upon these numbers by taking advantage of the fact that the distances will be inte-

gers and that we need not use all the digits of one word to specify a distance, but this requires some fancy programming.

Furthermore, in any particular case, one can easily reduce the number of cities by grouping subtours as one new "distance." In this way, one can quite quickly obtain approximations to the solutions of large scale problems.

One advantage of the dynamic programming approach is that one can readily incorporate all types of realistic constraints involving the order in which cities can be visited. For example, it simplifies the minimization process of (2) if the restriction $m \subset S(i)$ is imposed, where S is a set of cities determined by i, e.g., the ten nearest neighbors of i.

Finally, note the following points concerning the time and memory aspects of this problem. A straightforward enumerative comparison of paths would require $n!$ comparisons. We reduce this figure to the order of $n^2 2^{n-1}$, with about the same numbers of additions required. The memory requirements for the function being computed, the function stored, and the optimal policy being determined triple the estimates given above in (6). On the other hand, all types of constraints can be treated in a very simple fashion.

ACKNOWLEDGMENT

I express appreciation to Mario Juncosa for much helpful criticism and suggestions in the writing of this paper.

REFERENCES

1. DANTZIG, G. B.; FULKERSON, D. R., AND JOHNSON, S. M. On a linear programming combinatorial approach to the travelling salesman problem. *Operations Res.* 7 (1959), 1.
2. MILLER, C. E.; TUCKER, A. W., AND ZEMLIN, R. A. Integer programming formulation and travelling salesman problems. *J. ACM* 7 (1960), 326–329.
3. BELLMAN, R. *Dynamic Programming.* Princeton University Press, Princeton, New Jersey, 1957.