A COMPARISON OF LINEAR PROGRAMMING AND DYNAMIC PROGRAMMING

Stuart E. Dreyfus

P-885 pH

June 22, 1956

The RAND Corporation
1700 MAIN ST. • SANTA MONICA • CALIFORNIA

A COMPARISON OF LINEAR PROGRAMMING AND DYNAMIC PROGRAMMING

This paper considers the applications and interrelations of linear and dynamic programming and attempts to place each in a proper perspective so that efficient use can be made of the two techniques. ( )

§1. The Philosophies

Linear programming adopts an intentionally simple model. The complexity of economic formulations during the past hundred years has far outstripped the mathematician's ability to solve such problems computationally. Linear programming represents an attempt to reverse this trend and to view economic processes as interrelated, but fundamentally simple, activities. The optimization of these processes then becomes mathematically feasible.

Dynamic programming concerns itself with a class of functional relations that arise from multi-stage decision processes possessing certain definite structural characteristics. The characteristic properties are exploited to effect a reduction in the dimensionality of the mathematical problem, thereby making some complex processes amenable to analytic or computational techniques.

§2. The Models

The formulation of a problem in mathematical terms is the initial step towards its solution. In this, the model building stage, care must be taken to assure two conditions:

1) The model must be an adequate description of reality and

2) the model, if numerical results are required, must be amenable

to computational techniques.

As will be discussed in §4, care must be taken that condition one is fulfilled when considering a linear programming formulation. As noted in §5, computational feasibility represents the most important consideration when a dynamic programming model is contemplated.

The linear programming model always assumes the following form:

Minimize: $$c_1x_1 + c_2x_2 + \dots + c_nx_n \tag{1}$$

subject to the restrictions:

$$\left.\begin{aligned} &a_{11}x_1 + a_{12}x_2 + \dots a_{1n}x_n = b_1 \\ &\vdots \qquad\qquad\qquad\qquad \vdots \\ &a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n = b_m \end{aligned}\right\} \tag{2}$$

$$x_j \geq 0$$

A common economic interpretation of these equations results from considering the x_j to be activity levels, the a_{ij} to be the input of the i^{th} commodity into the j^{th} activity and the b_i to be upper limits on requirements and availability of commodities. Then the model states that we wish to carry on n productive activities, each using the same m limited resources in such a way as to minimize total cost or maximize profits. [1a]

The typical dynamic programming formulation appears as

$$f_N(S) = \operatorname*{Max}_P \left[R(S,P) + f_{N-1}(S'(P)) \right] \tag{3}$$

This is a recurrence relation and is therefore particularly suited to processes that occur over a sequence of time periods or stages.

A multi-stage process that is easily formulated in these terms is an allocation problem where, at each stage, resources are reinvested in such a way as to maximize total payoff over the length of the process. For example, suppose we are given a quantity x of a resource that may be divided into two non-negative parts y and x-y, During stage 1 we obtain from y a return $g(y)$, at the expense of reducing y to ay where $0 < a < 1$; from (x-y) we obtain a return $h(x-y)$ at the expense of reducing (x-y) to $b(x-y)$ where $0 < b < 1$. The process is now repeated with the new initial quantity $ay + b(x-y)$, and so on for N periods. How can one allocate at each stage so as to maximize the total return obtained over the entire process? The functional equation describing this particular process is

$$f_N(x) = \max_{0 \le y \le x} \left[g(y) + h(x-y) + f_{N-1}(ay + b(x-y)) \right] \quad (4)$$

Here $f_N(x)$ is defined as the return from an N-stage process, starting with resource x, and using an optimal policy. $g(y)+h(x-y)$ is the return from an allocation of resources into parts y and (x-y) at stage 1. $ay+b(x-y)$ represents the remaining resources and is the initial condition for the (N-1) stage process. The equation states that the return from an N-stage process is the sum of the return from stage 1 plus the return from the remaining (N-1) stages where the initial division of resources is chosen so as to maximize the sum. [2]

§3. The Nomenclature

In view of the above examples, it is easy to appreciate the choice of nomenclature. Programming, of course, means allocation in each case. In the linear programming model limited resources are

allocated to various activities. In dynamic programming resources are allocated at each of several time periods.

The term "linear" emphasizes that the applications concern situations where the inputs and outputs of various activities can be assumed to be proportional to the level of the activity.

Dynamic programming takes its name from the fact that the functional equation and its associated computational techniques are derived from, and adapted to, a process changing over a discrete or continuous time interval.

Two exceptions should be noted. It is possible to expand the static picture described in the discussion of the linear programming model to include several time periods by the addition of a new set of restrictions reflecting constraints felt at each time interval. It is important to note, however, that even where a dynamic situation is being considered, the entire process is described by one set of equations and solved as one large problem. [3] In a dynamic programming formulation an iteration of the functional equation corresponds to each time interval, so that a longer duration of the process only entails additional iterations. Hence doubling the length of the process merely requires twice the computation time but no additional formulational considerations. Secondly, dynamic programming does not necessarily involve processes changing over time. It is a multi-stage technique, but often the stages are artificially introduced by considering the component activities individually although they actually occur simultaneously in time. For an example, see ref. [4]

§4. The Model as a Description of Reality

Linear programming, representing a simple model, is necessarily restricted as to the generality of problems that can be attacked. However, if applicable, it has associated with it a very powerful computational device, the simplex algorithm, which can efficiently solve large systems containing hundreds of equations. Dynamic programming, on the other hand, is an extremely general technique of formulation, but carries with it a much more limited computational scheme. This section will discuss three aspects of problems which may make a linear programming model not applicable but which are handled with some ease by dynamic programming.

The linearity assumption is sometimes invalid when applied to industrial processes. The deviation from linearity, and the degree of exactness required from the model, then determine the applicability of linear programming. A problem involving set-up costs is a representative member of the class of non-linear processes. Here the situation is such that the cost of carrying on an activity at zero level is essentially zero. However the use of the activity at even a small level incurs a large cost associated perhaps with the retooling of a production line or administrative costs of placing an order. Once this penalty has been assessed the assumption of proportional costs may become valid. Turning now to the functional equation (3) of dynamic programming, the policy, P, may be thought of as the activity level. When $f_N(S)$ is computed, the policy space is searched for a maximizing policy P. If the return or cost function R(S,P) is given in tabular form a discontinuity at the origin represents no difficulty. If R(S,P) is a function applicable everywhere except

at the origin, the computer code must include a test for P = 0 and the substitution of a cost of zero for the functional cost in that special case.

A second consideration, discreteness of the solution, tends to complicate the linear programming formulation. Some problems of a number theoretic nature require that the activities assume only integral values. For example, in a cargo loading problem only the integral numbers of each item may be loaded and the optimal fitting together of the cargo subject to a weight or space restrictions then becomes a discrete number theory problem. This sort of a restriction is nicely suited to dynamic programming using a digital computer, for only integral numbers of the policy domain P need be considered and the function $f_N(S)$ need only be computed for fixed discrete values. [4]

Finally, there is a large family of problems where the return associated with an activity is known only as a stochastic function of the activity level. Here again, except in special cases, linear programming techniques are not applicable.

§5. Computational Solution of the Model

Just as the previous section was devoted almost entirely to areas where linear programming may not be applicable, this section will concern itself primarily with the computational pitfalls of dynamic programming. And just as it was asserted in the last section that almost any conceivable process can be formulated in terms of dynamic programming, it should now be understood that by means of the simplex and associated algorithms linear programming problems even of formidable size can be automatically

subdued and the solutions exhibited. [5] . This should be noted, since, the entirely successful aspects of each technique being little discussed, one might arrive at the mistaken conclusion that neither technique offered much of practical value. One need only refer to the well known contributions of the older and better understood linear programming approach or to section 7 to see the wide applicability of both techniques.

Let us reconsider the recurrence relation

$$f_N(x) = \max_{0 \le y \le x} \left[g(y) + h(x-y) + f_{N-1}(ay + b(x-y)) \right] \qquad (5)$$

Numerical solution is obtained by first observing that the return from a 1-stage process is simply the maximum of $g(y)+h(x-y)$ since f_o, the 0-stage return, is identically zero. A table of $f_1(z)$ is computed for all values of z from 0 to x. We are really saying that although we don't know what our resources will be when we find that the process has only one stage left (this depending on our as yet undetermined policy during the first (N-1) stages), for any initial resource, z, our final allocation and the return from it are known. Once $f_1(z)$ for all possible z is calculated, $f_2(z)$ can be computed using equation (5) which, for N = 2, relates f_2 to f_1. By working backwards thru the sequence of functions $f_i(z)$ we finally arrive at a table of $f_N(z)$, $0 \le z \le x$, where $f_N(x)$ is the desired solution, the N-stage return employing an optimal policy. Hence for an N-stage process the 1-stage computation is merely performed N times, which presents no difficulties for the digital computer programmer.

Once the number of time intervals has been relegated to its proper place, the problem is reduced to its true dimensions. Its true dimension is precisely the quantity of information needed to describe completely the situation at any one particular stage. In the above example, x, the quantity of resource available, defines the state of the system at any stage of the process. Since x is a scalar number, the problem is termed one dimensional and presents no computational difficulties.

However, it is easy for even this reduced dimensionality to become large. In the job-shop scheduling problem, discussed in §6, for example, the dimensionality is equal to the number of machines in the shop, since one must know the state of each machine in order to understand the state of the system. To achieve a solution a table of the function $f_1(S)$ must be computed for all possible states S and stored for use in computing $f_2(S)$. Where S is, say, 10 dimensional and can assume 100 values in each dimension, the tabular storage of $f_1(S)$ would require 100^{10} memory cells, far beyond the range of present generation computers. So the dimensionality of the process must be about 4 or less for dynamic programming to be applicable in a purely computational manner. The actual performance of the maximization presents further interesting difficulties, particularly where it is to be performed over a 2 or 3 dimensional region. Fortunately many apparently complex processes can be reduced to one or two dimensional problems and solved by dynamic programming while in some higher dimensional problems, the structure of the optimal policy may be deduced analytically, thus simplifying the computation.

§6. Formulation of a Sample Problem

The following is a variation of a problem discussed by Markowitz and Manne [6]. Consider a small scale job shop. There are 6 machines or resources. Twenty-one different kinds of items can be manufactured using the machines. Let x_j be the level of the j^{th} activity, i.e., the number of items of type j to be produced. a_{ij} is the amount of time on the i^{th} machine required to produce 1 of the j^{th} item. c_j is the return from the unit level of the j^{th} activity. b_i is the total time available on the i^{th} machine. Further, let us assume that it makes no sense to produce a fractional number of items. Finally, we assert that the purpose of the process is to maximize the total return by choosing an optimal number of each item to produce.

Viewed first on a linear programming model we have:

Maximize: $\sum_{j=1}^{21} c_j x_j$ (the return)

subject to the restrictions

$$\sum_{j=1}^{21} a_{ij} x_j \leq b_i \quad \text{for } i = 1, 2, 3, 4, 5, 6$$

where the activity levels x must be non-negative and integral.

As dynamic programming our model appears:

$$f_{21}(b_1,b_2,\ldots,b_6) = \text{Max} \left(x_1 c_1 + f_{20}(b_1 - a_{11}x_1, b_2 - a_{21}x_1, \ldots, b_5 - a_{61}x_1) \right)$$

$$x_1 \leq \text{Min}\left(\frac{b_1}{a_{11}}, \ldots, \frac{b_6}{a_{61}}\right)$$

x_1 an integer

The process involving 21 items has been expressed in terms of the first item and a process involving the remaining 20 items.

As discussed in the text, the linear programming solution is made difficult by the integral restriction, whereas dynamic programming is obstructed by the 6-dimensionality of the process.

§7. Some Applications

Contained in this section will be found the statement of several problems that have been solved by either linear programming or dynamic programming. References are made to papers containing more detailed discussions of the solutions. This section is included for two purposes: to demonstrate the catholic nature of both techniques and to aid the reader in deciding which approach, if either, might prove rewarding if applied to problems he may be facing.

Some problems solved by linear programming.

Transportation Problem:

There are N sources of an item and N sinks. The requirement of each sink is known, and the cost of shipment between each source and sink is given. A shipment policy is desired to meet the N demands at minimum shipment cost. [1b]

Assignment Problem:

N employees are to be assigned to N distinct jobs. The aptitude of each employee for each job is known. Which employee is assigned to which job in order to maximize the total aptitude of all employees for the jobs they are filling? [7]

Petroleum Blending Problem:

Crude oil produces gases, straight-run gasoline, 3 weights of distillate and residue. These products may be used for fuel oil blending or in several other processes. The value of the total output of the interrelated process is to be maximized. [8]

Traveling Salesman Problem:

To determine a path of minimum length touching 49 principal cities of the U. S. [9]

Network Problem:

Consider a network connecting two given points by way of a number of intermediate points, where each link of the network has a number assigned to it representing its capacity. Assuming a steady state condition, find a maximal flow from one given point to the other. [10]

Some problems solved by dynamic programming.

Cargo Loading Problem:

A vessel of weight capacity z is to be loaded with an assortment of items. Each item has a weight W_i and value V_i. The value may be a non-linear or stochastic function of number loaded. To determine an integral number of each item to load in order to maximize the value of cargo subject to the weight restriction. [4]

Missile Allocation Problem:

M missiles are to be allocated to N targets. Targets are of various values and defense levels. Probability of a missile surviving defenses and destroying target is an exponential function of the number allocated. How many missiles should be allocated

to each target to maximize total expected damage? [11]

Optimal Climb Problem:

What path should an airplane follow in order to minimize time-to-climb from takeoff to combat conditions? [12]

Early Warning Radar Net Problem:

Radar protection circles are constructed around each of a complex of airbases. Protection afforded each base is a function of the radius of its circle. A set of radii is desired which affords the base complex a fixed protection at minimum defense system perimeter. [13]

Eigenvalue Calculation Problem:

The eigenvalues associated with the differential equation $u''+\lambda^2\beta(t)u=0$ are obtained. We are interested in the values of λ^2 which yield non-trivial solutions u. [2]

§8. Conclusions

The potential user of either of these programming techniques should first consider the factors discussed in the preceding sections. If the problem is essentially linear in its assumptions, there is little doubt that linear programming offers the superior means of solution. If the process is of a multi-stage nature and its true, or reduced, dimension is not large, dynamic programming should be capable of providing a solution unperturbed by non-linear, stochastic, or integral restraints.

It should be noted, however, that both techniques, when artfully applied, have solved problems that at first glance did not fall within the limited domains described above.

Finally, it is important to realize that both linear and dynamic programming represent powerful analytic tools. Interesting theorems have been proved and structures of solutions which are not computationally obtainable have been demonstrated by mathematical application of the simplex or dual-simplex algorithms of linear programming and functional equation theory of dynamic programming.

[1] Activity Analysis of Production and Allocation, T. C. Koopmans, Editor, John Wiley and Sons, 1951

(a) Koopmans, T. C., "Analysis of Production as an Efficient Combination of Activities", Chapter III.

(b) Dantzig, G., "Application of the Simplex Method to a Transportation Problem", Chapter XXIII.

[2] Bellman, R., "The Theory of Dynamic Programming", Bulletin of the American Mathematical Society, November, 1954.

[3] Dantzig, G., Notes on Linear Programming: Part XXII, The RAND Corporation, Research Memorandum, RM-1475, April, 1955.

[4] Bellman, R. and S. Dreyfus, On the Computational Solution of Dynamic Programming Processes - II. On a Cargo Loading Problem. The RAND Corporation, Research Memorandum, RM -1746, July, 1956.

[5] Orchard-Hays, W., Evolution of Computer Codes for Linear Programming. The RAND Corporation, Paper P-810, March, 1956.

[6] Markowitz, H. M. and A. S. Manne, On the Solution of Discrete Programming Processes. The RAND Corporation Paper P-711, revised February, 1956.

[7] Votaw, D. F. and A. Orden, "The Personnel Assignment Problem", Project Scoop No. 10, DSC/Comptroller, Hq. USAF, April, 1952.

[8] Orchard-Hays, W., Computational Experience in Solving Linear Programming Problems, The RAND Corporation, Paper P-482, March, 1954.

[9] Dantzig, G., D. R. Fulkerson and S. M. Johnson, Solution of a Large Scale Traveling Salesman Problem, Journal of the Operations Research Society, November, 1954.

[10] Fulkerson, D. R., and G. Dantzig, Computation of Maximal Flows in Networks, The RAND Corporation, Paper P-677, April, 1955.

[11] Bellman, R., and S. Dreyfus, On the Computational Solution of Dynamic Programming Processes - III. On the Optimal Use of Guided Missiles Against a Fixed Target System - Maximum Expected Damage, The RAND Corporation, Research Memorandum RM-1747, July, 1956.

[12] Cartaino, T. F., and S. Dreyfus, Application of Dynamic Programming to the Airplane Minimum Time-to-Climb Problem, The RAND Corporation, Paper P-834, March, 1956.

[13] Bellman, R., and S. Dreyfus, On the Computational Solution of Dynamic Programming Processes - VII. Early Warning Radar Nets, The RAND Corporation, Research Memorandum RM-1751, July, 1956.

Important forthcoming contributions are:

[14] Bellman, R., "Principles of Dynamic Programming", Princeton University Press, (to appear).

[15] Dantzig, G., (Book in process).