STUDIES IN DISCRETE DYNAMIC PROGRAMMING

by

RONALD ARTHUR HOWARD

S.B., Massacuhsetts Institute of Technology
(1955)

S.M., Massachusetts Institute of Technology
(1956)

E.E., Massachusetts Institute of Technology
(1957)

SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
June, 1958

Signature of Author____________________________
Department of Electrical Engineering

Certified by____________________________
Thesis Supervisor

Accepted by____________________________
Chairman, Departmental Committee on Graduate Students

May 16, 1958

Mr. Ronald A. Howard
Room 10-397B
M.I.T.

Dear Mr. Howard:

This letter is to give you permission to print additional copies of your thesis by the Multilith process, and to submit to the Department of Electrical Engineering copies thus produced, in lieu of the typed copies normally required.

A copy of this letter is to be reproduced by the same process and is to be placed in each copy of the thesis immediately following its title page.

Sincerely yours,

S. H. Caldwell
for the
Department Graduate Committee

SHC:mp
cc: Professor Morse

STUDIES IN DISCRETE DYNAMIC PROGRAMMING

by
RONALD A. HOWARD

Submitted to the Department of Electrical Engineering on May 19, 1958, in partial fulfillment of the requirements for the degree of Doctor of Science.

ABSTRACT

A policy iteration method for solving sequential decision processes of long duration is presented. Consider a system with a discrete number of states, N, and a set of transition probabilities p_{ij} for movement from state i to state j. There is also associated with the system a reward r_{ij} attached to each transition. In each state i, there is a choice of several sets of transition probabilities that could be used as the i^{th} row of the $[p_{ij}]$ matrix. Each set of probabilities has an associated set of rewards, r_{ij}, and the combination is called an alternative in the i^{th} state. A policy for the system is a choice of one alternative in each state so that probability and return matrices are defined. Once the system is making transitions under a given policy, it exhibits all the characteristics of a Markov process; however, it is generating a sequence of returns from its transitions. The problem consists of finding that policy which will cause the system to have the highest average earnings after it has reached statistical equilibrium.

The policy iteration method for finding the optimal policy is based upon a two-part iteration cycle. The entire procedure rests upon a proof that V_i^n, the total expected return in n moves starting from state i, can be represented in the form $v_i + ng_i$ for very large n. The transient values v_i and the gains g_i depend only on the starting state i. In most practical cases g_i is independent of i and may be given the symbol g. The quantity g is called the gain of the policy; it is the average return per transition after a large number of moves. The optimal policy is the policy with highest gain.

The first part of the iteration cycle is a procedure which finds the v_i and g_i pertinent to a particular policy. The procedure may be carried out either by solving a set of N by N linear simultaneous equations or by a simulation approach using Monte Carlo methods.

The second part of the iteration cycle is a policy improvement routine which will find a policy of higher gain if such a policy exists. Convergence on the optimal policy is guaranteed.

Problems in baseball strategy and replacement theory are shown to illustrate the power of the method. This procedure should make possible the formulation and solution of many important decision-making problems.

Thesis supervisor: Philip M. Morse

Title: Professor of Physics

To Polly

ACKNOWLEDGMENT

I would like to express my appreciation to those who have made possible the completion of this thesis. Prof. George E. Kimball has been so helpful in the course of this work that it is almost impossible to number his contributions. My thesis committee, Profs. Philip M. Morse, Samuel J. Mason and Bernard Widrow, have given freely of their time, advice, and encouragement. My interest in this field of research was aroused by Prof. Morse and encouraged by Prof. Kimball. The conversations I have had with Profs. Mason and Widrow have increased my understanding of this area in countless ways.

This work has been supported by fellowships from the Schlumberger Foundation, the International Business Machines Corporation, and the Ramo-Wooldridge Corporation; to them I express my gratitude. My relationship with the Operations Research Group of Arthur D. Little, Inc., has considerably enhanced my graduate education and aided in the preparation of this thesis.

The author would like to thank Dr. Jerome D. Herniter for his careful review of the text and helpful suggestions. To Miss Marion Carter goes credit for the sine qua non of thesis preparation, the typing of the final manuscript.

This work was done in part at the Massachusetts Institute of Technology Computation Center, Cambridge, Massachusetts. It was reproduced under Department of the Navy, Bureau of Ships Contract NObsr 72716, which is administered by the Research Laboratory of Electronics.

TABLE OF CONTENTS

TABLE OF CONTENTS

ILLUSTRATIONS

Introduction

The theory of dynamic programming has existed for several years without finding extensive practical application. In some measure this is due to the fact that the specialization of the theory to individual problems has always required a considerable amount of ingenuity. The systems analyst is more likely to spend his time modifying conventional methods to fit his problem than he is to develop a new method, even if he has overlying principles to guide him. Up to the present time, dynamic programming has been in this category of providing general principles in solving a problem without yielding a complete solution.

It is our feeling that dynamic programming can be developed to the point where a broad class of problems can be solved without the need for excessive amounts of ingenuity. If this can be done, dynamic programming should join the arsenal of powerful, convenient techniques available to the systems analyst.

The Essence of Dynamic Programming

The concept of dynamic programming is useful in multistage decision processes where the result of a decision at one stage affects the decision to be made in the succeeding stage. In other words, if the decision-maker is interested not only in the immediate effect of his decision but also in the long-run effect, then he (or it) has become involved in a dynamic programming problem.

The two main types of dynamic programming problems are deterministic and stochastic problems. Deterministic problems have the property that the result of any decision is known to the decision-maker before the decision is made. Stochastic problems specify the probability of each of the various outcomes of the decision. We see that deterministic problems may always be treated as a special case of stochastic problems although it may not be fruitful to handle them in this way.

Regardless of the deterministic or stochastic nature of the problem, the following properties must be found in the problem before it can be successfully treated by dynamic programming.

1) The states of the system must be describable by a small number of variables.

2) The function of a decision must be to assign to these system variables different numerical values.

3) The history of the system must have no influence on future behavior.

The last property (or Markovian property) can usually be achieved by adding more variables to the state description, but if too many extra variables are required, this approach may be self-defeating.

The sequence of decisions in a problem is called a policy; a policy which is most desirable according to some criterion is called an optimal policy. Reflection on the sequential nature of the process reveals a recurrence relation stated by Bellman[1] as the:

> "Principle of Optimality: An optimal policy has the property that, whatever the initial state and the initial decision, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision."

Although the principle of optimality has been the traditional basis on which sequential decision problems are solved, it is possible to formulate the process of solution for the optimal policy in such a way that this principle never appears explicitly.

It is possible to describe the sequential decision process using either discrete or continuous state variables. Since very few such processes can be solved analytically, it is usually necessary to revert to electronic computation in order to obtain a solution. The selection of digital computation ultimately requires a discrete formulation of the problem. Because this is the end result in so many applications, it appears especially useful to explore the sequential decision process from a discrete point of view. The discrete formulation is general enough to include a wide class of practical problems; it is limited only by the size and speed of existing computational facilities.

Discrete Formulation

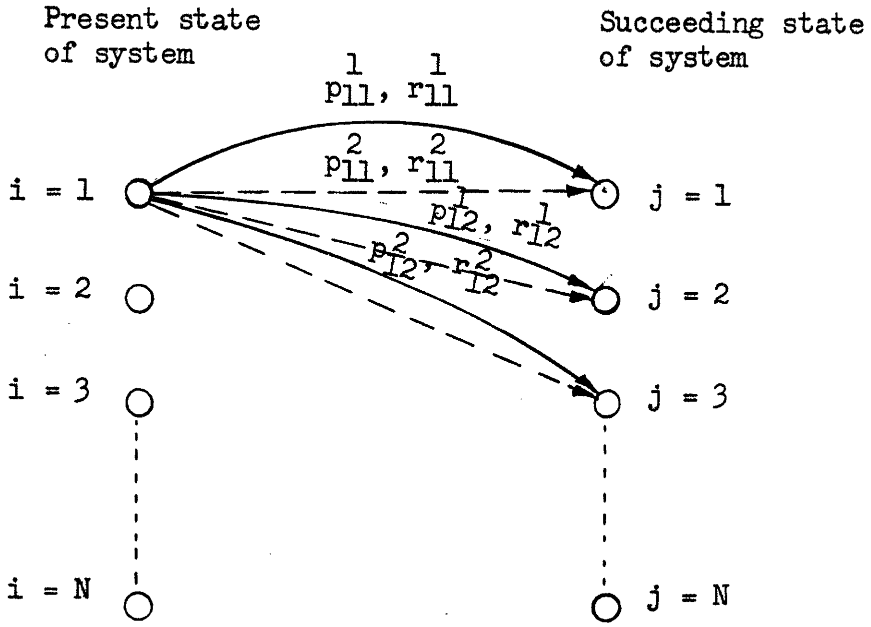
Consider a system with N states. As above, the states must be describable by a small number of parameters, but these parameters may take only discrete values. The number of states in the system, N, is constant for the duration of the problem. We may now consider transformations of the system at discrete intervals of time. Each transition of the system will in general cause the system to move from one state to another. If the initial state of the process is state i or S_i, and the succeeding state is state j or S_j, then we may symbolically write the transformation process as $S_j = T(S_i)$ where T is the symbol for the transformation.

If we define a conditional probability p_{ij} that the system will occupy S_j given that it previously occupied S_i, with $p_{ij} \geqslant 0$ and $\sum_{j=1}^{N} p_{ij} = 1$, then we have a Markov process. The generalization of this process that we shall consider differs from the standard Markov process in two ways:

1) A reward or return r_{ij} is assigned to the transition from S_i to S_j.

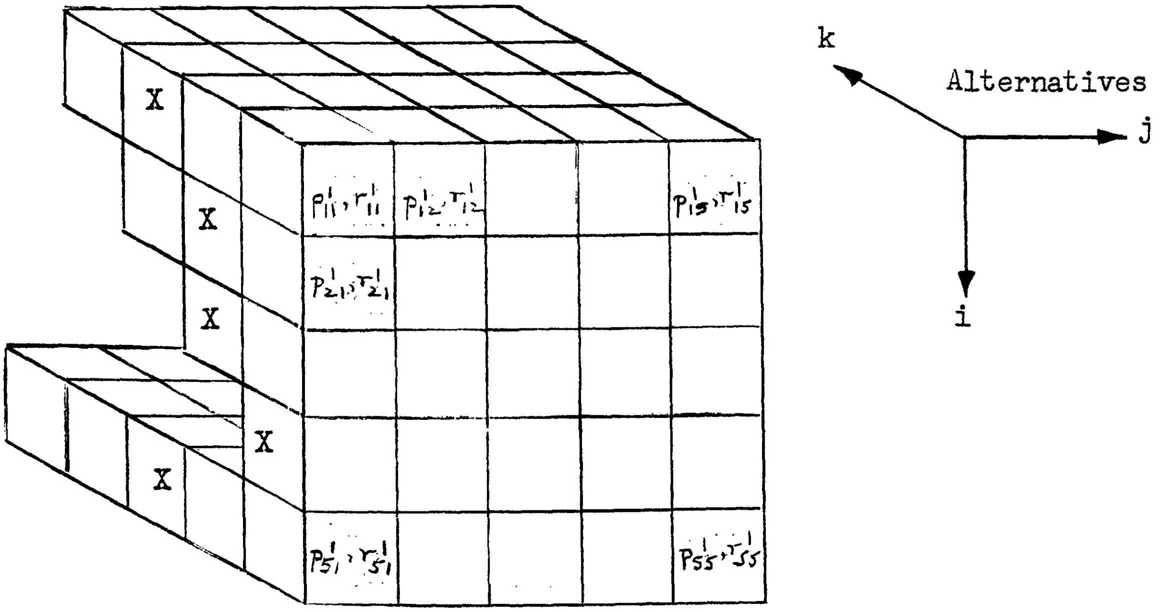
2) The probabilities p_{ij} and the rewards r_{ij} to be associated with transitions from state i are a function of the decision we make in the i^{th} state.

The first addition to the Markov process assigns a random variable r_{ij} to each transition. The second addition allows us to characterize the multistage decision process. In order to see more clearly the character of the process, let us define p_{ij}^k and r_{ij}^k as the probability and reward parameters to be attached to the transition i ⟶ j if we are operating under the k^{th} alternative in state i. An alternative is the name given one of the options in the i^{th} state. Before proceeding any further, let us clarify these remarks with a diagram:



In this diagram, two alternatives have been allowed in the first state. If we pick alternative 1 ($k = 1$), then the transition from state 1 to state 1 will be governed by the probability p_{11}^{1}, the transition from state 1 to state 2 will be governed by p_{12}^{1}, from 1 to 3 by p_{13}^{1}, etc. The rewards associated with these transitions are r_{11}^{1}, r_{12}^{1}, r_{13}^{1}, etc. If the second alternative in state 1 is chosen ($k = 2$), then p_{11}^{2}, p_{12}^{2}, p_{13}^{2}, . . ., p_{1N}^{2} and r_{11}^{2}, r_{12}^{2}, r_{13}^{2}, . . ., r_{1N}^{2}, etc., would be the pertinent probabilities and returns. In the diagram above we see that if alternative 1 in state 1 is selected, we make transitions according to the solid lines; if alternative 2 is chosen, transitions are made according to the dashed lines. The number of alternatives in any state may be of any finite size, and the number of alternatives in each state may be different from the numbers in other states.

A convenient way to visualize the states and their alternatives is by means of a three-dimensional array:



A Possible Five-State Problem

The array as drawn illustrates a five-state problem which has four alternatives in the first state, three in the second, two in the third, one in the fourth, and five in the fifth. Entered on the face of the array are the parameters for the first alternative in each state, the second row in depth of the array contains the parameters for the second alternative in each state, etc. An "X" indicates that we have chosen a particular alternative in a state for operation of the system; the alternative thus selected is called the decision for that state. The set of X's or the set of decisions for all states is called a "policy." The policy indicated in the diagram requires that the probability and reward matrices for the system be composed of the first alternative in state four, the second alternative in states two and three, and the third alternative in states one and five. It is possible to describe the policy by a decision vector D whose elements represent the number of the alternative selected in each state. In this case

$$D = \begin{bmatrix} 3 \\ 2 \\ 2 \\ 1 \\ 3 \end{bmatrix}$$

An optimal policy is defined as a policy which maximizes some performance index. In the five-state problem diagrammed above, there are $4 \times 3 \times 2 \times 1 \times 5 = 120$ different policies. It is conceivable that one could find the performance index for each of these policies and then compare the indices to find the policy which had the largest one. However feasible this may be for 120 policies, it becomes unfeasible for very large problems. For example, a problem with 50 states and 50 alternatives in each state contains $50^{50} \approx 10^{85}$ policies.

It is clear that a direct selection of the optimal policy becomes unfeasible even with the fastest of present computers.

Bellman[2] has proposed one method for circumventing the combinatorial complexity of the sequential decision problem; it will be illustrated in the following section. We shall then propose an alternative method which has certain advantages over Bellman's procedure, especially in the case where the system is permitted a very large number of transitions.

A Simple Example - Bellman's Iteration Procedure

In order to fix ideas, let us consider a simple coin-tossing example. Suppose that we have a choice of tossing either of two biased coins. Coin 1 has probability of heads 3/4 and probability of tails 1/4; coin 2 has corresponding probabilities of 1/3 and 2/3. Profits from playing the game are calculated as follows: If heads are obtained twice in a row, three dollars is won; if tails follow tails, one dollar. If either heads follow tails or tails follow heads, a sum of two dollars is lost.

Consider heads to be state 1 and tails to be state 2. In each state there are two alternatives: flip coin 1 or coin 2; thus

$$\left[p_{1j}^{1}\right] = \left[p_{2j}^{1}\right] = \left[3/4 \quad 1/4\right] \text{ and } \left[p_{1j}^{2}\right] = \left[p_{2j}^{2}\right] = \left[1/3 \quad 2/3\right].$$

The returns are the same for all alternatives so that there is a unique return matrix for the system, $\left[r_{ij}\right] = \begin{bmatrix} 3 & -2 \\ -2 & 1 \end{bmatrix}$. We are seeking to determine that policy which will maximize our expected return in n moves.

Bellman's iteration procedure may be described in the following way. Let f_i^n be the total expected return in n moves starting from state i under an optimal policy. Then from the principle of optimality, the recurrence relation

$$f_i^{n+1} = \operatorname*{Max}_k \sum_{j=1}^{N} \left[p_{ij}^k (r_{ij}^k + f_i^n) \right]$$

is obtained. The boundary values f_i^o when there are no moves remaining may be set arbitrarily to zero in the absence of any specific values. At each stage of the process, the best policy to use at that stage is developed from the recurrence relation.

If the Bellman technique is applied to the coin-tossing problem, we obtain the following table of calculations for ten stages:

n	f_1^n	$f_1^n - f_1^{n-1}$	f_2^n	$f_2^n - f_2^{n-1}$	$f_1^n - f_2^n$
0	0.0000	--	0.0000	--	0.0000
1	1.7500	1.7500	0.0000	0.0000	1.7500
2	3.0625	1.3125	0.5833	0.5833	2.4792
3	4.1927	1.1302	1.4097	0.8264	2.7830
4	5.2470	1.0543	2.3373	0.9276	2.9097
5	6.2696	1.0226	3.3072	0.9699	2.9624
6	7.2790	1.0094	4.2946	0.9874	2.9844
7	8.2829	1.0039	5.2893	0.9947	2.9936
8	9.2845	1.0016	6.2871	0.9978	2.9974
9	10.2852	1.0007	7.2862	0.9991	2.9990
10	11.2855	1.0003	8.2858	0.9996	2.9997

The best policy at any stage is to flip coin 1 when the last flip produced heads, and coin 2 when it produced tails. The values of f_1^n and f_2^n for each n are plotted in the following graph, Figure 1. Note that the curves become linear and parallel for large n. From the table of calculations, it appears that in the limit as $n \to \infty$, $f_1^n - f_1^{n-1}$ and

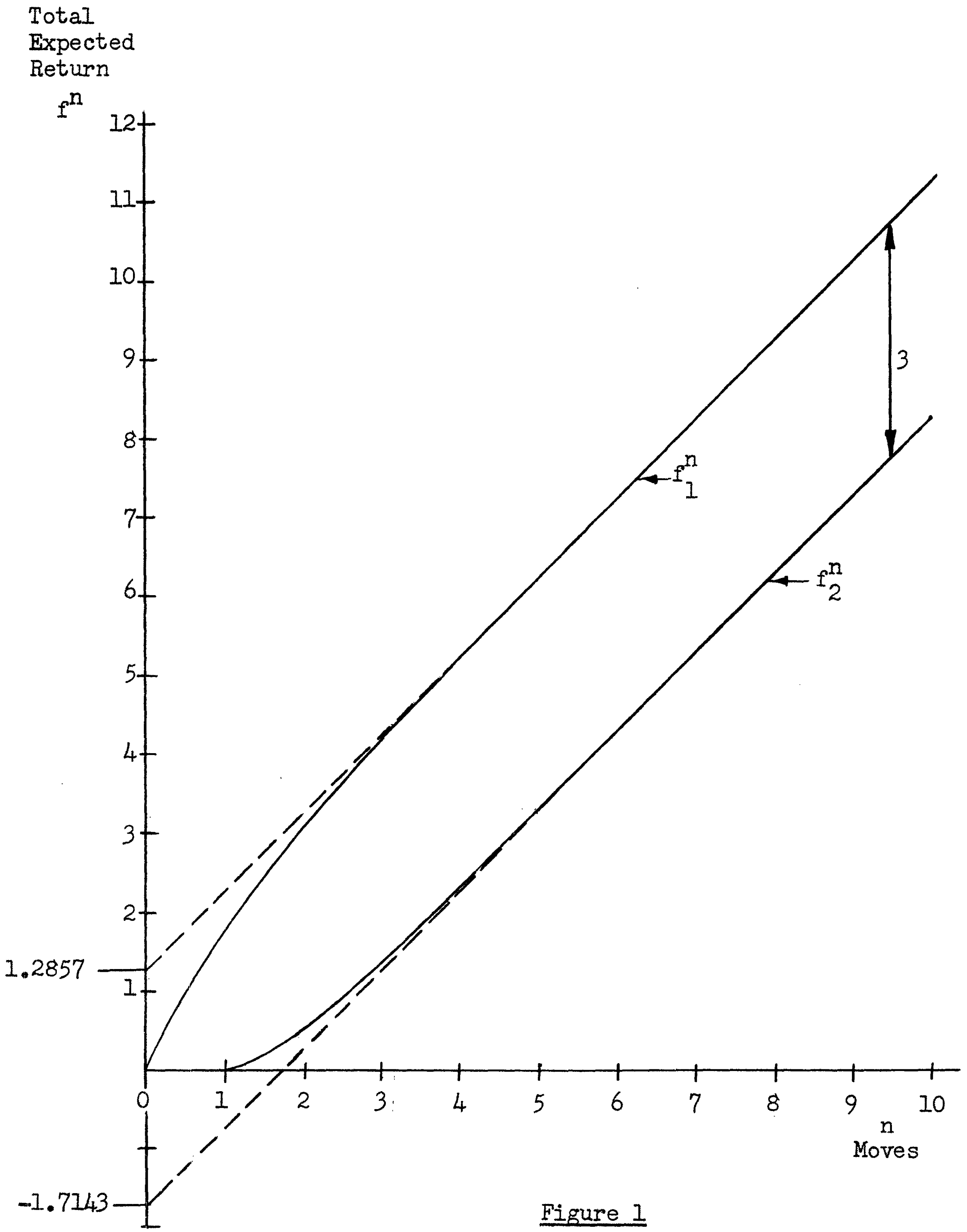


Figure 1

COIN TOSSING PROBLEM

Total Expected Return in n Moves

$f_2^n - f_2^{n-1}$ will approach 1, whereas $f_1^n - f_2^n$ will approach 3. The equations for the asymptotes to the curves are $f_1 = 1.2857 + n$ and $f_2 = - 1.7143 + n$ as $n \rightarrow \infty$. Thus, the Bellman procedure has found the best policy and the amount we expect to make per transition as $n \rightarrow \infty$, namely about $1. It has also found that the state "heads" is worth about $3 more than the state "tails" after a large number of moves. However, these results are found, theoretically, only in the limit as $n \rightarrow \infty$. We have to perform many calculations for small n before they appear. It would be very useful to have a procedure which we could use if we restricted our answer to be valid only for large n. It is just such a procedure which is described in this thesis.

The policy iteration method to be proposed has the following properties:

1. The solution of the sequential decision process is reduced to solving sets of linear simultaneous equations and subsequent comparisons.

2. Each succeeding policy found in the iteration has a higher expected return per transition than the previous one.

3. The procedure will find the best long-run policy; namely, that policy which has the largest expected return per transition attainable within the realm of the problem. It will find it in a finite (usually small) number of iterations.

4. It is not necessary to apply the principle of optimality explicitly.

This policy iteration procedure can be divided into two parts, the value determination operation and the policy improvement routine; they will be described in the following sections.

The Value Determination Operation

Consider the system to be operating under a given policy. Since a policy has been selected, we may drop the subscript k and speak of probability and return matrices $\left[p_{ij}\right]$ and $\left[r_{ij}\right]$. The elements of these matrices are calculated according to the rule $p_{ij} = p_{ij}^{k}$, $r_{ij} = r_{ij}^{k}$, with $k = D_i$.

Suppose that we are going to allow the system to make transitions indefinitely and that we seek to find the policy which will maximize the average return we shall receive per transition; from now on this is by definition the optimal policy.

For any policy under which we operate, we know that the system must exhibit the behavior of a Markov process. In particular, after a large number of moves the state occupancy probabilities must converge.* Since the returns depend on the state occupancy probabilities, we expect the average return per transition to approach a limit, g, as the number of moves becomes very large. The nature of g will be more completely explained below, but suffice it to say at this time that we know g is bounded because $g \leq \underset{i,j}{\text{Max}}\, r_{ij}$, and the r_{ij} are finite. The value of g is a function of the policy or the set $\left[D_i\right]$; it may be called the gain of the policy. We seek to find that policy which maximizes g.

We shall only obtain an average return g if the system is truly in the steady state; i.e., has made n transitions, where $n \longrightarrow \infty$. If we consider the system for a finite number of moves, n, then the average

* In order to assure convergence, assume temporarily that all $p_{ij} > 0$. However, it is sufficient that the rank of $(I - P)$ be $N - 1$, where I is the unity matrix.

return per move will be different from g and will depend upon the state in which the system is started. Let us define V_i^n as the total return to be expected from operating the system for n moves starting from the state i under the given policy. The quantity V_i^n will in general be composed of two parts, a steady state part ng resulting from the behavior as $n \rightarrow \infty$, and a transient part v_i which depends only on the starting state, so that

$$V_i^n = v_i + ng \qquad \text{for large } n.$$

V_i^n may be called the value of starting the system in state i with n moves remaining; v_i is the transient value of starting the system in state i. The proof that V_i^n has the prescribed form is temporarily deferred for expository purposes.

Let us consider the operation of the system for n moves under a given policy. Remembering the definition of V_i^n, we obtain the recurrence equation

$$V_i^n = \sum_{j=1}^{N} p_{ij}(r_{ij} + V_j^{n-1})$$

This equation states that the value of being in state i with n moves remaining is equal to the weighted average of the sum of the return from a transition to the j^{th} state and the value of being in state j with $n - 1$ moves remaining. The weighting is performed with the probabilities p_{ij}, as expected. If the limiting expression for V_i^n is substituted in this equation, it becomes:

$$v_i + ng = \sum_{j=1}^{N} p_{ij}\left[r_{ij} + v_j + (n - 1)g\right]$$

or

$$v_i + ng = \sum_{j=1}^{N} p_{ij}(r_{ij} + v_j) + (n - 1)g \sum_{j=1}^{N} p_{ij}$$

However $\sum_{j=1}^{N} p_{ij} = 1$ by definition. Therefore

$$v_i + ng = \sum_{j=1}^{N} p_{ij}(r_{ij} + v_j) + (n - 1)g$$

or

$$g + v_i = \sum_{j=1}^{N} p_{ij}(r_{ij} + v_j) \qquad \text{for } i = 1, 2, \ldots, N$$

A set of N equations relating the gain and transient values to the probabilities and returns has now been obtained. However, a count of unknowns reveals that there are N v_i's to be determined plus one g, a total of N + 1 unknowns contained in the N equations. This difficulty is surmounted if we examine the results of adding a constant, a, to all v_i's.

$$g + v_i + a = \sum_{j=1}^{N} p_{ij}(r_{ij} + v_j + a)$$

$$g + v_i + a = \sum_{j=1}^{N} p_{ij}(r_{ij} + v_j) + a \sum_{j=1}^{N} p_{ij}$$

$$g + v_i = \sum_{j=1}^{N} p_{ij}(r_{ij} + v_j)$$

The addition of a constant to all v_i's leaves the equations unchanged. This implies that only the differences between v_i's are important, and that the absolute level of the v_i's is arbitrary (as in the case of gravitational potential energy and node pair voltages). Realizing this situation we may arbitrarily, as far as these equations are concerned,* set one

* The transient values which appear in the limiting expression for V_i^n are generally different from those obtained by solving the simultaneous equations. The former are called "absolute" transient values; the latter, relative. The distinction, which is mainly of theoretical importance, will be

of the v_i's to zero, say v_N. We now have N equations in N unknowns which may be solved for g and the remaining v_i's. The v_i's now have the physical interpretation that at any stage of the process $v_i - v_{i_s}$ represents the increase in expected return due to entering the system in state i rather than in some standard state i_s. This is seen by considering $V_i^n - V_{i_s}^n = v_i - v_{i_s} + ng - ng$ or $V_i^n - V_{i_s}^n = v_i - v_{i_s}$, independent of n. From now on it will be convenient to call the v_i's relative transient values (relative to the standard state i_s for which v_{i_s} is arbitrarily set to zero); these quantities will be called simply values in situations in which no ambiguity can arise.

The equation

$$g + v_i = \sum_{j=1}^{N} p_{ij}(r_{ij} + v_j)$$

can be written

$$g + v_i = \sum_{j=1}^{N} p_{ij}r_{ij} + \sum_{j=1}^{N} p_{ij}v_j$$

$$g + v_i = q_i + \sum_{j=1}^{N} p_{ij}v_j$$

where $q_i = \sum_{j=1}^{N} p_{ij}r_{ij}$ is the expected return from a single transition in the i^{th} state. Thus the solution of these equations depends only on the N quantities q_i and on the N by N $[p_{ij}]$ matrix. Since the q_i's and p_{ij}'s are functions only of the policy, we now have a system of equations which generates the g and the relative transient v_i's pertaining to a particular policy. Let us call the generation of the gain and values under a policy the value determination operation.

The Policy Improvement Routine

Having examined the value determination operation, we are ready to explore the mechanism by which a given policy is replaced by a better one. Consider our basic set of equations

$$g + v_i = \sum_{j=1}^{N} p_{ij}(r_{ij} + v_j)$$

It is possible to write an expression for g as

$$g = \sum_{j=1}^{N} p_{ij}(r_{ij} + v_j) - v_i$$

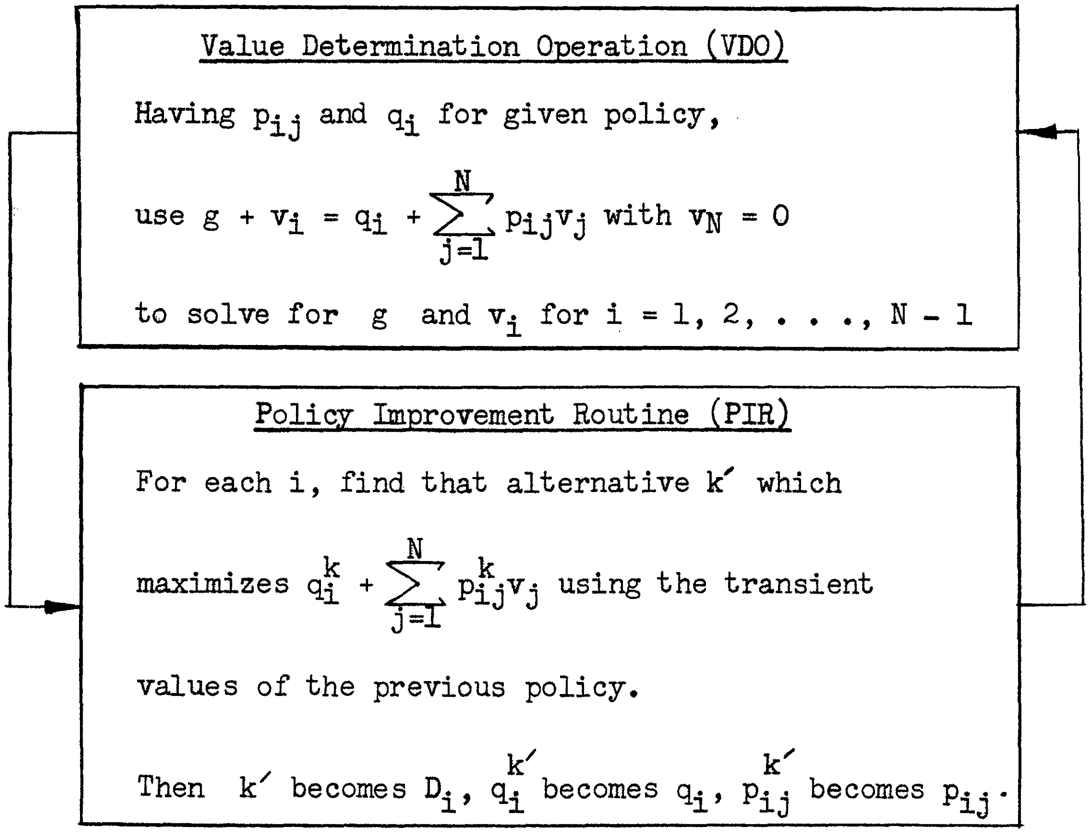
We are attempting to find a better policy, that is, one with a larger g, so that it is reasonable to attempt a maximization of the right-hand side of the above equations. Since the v_i's have been determined under the present policy, only the alternatives in each state may be varied in the maximization procedure. In each state i, we could then find the alternative k which maximizes $\sum_{j=1}^{N} p_{ij}^k(r_{ij}^k + v_j) - v_i$. Since the term v_i is independent of the maximization over the alternatives, it may be dropped and the policy improvement routine given as:

For each state i, find the alternative k which maximizes the quantity $\sum_{j=1}^{N} p_{ij}^k(r_{ij}^k + v_j)$ using the v_i's determined under the old policy. This k now becomes D_i, the decision in the i^{th} state. A new policy has been determined when this procedure has been performed for every state. If we define $q_i^k = \sum_{j=1}^{N} p_{ij}^k r_{ij}^k$ as the expected immediate return in a transition from the i^{th} state, then $q_i^k + \sum_{j=1}^{N} p_{ij}^k v_j^k$ is the quantity to be maximized.

Further properties of the policy improvement routine including the proof that it leads to a policy of higher gain will be found in a later section.

The Iteration Cycle

The basic iteration cycle may be diagrammed as follows:



Value Determination Operation (VDO)

Having p_{ij} and q_i for given policy,

use $g + v_i = q_i + \sum_{j=1}^{N} p_{ij} v_j$ with $v_N = 0$

to solve for g and v_i for $i = 1, 2, \ldots, N - 1$

Policy Improvement Routine (PIR)

For each i, find that alternative k' which

maximizes $q_i^k + \sum_{j=1}^{N} p_{ij}^k v_j$ using the transient

values of the previous policy.

Then k' becomes D_i, $q_i^{k'}$ becomes q_i, $p_{ij}^{k'}$ becomes p_{ij}.

The upper box, the value determination operation or VDO, yields the g and v_i corresponding to a given choice of q_i and p_{ij}. The lower box yields the p_{ij} and q_i which maximize the gain for a given set of v_i. In other words, the VDO yields values as a function of policy, whereas the PIR yields the policy as a function of the values.

We may enter the iteration cycle in either box. If the VDO is chosen as the entrance point, an initial policy must be selected. If the cycle is to start in the PIR, then a starting set of values is necessary. If there is no a priori reason for selecting a particular initial policy or for choosing a certain starting set of values, then it is often convenient to start the process in the PIR with all $v_i = 0$. In this case, the PIR will select a policy as follows:

For each i, find the alternative k' which maximizes q_i^k. Then set $D_i = k'$.

This starting procedure will consequently cause the PIR to select as an initial policy the one which maximizes the expected immediate return in each state. The iteration will then proceed to the VDO with this policy and the iteration cycle will begin. The selection of an initial policy which maximizes expected immediate return is quite satisfactory in the majority of cases.

At this point it would be wise to say a few words about how to stop the iteration cycle once it has done its job. The rule is quite simple: the optimal policy has been reached (g is maximized) when the policies on two successive iterations are identical. In order to prevent the PIR from quibbling over equally good alternatives in a particular state, it is only necessary to require that the old D_i be left unchanged if the alternative for that D_i is as good as any other alternative in the new policy determination.

An Example - Taxicab Operation

Before introducing more theoretical results, it should be interesting to investigate a simple example.

The coin-tossing problem is too simple from a policy point of view to serve as a real test of the method; its solution is left to the reader and to a later section.

Let us consider here the problem of a taxi driver whose area encompasses three towns, A, B, and C. If he is in town A, he has three alternatives:

1) He can cruise in the hope of picking up a passenger by being hailed.

2) He can drive to the nearest cab stand and wait in line.

3) He can pull over and wait for a radio call.

If he is in town C he has the same three alternatives, but if he is in town B the last alternative is not present because there is no radio cab service in that town. For a given town and given alternative there is a probability that the next trip will go to each of the towns A, B, and C, and a corresponding return in monetary units associated with each such trip. This return represents the income from the trip after all necessary expenses have been deducted. For example, in the case of alternatives 1 and 2, the cost of cruising and of driving to the nearest stand must be included in calculating the returns. The probabilities of transition and the returns depend upon the alternative because different customer populations will be contacted under each alternative.

If we identify being in towns A, B, and C with S_1, S_2, and S_3, respectively, then we have

State	Alternative	Probability			Return			Expected Immediate Return
i	k	p_{ij}^k			r_{ij}^k			$q_i^k = \sum_{j=1}^{N} p_{ij}^k r_{ij}^k$
		j = 1	2	3	j = 1	2	3	
1	1	1/2	1/4	1/4	10	4	8	8
	2	1/16	3/4	3/16	8	2	4	2.75
	3	1/4	1/8	5/8	4	6	4	4.25
2	1	1/2	0	1/2	14	0	18	16
	2	1/16	7/8	1/16	8	16	8	15
3	1	1/4	1/4	1/2	10	2	8	7
	2	1/8	3/4	1/8	6	4	2	4
	3	3/4	1/16	3/16	4	0	8	4.5

The return is measured in some arbitrary monetary unit; the above numbers are chosen more for ease of calculation than for any other reason.

In order to start the decision-making process, suppose we make v_1, v_2, and $v_3 = 0$, so that the policy improvement will choose the policy which maximizes expected immediate return. By examining the q_i^k we see that this policy consists of choosing the first alternative in each state. In other words, $D_1 = 1$, $D_2 = 1$, $D_3 = 1$ or we can speak of an N-dimensional column policy vector D with elements D_i, so that we can say that the policy is

$$D = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \qquad \text{namely, always cruise.}$$

The transition probabilities and immediate returns corresponding to this policy are

$$\left[p_{ij}\right] = \begin{bmatrix} 1/2 & 1/4 & 1/4 \\ 1/2 & 0 & 1/2 \\ 1/4 & 1/4 & 1/2 \end{bmatrix} \qquad \left[q_i\right] = \begin{bmatrix} 8 \\ 16 \\ 7 \end{bmatrix}$$

Now the value determination operation is entered and we solve the equations

$$g + v_i = q_i + \sum_{j=1}^{N} p_{ij} v_j \qquad i = 1, 2, \ldots, N$$

In this case we have

$$g + v_1 = 8 + 1/2\, v_1 + 1/4\, v_2 + 1/4\, v_3$$

$$g + v_2 = 16 + 1/2\, v_1 + 0\, v_2 + 1/2\, v_3$$

$$g + v_3 = 7 + 1/4\, v_1 + 1/4\, v_2 + 1/2\, v_3$$

Setting $v_3 = 0$ arbitrarily and solving these equations, we obtain

$$v_1 = 1.33$$
$$v_2 = 7.47$$
$$v_3 = 0$$
$$g = 9.2$$

Under a policy of always cruising, the driver will make 9.2 units per trip on the average.

Returning to the PIR, we calculate the quantities $q_i^k + \sum_{j=1}^{N} p_{ij}^k v_j$ for all i and k:

i	k	$q_i^k + \sum_{j=1}^{N} p_{ij}^k v_j$
1	1	10.50 *
	2	8.43
	3	5.51
2	1	16.67
	2	21.75 *
3	1	9.20
	2	9.66 *
	3	6.75

We see that for $i = 1$, the quantity in the right-hand column is maximized when $k = 1$. For $i = 2$ or 3, it is maximized when $k = 2$. In other words, our new policy is

$$D = \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix}$$

This means that if the cab is in town A, it should cruise; if it is in town B or C, it should drive to the nearest stand.

We now have

$$[p_{ij}] = \begin{bmatrix} 1/2 & 1/4 & 1/4 \\ 1/16 & 7/8 & 1/16 \\ 1/8 & 3/4 & 1/8 \end{bmatrix} \qquad [q_i] = \begin{bmatrix} 8 \\ 15 \\ 4 \end{bmatrix}$$

Returning to the VDO, we solve the equations:

$$g + v_1 = 8 + 1/2\ v_1 + 1/4\ v_2 + 1/4\ v_3$$

$$g + v_2 = 15 + 1/16\ v_1 + 7/8\ v_2 + 1/16\ v_3$$

$$g + v_3 = 4 + 1/8\ v_1 + 3/4\ v_2 + 7/8\ v_3$$

Again with $v_3 = 0$ we obtain

$$v_1 = 3.88$$

$$v_2 = 12.85$$

$$v_3 = 0$$

$$g = 13.15$$

Note that g has increased from 9.2 to 13.15 as desired, so that the cab earns 13.15 units per trip on the average. Entering the PIR with these values,

i	k	$q_i^k + \sum_{j=1}^{N} p_{ij}^k v_j$
1	1	9.27
	2	12.14 *
	3	4.91
2	1	14.06
	2	26.00 *
3	1	9.26
	2	12.02 *
	3	2.37

The new policy is thus

$$D = \begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix}$$

The cab should drive to the nearest stand regardless of the town in which it finds itself.

With this policy

$$\left[p_{ij}\right] = \begin{bmatrix} 1/16 & 3/4 & 3/16 \\ 1/16 & 7/8 & 1/16 \\ 1/8 & 3/4 & 1/8 \end{bmatrix} \qquad \left[q_i\right] = \begin{bmatrix} 2.75 \\ 15 \\ 4 \end{bmatrix}$$

Entering the VDO

$$g + v_1 = 2.75 + 1/16\ v_1 + 3/4\ v_2 + 3/16\ v_3$$
$$g + v_2 = 15 + 1/16\ v_1 + 7/8\ v_2 + 1/16\ v_3$$
$$g + v_3 = 4 + 1/8\ v_1 + 3/4\ v_2 + 1/8\ v_3$$

With $v_3 = 0$,

$$v_1 = -\ 1.18$$
$$v_2 = 12.66$$
$$v_3 = 0$$
$$g = 13.34$$

Note that there has been a small but definite increase in g from 13.15 to 13.34.

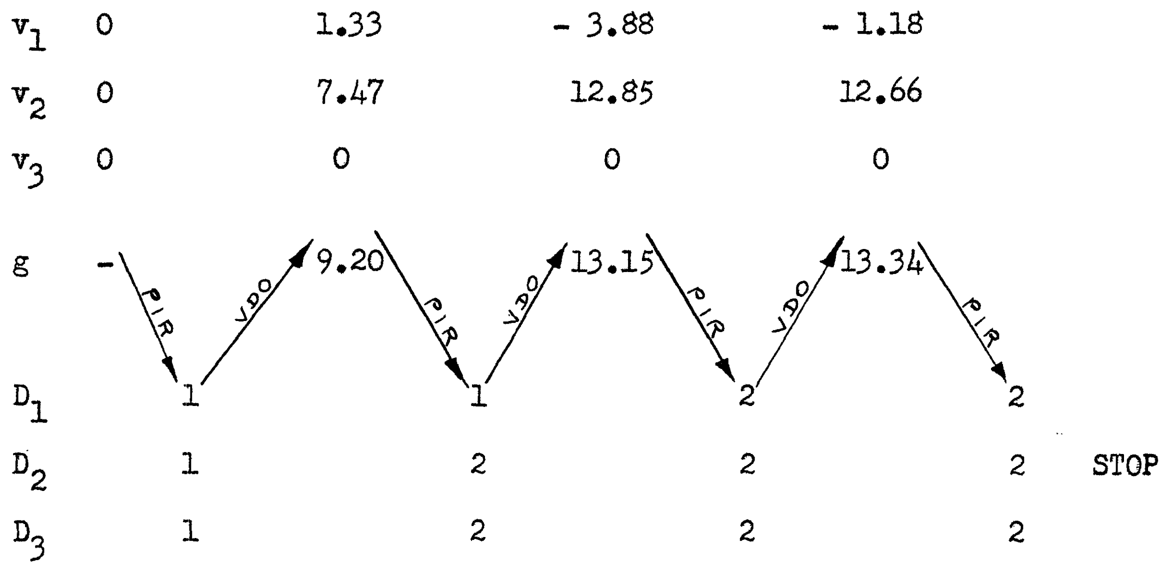
Trying the PIR

i	k	$q_i^k + \sum_{j=1}^{N} p_{ij}^k v_j$
1	1	10.57
	2	12.16 *
	3	5.53
2	1	15.41
	2	26.00 *
3	1	9.86
	2	13.33 *
	3	5.40

The new policy is

$$D = \begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix}$$

but this is equal to the last policy, so that the process has converged and g has attained its maximum; namely, 13.34. The cab driver should drive to the nearest stand in any city. Following this policy will yield a return of 13.34 units per trip on the average, almost half as much again as the policy of always cruising found by maximizing immediate return. Summarizing the calculations (to two decimal places)



v_1	0	1.33	- 3.88	- 1.18	
v_2	0	7.47	12.85	12.66	
v_3	0	0	0	0	
g	-	9.20	13.15	13.34	
D_1	1	1	2	2	
D_2	1	2	2	2	STOP
D_3	1	2	2	2	

Notice that the optimal policy of always driving to a stand is the <u>worst</u> policy in terms of immediate return. It often happens in the sequential decision process that the birds in the bush are worth more than the one in the hand.

Properties of the Value Determination Operation

Having been enlightened with an example, we may return to theoretical considerations in order to gain more insight into the value determination operation.

We must solve the following equation for the v_i's and the g.

$$g + v_i = q_i + \sum_{j=1}^{N} p_{ij} v_j \qquad i = 1, 2, \ldots, N$$

Rearranging

$$v_i - \sum_{j=1}^{N} p_{ij} v_j + g = q_i$$

Set $v_N = 0$, arbitrarily,

$$\sum_{j=1}^{N-1} (\delta_{ij} - p_{ij}) v_j + g = q_i$$

where δ_{ij} is the Kronecker delta; $\delta_{ij} = 0$ if $i \neq j$, $\delta_{ij} = 1$ if $i = j$. If we define a matrix $A = \left[a_{ij}\right]$, where

$$\begin{cases} a_{ij} = \delta_{ij} - p_{ij} & \text{for } j < N \\ a_{iN} = 1 \end{cases}$$

then

$$A = \begin{bmatrix} 1-p_{11} & -p_{12} - \cdots - p_{1,N-1} & & 1 \\ -p_{21} & & & 1 \\ \cdot & & & \\ \cdot & & & \\ -p_{N1} & & -p_{N,N-1} & 1 \end{bmatrix}$$

Note that the matrix A is formed by taking the $\left[p_{ij}\right]$ matrix, making all elements negative, adding ones to the main diagonal, and replacing the last column by ones.

If we also define a matrix V where

$$\begin{cases} v_i = v_i & i < N \\ v_N = g & \end{cases}$$

then

$$V = \begin{bmatrix} v_1 \\ v_2 \\ \cdot \\ \cdot \\ v_{N-1} \\ g \end{bmatrix}$$

Finally, let $Q = \begin{bmatrix} q_i \end{bmatrix}$ so that

$$Q = \begin{bmatrix} q_1 \\ q_2 \\ \cdot \\ \cdot \\ q_N \end{bmatrix}$$

The above equation in the v_i's and the g can then be written in matrix form as

$$A\ V = Q$$

or

$$V = A^{-1}\ Q$$

Thus by inverting A to obtain A^{-1}, and then postmultiplying A^{-1} by Q, v_i for $1 \leq i \leq N-1$ and g will be determined.

Consider matrices T, S:

$$T = \begin{bmatrix}1\\1\\1\\\cdot\\\cdot\\\cdot\\1\\1\end{bmatrix} \Big\updownarrow N \qquad\qquad S = \begin{bmatrix}0\\0\\0\\\cdot\\\cdot\\\cdot\\0\\1\end{bmatrix} \Big\updownarrow N$$

By inspection

$$T = A\ S$$

or

$$A^{-1}\ T = S$$

The sum of the rows of $A^{-1} = 0$, except for the last row whose sum is 1.

If $Q = a\ T$, where a is a constant, then

$$V = A^{-1}\ Q = A^{-1}\ a\ T = a\ A^{-1}\ T = a\ S$$

Therefore all $v_i = 0$, and $g = a$. If all expected immediate returns are equal, then all values are zero, and g is equal to the immediate return. This result checks with our interpretation of the g and v_i's.

Let us consider the effect of subjecting all returns to the linear transformation

$$r'_{ij} = ar_{ij} + b.$$

Since

$$q_i = \sum_j p_{ij}r_{ij};\ q'_i = \sum_j p_{ij}r'_{ij} = \sum_j p_{ij}(ar_{ij} + b)$$

$$q'_i = a \sum_j p_{ij}r_{ij} + b \sum_j p_{ij}$$

or

$$q'_i = aq_i + b \qquad \text{since} \sum_j p_{ij} = 1$$

In matrix form

$$Q' = a\ Q + b\ T$$

$$V' = A^{-1}\ Q' = A^{-1}\ a\ Q + b\ T$$

$$V' = a\ A^{-1}\ Q + b\ A^{-1}\ T$$

$$V' = a\ V + b\ S$$

or $$v_i' = a\ v_i \qquad \text{for all } i$$

and $$g' = a\ g + b$$

We can thus make any scale change on the returns, solve the problem with simpler numbers, and rescale the g and v_i's to correspond to the original problem.

Let us assume that all $p_{ij} > 0$, so that all states are recurrent. Further, let π_i be the probability of finding the system in state i after n transitions where $n \rightarrow \infty$. In other words, the π_i are the steady-state probabilities of the system.

The basic balancing relations on the π_i are

$$\begin{aligned}
\pi_1 &= \pi_1 p_{11} + \pi_2 p_{21} + \pi_3 p_{31} + \cdot\ \cdot\ \cdot + \pi_N p_{N1} \\
\pi_2 &= \pi_1 p_{12} + \pi_2 p_{22} + \qquad\qquad\quad + \pi_N p_{N2} \\
&\vdots \qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
\pi_N &= \pi_1 p_{1N} + \qquad\qquad\qquad\qquad + \pi_N p_{NN}
\end{aligned}$$

or $$\pi_j = \sum_i \pi_i p_{ij} \qquad \text{for all } j = 1, 2, \ldots, N.$$ N equations, N unknowns.

Summing over j

$$\sum_j \pi_j = \sum_j \sum_i \pi_i p_{ij}$$

$$\sum_j \pi_j = \sum_i \pi_i \sum_j p_{ij}$$

$$\sum_j \pi_j = \sum_i \pi_i$$

This result indicates that we do not have sufficient equations to determine the π_i. However, we are fortunate in having an equation we have not used; namely, $\sum_i \pi_i = 1$.

Replacing the last equation of the above set by the condition that the sum of the steady-state probabilities is 1, and rearranging yields:

$$\begin{aligned}
(1 - p_{11})\pi_1 - p_{21}\pi_2 - p_{31}\pi_3 \cdots\cdots\cdots - p_{N1}\pi_N &= 0 \\
- p_{12}\pi_1 + (1 - p_{22})\pi_2 - p_{32}\pi_3 \cdots\cdots\cdots - p_{N2}\pi_N &= 0 \\
&\vdots \\
- p_{1,N-1}\pi_1 - \cdots\cdots\cdots\cdots - p_{N,N-1}\pi_N &= 0 \\
\pi_1 + \pi_2 + \pi_3 + \cdots\cdots\cdots\cdots + \pi_N &= 1
\end{aligned}$$

In matrix form

$$\begin{bmatrix} 1-p_{11} & -p_{21} \cdots\cdots & -p_{N1} \\ -p_{12} & & -p_{N2} \\ \vdots & & \vdots \\ -p_{1,N-1} & & -p_{N,N-1} \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \pi_1 \\ \pi_2 \\ \vdots \\ \cdot \\ \pi_N \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

If

$$\Pi = [\pi_i] = \begin{bmatrix} \pi_1 \\ \pi_2 \\ \vdots \\ \pi_N \end{bmatrix}$$ and we notice that the matrix premultiplying Π is A^T,

then
$$A^T \Pi = S$$
or
$$\Pi = (A^{-1})^T S$$
$$\Pi^T = S^T A^{-1}$$

The matrix Π^T is thus equal to the last row of A^{-1} so that $\Pi_j = a^{-1}_{Nj}$ for $1 \leq j \leq N$.

In other words, we have shown that if we invert the matrix A, we have solved both the sequential decision problem and the Markov process.

Furthermore, since
$$V = A^{-1} Q$$
$$S^T V = S^T A^{-1} Q$$
but
$$S^T V = g \quad \text{and} \quad S^T A^{-1} = \Pi^T$$
therefore
$$g = \Pi^T Q$$
or
$$g = \sum_{i=1}^{N} \pi_i q_i$$

The exact nature of g is now clear. The gain is equal to the sum of the expected immediate returns weighted by the steady-state probabilities.

Let us refer to the last iteration of the taxi problem to illustrate the above results. At this stage,
$$D = \begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix}$$
and we seek to find the corresponding g and v_i.

For the above policy

$$\left[p_{ij}\right] = \begin{bmatrix} 1/16 & 3/4 & 3/16 \\ 1/16 & 7/8 & 1/16 \\ 1/8 & 3/4 & 1/8 \end{bmatrix} \qquad \left[q_i\right] = \begin{bmatrix} 2.75 \\ 15 \\ 4 \end{bmatrix}$$

$$A = \begin{bmatrix} 15/16 & -3/4 & 1 \\ -1/16 & 1/8 & 1 \\ -1/8 & -3/4 & 1 \end{bmatrix} \qquad A^{-1} = \begin{bmatrix} 16/17 & 0 & -16/17 \\ -8/119 & 8/7 & -128/119 \\ 8/119 & 6/7 & 9/119 \end{bmatrix}$$

$$V = A^{-1} Q = \begin{bmatrix} 16/17 & 0 & -16/17 \\ -8/119 & 8/7 & -128/119 \\ 8/119 & 6/7 & 9/119 \end{bmatrix} \begin{bmatrix} 2.75 \\ 15 \\ 4 \end{bmatrix} = \begin{bmatrix} -1.17647 \\ 12.65546 \\ 13.34454 \end{bmatrix}$$

so that $v_1 = 1.17647$, $v_2 = 12.65546$, $v_3 = 0$ (by definition), and $g = 13.34454$ as we found in our previous calculations.

We also know that $\left[\pi_1 \quad \pi_2 \quad \pi_3\right] = \left[8/119 \quad 6/7 \quad 9/119\right]$ or $\pi_1 = 0.0672$, $\pi_2 = 0.8571$ and $\pi_3 = 0.0757$.

When the driver is following the optimal policy of always driving to a stand, he will make 13.34 units per trip on the average <u>and</u> he will spend about 86% of his time in town B, and about 7% of his time in each of towns A and C. This information is useful in interpreting the nature of the optimal policy.

Properties of the Policy Improvement Routine

When the policy improvement routine was first described, it was asserted that the scheme for improving policies would lead to increases in gain. This assertion will now be proved.

The rule for policy improvement is: find k' such that

$$q_i^{k'} + \sum_{j=1}^{N} p_{ij}^{k'} v_j \geqslant q_i^k + \sum_{j=1}^{N} p_{ij}^k v_j \qquad \text{for } k \neq k'$$

Let us denote $q_i^{k'}$ by q_i', $p_{ij}^{k'}$ by p_{ij}', q_i^k by q_i, p_{ij}^k by p_{ij} in order to simplify notation. The prime always indicates the new policy, while lack of a prime indicates the former policy.

$$q_i' + \sum_j p_{ij}' v_j \geqslant q_i + \sum_j p_{ij} v_j$$

$$q_i' + \sum_j p_{ij}' v_j = q_i + \sum_j p_{ij} v_j + \gamma_i$$

where all $\gamma_i \geqslant 0$.

In addition, for any quantity x, let $x^* = x' - x$

$$q_i' - q_i + \sum_j p_{ij}' v_j - \sum_j p_{ij} v_j = \gamma_i$$

$$q_i^* + \sum_j p_{ij}' v_j - \sum_j p_{ij} v_j = \gamma_i$$

The equation for the v_i's and g under the old policy were

$$g + v_i = q_i + \sum_j p_{ij} v_j$$

The equation for the v_i''s and g' under the new policy are

$$g' + v_i' = q_i' + \sum_j p_{ij}' v_j'$$

We seek to prove $g' \geqslant g$, or $g^* \geqslant 0$.

Subtracting the above two equations,

$$g' - g + v_i' - v_i = q_i' - q_i + \sum_j p_{ij}'v_j' - \sum_j p_{ij}v_j$$

$$g^* + v_i^* = q_i^* + \sum_j p_{ij}'v_j' - \sum_j p_{ij}v_j$$

Substituting the equation relating q_i^* to γ_i

$$g^* + v_i^* = \gamma_i - \sum_j p_{ij}'v_j + \sum_j p_{ij}v_j + \sum_j p_{ij}'v_j' - \sum_j p_{ij}v_j$$

$$g^* + v_i^* = \gamma_i + \sum_j p_{ij}'v_j^*$$

Notice the similarity between this equation and the equation for the g' and v_i''s under the new policy. If we define vectors

$$V^* = \begin{bmatrix} v_1^* \\ v_2^* \\ \cdot \\ \cdot \\ \cdot \\ v_{N-1}^* \\ g^* \end{bmatrix} \qquad \text{and } \Gamma = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \cdot \\ \cdot \\ \cdot \\ \gamma_N \end{bmatrix}$$

and furthermore set $v_n^* = 0$; that is, use S_N as the standard state, then

$$A' V^* = \Gamma$$

$$V^* = A'^{-1}\Gamma$$

Since $S^T A^{-1} = \Pi^T$ as shown earlier

and $\quad S^T V^* = S^T A'^{-1}\Gamma$

then $\quad g^* = \Pi'^T \Gamma$

or $\quad g^* = \sum_{i=1}^{N} \pi_i' \gamma_i$

Now, if all $p'_{ij} > 0$, then all $\pi'_i > 0$, and since all $\gamma_i \geq 0$, then

$$g^* > 0 \qquad \text{If } \underline{\text{any}} \; \gamma_i > 0$$

$$\text{and} \quad g^* = 0 \qquad \text{If all } \gamma_i = 0$$

If an improvement can be made in any state using the PIR, then the gain of the system must increase. The proof that the PIR finds the policy with the highest gain attainable within the realm of the problem is found in a later section.

To show that

$$g^* = \sum_{i=1}^{N} \pi'_i \gamma_i$$

let us return once more to the last iteration of the taxi problem where we have already determined the π_i. Let

$$W_i^k = q_i^k + \sum_{j=1}^{N} p_{ij}^k v_j$$

Consider the case where the policy changed from $\begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix}$ to $\begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix}$ and examine the PIR calculation in which the change was discovered. Here

$W_1^1 = 9.27$ corresponds to the old decision in state 1

$W_1^2 = 12.14$ corresponds to the new decision in state 1

$$\gamma_1 = W_1^2 - W_1^1 = 12.14 - 9.27 = 2.87$$

γ_2 and γ_3 are both zero, since the decision in those states is unchanged. Therefore

$$\begin{aligned} g^* &= \pi'_1 \gamma_1 \\ &= (0.0672)(2.87) \\ &= 0.19 \end{aligned}$$

We know $g' = 13.34$ while $g = 13.15$, so that $g' - g = 0.19$ as expected.

The above development rests on the assumption that all states are recurrent, but we know that transient states often occur. In order to make our remarks apply to situations with transient states, let us consider the following problem. Suppose S_N is a recurrent state and an independent chain so that $p_{Nj} = 0$ for $j \neq N$ and $p_{NN} = 1$. Furthermore, let there be no recurrent states among the remaining N-1 states of the problem.

We know that

$$V = A^{-1} Q$$

where A assumes the special form

$$A = \left[\begin{array}{cccc|c} 1-p_{11} & -p_{12} & \cdots & -p_{1,N-1} & 1 \\ -p_{21} & & & & 1 \\ \vdots & & & & 1 \\ -p_{N-1,1} & - & - & 1-p_{N-1,N-1} & 1 \\ \hline 0 & 0 & 0 \cdots & 0 & 1 \end{array}\right]$$

Also since

$$S = \begin{array}{cc} \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} & \updownarrow N \end{array}$$

$$S^T A = S^T$$

or

$$S^T = S^T A^{-1}$$

which means that the last rows of A and A^{-1} are identical.

Let A be partitioned as follows

$$A = \left[\begin{array}{ccccc|c} & & B & & & F \\ \hline 0 & 0 & \cdots & 0 & 0 & 1 \end{array}\right]$$

where the nature of B and F are evident by comparison with the A defined above.

We know that the last row of A^{-1} is S^T. From the relations for partitioned matrices

$$A^{-1} = \left[\begin{array}{cccc|c} & & B^{-1} & & -B^{-1}F \\ \hline 0 & 0 & \cdots & 0 & 1 \end{array}\right]$$

The elements in the first N-1 rows of the last column are equal to the negative sum of the first N-1 elements in each row. Also, $g = q_N$ as expected.

What is the significance of B^{-1} and $B^{-1}F$? Let us consider the relations for the number of times the system enters each transient state before it is absorbed by the recurrent state. Let u_i equal the expected number of times the system will enter S_i before it enters S_N.

The balancing relations for the u_i are

$$\begin{aligned} u_1 &= p_{11}u_1 + p_{21}u_2 + \cdots + p_{N1}u_N + \delta_{1i_o} \\ u_2 &= p_{12}u_1 + p_{22}u_2 + \cdots + p_{N2}u_N + \delta_{2i_o} \\ &\vdots \\ u_i &= p_{1i}u_1 + p_{2i}u_2 + \cdots + p_{Ni}u_N + \delta_{ii_o} \\ &\vdots \\ u_N &= p_{1N}u_1 + p_{2N}u_2 + \cdots + p_{NN}u_N + \delta_{Ni_o} \end{aligned}$$

where

$$\delta_{ii_o} \begin{cases} = 1 & \text{if } i = i_o \\ = 0 & \text{otherwise} \end{cases}$$ and S_{i_o} is the state in which the system is started

Since $p_{Ni} = 0$ for $1 \leqslant i \leqslant N-1$, the first N-1 equations determine $u_1, u_2, \ldots, u_{N-1}$ uniquely. The determinant of this set of equations will be non-singular if no recurrent states exist in this set, and this is true by assumption. Rearranging the equations and writing them in matrix form,

$$\begin{bmatrix} 1-p_{11} & -p_{21} - p_{31} \cdots - p_{N-1,1} \\ -p_{12} & \\ \vdots & \\ -p_{1,N-1} \cdots & \cdots - p_{N-1,N-1} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{N-1} \end{bmatrix} = \begin{bmatrix} \delta_{1i_o} \\ \delta_{2i_o} \\ \vdots \\ \delta_{N-1,i_o} \end{bmatrix}$$

$$B^T U = \triangle$$

$$U = (B^T)^{-1} \triangle$$

$$U^T = \triangle^T B^{-1}$$

If the system is started in state i_o, the i_oth row of B^{-1} will yield the average number of times the system will occupy the states S_i, i = 1, 2, . . ., N-1.

The element $-a^{-1}_{iN} = \sum_{j=1}^{N-1} a^{-1}_{ij}$ is the expected number of moves in all transient states before absorption.

$$V = A^{-1} Q$$

$$v_{i_o} = \sum_{j=1}^{N-1} a^{-1}_{i_oj} q_j - g \sum_{j=1}^{N-1} a^{-1}_{i_oj}$$

$$v_{i_o} = \sum_{j=1}^{N} u_j q_j - g \sum_{j=1}^{N} u_j$$

where the u_j are calculated under the assumption that the system is started in state i_o.

An interpretation for the v_i is now possible. The v_i represent the sum of the expected number of times the system will enter each state j times the immediate expected return in that state less the total number of times any state other than S_N will be entered times the gain for S_N, all given that the system started in state i. This is a reasonable interpretation for the v_i in this special case.

In particular, if $g = 0$ for the recurrent state and all $q_i \geq 0$ for $i < N$, then

$$v_i = \sum_{j=1}^{N-1} u_j q_j \geq 0 \qquad \text{for all } i.$$

Suppose that there is only one recurrent state S_N and that a maximum g has been found in that state. Then $g^* = 0$.

From the development for the all-states recurrent case,

$$g^* + v_i^* = \gamma_i + \sum_j p'_{ij} v_j^*$$

$$V^* = (A')^{-1} \Gamma$$

since $g^* = 0$, $\gamma_N = 0$ and $v_i^* = \sum_{j=1}^{N-1} a_{ij}^{-1} \gamma_j \qquad i = 1, 2, \ldots, N-1$

We have shown above that $a_{ij}^{-1} \geq 0$ for $i, j < N-1$. Since $\gamma_j \geq 0$, $v_i^* > 0$ for $i = 1, 2, \ldots, N-1$, if any $\gamma_j > 0$.

The result is that in a situation where only S_N is recurrent, the PIR will maximize the v_i's after it has maximized the g. This property is important in applications and is essential to one of the examples presented below.

If the words "independent chain with gain g" are substituted for "single recurrent state S_N," the above development is still correct. The policy improvement routine will not only maximize the g of the independent chain all of whose states are recurrent; it will also maximize the value of the transient states which run into that chain.

The Simultaneous Equation Approach - A Baseball Problem

At this point it would be interesting to explore various methods of solving the discrete sequential decision problem. The policy improvement routine is a simple computational problem compared to the value determination operation. In order to determine the gain and the values, it is necessary to solve a set of simultaneous equations which may be quite large. In this section the advantages and disadvantages of tackling the solution of these equations by conventional methods will be investigated. In the following section a different method which makes use of certain properties of the basic sequential decision process will be described.

A 704 program for solving the problem we have been discussing has been developed as an instrument of research. This program contains both the PIR and VDO, and performs the VDO by solving a set of simultaneous equations using the Gauss-Jordan reduction. Problems possessing up to 50 states and with up to 50 alternatives in each state may be solved.

When this program was used to solve the taxicab problem, it of course yielded the same solutions we obtained earlier, but with more significant figures. The power of the technique can only be appreciated in a more complex problem possessing several states. As an illustration of such a problem, let us analyze the game of baseball using suitable simplifying assumptions to make the problem manageable.

Consider the half of an inning of a baseball game when one team is at bat. This team is unusual because all its players are identical in athletic ability and their play is unaffected by the tensions of the game. The manager makes all decisions regarding the strategy of the team and his alternatives are limited in number. He may tell the batter to hit or bunt, tell a man on first to steal second, a man on second to steal third, or a man on third to steal home. For each situation during the inning and for each alternative there will be a probability of reaching each other situation that could exist and an associated reward expressed in runs. Let us specify the probabilities of transition under each alternative as follows:

1. Manager tells player at bat to try for a hit

Outcome	Probability of Outcome	Batter Goes To	Player on First Goes To	Player on Second Goes To	Player on Third Goes To
Single	0.15	1	2	3	H
Double	0.07	2	3	H	H
Triple	0.05	3	H	H	H
Home run	0.03	H	H	H	H
Base on balls	0.10	1	2	3 (if forced)	H (if forced)
Strike out	0.30	out	1	2	3
Fly out	0.10	out	1	2	H (if less than 2 outs)
Ground out	0.10	out	2	3	H (if less than 2 outs)
Double play	0.10	out	The player nearest first is out		

The interpretation of these outcomes is not described in detail. For instance, if there are no men on base, then hitting into a double play is counted simply as making an out.

2. Manager tells player at bat to bunt

Outcome	Probability	Effect
Single	.05	Runners advance one base
Sacrifice	.60	Batter out; runners advance one base
Fielder's choice	.20	Batter safe; runner nearest to making run is out, other runners stay put unless forced
Strike or foul out	.10	Batter out; runners do not advance
Double play	.05	Batter and player nearest first are out

3. Manager tells player on first to steal second

4. Manager tells player on second to steal third

In either case, the attempt is successful with probability 0.4, the player's position is unchanged with probability 0.2, and the player is out with probability 0.4.

5. Manager tells player on third to steal home

The outcomes are the same as those above, but the corresponding probabilities are 0.2, 0.1, and 0.7.

Baseball fans please note: No claim is made for the validity of either assumptions or data.

The state of the system depends upon the number of outs and upon the situation on the bases. We may designate the state of the system by a four-digit number $d_1d_2d_3d_4$, where d_1 is the number of outs--0, 1, 2, or 3--and the digits $d_2d_3d_4$ are 1 or 0 corresponding to whether there is or is not a player on bases 3, 2, and 1, respectively. Thus the state designation 2110 would identify the situation "2 outs; players on second and third," whereas 1111 would mean "1 out; bases loaded." The states are also given a decimal number equal to 1 + 8d_1 + (decimal number corresponding to binary number $d_2d_3d_4$). The state 0000 would be state 1 and the state 3000 would be state 25, 2110 corresponds to 23, 1111 to 16. There are eight base situations possible for each of the three out situations 0, 1, 2. There is also the three out case 3--where the situation on base is irrelevant and we may arbitrarily call 3--the state 3000. Therefore we have a 25-state problem.

The number of alternatives in each state is not the same. State 1000 or 9 has no men on base so that none of the stealing alternatives are applicable and only the hit or bunt options are present. State 0101 or 6 has four alternatives: hit, bunt, steal second, or steal home. State 3000 or 25 has only 1 alternative, and that alternative causes it to return to itself with probability one and return zero. State 25 is a trapping state; it is the only recurrent state in the system.

To fix ideas still more clearly, let us explicitly list the transition probabilities p_{ij}^k and rewards r_{ij}^k for a typical state, say 0011 or 4. In state 4 (i = 4), three alternatives apply: hit, bunt, steal third. Only non-zero p_{ij}^k are listed.

a. Hit k = 1

Next state	j	p_{4j}^1	r_{4j}^1
0000	1	.03	3
0100	5	.05	2
0110	7	.07	1
0111	8	.25	0
1011	12	.40	0
1110	15	.10	0
2010	19	.10	0

$q_4^1 = .26$

b. Bunt $k = 2$

Next state	j	p^2_{4j}	r^2_{4j}
0111	8	.05	0
1011	12	.25	0
1110	15	.60	0
2010	19	.10	0

$q^2_4 = 0$

c. Steal third $k = 3$

Next state	j	p^3_{4j}	r^3_{4j}
0011	4	.20	0
0101	6	.40	0
1001	10	.40	0

$q^3_4 = 0$

The highest expected immediate return in this state would be obtained by following alternative 1, hit.

Table I, entitled "Summary of Baseball Problem Input," shows for each state i the state description, the alternative open to the manager in that state, and q^k_i, the expected immediate return (in runs) from following alternative k in state i. The final column shows the policy that would be obtained by maximizing expected immediate return in each state. This policy is to bunt in states 5, 6, 13, and 14, and to hit in all others. States 5, 6, 13, and 14 may be described as those states with a player on third, none on second, and with less than two outs.

The foregoing data was used as an input to the 704 program described earlier. Since the program chooses an initial policy by maximizing expected immediate return, the initial policy was the one mentioned above. The

TABLE I

SUMMARY OF BASEBALL PROBLEM INPUT

State Description					Alternative 1		Alternative 2		Alternative 3		Alternative 4		Number of Alternatives in State i	Initial Policy D_i if v_i set = 0
					k = 1		k = 2		k = 3		k = 4			
i		3	2	1	q_i^1		q_i^2		q_i^3		q_i^4			
1	0	0	0	0	.03	HIT	-		-		-		1	1
2	0	0	0	1	.11	HIT	0	BUNT	0	STEAL 2	-		3	1
3	0	0	1	0	.18	HIT	0	BUNT	0	STEAL 3	-		3	1
4	0	0	1	1	.26	HIT	0	BUNT	0	STEAL 3	-		3	1
5	0	1	0	0	.53	HIT	.65	BUNT	.20	STEAL H	-		3	2
6	0	1	0	1	.61	HIT	.65	BUNT	0	STEAL 2	0	STEAL H	4	2
7	0	1	1	0	.68	HIT	.65	BUNT	.20	STEAL H	-		3	1
8	0	1	1	1	.86	HIT	.65	BUNT	.20	STEAL H	-		3	1
9	1	0	0	0	.03	HIT	-		-		-		1	1
10	1	0	0	1	.11	HIT	0	BUNT	0	STEAL 2	-		3	1
11	1	0	1	0	.18	HIT	0	BUNT	0	STEAL 3	-		3	1
12	1	0	1	1	.26	HIT	0	BUNT	0	STEAL 3	-		3	1
13	1	1	0	0	.53	HIT	.65	BUNT	.20	STEAL H	-		3	2
14	1	1	0	1	.61	HIT	.65	BUNT	0	STEAL 2	.20	STEAL H	4	2
15	1	1	1	0	.68	HIT	.65	BUNT	.20	STEAL H	-		3	1
16	1	1	1	1	.86	HIT	.65	BUNT	.20	STEAL H	-		3	1
17	2	0	0	0	.03	HIT	-		-		-		1	1
18	2	0	0	1	.11	HIT	0	BUNT	0	STEAL 2	-		3	1
19	2	0	1	0	.18	HIT	0	BUNT	0	STEAL 3	-		3	1
20	2	0	1	1	.26	HIT	0	BUNT	0	STEAL 3	-		3	1
21	2	1	0	0	.33	HIT	.05	BUNT	.20	STEAL H	-		3	1
22	2	1	0	1	.41	HIT	.05	BUNT	0	STEAL 2	.20	STEAL H	4	1
23	2	1	1	0	.48	HIT	.05	BUNT	.20	STEAL H	-		3	1
24	2	1	1	1	.66	HIT	.05	BUNT	.20	STEAL H	-		3	1
25	3	-	-	-	0	TRAPPED	-		-		-		1	1

machine had to solve the equations only twice to reach a solution. Its results are summarized in Table II.

The optimal policy is to hit in every state. The v_i have the interpretation of being the expected number of runs that will be made if the game is now in state i and it is played until three outs are incurred. Since a team starts each inning in state 1, or "no outs, no men on," then v_1 may be interpreted as the expected number of runs per inning under the given policy. The initial policy yields 0.75034 for v_1, whereas the optimal policy yields 0.81218. In other words, the team will earn about .06 more runs per inning on the average if it uses the optimal policy rather than the policy which maximizes immediate expected return.

Note that under both policies the gain was zero as expected since after an infinite number of moves the system will be in state 25 and will always make return zero. Note also that in spite of the fact that the gain could not be increased, the policy improvement routine yielded values for the optimal policy which are all greater than or equal to those for the initial policy. This gratifying result was proved as the last proof in the section entitled "Properties of the Policy Improvement Routine." That proof clearly applies here because state 25 is a recurrent state, its gain is zero, and all other states of the system are transient.

The values v_i can be used in comparing the usefulness of states. For example, under either policy the manager would rather be in a position with two men out and bases loaded than be starting a new inning (compare v_{24} with v_1). However, he would rather start a new inning than have two men out and men on second and third (compare v_{23} with v_1). Many other interesting comparisons can be made. Under the optimal policy, having no men out and a

TABLE II

RESULTS OF USING SIMULTANEOUS EQUATION APPROACH ON BASEBALL PROBLEM

Iteration 1				Iteration 2			
$g = 0$				$g = 0$			
State	Description	Decision	Value, v_i	State	Description	Decision	Value, v_i
1	0000	Hit	0.75034	1	0000	Hit	0.81218
2	0001	"	0.18284	2	0001	"	1.24726
3	0010	"	1.18264	3	0010	"	1.34743
4	0011	"	1.82094	4	0011	"	1.88536
5	0100	Bunt	1.18079	5	0100	"	1.56106
6	0101	"	1.56329	6	0101	"	2.06786
7	0110	Hit	2.00324	7	0110	"	2.16803
8	0111	"	2.67094	8	0111	"	2.73536
9	1000	"	0.43383	9	1000	"	0.45604
10	1001	"	0.74878	10	1001	"	0.77099
11	1010	"	0.78970	11	1010	"	0.85999
12	1011	"	1.21278	12	1011	"	1.23499
13	1100	Bunt	0.88487	13	1100	"	1.10629
14	1101	"	1.10228	14	1101	"	1.44499
15	1110	Hit	1.46370	15	1110	"	1.53399
16	1111	"	1.93278	16	1111	"	1.95499
17	2000	"	0.17349	17	2000	"	0.17349
18	2001	"	0.33979	18	2001	"	0.33979
19	2010	"	0.39949	19	2010	"	0.39949
20	2011	"	0.58979	20	2011	"	0.58979
21	2100	"	0.50749	21	2100	"	0.50749
22	2101	"	0.67979	22	2101	"	0.67979
23	2110	"	0.73949	23	2110	"	0.73949
24	2111	"	0.98979	24	2111	"	0.98979
25	3000	"	0.	25	3000	"	0.

player on first is just about as valuable a position as having one man out and players on first and second (compare v_2 with v_{12}). It is interesting to see how the comparisons made above compare with our intuitive notions of the relative values of baseball positions.

Unfortunately, not all problems can be solved using a straightforward simultaneous equation approach. For example, there may be systems with two or more independent recurrent chains. Each chain would in general have its own gain, and the simultaneous equations could not be solved because their determinant would be singular. The extension of the simultaneous equation method to this case is found in a later section.

Another kind of difficulty arises when the equations are poorly determined or when the number of equations is so large that accurate solutions cannot be obtained. There are certain computational techniques that can help when these situations are encountered.

Rather than try to deal with these difficulties as separate and distinct problems, it would be useful to have a method that could handle the general case without apology. The simulation approach described in the following section is such a method.

The Simulation Approach

In order to describe the simulation technique, it will be helpful to examine its basic building blocks individually. These blocks are the main line simulation, branch line simulation, recurrent state search, and the policy improvement routine.

The Main Line Simulation

The function of the main line simulation is to determine the gain and the value of the most frequently entered states in a recurrent chain. It operates as follows. A standard state i_s which is assumed to be recurrent is used as a base of operations. If the system is started in state i_s and allowed to make transitions according to the probability distribution associated with a given policy, then sooner or later the system will again enter state i_s. The sequence of transitions from i_s to i_s is called a "period" of the system. In the course of a period every time the system enters a state i, a "run" is said to be started in state i. A run is terminated at the end of a period when the system enters state i_s. The number of moves, n_i, and the total amount of return, T_i, in all runs from state i are recorded. The number of runs from state i is given the symbol m_i. Let T_{ir} and n_{ir} be the return and the number of moves in the r^{th} run from state i. The expected return from n moves starting from state i is given by

$$V_i^n = v_i + ng$$

the defining equation for V_i^n.

Similarly for i_s,

$$V_{i_s}^n = v_{i_s} + ng$$

Suppose it requires n_{ir} moves to reach i_s from i during the r^{th} run. Then T_{ir} is an estimate of

$$V_i^n - V_{i_s}^{n-n_{ir}} = \left[v_i + ng\right] - \left[v_{i_s} + (n - n_{ir})g\right]$$

or

$$T_{ir} \approx v_i - v_{i_s} + n_{ir}g$$

If v_{i_s} is set = 0, arbitrarily, then

$$T_{ir} \approx v_i + n_{ir}g$$

If this equation is summed from r = 1 to m_i, then

$$T_i = m_i v_i + n_i g$$

where the T_i, m_i, and n_i are as defined above. In order to improve the estimation by T_i, the system is allowed to run for several periods and the T_i, n_i, and m_i are summed for all periods before any calculations of g and v_i's are performed. m_i, n_i, and T_i will now be the number of runs from state i, the number of moves in those runs, and the total return in those runs, respectively, for an arbitrary number of periods. Since $v_{i_s} = 0$, then $T_{i_s} = n_{i_s}g$, or g can be computed as the ratio of T_{i_s} to n_{i_s}. T_{i_s} is the total amount of return the system has made in all its moves, and n_{i_s} is the total number of moves that it has made.

For any $i \neq i_s$,

$$T_i = m_i v_i + n_i g$$

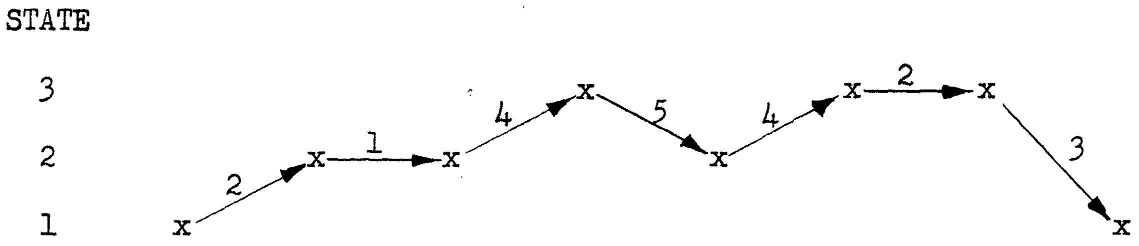
or

$$v_i = \frac{T_i - n_i g}{m_i}$$

The main line simulation has thus calculated estimates of g and the v_i's by observing the behavior of the system under the given policy. In order to avoid estimating a v_i for a state which has not been entered very often, a minimum number of runs is set, such that if m_i is below this minimum a v_i will not be calculated. If m_i is above this minimum, v_i is calculated, and the state i is said to be "established." The minimum itself is called "the number of runs to fix establishment." After the main line simulation has evaluated all established states, it goes to the branch line simulation.

It is possible that the state i_s which was assumed to be recurrent is in fact not recurrent, so that the system will be unable to complete a period. A parameter called "the number of moves in the transient state test" is specified such that if the number of moves in a period exceeds this parameter, the system does not try to complete its period but instead goes to the recurrent state search.

Let us consider a typical period of a three-state system as an example. A diagram of such a period might look as follows, if $i_s = 1$.



The numbers on the arrows represent the returns from the transitions. The return matrix for the system would look like

$$\left[r_{ij}\right] = \begin{bmatrix} - & 2 & - \\ - & 1 & 4 \\ 3 & 5 & 2 \end{bmatrix}$$

where dashes represent returns for transitions which do not occur in this period. The method of computation of the g and v_i's is shown in the following table.

STATE	No. of runs from state i = number of times state i is entered	Run No.	Move Sequence	No. of moves in run	No. of moves in all runs from state i	Return series for run	Total return from run	Total return in all runs from state i
i	m_i				n_i			T_i
1	1	1	12232331	7	7	2+1+4+5+4+2+3	21	21
2	3	1	2232331	6	14	1+4+5+4+2+3	19	46
		2	232331	5		4+5+4+2+3	18	
		3	2331	3		4+2+3	9	
3	3	1	32331	4	7	5+4+2+3	14	22
		2	331	2		2+3	5	
		3	31	1		3	3	

From the table,

$$g = \frac{T_{i_s}}{n_{i_s}} = \frac{T_1}{n_1} = \frac{21}{7} = 3 \qquad v_1 = 0 \text{ by definition}$$

$$v_2 = \frac{T_2 - n_2 g}{m_2} = \frac{46 - 3\text{x}14}{3} = 4/3$$

$$v_3 = \frac{T_3 - n_3 g}{m_3} = \frac{22 - 7\text{x}3}{3} = 1/3$$

Of course, it would be unwise to calculate g and the v_i's on the basis of such a limited amount of data. Hundreds of periods may be necessary before meaningful results can be obtained.

Since we are treating the general case in which several independent recurrent chains may exist, it will prove advantageous to attach a gain, g_i, to each state of the system. All states which are established in a single main line simulation will have the same g_i. Since all states will not in general be established in a single main line simulation, some method of evaluating the unestablished states is necessary. Such a method is afforded by the branch line simulation.

The Branch Line Simulation

Consider the general equation for the expected return in p moves starting from the state i

$$V_i^p = v_i + pg$$

Suppose that after n moves, $n < p$, the system reaches a state i_e whose value has been established. If $r = p - n$,

$$V_i^p = V_i^n + V_{i_e}^r$$

but
$$V_{i_e}^r = v_{i_e} + rg$$

Substituting

$$v_i + (r + n)g = V_i^n + v_{i_e} + rg$$

or
$$v_i = V_i^n + v_{i_e} - ng$$

Let the system be placed in an unestablished state i and allowed to run until an established state is reached. Such an event will be called a run from state i. A record is kept of the total amount of return from transitions and to this is added the value of the established state; call the record T. Then, in a single run of n moves, an estimate of v_i will be

$$v_i = T - ng$$

where g is the gain of the established state, g_{i_e}. If m runs are made to insure statistical regularity, and if the T and n counters are made cumulative, then

$$v_i = \frac{T - ng}{m} \qquad \text{and} \qquad g_i = g_{i_e}$$

Thus we have a method for finding the values of the unestablished states in terms of the values of the established states. Because T contains returns as well as v_{i_e}'s, the values of the states established using the branch line simulation will in general be known more accurately than the values of the established states into which they run. The branch line simulation is a powerful tool for evaluation of transient states or for evaluating recurrent states which have a low a priori probability. For consistency, the branch line simulation makes a number of runs equal to the "number of runs to fix establishment."

In normal operation, the branch line simulation will proceed sequentially through all states, evaluating those that have not been established in the main line simulation. Once all states are established, it prints out all gains and values and enters the policy improvement routine. It may occur that in trying to evaluate a certain state the branch line simulation will find itself in a recurrent chain that has not previously been evaluated in a main line simulation. In order to have it perform properly in this situation, there is a counter which counts the number of moves the system makes in a single run. If this counter exceeds a constant called "the number of moves in the transient state test," the branch line simulation goes to the recurrent state search.

Because of the presence of independent chains, it is possible that a state to be evaluated will have runs which end in states possessing different g values. In this case the branch line simulation will assign the state a g which is the average of all gains reached with weightings proportional to the number of runs in which each g_{i_e} is attained. The value is calculated from the equation given above, but using this average g.

The Recurrent State Search

The recurrent state search is a simple block which allows the system to make a fixed number of moves from its present position. The number is called, appropriately enough, "the number of moves in the recurrent state search." At the conclusion of these moves, this routine chooses the most frequently entered state, makes it the new standard state, and exits to the main line simulation. The sole function of this routine is to find a recurrent state for use as the standard state of the main line simulation.

The Policy Improvement Routine

The policy improvement routine is almost exactly like that described before. Its one modification is prompted by the fact that some states may have transitions to chains with different gains. Since an increase in g_i always overrides an increase in v_i, the decision in each state should be made on the basis of gain, if possible. However, if all alternatives yield transitions to states having the same gain, then the decision must be made according to value, as before. The policy improvement routine implements these remarks.

The final function of the policy improvement routine is to ask whether the new policy is equal to the old one. If this is the case, the program stops; otherwise, it exits to the main line simulation. The flow chart of the entire simulation is given in Figure 2. In order to see the system

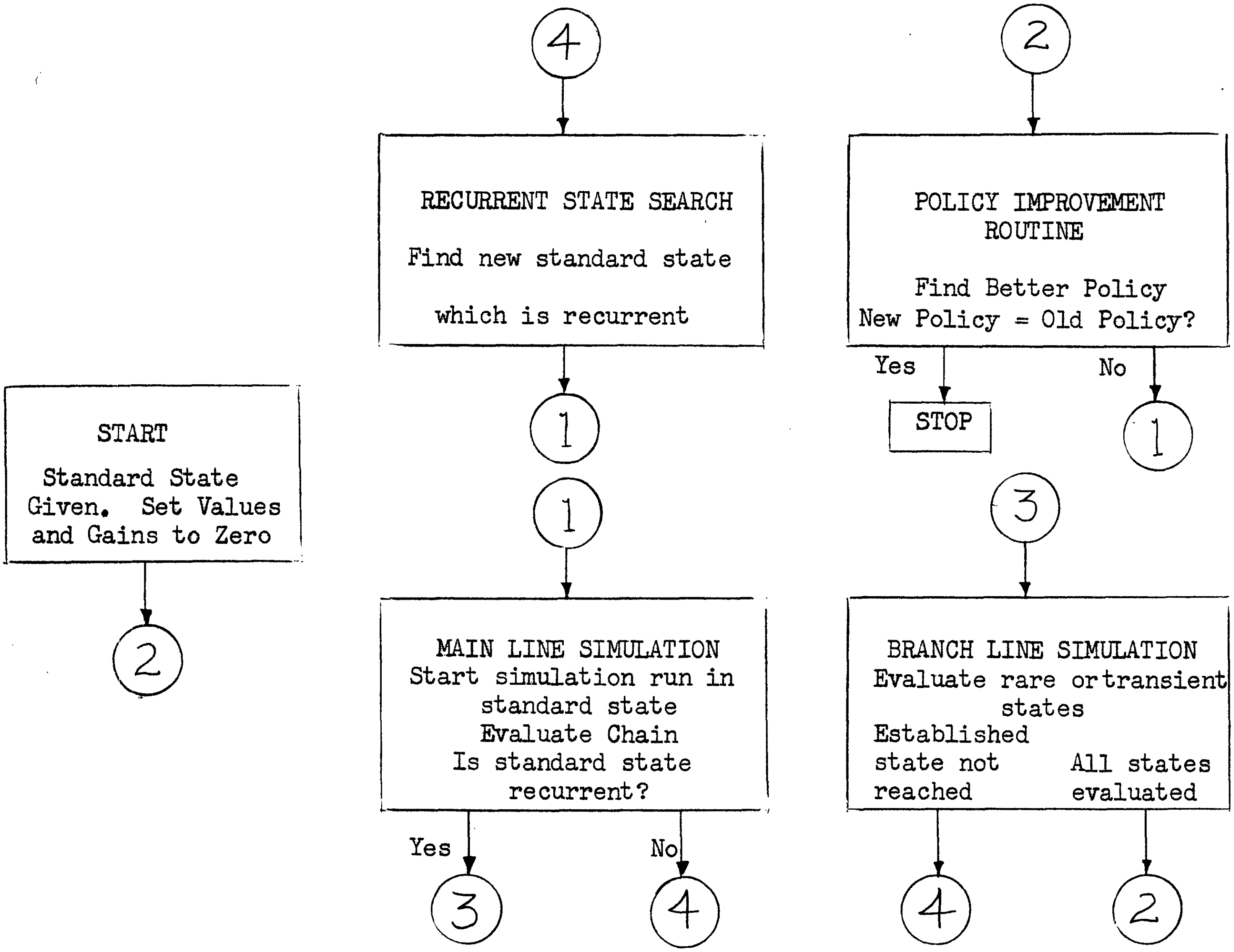
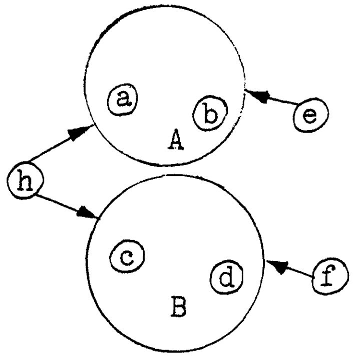

START
Standard State Given. Set Values and Gains to Zero
2
4
RECURRENT STATE SEARCH
Find new standard state which is recurrent
1
1
MAIN LINE SIMULATION
Start simulation run in standard state
Evaluate Chain
Is standard state recurrent?
Yes
No
3
4
2
POLICY IMPROVEMENT ROUTINE
Find Better Policy
New Policy = Old Policy?
Yes
No
STOP
1
3
BRANCH LINE SIMULATION
Evaluate rare or transient states
Established state not reached
All states evaluated
4
2

Figure 2

SIMULATION FLOW CHART

in action as a whole, let us examine the way it would solve the general two-chain problem.



A and B are two independent recurrent chains with different g values. States a and b are two members of A, but b occurs very infrequently compared to a. Corresponding remarks apply to states c and d of chain B. State e is a transient state running into A; f is a transient state running into B. State h is a "split" transient state which may enter either A or B.

Suppose a is picked as the standard state. Then the states of A will be evaluated in the main line simulation, with the possible exception of states like b. States b and e would be evaluated in the branch line simulation. However, if a branch line simulation is tried on f, no state of known value will be reached and the recurrent state search will be entered. It will pick a recurrent state in B, say c, as the standard state and the main line simulation will be entered once more, f will be established and so will h. State d could be established in either the main line or the branch line.

It is possible to follow the operation of the simulation method in a situation by means of the flow chart. For example, suppose e is initially picked as the standard state. Then the main line simulation will enter the recurrent state search and the main line simulation will be entered again using some recurrent state in A like a as the standard state. The simulation approach is effective in unraveling the complexities of the general problem.

A 704 program has been developed to implement the logic outlined above. It is capable of solving problems with the same numbers of states and alternatives as the simultaneous equation program, but it can work on arbitrarily complicated chain interconnections.

The taxicab problem solved earlier was solved using the simulation program. The results were as follows:

		Exact	Simulation		
Iteration		Solution	Run 1	Run 2	Run 3
1	v_1	1.33333	1.24	1.59	1.30
	v_2	7.46667	7.37	7.65	7.46
	g	9.20000	9.18	9.19	9.21
2	v_1	-3.87879	-5.01	-4.79	-4.61
	v_2	12.84849	13.18	12.91	13.23
	g	13.15152	13.08	13.26	13.58
3	v_1	-1.17647	-1.24	-1.48	-1.19
	v_2	12.65547	12.70	12.50	12.98
	g	13.34454	13.45	13.31	13.55

v_3 is set equal to zero arbitrarily. The simulation program made the same policy changes as the simultaneous equation program in every run. The number of periods in the main line simulation and the number of runs to fix establishment were 1000. The accuracy of results is within a few percent. The subsidiary quantities T_i, n_i, and m_i observed in the runs were compared with exact values; in this case agreement was within one percent.

In order to test some of the operating features of the simulation program, it was used on the baseball data. The standard state was given as state 1; a state which is far from being recurrent. The simulation program discovered that state 25 is indeed the only recurrent state and made state 25 the standard state. It then proceeded to use the branch line simulation to evaluate all other states. The simulation approach produced the same solution as the simultaneous equation approach to within two significant figures.

The simulation technique should prove to be a powerful computational tool; the ability to handle complicated interconnections makes it extremely flexible. By varying the parameters of the program, it is possible to achieve a wide range of behavior. For instance, relatively short runs could be made and the values roughly determined for a fast improvement of the policy. When the policy was close to optimal, longer runs could be made to pin down the values and gains exactly, or the simultaneous equation approach could be used for the final stage since the structure of the problem would be known. One of the attractive features of the simulation approach is that the states which are entered most frequently are those whose values are most accurate; it gives the best answers to the most important questions.

Concerning the Gain, g

In an earlier section the gain, g, was introduced in a heuristic fashion, and then given substance by the result that $g = \sum_{i=1}^{N} \pi_i q_i$; namely that the gain is the sum of the expected immediate returns weighted by the absolute state probabilities. Later, in discussing the simulation approach, a subscript i is attached to each g and it is stated that all states belonging to one chain have the same gain, whereas different chains have different gains. At this point, it would seem wise to amplify the previous remarks by examining the process on a more fundamental level. In particular, the existence of the limit

$$\lim_{n \to \infty} \frac{V_i^n}{n} = g_i$$

will be investigated, and the properties of the limits g_i will be determined.

Consider the basic recurrence relation

$$V_i^{n+1} = q_i + \sum_{j=1}^{N} p_{ij} V_j^n \quad .$$

Suppose the final value distribution V_j^o is given. Then

$$V_i^1 = q_i + \sum_{j=1}^{N} p_{ij} V_j^o$$

and

$$V_i^2 = q_i + \sum_{j=1}^{N} p_{ij} V_j^1$$

or

$$V_i^2 = q_i + \sum_{m=1}^{N} p_{im} \left[q_m + \sum_{j=1}^{N} p_{mj} V_j^o \right]$$

$$= q_i + \sum_{m=1}^{N} p_{im} q_m + \sum_{m=1}^{N} p_{im} p_{mj} V_j^o$$

where appropriate changes in indices have been made. According to our matrix definitions for P, Q, and

$$V^n = \begin{bmatrix} V_1^n \\ V_2^n \\ \cdot \\ \cdot \\ \cdot \\ V_N^n \end{bmatrix}$$

$$V^2 = Q + PQ + P^2V^o$$

$$V^3 = Q + PQ + P^2Q + P^3V^o$$

In general,

$$V^n = Q + PQ + \ldots + P^{n-1}Q + P^nV^o$$

$$V^n = \left[I + P + \ldots + P^{n-1}\right] Q + P^nV^o$$

$$V^n = \left[P + \ldots + P^{n-1} + P^n\right] Q + P^n\left[V^o - Q\right] + Q$$

$$\frac{V^n}{n} = \frac{1}{n}\sum_{m=1}^{n} P^mQ + \frac{1}{n} P^n\left[V^o - Q\right] + \frac{1}{n} Q$$

$$\lim_{n\to\infty} \frac{V^n}{n} = \left[\lim_{n\to\infty} \frac{1}{n}\sum_{m=1}^{n} P^m\right] Q$$

since the last two terms on the right vanish as n→∞. The limit on the right-hand side is a Cesàro limit; it exists for any stochastic matrix without restriction. Let this limiting N by N stochastic matrix be given the symbol F.

$$F = \lim_{n\to\infty} \frac{1}{n}\sum_{m=1}^{n} P^m$$

Then

$$\lim_{n\to\infty} \frac{V^n}{n} = FQ$$

or

$$\lim_{n\to\infty} \frac{1}{n} V_i^n = \sum_{j=1}^{N} f_{ij}q_j$$

The matrix F is the limit transition probability matrix and has the properties F = FP = PF and $F = F^2$. If a matrix $G = \lim_{n \to \infty} \frac{1}{n} V_i^n$ is defined, where

$$G = \begin{bmatrix} g_1 \\ g_2 \\ \cdot \\ \cdot \\ \cdot \\ g_N \end{bmatrix}$$

then G = FQ, $g_i = \sum_{j=1}^{N} f_{ij} q_i$. By referring to the properties of the F matrix developed in Doob,[3] it is possible to infer the properties of the G matrix, or in other words, the properties of the gains of the system.

The limit matrix F has all rows equal if and only if there is only one recurrent chain in the system. In this case all elements of the g matrix are equal and there exists a unique gain for the system. The value of g is equal to $\sum_{j=1}^{N} f_{ij} q_j$ for any i, where the f_{ij} now have an interpretation as the absolute state probabilities π_j and $g = \sum_{j=1}^{N} \pi_j q_j$ as before. Furthermore, if there are no transient states, then all $\pi_j > 0$.

If there are two or more recurrent chains in the system, then the rows of the F matrix corresponding to the states of a single chain A will all be equal so that all g_i for these states will be equal to, say, g_A. Similarly the rows corresponding to states of chain B will have a g_B, etc. If a transient state always enters the same recurrent chain, then its row of F will be equal to the F rows corresponding to the chain, and it will share the gain of that chain. If there are split transient states which enter two or more chains, the F rows corresponding to the split transient states will be linear combinations of the typical F rows for each recurrent chain. Consequently, the gain for such states will be a linear combination of the gains of the chains into which they run. For example, if state 4 is a split transient state which may enter chain A with probability 1/3 and chain B with probability 2/3, then $g_4 = 1/3\ g_A + 2/3\ g_B$.

The limit matrix F exists even in the case where there are periodic chains in the system. In this case the division by n as the limit is taken performs essentially a time average, with the result that the rows of F corresponding to such chains represent the fraction of the time the system spends in each of the respective states of such a chain. This interpretation is quite useful for our purposes, since it reveals the nature of the gain for periodic chains.

Before leaving this section, it is worth while to mention two other properties of the matrix F. First, if $p_{ij} = p_{ji}$ so that P is symmetric, then F is symmetric, and there can be no periodic chains or transient states. Furthermore, if there are N_A states in chain A, then $\pi_i = \frac{1}{N_A}$ for all states i which are in chain A. In particular, if there is only one recurrent chain, $\pi_i = \frac{1}{N}$ for all i. Second, if P is not necessarily symmetric, but is a doubly stochastic matrix so that $\sum_{i=1}^{N} p_{ij} = 1$, then the π_i follow the same rules as for symmetric P matrices.

Now that the exact nature of the g_i are understood, let us return to the basic recurrence equation with the more rigorous limiting expression for V_i^n for large n, namely

$$V_i^n = v_i + ng_i$$

$$V_i^{n+1} = q_i + \sum_{j=1}^{N} p_{ij} V_j^n$$

$$v_i + (n + 1)g_i = q_i + \sum_{j=1}^{N} p_{ij}\ v_j + ng_j$$

$$v_i + ng_i + g_i = q_i + \sum_{j=1}^{N} p_{ij}v_j + n\sum_{j=1}^{N} p_{ij}g_j$$

Since this equation must hold for all sufficiently large n, two sets of equations may be written:

$$g_i = \sum_{j=1}^{N} p_{ij}g_j \qquad i = 1, 2, \ldots, N$$

$$v_i + g_i = q_i + \sum_{j=1}^{N} p_{ij}v_j \qquad i = 1, 2, \ldots, N$$

There now exist 2N equations in the 2N unknowns v_i and g_i. Note that the second set of equations is identical with the equations for transient values which were derived earlier with the exception that there is now a gain associated with each state. The first N equations in the g_i's are especially interesting because they are far from independent. They may be written in the homogeneous form

$$\sum_{j=1}^{N} \left[\delta_{ij} - p_{ij}\right]g_j = 0$$

If the rank of the coefficient matrix is R, then the solution will involve N - R undetermined constants. R can never be equal to N because the matrix $\left[\delta_{ij} - p_{ij}\right]$ is always singular. If there is only one recurrent chain, R = N - 1, and there is only one undetermined constant, say g, which is the gain of the entire system. In this case all $g_i = g$. In general, if there are k independent chains in the system, then R = N - k and there will be k different g_i's. The transient states will have the gain of the chain into which they run, whereas split transient states will have a gain which is a linear combination of the gains of the relevant chains.

By way of illustration, let us consider our old friend, the final transition matrix of the taxicab problem.

$$\left[p_{ij}\right] = \begin{bmatrix} 1/16 & 3/4 & 3/16 \\ 1/16 & 7/8 & 1/16 \\ 1/8 & 3/4 & 1/8 \end{bmatrix} \quad \text{which yields the equations} \quad \begin{aligned} 15/16g_1-3/4g_2-3/16g_3 &= 0 \\ -1/16g_1+1/8g_2-1/16g_3 &= 0 \\ -1/8g_1 -3/4g_2+7/8g_3 &= 0 \end{aligned}$$

The rank of the coefficient matrix is equal to 2 so that the solution of the equations is $g_1 = g_2 = g_3$, or all g_i can be replaced by g and there is only one recurrent chain.

As a second example consider the following matrix

$$\left[p_{ij}\right] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1/2 & 1/2 & 0 \\ 1/3 & 2/3 & 0 & 0 \end{bmatrix}$$

where states 1 and 2 are each independent recurrent chains, state 3 is a transient state which enters state 2, and state 4 is a split transient state which may enter either chain. In this case the coefficient matrix of the g_i equations has rank 2 and the solutions are $g_3 = g_2$; $g_4 = 1/3g_1 + 2/3g_2$. The gains g_1 and g_2 are independent and the gains for the two types of transient states are related to them as described above.

If there are k independent chains in the system, then there will be k independent g_i's to be determined from the transient value equations. With N v_i's and k g_i's there is a total of N + k unknowns to be determined from N equations. The modification of our earlier procedure now required is that we may set to zero arbitrarily the value of one state in each of the independent chains. The equations may then be solved for the remaining N - k values and the k gains.

The conclusions drawn from examination of the g_i equations are exactly the same as those obtained from the properties of the F matrix. The added advantage of the equation approach is that it offers a method of solution for gains and values even in the case of independent chains and split transient states.

Concerning the Transient Values, v_i

In the initial exposition it was stated that for large n the total expected return in n moves starting from state i, V_i^n, could be expressed as the sum of two quantities: $V_i^n = v_i + ng$. The v_i, or transient values, are independent of n but dependent on i, whereas the ng term depends only on n and not on i. In the light of the previous section, this relation should be modified to $V_i^n = v_i + ng_i$, where all g_i are the same if there is only one recurrent chain in the system. It has already been shown above that $\lim\limits_{n \to \infty} \frac{V_i^n}{n} = g_i$. In order to place the defining equation for transient values on really firm ground, it remains only to show the existence of the limit:

$$\lim_{n \to \infty} (V_i^n - ng_i) = v_i$$

as follows.

Suppose X_k is the k^{th} left-sided characteristic vector of the P matrix and that it corresponds to the characteristic value λ_k; $X_k P = \lambda_k X_k$. Further, assume that all characteristic values are distinct so that the characteristic vectors are independent. It is a readily verified property of a stochastic matrix that at least one characteristic value must equal one, and that no characteristic value may be larger than one in magnitude.

Consider some arbitrary initial state vector Π^{o} which represents the probability that the system initially occupies each of its states. It is possible to express Π^{o} in the form

$$\Pi^{o} = \sum_{k} c_k X_k$$

where the c_k are appropriately chosen constants. From the properties of the transition matrix P,

$$\begin{aligned} \Pi^{1} &= \Pi^{o} P \\ \Pi^{2} &= \Pi^{1} P = \Pi^{o} P^{2} \\ \Pi^{n+1} &= \Pi^{o} P^{n} \end{aligned}$$

where Π^{n} represents the probability that the system will occupy each of its states on the n^{th} move. Also,

$$\Pi^{n+1} = \Pi^{o} P^{n} = \sum_{k} c_k X_k P^{n}$$

Since

$$\begin{aligned} X_k P &= \lambda_k X_k \\ X_k P^{2} &= \lambda_k X_k P = \lambda_k^{2} X_k \\ X_k P^{n} &= \lambda_k^{n} X_k \end{aligned} \quad ,$$

$$\Pi^{n+1} = \sum_{k} c_k \lambda_k^{n} X_k$$

Since

$$\lim_{n \to \infty} \lambda_k^{n} = 0 \qquad \text{if } |\lambda_k| < 1 \quad ,$$

$$\lim_{n \to \infty} \Pi^{n} = \sum_{K_1} c_k X_k$$

where K_1 are those indices corresponding to characteristic values which equal one. This incidentally proves the existence of and defines the absolute state probabilities of the system. We are more interested, however, in the existence of transient values than in the state probabilities. If r^n is the expected return on the n^{th} move if the system starts with a state probability distribution Π^o, $r^{n+1} = \Pi^{n+1}Q = \sum_i \pi_i^{n+1} q_i$.

$$r^{n+1} = \sum_i \sum_k c_k \lambda_k^n x_{ki} q_i$$

where x_{ki} is the i^{th} component of characteristic vector X_k.

$$r^{n+1} = \sum_k c_k \lambda_k^n \sum_i x_{ki} q_i$$

$$= \sum_k c_k \lambda_k^n r_k$$

where r_k is the expected reward corresponding to characteristic vector X_k; $r_k = \sum_i x_{ki} q_i = X_k Q$.

As mentioned above, r^n is the expected return on the n^{th} move; however, we are really seeking V^n, the total expected return in n moves if the system has an initial state probability distribution Π^o--we may call V^n the value of starting with Π^o.

$$V^n = \sum_{j=1}^{n} r^j = \sum_{j=1}^{n} \sum_k c_k \lambda_k^{j-1} r_k$$

$$= \sum_k c_k r_k \sum_{j=1}^{n} \lambda_k^{j-1}$$

$$= n \sum_{K_1} c_k r_k + \sum_{K_2} c_k r_k \frac{1 - \lambda_k^n}{1 - \lambda_k}$$

$$(\lambda_k = 1) \qquad (|\lambda_k| < 1)$$

As noted, K_1 are those indices k for which $\lambda_k = 1$, whereas K_2 are those indices for which $|\lambda_k| < 1$. We have now obtained the expected return in a finite number of moves n starting from Π^o.

Since

$$\lim_{n \to \infty} \lambda_k^n = 0 \text{ if } |\lambda_k| < 1 \qquad ,$$

$$\lim_{n \to \infty} V^n = n \sum_{K_1} c_k r_k + \sum_{K_2} c_k r_k \frac{1}{1 - \lambda_k}$$

The existence and nature of the limit for which we have been searching has now been shown. If the system is started according to Π^o, the total expected return V^n will have the form

$$V^n = ng + v$$

for large n. The gain of Π^o is g and its transient value is v. These quantities are defined as

$$g = \sum_{K_1} c_k r_k \qquad\qquad v = \sum_{K_2} c_k r_k \frac{1}{1 - \lambda_k}$$
$$(\lambda_k = 1) \qquad\qquad (|\lambda_k| < 1)$$

In terms of the gains of each of the N states, $\sum_{i=1}^{N} \pi_i^o g_i = g$, and in terms of the transient values, $\sum_{i=1}^{N} \pi_i^o v_i = v$. We shall usually use these relations when the system is started in a particular state i rather than according to some distribution of state probabilities. In this case, Π^o will have a 1 in the element corresponding to the ith state and zeros elsewhere. In this case $v_i = v$, $g_i = g$, and it is appropriate to write

$$V_i^n = ng_i + v_i$$

for large n.

As an illustration of this proof for a particular case, let us return once more to the final stage of the taxicab problem where:

$$P = \left[p_{ij}\right] = \begin{bmatrix} 1/16 & 3/4 & 3/16 \\ 1/16 & 7/8 & 1/16 \\ 1/8 & 3/4 & 1/8 \end{bmatrix} \qquad Q = \left[q_i\right] = \begin{bmatrix} 11/4 \\ 15 \\ 4 \end{bmatrix}$$

The characteristic values found by setting the determinant of $P - \lambda$ equal to zero are 1, 1/8 and -1/16, or

$$\lambda_1 = 1 \qquad \lambda_2 = 1/8 \qquad \lambda_3 = -1/16$$

Solving the equation $X_kP = \lambda_kP$ yields corresponding characteristic vectors

$$X_1 = 8 \quad 102 \quad 9 \qquad X_2 = 1 \quad -3 \quad 2 \qquad X_3 = 1 \quad 0 \quad -1$$

Also, since

$$r_k = X_kQ$$

$$r_1 = 1588 \qquad r_2 = -137/4 \qquad r_3 = -5/4$$

To find v_1 and g_1, set $\Pi^o = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$. Then

$$\Pi^o = c_1X_1 + c_2X_2 + c_3X_3$$

Solving

$$c_1 = 1/119 \qquad c_2 = 2/7 \qquad c_3 = 77/119$$

$$g_1 = \sum_{K_1} c_kr_k = c_1r_1 \qquad v_1 = \sum_{K_2} c_kr_k \frac{1}{1 - \lambda_k} = c_2r_2 \frac{1}{1 - \lambda_2} + c_3r_3 \frac{1}{1 - \lambda_3}$$

$$(\lambda_k = 1) \qquad (|\lambda_k| < 1)$$

Therefore

$$g_1 = 1588/119 = 13.34 \qquad v_1 = -11.95$$

To find v_2 and g_2, set $\Pi^o = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$

$$\Pi^o = c_1X_1 + c_2X_2 + c_3X_3$$

Solving,

$$c_1 = 1/119 \qquad c_2 = -1/21 \qquad c_3 = -1/51$$

$$g_2 = c_1 r_1 \qquad v_2 = c_2 r_2 \frac{1}{1 - \lambda_2} + c_3 r_3 \frac{1}{1 - \lambda_3}$$

$$g_2 = 13.34 \qquad v_2 = 1.89$$

To find v_3 and g_3, set $\Pi^o = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$. Then

$$c_1 = 1/119 \qquad c_2 = 34/119 \qquad c_3 = -42/119$$

$$g_3 = c_1 r_1 \qquad v_3 = c_2 r_2 \frac{1}{1 - \lambda_2} + c_3 r_3 \frac{1}{1 - \lambda_3}$$

$$g_3 = 13.34 \qquad v_3 = -10.76$$

In summary,

$$g = g_1 = g_2 = g_3 = 13.34$$

$$v_1 = -11.95$$

$$v_2 = 1.89$$

$$v_3 = -10.76$$

The gain agrees with our earlier results and so do the values if v_3 is subtracted from each of them to make $v_3 = 0$. However, the fact that we obtained transient values without making some arbitrary choice of a standard state should give us pause for reflection. The above development which proved the existence of the v_i makes no mention of any arbitrariness in their numerical values. As a matter of fact, the transient values are explicitly defined and are definitely <u>not</u> arbitrary. Yet we have been saying that one transient value in each chain may be chosen arbitrarily. How is this inconsistency explained? The answer is that the transient values are <u>relative</u> (one value in each chain may be picked arbitrarily) <u>as far as the value equations</u>

$$v_i + g_i = q_i + \sum_{j=1}^{N} p_{ij} v_j$$

are concerned, but the transient values are absolute as far as the limiting form

$$V_i^n = v_i + ng_i$$

is concerned. Since the policy improvement routine depends only on the differences in transient values, the use of relative transient values is permissible in the iteration cycle, and there is no need to find absolute transient values. If the absolute transient values v_i^a are desired for some reason and the relative transient values v_i^r are known, only a simple operation is required.

First, we shall show that $\sum_{i=1}^{N} \pi_i v_i^a = 0$, where $[\pi_i] = \Pi$, are the absolute state probabilities of the system. Suppose the system is started with a state probability distribution $\Pi^o = \Pi$; then

$$\Pi^o = \Pi = \lim_{n \to \infty} \Pi^n = \sum_k c_k x_k$$

$$(\lambda_k = 1)$$

as shown earlier. Therefore all c_k for which $|\lambda_k| < 1$ must be zero, the absolute transient value of the distribution Π must be zero and hence

$$\sum_{i=1}^{N} \pi_i v_i^a = 0$$

Another way in which this result may be seen is the following. Suppose a large number of runs of length n were made using each state as a starting state for a fraction of the runs equal to its absolute state probability. Then, in a sense, the system is always in the steady state and a return g is expected per move. Thus

$$\sum_{i=1}^{N} \pi_i V_i^n = ng$$

but

$$V_i^n = v_i^a + ng$$

and

$$\sum_{i=1}^{N} \pi_i V_i^n = \sum_{i=1}^{N} \pi_i v_i^a + \sum_{i=1}^{N} \pi_i ng$$

$$\sum_{i=1}^{N} \pi_i V_i^n = \sum_{i=1}^{N} \pi_i v_i^a + ng$$

and thus $\sum_{i=1}^{N} \pi_i v_i^a = 0$ as before. The above arguments are modified in an obvious way if there is more than one recurrent chain.

If the v_i^r are known, finding the v_i^a is quite simple. Suppose $\sum_{i=1}^{N} v_i^r = b$, a constant. Then $v_i^a = v_i^r - b$ for all i assures that $\sum_{i=1}^{N} \pi_i v_i^a = 0$. It is worth repeating that it is not necessary to know the absolute values in order to solve the sequential decision problem.

In the preceding example $\Pi = c_1 X_1 = \begin{bmatrix} 8/119 & 102/119 & 1/119 \end{bmatrix}$ are the steady state probabilities, and $v_1^a = -11.95$, $v_2^a = 1.89$, $v_3^a = -10.76$ are the values. $\sum_{i=1}^{3} \pi_i v_i^a = 0$ as expected.

Multiple Characteristic Values and Certain Other Cases

The foregoing proof concerning the existence of the v_i rests on the assumption that all characteristic values of the P matrix are distinct. We shall now show the extension of the proof to the situation in which this is not the case. Suppose that two characteristic values of the P matrix are identical. In this case the characteristic vectors obtained in a straightforward way will not in general be sufficient to express an arbitrary Π^o in the form $\sum_k c_k X_k$. There now exists the problem of finding the missing characteristic vector. It is shown by Friedman[4] that such a characteristic vector may be found by solving $X(P - \lambda I)^2 = 0$ using the repeated λ. If X_m is the characteristic vector obtained in a straightforward manner from $X_m(P - \lambda_m I) = 0$ and $\lambda_m = \lambda_n$, then $X_n(P - \lambda_n I)^2 = 0$ yields X_n, and $X_n(P - \lambda_n I) \neq 0$. It is convenient to specify $X_n(P - \lambda_n I) = X_m$ so that $X_n P = \lambda_n X_n + X_m$. X_n is an unusual kind of characteristic vector called a "generalized characteristic vector of rank 2." An example will indicate how our proof is modified in cases where such generalized characteristic values arise. Consider the following system.

$$P = [p_{ij}] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 1/2 \\ 1/2 & 0 & 1/2 \end{bmatrix} \qquad Q = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

By setting the determinant of $P - \lambda I$ equal to zero, the characteristic values are found to be $\lambda_1 = 1$, $\lambda_2 = \lambda_3 = 1/2$. The characteristic vectors obtained from $X_k P = \lambda_k X_k$ are

$$X_1 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \qquad X_2 = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

Also, from $r_k = X_k Q$

$$r_1 = 1 \qquad r_2 = -2$$

Let us first find v_1 and g_1, so that $\Pi^o = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$. In this case

$$\Pi^o = X_1 \qquad \text{so that} \qquad c_1 = 1,\ c_2 = c_3 = 0$$

$$g = \sum_k c_k r_k \qquad v = \sum_k c_k r_k \text{ (sum of } \lambda_k \text{ series)}$$

$$(\lambda_k = 1) \qquad (|\lambda_k| < 1)$$

$$g_1 = c_1 r_1 = 1 \qquad v_1 = 0$$

Next we try to find v_2 and g_2 with $\Pi^o = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$, and now we are in trouble since Π^o cannot be expressed as a linear combination of X_1 and X_2. It is time to find X_3 from $X_3(P - 1/2\ I)^2 = 0$.

$$(P - 1/2\ I)^2 = \begin{bmatrix} 1/4 & 0 & 0 \\ 1/4 & 0 & 0 \\ 1/4 & 0 & 0 \end{bmatrix}$$

and X_3 may be taken as $\begin{bmatrix} 0 & -2 & 2 \end{bmatrix}$; consequently $r_3 = 2$.

Now $\Pi^o = X_1 - X_2 - 1/2\ X_3$ so that $c_1 = 1$, $c_2 = -1$, $c_3 = -1/2$. At this point a return to the original proof is necessary in order to introduce the necessary modifications.

$$\Pi^{n+1} = \Pi^o P^n = \sum_k c_k X_k P^n$$

Now $$X_1 P^n = \lambda_1^n X_1$$

and $$X_2 P^n = \lambda_2^n X_2$$

but since $$X_3 P = \lambda_3 X_3 + X_2$$

$$X_3 P^n = \lambda_3^n X_3 + n \lambda_3^{n-1} X_2$$

$$\Pi^{n+1} = c_1 X_1 \lambda_1^n + c_2 X_2 \lambda_2^n + c_3 X_3 \lambda_3^n + c_3 X_2 n \lambda_3^{n-1}$$

$$\Pi^{n+1} = c_1 \lambda_1^n X_1 + c_2 \lambda_2^n X_2 + c_3 n \lambda_3^{n-1} X_2 + c_3 \lambda_3^n X_3$$

similarly

$$r^{n+1} = c_1 \lambda_1^n r_1 + c_2 \lambda_2^n r_2 + c_3 n \lambda_3^{n-1} r_2 + c_3 \lambda_3^n r_3$$

$$V^n = c_1 r_1 \sum_{j=1}^{n} \lambda_1^{j-1} + c_2 r_2 \sum_{j=1}^{n} \lambda_2^{j-1} + c_3 r_2 \sum_{j=1}^{n} (j-1) \lambda_3^{j-2} + c_3 r_3 \sum_{j=1}^{n} \lambda_3^{j-1}$$

As $n \rightarrow \infty$

$$V^n = nc_1r_1 + c_2r_2\sum_{j=1}^{\infty}\lambda_2^{j-1} + c_3r_2\sum_{j=1}^{\infty}(j-1)\lambda_3^{j-2} + c_3r_3\sum_{j=1}^{\infty}\lambda_3^{j-1}$$

$$= nc_1r_1 + r_2\left[\frac{c_2}{1-\lambda_2} \quad \frac{c_3}{(1-\lambda_3)^2}\right] + c_3r_3\frac{1}{1-\lambda_3}$$

so that $g = c_1r_1$

$$v = \frac{c_2r_2 + c_3r_3}{1-\lambda_o} + \frac{c_3r_2}{(1-\lambda_o)^2} \qquad \text{where } \lambda_o = \lambda_2 = \lambda_3$$

The modification in the proof which occurs in the case of multiple roots is now clear. Such roots cause terms of the form $n\lambda^n$, $n^2\lambda^n$, etc., in the infinite sums for characteristic values with magnitude less than 1. Such sums always converge so that the limit exists as before. The only difference is that the limit is more difficult to compute.

Concluding the example, for the case $\Pi^{\circ} = [0 \quad 1 \quad 0]$ where we found $c_1 = 1$, $c_2 = -1$, $c_3 = -1/2$; $r_1 = 1$, $r_2 = -2$, $r_3 = 2$; and $\lambda_o = 1/2$,

$$g_2 = 1$$

$$v_2 = 6$$

Finally, considering $\Pi^{\circ} = [0 \quad 0 \quad 1]$, then $c_1 = 1$, $c_2 = -1$, $c_3 = 0$, and

$$g_3 = 1$$

$$v_3 = 4$$

In summary, $g = g_1 = g_2 = g_3 = 1$; $v_1 = 0$, $v_2 = 6$, $v_3 = 4$. These are exactly the same results obtained if the value determination operation for the process is performed with $v_1 = 0$.

It might appear that multiple roots at $\lambda = 1$ might cause trouble. A simple example will remove any mystery surrounding this case.

Consider

$$P = \begin{bmatrix} 1/2 & 1/4 & 1/4 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad Q = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

The characteristic values of P are $\lambda_1 = \lambda_2 = 1$, $\lambda_3 = 1/2$. There are two characteristic values corresponding to $\lambda = 1$ found in the straightforward way to be $X_1 = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$ and $X_2 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$; similarly $X_3 = \begin{bmatrix} 2 & -1 & -1 \end{bmatrix}$. Consequently $r_1 = 2$, $r_2 = 3$, $r_3 = -3$.

$$g = c_1 r_1 + c_2 r_2$$

$$v = \frac{c_3 r_3}{1 - \lambda_3}$$

If $\Pi^o = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$ $\quad c_1 = 1/2 \quad c_2 = 1/2 \quad c_3 = 1/2$

and $\quad g_1 = 2.5$

$v_1 = -3$

If $\Pi^o = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$ $\quad c_1 = 1 \quad c_2 = 0 \quad c_3 = 0$

and $\quad g_2 = 2$

$v_2 = 0$

If $\Pi^o = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$ $\quad c_1 = 0 \quad c_2 = 1 \quad c_3 = 0$

and $\quad g_3 = 3$

$v_3 = 0$

These results are identical with those obtained from the value determination operation. Multiple roots at $\lambda = 1$ do not cause trouble because they generate a complete set of characteristic vectors. Incidentally, the number of independent chains is equal to the number of characteristic values which equal 1.

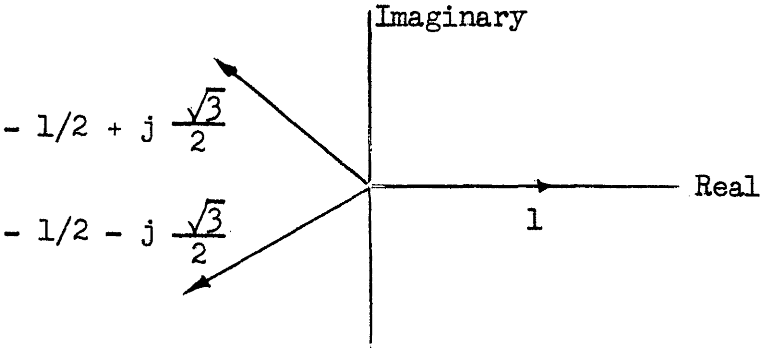
Another case which may cause concern is that in which the characteristic values are complex. Here again, an example will show the interpretation of this case.

Consider

$$P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \qquad Q = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

The characteristic values of P are $\lambda_1 = 1$, $\lambda_2 = -1/2 + j\frac{\sqrt{3}}{2}$, $\lambda_3 = -1/2 - j\frac{\sqrt{3}}{2}$

Plot of Characteristic Values



The transition matrix is periodic with period 3. Transitions of the system may be viewed as rotations of the characteristic value complex vectors. The characteristic vectors are

$$X_1 = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \qquad X_2 = \begin{bmatrix} 1 & \left(-1/2 - j\frac{\sqrt{3}}{2}\right) & \left(-1/2 + j\frac{\sqrt{3}}{2}\right) \end{bmatrix}$$

$$X_3 = \begin{bmatrix} 1 & \left(-1/2 + j\frac{\sqrt{3}}{2}\right) & \left(-1/2 - j\frac{\sqrt{3}}{2}\right) \end{bmatrix}$$

Also

$$r_1 = 6 \qquad r_2 = -3/2 + j\frac{\sqrt{3}}{2} \qquad r_3 = -3/2 - j\frac{\sqrt{3}}{2}$$

In general $$V^{n} = \sum_{k} c_k r_k \sum_{j=1}^{n} \lambda_k^{j-1}$$

$$V^{n} = nc_1r_1 + c_2r_2 \sum_{j=1}^{n} \lambda_2^{j-1} + c_3r_3 \sum_{j=1}^{n} \lambda_3^{j-1}$$

For any integer t,

$$\begin{array}{llll} \text{if} \quad n = 3t & \sum_{i=1}^{n} \lambda_2^{j-1} = 0 & & \sum_{i=1}^{n} \lambda_3^{j-1} = 0 \\ n = 3t + 1 & \sum_{i=1}^{n} \lambda_2^{j-1} = 1 & & \sum_{i=1}^{n} \lambda_3^{j-1} = 1 \\ n = 3t + 2 & \sum_{i=1}^{n} \lambda_2^{j-1} = 1/2 & + j\frac{\sqrt{3}}{2} & \sum_{i=1}^{n} \lambda_3^{j-1} = 1/2 - j\frac{\sqrt{3}}{2} \end{array}$$

If

$$\begin{array}{ll} n = 3t & V^{n} = nc_1r_1 \\ n = 3t + 1 & V^{n} = nc_1r_1 + c_2r_2 + c_3r_3 \\ n = 3t + 2 & V^{n} = nc_1r_1 + c_2r_2\ 1/2 + j\frac{\sqrt{3}}{2} + c_3r_3\ 1/2 - j\frac{\sqrt{3}}{2} \end{array}$$

As $n \longrightarrow \infty$, $V^{n} = ng + v(n)$, and the limit v depends periodically on n although g is always equal to c_1r_1. However, let us proceed retaining distinct $v(n)$.

If $\Pi^{o} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$ $\quad c_1 = c_2 = c_3 = 1/3 \quad g = c_1r_1 = 2$

$$\begin{array}{lll} \text{If} & n = 3t & v(n) = 0 \\ & n = 3t + 1 & v(n) = -1 \\ & n = 3t + 2 & v(n) = -1 \end{array}$$

It now seems reasonable to define v_1 as the time average of $v(n)$. If this is done, $v_1 = -2/3$.

If $\Pi^{o} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$ $\quad c_1 = 1/3 \quad c_2 = -1/6(1 - \sqrt{3}j) \quad c_3 = -1/6(1 + \sqrt{3}j)$

$$g = c_1r_1 = 2$$

$$\begin{array}{lll} \text{If} & n = 3t & v(n) = 0 \\ & n = 3t + 1 & v(n) = 0 \\ & n = 3t + 2 & v(n) = 1 \end{array}$$

Again defining v_2 as the time average of $v(n)$, $v_2 = 1/3$.

$$\text{If } \Pi^o = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \qquad c_1 = 1/3 \qquad c_2 = -1/6\,(1 + \sqrt{3}\,j) \qquad c_3 = -1/6\,(1 - \sqrt{3}\,j)$$

$$g = c_1 r_1 = 2$$

$$\begin{array}{lll} \text{If} & n = 3t & v(n) = 0 \\ & n = 3t + 1 & v(n) = 1 \\ & n = 3t + 2 & v(n) = 0 \end{array}$$

so that $v_3 = 1/3$ by time averaging.

$$\begin{array}{lll} \text{In summary,} & g = 2 & v_1 = -2/3 \\ & & v_2 = 1/3 \\ & & v_3 = 1/3 \end{array}$$

All the v_i are to be interpreted as time averages. If we agree on this interpretation, the value determination operation may be used without apology to find the gain and values. Of course, the values obtained in that way check with those above. Periodic chains introduce no essential difficulty in the iteration cycle; all that is required is circumspection in interpretation.

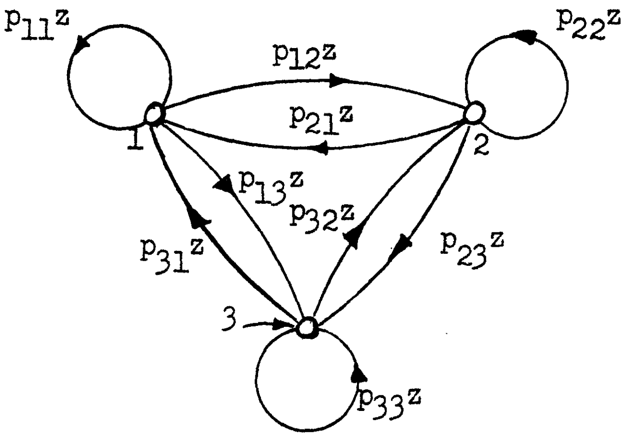
A Flow Graph Interpretation of System Operation

Markov processes have been analyzed using systems concepts and flow graphs by R. W. Sittler.[5] This approach is essentially one which uses generating functions for the time sequences of probabilities. The main advantage of the method is that it permits a high degree of visualization. In the following we shall extend Sittler's work to include the analysis of Markov processes with associated returns.

According to Sittler, the Markov process defined by

$$\begin{bmatrix} p_{ij} \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix}$$

may be represented in the frequency or transform domain by the flow graph:



The z appended to every branch represents the unit delay that occurs at each transition. The general transform relation is $F(z) = \sum_{n=o}^{\infty} f(n)z^n$, where f(n) is the probability of an event at a time equal to n.

The probability that the system will occupy each of the states after n moves, Π^n, is equal to Π^o multiplied by the nth power of the transition matrix P.

$$\Pi^n = \Pi^o P^n$$

Let $\Pi(z) = \sum_{n=o}^{\infty} \Pi^n z^n$; $\quad P(z) = \sum_{n=o}^{\infty} P^n z^n$.

$\Pi(z)$ is a column matrix of size N, and P(z) is an N by N square matrix. If both sides of the equation for Π^n are transformed,

$$\Pi(z) = \Pi^o P(z)$$

P(z) may be evaluated directly by multiplying its defining equation by the matrix (I - zP).

$$P(z) = \sum_{n=o}^{\infty} P^n z^n$$

$$(I - zP)P(z) = (I - zP) \sum_{n=o}^{\infty} P^n z^n$$

$$= \sum_{n=o}^{\infty} P^n z^n - \sum_{n=o}^{\infty} P^{n+1} z^{n+1}$$

$$= I$$

or

$$P(z) = (I - zP)^{-1}$$

The solution of the Markov process is then given in transform form by

$$\Pi(z) = \Pi^o (I - zP)^{-1}$$

If $(I - zP)^{-1}$ is given the symbol $H(z)$, then

$$\Pi(z) = \Pi^o H(z)$$

$H(z)$ may be interpreted as the matrix of node-to-node transfer functions of the process. In simple cases it is feasible to find the node transfer functions by flow graph reduction, and to compose the $H(z)$ matrix in this way. Π^o may be interpreted as the excitation applied to each node of the system. If Π^o has a one as its i^{th} element and zeros elsewhere, the i^{th} row of $H(z)$ may be interpreted as the transforms of the output at each node of the graph for a unit sample input at node i as excitation.

There are many interesting and novel ways[5] in which $H(z)$ may be found from the flow graph. These need not concern us here because we are interested only in the nature of $H(z)$. As Sittler shows, $H(z)$ must have at least one pole on the unit circle in the z-plane and no poles within the unit circle. These statements are equivalent to those made regarding the characteristic values of stochastic matrices. The poles on the unit circle correspond to the absolute state probabilities, whereas those outside the unit circle correspond to the transients in state probabilities which ultimately die away. Thus $H(z)$ may be divided into two parts by partial fraction expansion.

$$H(z) = S(z) + T(z)$$

where $S(z)$ is that part of $H(z)$ with poles on the unit circle (steady-state component) and $T(z)$ is that part with poles outside the unit circle (transient component). The i^{th} row of $S(0)$ represents the absolute state probabilities of the system if it is started in state i.

From our previous development

$$r^n = \Pi^n Q$$

$$V^n = \sum_{j=1}^{n} r^j = \sum_{j=1}^{n} \Pi^j Q$$

or

$$V^n = \lim_{n \to \infty} \sum_{j=1}^{n} \Pi^j Q$$

where $\lim_{n \to \infty}$ is taken to mean "for large n."

$$\lim_{n \to \infty} \sum_{j=1}^{n} p^j = \lim_{z \to 1} \Pi(z) = \Pi^o \lim_{z \to 1} H(z) = \Pi^o \lim_{z \to 1} S(z) + \lim_{z \to 1} T(z)$$

$\lim_{z \to 1} S(z)$ does not exist for infinite n because the absolute probabilities do not die away. However for any finite n, $\lim_{z \to 1} S(z)$ can be replaced by $nS(0)$. Further, $\lim_{z \to 1} T(z)$ always exists.

$$\lim_{n \to \infty} \sum_{j=1}^{n} \Pi^j = \Pi^o nS(0) + \Pi^o T(1)$$

For large n,

$$V^n = n\Pi^o S(0)Q + \Pi^o T(1)Q$$

The gain of the initial distribution Π^o is thus $\Pi^o S(0)Q$, whereas its value is $\Pi^o T(1)Q$. We may define column vectors $G = S(0)Q$ and $V = T(1)Q$, so that

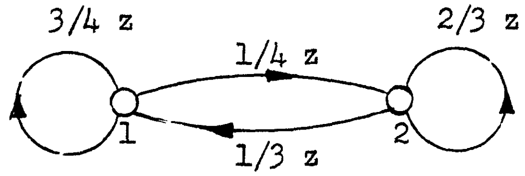
$$V^n = n\Pi^o G + \Pi^o V$$

We have then that the gain and absolute transient value of the i^{th} state are given by the i^{th} elements of G and V, respectively. This concludes an alternate proof concerning the existence and definition of the absolute values v_i^a. The above method of calculation of g and v_i is no more practical than the characteristic vector method (to which it is entirely equivalent) because of the amount of computation involved. In a sense it provides too much information because it prescribes just how the transient values approach their limit rather than directly finding the limit itself. It is for this reason that the simultaneous equation method which yields relative values is the most convenient for computation of the sequential decision problem.

It is interesting to apply the transform method to some typical problems to see just how the computational procedure is carried out. Consider the following simple problem, which is the earlier coin-tossing problem using the optimal strategy: toss coin 1 if last toss produced heads; toss coin 2 if it produced tails.

$$P = \left[p_{ij}\right] = \begin{bmatrix} 3/4 & 1/4 \\ 1/3 & 2/3 \end{bmatrix} \qquad \left[q_i\right] = \begin{bmatrix} 7/4 \\ 0 \end{bmatrix}$$

The associated flow graph is



$$(I - Pz) = \begin{bmatrix} (1 - 3/4z) & - 1/4z \\ - 1/3z & (1 - 2/3z) \end{bmatrix}$$

$$(I - Pz)^{-1} = \begin{bmatrix} \dfrac{1 - 2/3z}{(1-z)(1-5/12z)} & \dfrac{1/4z}{(1-z)(1-5/12z)} \\[2ex] \dfrac{1/3z}{(1-z)(1-5/12z)} & \dfrac{1 - 3/4z}{(1-z)(1-5/12z)} \end{bmatrix}$$

$$= \begin{bmatrix} \dfrac{4/7}{1-z} + \dfrac{3/7}{1-5/12z} & \dfrac{3/7}{1-z} + \dfrac{-3/7}{1-5/12z} \\[2ex] \dfrac{4/7}{1-z} + \dfrac{-4/7}{1-5/12z} & \dfrac{3/7}{1-z} + \dfrac{4/7}{1-5/12z} \end{bmatrix}$$

using a partial fraction expansion.

$$H(z) = (I - Pz)^{-1} = \begin{bmatrix} \dfrac{4/7}{1-z} & \dfrac{3/7}{1-z} \\[2ex] \dfrac{4/7}{1-z} & \dfrac{3/7}{1-z} \end{bmatrix} + \begin{bmatrix} \dfrac{3/7}{1-5/12z} & \dfrac{-3/7}{1-5/12z} \\[2ex] \dfrac{-4/7}{1-5/12z} & \dfrac{4/7}{1-5/12z} \end{bmatrix}$$

$$= S(z) \qquad + \qquad T(z)$$

where $S(z)$ has all poles on the unit circle in the z-plane and $T(z)$ has all poles outside the unit circle in the z-plane.

$$S(0) = \begin{bmatrix} 4/7 & 3/7 \\ 4/7 & 3/7 \end{bmatrix}$$

is the matrix of absolute probabilities; all rows are equal because there is only one recurrent chain.

$$T(1) = \begin{bmatrix} 36/49 & -36/49 \\ -48/49 & 48/49 \end{bmatrix}$$

is the sum matrix for the transient components of state probability.

$$G = S(0)Q = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \qquad\qquad V = T(1)Q = \begin{bmatrix} 9/7 \\ -12/7 \end{bmatrix}$$

Thus

$$g_1 = 1 \qquad v_1^a = 9/7 \qquad \pi_1 = 4/7$$

$$g_2 = 1 \qquad v_2^a = -12/7 \qquad \pi_2 = 3/7$$

Note that $\sum_{i=1}^{2} \pi_i v_i^a = 0$, as required. This result has a simple proof in flow graph form. Suppose the system is excited with the steady-state probabilities Π. Then the initial conditions will be exactly matched and there will be no transient in the state probabilities. Hence, $T(z) = 0$, $T(1) = 0$, and the transient value of the initial distribution $\Pi^o = \Pi$ is zero. Therefore

$$\sum_{i=1}^{N} \pi_i v_i^a = 0$$

The relationship of the results obtained above to our earlier results on the coin-tossing problem is worth investigating. First, for large n we know that

$$V_i^n = v_i^a + ng_i$$

From above,

$$V_1^n = 9/7 + n = 1.2857 + n$$

$$V_2^n = -12/7 + n = -1.7143 + n$$

These are just the equations for the asymptotes to f_1^n and f_2^n found earlier. Furthermore, it is possible to find the <u>entire</u> functions f_1^n and f_2^n using the flow graph-transform approach.

The matrix $H(z)$ represents the transform of the probabilities that the system will occupy state j at time n if it is started in state i. For example, $H_{11}(z) = \frac{4/7}{1-z} + \frac{3/7}{1-5/12z}$ is the transform of π_{11}^n, the probability that the system will be in state 1 after the nth transaction given that it was started in state 1. In general the return on the nth move if the system is started in i is $r_i^n = \sum_{j=1}^{N} \pi_{ij}^n q_j$. Consequently,

$$V_i^n = \sum_{m=1}^{n} r_i^m = \sum_{m=1}^{n} \sum_{j=1}^{N} \pi_{ij}^m q_j = \sum_{j=1}^{N} \sum_{m=1}^{n} \pi_{ij}^m q_j,$$

and this relation holds for any n.

Since in this problem $q_1 = 7/4$ and $q_2 = 0$,

$$V_1^n = 7/4 \sum_{m=1}^{n} \pi_{11}^{m}$$

$$V_2^n = 7/4 \sum_{m=1}^{n} \pi_{21}^{m}$$

Since $H_{11}(z) = \frac{4/7}{1-z} + \frac{3/7}{1-5/12z}$ and $H_{21}(z) = \frac{4/7}{1-z} + \frac{-4/7}{1-5/12z}$, by inverse transformation,

$$\pi_{11}^{n} = 4/7 + 3/7(5/12)^n \qquad \pi_{21}^{n} = 4/7 - 4/7(5/12)^n$$

$$\sum_{m=1}^{n} \pi_{11}^{m} = 4/7n + 3/7 \left[\frac{1-(5/12)^n}{1-5/12}\right] \qquad \sum_{m=1}^{n} \pi_{21}^{n} = 4/7n - 4/7 \left[\frac{1-(5/12)^n}{1-5/12}\right]$$

Therefore

$$V_1^n = n + 3/4 \left[\frac{1-(5/12)^n}{1-5/12}\right] \qquad V_2^n = n - \left[\frac{1-(5/12)^n}{1-5/12}\right]$$

$$V_1^n = n + 9/7 \left[1 - (5/12)^n\right] \qquad V_2^n = n - 12/7 \left[1 - (5/12)^n\right]$$

These expressions for V_1^n and V_2^n hold true <u>for any n</u>, even for $n = 0$. Note that they approach the asymptotic equations derived above as $n \rightarrow \infty$. Since in this problem the same policy is used throughout the computation when the Bellman technique is applied, in this case $V_1^n = f_1^n$ and $V_2^n = f_2^n$. In other words, the total expected return graphs determined earlier for the coin-tossing problem using the Bellman iteration approach could be calculated analytically from the equations

$$f_1^n = n + 9/7 \left[1 - (5/12)^n\right]$$

$$f_2^n = n - 12/7 \left[1 - (5/12)^n\right]$$

as the reader may verify.

As a second example, let us solve the problem which previously caused trouble because it had multiple characteristic values.

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 1/2 \\ 1/2 & 0 & 1/2 \end{bmatrix} \qquad Q = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

$$(I - Pz) = \begin{bmatrix} 1-z & 0 & 0 \\ 0 & 1-1/2z & -1/2z \\ -1/2z & 0 & 1-1/2z \end{bmatrix}$$

$$(I - Pz)^{-1} = \begin{bmatrix} \frac{1}{1-z} & 0 & 0 \\ \frac{1/4z^2}{(1-z)(1-1/2z)^2} & \frac{1}{1-1/2z} & \frac{1/2z}{(1-1/2z)^2} \\ \frac{1/2z}{(1-z)(1-1/2z)} & 0 & \frac{1}{1-1/2z} \end{bmatrix}$$

$$H(z) = (I - Pz)^{-1} = S(z) + T(z) =$$

$$\begin{bmatrix} 1/1-z & 0 & 0 \\ 1/1-z & 0 & 0 \\ 1/1-z & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ \frac{-1}{(1-1/2z)^2} & \frac{1}{1-1/2z} & \frac{1/2z}{(1-1/2z)^2} \\ \frac{-1}{1-1/2z} & 0 & \frac{1}{1-1/2z} \end{bmatrix}$$

$$S(0) = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \qquad T(1) = \begin{bmatrix} 0 & 0 & 0 \\ -4 & 2 & 2 \\ -2 & 0 & 2 \end{bmatrix}$$

$$G = S(0)Q = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \qquad V = T(1)Q = \begin{bmatrix} 0 \\ 6 \\ 4 \end{bmatrix}$$

so that

$$\begin{array}{lll} g_1 = 1 & v_1 = 0 & \pi_1 = 1 \\ g_2 = 1 & v_2 = 6 & \pi_2 = 0 \\ g_3 = 1 & v_3 = 4 & \pi_3 = 0 \end{array}$$

These are the same results obtained from characteristic vector considerations; however, at no place in the above example was it necessary to make any special provisions for the multiple characteristic values. The transform approach circumvents this difficulty.

As a final example let us consider the case in which there are two independent chains and a split transient state.

$$P = \begin{bmatrix} 1/2 & 1/4 & 1/4 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad Q = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

$$(I - Pz) = \begin{bmatrix} 1-1/2z & -1/4z & -1/4z \\ 0 & 1-z & 0 \\ 0 & 0 & 1-z \end{bmatrix}$$

$$(I - Pz)^{-1} = \begin{bmatrix} \frac{1}{1-1/2z} & \frac{1/4z}{(1-1/2z)(1-z)} & \frac{1/4z}{(1-1/2z)(1-z)} \\ 0 & \frac{1}{1-z} & 0 \\ 0 & 0 & \frac{1}{1-z} \end{bmatrix}$$

$$H(z) = (I - Pz)^{-1} = S(z) + T(z) =$$

$$\begin{bmatrix} 0 & \frac{1/2}{1-z} & \frac{1/2}{1-z} \\ 0 & \frac{1}{1-z} & 0 \\ 0 & 0 & \frac{1}{1-z} \end{bmatrix} + \begin{bmatrix} \frac{1}{1-1/2z} & \frac{-1/2}{1-1/2z} & \frac{-1/2}{1-1/2z} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$S(0) = \begin{bmatrix} 0 & 1/2 & 1/2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad T(1) = \begin{bmatrix} 2 & -1 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$G = S(0)Q = \begin{bmatrix} 2.5 \\ 2 \\ 3 \end{bmatrix} \qquad V = T(1)Q = \begin{bmatrix} -3 \\ 0 \\ 0 \end{bmatrix}$$

$$g_1 = 2.5 \qquad v_1 = -3$$

$$g_2 = 2 \qquad v_2 = 0$$

$$g_3 = 3 \qquad v_3 = 0$$

Thus the case of two independent chains does not produce any difficulty; the results agree completely with those obtained before. The case of periodic chains can also be handled by the transform method, but as expected some concept of time averaging must be introduced.

The transform method is much more satisfying than the characteristic vector method because it retains physical insight for the process and avoids certain technical difficulties associated with the characteristic vector approach. However, either method suffices to prove the existence of the transient values and to derive their properties.

More on the Policy Improvement Routine

In earlier sections a plausibility argument and proof for the policy improvement routine were given. At this point a supplementary argument will be presented. Consider two policies A and B which are to be compared. Suppose that in making n moves it is decided to use policy A for the last n-1 moves, and to make the first move according to the policy which will maximize return for the entire n moves. Then if policy B is the policy to use for the first move in order to maximize return for the first n moves, by induction it would be a better policy to use for all numbers of moves larger than n.

Thus since

$$V_i^{A^n} = q_i^A + \sum_{j=1}^{N} p_{ij}^A V_j^{A^{n-1}}$$

$$V_i^{B^n} = q_i^B + \sum_{j=1}^{N} p_{ij}^B V_j^{A^{n-1}}$$

Policy B is better than policy A if $V_i^{B^n} > V_i^{A^n}$.

or if $$q_i^B + \sum_{j=1}^{N} p_{ij}^B V_j^{A^{n-1}} > q_i^A + \sum_{j=1}^{N} p_{ij}^A V_j^{A^{n-1}}$$

Since $V_j^A = v_j^A + ng_j$ for large n, B is better than A if

$$q_i^B + \sum_{j=1}^{N} p_{ij}^B \left[v_j^A + (n-1)g_j^A \right] > q_i^A + \sum_{j=1}^{N} p_{ij}^A \left[v_j^A + (n-1)g_j^A \right]$$

$$q_i^B + \sum_{j=1}^{N} p_{ij}^B v_j^A + (n-1) \sum_{j=1}^{N} p_{ij}^B g_j^A > q_i^A + \sum_{j=1}^{N} p_{ij}^A v_j^A + (n-1) \sum_{j=1}^{N} p_{ij}^A g_j^A$$

Since n is large, B will be better than A if

$$\sum_{j=1}^{N} p_{ij}^B g_j^A > \sum_{j=1}^{N} p_{ij}^A g_j^A = g_i^A$$

This is the mathematical form of the statement that the policy improvement routine should make its decision based on gain if a difference in gain exists. Thus B should be chosen over A in state i if

$$\sum_{j=1}^{N} p_{ij}^B g_j^A > g_i^A$$

In the case where this relation is an equality (for example where there is only one recurrent chain and all $g_i = g$), the policy improvement routine must make its decision based on transient values. Thus it will decide that B is better than A if

$$q_i^B + \sum_{j=1}^{N} p_{ij}^B v_j^A > q_i^A + \sum_{j=1}^{N} p_{ij}^A v_j^A$$

This is exactly the criterion used in the policy improvement routine in the earlier development where it was assumed that there was only one recurrent chain in the system. Another formal proof of the fact that this criterion improves the policy is given there.

Continuing, if the above policy improvement routine is used and policy B is better than policy A,

$$V_i^{B^n} > V_i^{A^n}$$

$$\text{or} \qquad V_i^B > ng_i^A + v_i^A$$

According to this result, if we make a policy improvement in state i, g_i must stay the same or increase. If g_i remains the same, then v_i must increase. The precise conditions under which each of these alternatives will occur have been given previously.

It is a matter of real concern whether the policy improvement routine can reach a relative maximum and pick as a final policy one which is less than optimal. It is quite easy to show that the policy improvement routine will always discover a policy which has a higher gain than the current one, if such a better policy exists.

Suppose that policy B is better than policy A in the sense that $g^B > g^A$ (assume one recurrent chain for simplicity). If an asterisk is used to denote the difference in corresponding quantities for the two policies,

$g^* = g^B - g^A$ and g^* is by assumption greater than zero.

Also

$$v_i^A + g^A = q_i^A + \sum_{j=1}^{N} p_{ij}^A v_j^A$$

$$v_i^B + g^B = q_i^B + \sum_{j=1}^{N} p_{ij}^B v_j^B$$

Let us suppose also that the policy improvement routine has not found that policy B is better than A, so that

$$q_i^A + \sum_{j=1}^{N} p_{ij}^A v_j^A \geqslant q_i^B + \sum_{j=1}^{N} p_{ij}^B v_j^A$$

Let

$$\gamma_i = q_i^A + \sum_{j=1}^{N} p_{ij}^A v_j^A - q_i^B - \sum_{j=1}^{N} p_{ij}^B v_j^A$$

where all $\gamma_i \geqslant 0$.

Subtracting the two value equations

$$g^B - g^A + v_i^B - v_i^A = q_i^B - q_i^A + \sum_{j=1}^{N} p_{ij}^B v_j^B - \sum_{j=1}^{N} p_{ij}^A v_j^A$$

Substituting for $q_i^B - q_i^A$

$$g^B - g^A + v_i^B - v_i^A = -\gamma_i + \sum_{j=1}^{N} p_{ij}^A v_j^A - \sum_{j=1}^{N} p_{ij}^B v_j^A + \sum_{j=1}^{N} p_{ij}^B v_j^B - \sum_{j=1}^{N} p_{ij}^A v_j^A$$

$$g^* + v_i^* = -\gamma_i + \sum_{j=1}^{N} p_{ij}^B v_j^*$$

The solution of these equations has been shown to be

$$g^* = -\sum_{i=1}^{N} \pi_i^B \gamma_i$$

where the π_i^B are the absolute state probabilities for policy B. Since

$\pi_i \geq 0$ and $\gamma_i \geq 0$, the maximum value of $g^* = 0$. However, we have assumed that $g^* > 0$. Hence we have a proof by contradiction that the policy improvement routine must achieve the largest possible g permitted by the problem; it has already been shown that this maximum is achieved monotonically in a finite number of iterations.

The Replacement Problem

The examples of the policy iteration method presented up to this point have been somewhat far removed from the realm of practical problems. It would be extremely interesting to see the method applied to a problem which is of major importance to industry. As an example of such a practical application, the replacement problem was chosen. This is the problem of when to replace a piece of capital equipment which deteriorates with time. The question to be answered is this: If we now own a machine of a certain age, should we keep it or should we trade it in; further, if we trade it in, how old a machine should we buy.

In order to fix ideas, let us consider the problem of automobile replacement over a time interval of ten years. We agree to review our current situation every three months and to make a decision on keeping our present car or trading it in at that time. The state of the system, i, is described by the age of the car in three-month periods; i may run from 1 to 40. In order to keep the number of states finite, a car of age 40 remains a car of age 40 forever (it is considered to be essentially worn out). The alternatives available in each state are these: The first alternative, $k = 1$, is to keep the present car for another quarter. The other alternatives, $k > 1$, are to buy a car of age $k-2$, where $k-2$ may be as large as 39. We have then forty states with forty-one alternatives in each state so that there are 41^{40} possible policies.

The data supplied are the following:

C_i, the cost of buying a car of age i

T_i, the trade-in value of a car of age i

E_i, the expected cost of operating a car of age i until it reaches age $i+1$

p_i, the probability that a car of age i will survive to be age $i+1$ without incurring a prohibitively expensive repair

The probability defined above is necessary to limit the number of states. A car of any age that has a hopeless breakdown is immediately sent to state 40. Naturally, $p_{40} = 0$.

The basic equations governing the system when it is in state i are:

If $k = 1$ (keep present car)

$$g + v_i = - E_i + p_i v_{i+1} + (1 - p_i)v_{40}$$

If $k > 1$ (trade for car of age $k-2$)

$$g + v_i = T_i - C_{k-2} - E_{k-2} + p_{k-2}v_{k-1} + (1 - p_{k-2})v_{40}$$

It is simple to phrase these equations in terms of our earlier notation. For instance,

$$q_i^k = -E_i \text{ for } k = 1 \qquad q_i^k = T_i - C_{k-2} - E_{k-2} \text{ for } k > 1$$

$$p_{ij}^k = \left\{\begin{matrix} p_i & j = i+1 \\ 1-p_i & j = 40 \\ 0 & \text{other } j \end{matrix}\right\} \text{for } k = 1 \qquad p_{ij}^k = \left\{\begin{matrix} p_{k-2} & j = k-1 \\ 1-p_{k-2} & j = 40 \\ 0 & \text{other } j \end{matrix}\right\} \text{for } k > 1$$

The actual data used in the problem are listed in Table III and graphed in Figure 3. The discontinuities in the cost and trade-in functions were introduced in order to characterize typical model year effects.

TABLE III

REPLACEMENT PROBLEM DATA

Age in Periods i	Cost C_i	Trade-in Value T_i	Operating Expense E_i	Survival Probability p_i
0	\$2,000	\$1,600	\$ 50	1.000
1	1,840	1,460	53	0.999
2	1,680	1,340	56	0.998
3	1,560	1,230	59	0.997
4	1,300	1,050	62	0.996
5	1,220	980	65	0.994
6	1,150	910	68	0.991
7	1,080	840	71	0.988
8	900	710	75	0.985
9	840	650	78	0.983
10	780	600	81	0.980
11	730	550	84	0.975
12	600	480	87	0.970
13	560	430	90	0.965
14	520	390	93	0.960
15	480	360	96	0.955
16	440	330	100	0.950
17	420	310	103	0.945
18	400	290	106	0.940
19	380	270	109	0.935
20	360	255	112	0.930
21	\$345	\$240	\$115	0.925
22	330	225	118	0.919
23	315	210	121	0.910
24	300	200	125	0.900
25	290	190	129	0.890
26	280	180	133	0.880
27	265	170	137	0.865
28	250	160	141	0.850
29	240	150	145	0.820
30	230	145	150	0.790
31	220	140	155	0.760
32	210	135	160	0.730
33	200	130	167	0.660
34	190	120	175	0.590
35	180	115	182	0.510
36	170	110	190	0.430
37	160	105	205	0.300
38	150	95	220	0.200
39	140	87	235	0.100
40	130	80	250	0

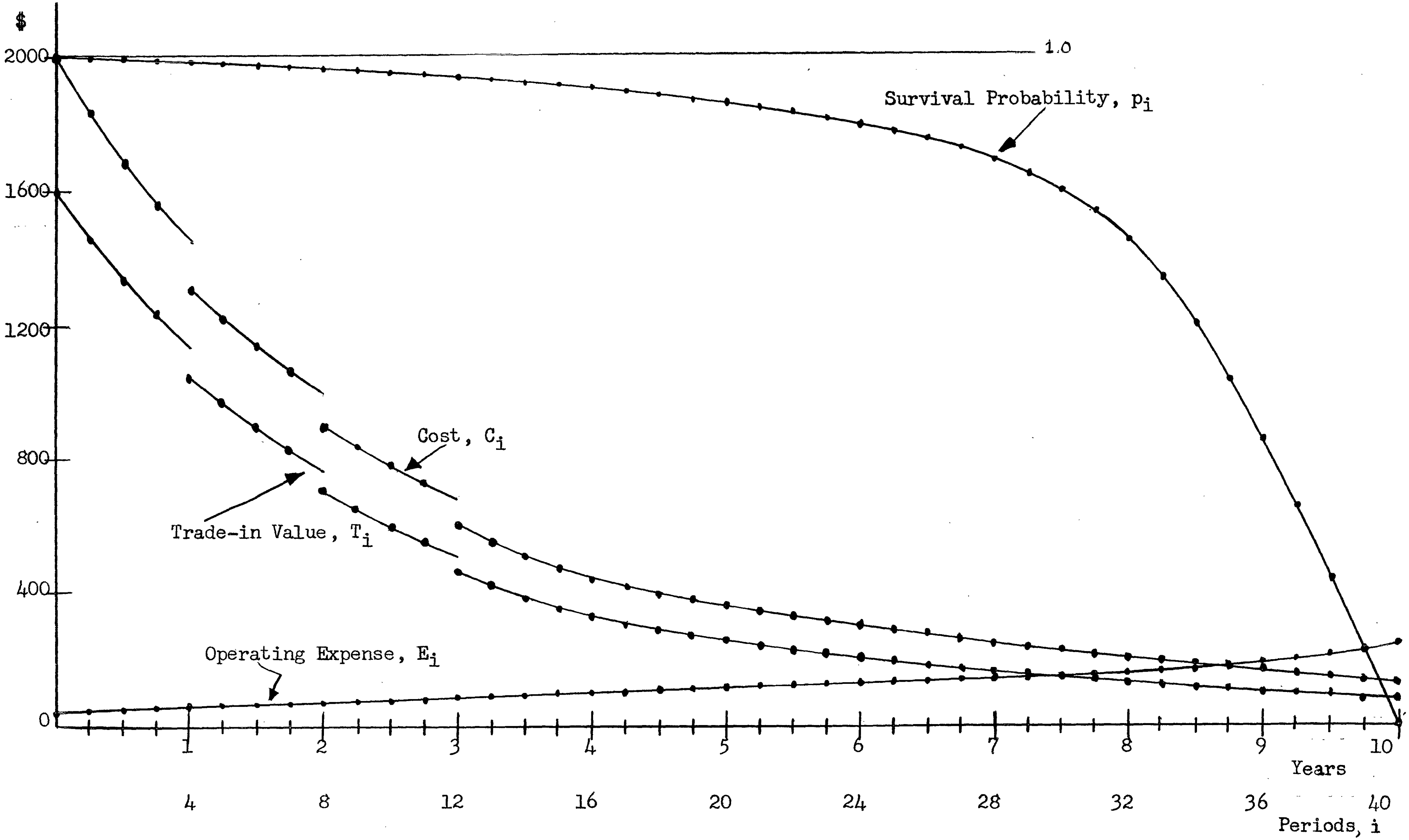


Figure 3

AUTOMOBILE REPLACEMENT DATA

The automobile replacement problem was solved by the simultaneous equation method in seven iterations. The sequence of policies, gains and values is shown in Tables IV, V and VI. The optimal policy given by iteration 7 is this: If you have a car which is more than 1/2 year old but less than 6-1/2 years old, keep it. If you have a car of any other age, trade it in on a 3-year-old car. This seems to correspond quite well with our intuitive notions concerning the economics of automobile ownership. It is satisfying to note that the program at any iteration requires that if we are going to trade, we must trade for a car whose age is independent of our present car's age. This is just the result that the logic of the situation would dictate.

If we follow our optimal policy, we will keep a car until it is 6-1/2 years old and then buy a 3-year-old car. Suppose, however, that when our car is four years old, a friend offers to swap his 1-year-old car for ours for an amount a. Should we take up his offer? In order to answer this question, we must look at the transient values.

In each of the iterations, the value of state 40 was set equal to zero, for computational purposes. Table VI also shows the values under the best policy when the value of state 40 is set equal to \$80, the trade-in value of a car of that age. When this is done, each v_i represents the value of a car of age i to a person who is following the optimal policy. In order to answer the question posed above, we must compare the value of a 1-year-old car, $v_4 = \$1,151.93$, with the value of a 4-year-old car, $v_{16} = \$421.80$. If his asking price, a, is less than $v_4 - v_{16} \approx \$730$, we should make the trade; otherwise, not. It is, of course, not necessary to change v_{40} from zero in order to answer this problem; however, making $v_{40} = \$80$ does give the values an absolute physical interpretation as well as a relative one.

TABLE IV

AUTOMOBILE REPLACEMENT RESULTS

	Iteration 1		Iteration 2		Iteration 3	
	Gain -250.00		Gain -193.89		Gain -162.44	
State	Decision	Value	Decision	Value	Decision	Value
1	Buy 36	$1373.61	Buy 20	$1380.00	Buy 19	$1380.00
2	"	1253.61	"	1260.00	"	1260.00
3	"	1143.61	"	1150.00	"	1150.00
4	"	963.61	"	970.00	Keep	1036.63
5	"	893.61	"	900.00	"	939.95
6	"	823.61	"	830.00	"	847.60
7	"	753.61	"	760.00	Buy 19	760.00
8	"	623.61	"	630.00	Keep	695.44
9	"	563.61	"	570.00	"	617.26
10	"	513.61	"	520.00	"	542.04
11	"	463.61	"	470.00	Buy 19	470.00
12	"	393.61	"	400.00	"	400.00
13	"	343.61	"	350.00	Keep	575.00
14	"	303.61	"	310.00	"	520.79
15	"	273.61	"	280.00	"	470.15
16	"	243.61	"	250.00	"	422.74
17	"	223.61	"	230.00	"	379.26
18	"	203.61	"	210.00	"	338.44
19	"	183.61	"	190.00	"	300.00
20	"	168.61	Keep	280.00	"	263.70
21	Keep	875.93	"	213.02	"	229.32
22	"	801.00	Buy 20	145.00	"	196.62
23	"	727.97	"	130.00	"	165.60
24	"	658.21	"	120.00	"	136.44
25	"	592.45	"	110.00	Buy 19	110.00
26	"	529.72	"	100.00	"	100.00
27	"	469.00	"	90.00	"	90.00
28	"	411.56	"	80.00	"	80.00
29	"	355.95	"	70.00	"	70.00
30	"	306.04	"	65.00	"	65.00
31	"	260.81	"	60.00	"	60.00
32	"	218.18	"	55.00	"	55.00
33	"	175.58	"	50.00	"	50.00
34	"	140.28	"	40.00	"	40.00
35	"	110.64	"	35.00	"	35.00
36	"	83.61	"	30.00	"	30.00
37	"	54.90	"	25.00	"	25.00
38	"	33.00	"	15.00	"	15.00
39	"	15.00	"	7.00	"	7.00
40	"	0.00	"	0.00	"	0.00

TABLE V

AUTOMOBILE REPLACEMENT RESULTS

	Iteration 4		Iteration 5		Iteration 6	
	Gain	-157.07	Gain	-151.05	Gain	-150.99
State	Decision	Value	Decision	Value	Decision	Value
1	Buy 12	$1380.00	Buy 12	$1380.00	Buy 12	$1380.00
2	"	1260.00	"	1260.00	"	1260.00
3	"	1150.00	"	1150.00	"	1150.00
4	"	970.00	Keep	1002.62	Keep	1072.26
5	"	900.00	"	917.24	"	987.22
6	"	830.00	"	836.21	"	906.67
7	"	760.00	Buy 12	760.00	"	831.16
8	"	630.00	Keep	760.54	"	760.30
9	"	570.00	"	694.91	"	694.73
10	"	520.00	"	632.62	"	632.50
11	"	470.00	"	574.05	"	573.99
12	Keep	520.00	"	520.00	"	520.00
13	"	463.84	"	470.05	"	470.12
14	"	411.16	"	423.84	"	423.97
15	"	361.55	"	381.03	"	381.23
16	"	314.63	"	341.34	"	341.61
17	"	271.11	"	305.57	"	305.92
18	"	229.67	"	272.50	"	272.95
19	Buy 12	190.00	"	241.97	"	242.51
20	"	175.00	"	213.82	"	214.46
21	"	160.00	"	187.93	"	188.68
22	"	145.00	"	164.19	"	165.07
23	"	130.00	"	142.70	"	143.73
24	"	120.00	"	123.79	"	124.99
25	"	110.00	"	108.60	Buy 12	110.00
26	"	100.00	"	97.25	"	100.00
27	"	90.00	Buy 12	90.00	"	90.00
28	"	80.00	"	80.00	"	80.00
29	"	70.00	"	70.00	"	70.00
30	"	65.00	"	65.00	"	65.00
31	"	60.00	"	60.00	"	60.00
32	"	55.00	"	55.00	"	55.00
33	"	50.00	"	50.00	"	50.00
34	"	40.00	"	40.00	"	40.00
35	"	35.00	"	35.00	"	35.00
36	"	30.00	"	30.00	"	30.00
37	"	25.00	"	25.00	"	25.00
38	"	15.00	"	15.00	"	15.00
39	"	7.00	"	7.00	"	7.00
40	"	0.00	"	0.00	"	0.00

TABLE VI

AUTOMOBILE REPLACEMENT RESULTS

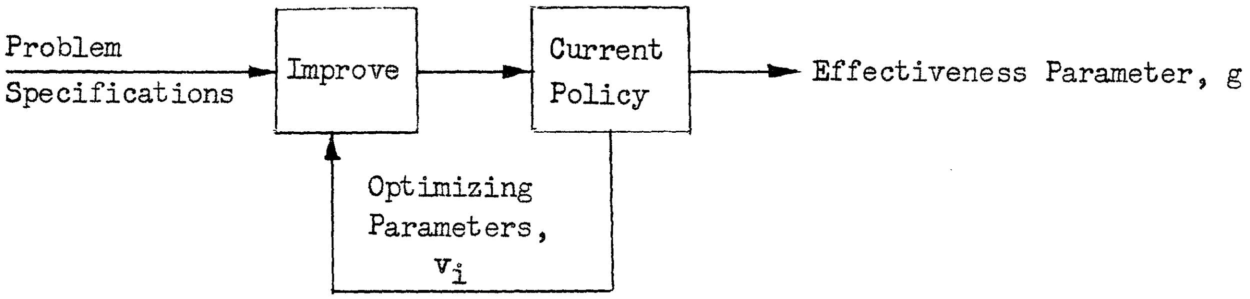
	Iteration 7		Iteration 7 Values
	Gain -150.95		$V_{40} = \$80$, Trade-in Value
State	Decision	Value	
1	Buy 12	$1380.00	$1460.00
2	"	1260.00	1340.00
3	Keep	1160.66	1240.66
4	"	1071.93	1151.94
5	"	986.93	1066.93
6	"	906.43	986.43
7	"	830.96	910.96
8	"	760.13	840.13
9	"	694.61	774.61
10	"	632.41	712.41
11	"	573.95	653.95
12	"	520.00	600.00
13	"	470.16	550.16
14	"	424.05	504.05
15	"	381.36	461.36
16	"	341.80	421.80
17	"	306.16	386.16
18	"	273.24	353.24
19	"	242.87	322.87
20	"	214.89	294.89
21	"	189.19	269.19
22	"	165.67	245.67
23	"	144.42	224.42
24	"	125.80	205.80
25	"	110.95	190.95
26	Buy 12	100.00	180.00
27	"	90.00	170.00
28	"	80.00	160.00
29	"	70.00	150.00
30	"	65.00	145.00
31	"	60.00	140.00
32	"	55.00	135.00
33	"	50.00	130.00
34	"	40.00	120.00
35	"	35.00	115.00
36	"	30.00	110.00
37	"	25.00	105.00
38	"	15.00	95.00
39	"	7.00	87.00
40	"	0.00	80.00

If the optimal policy is followed, the yearly cost of transportation is about $604 (4 x $150.95). If the policy of maximizing immediate return shown in iteration 1 were followed, the yearly cost would be $1,000. Thus, following a policy which maximizes future return rather than immediate return has resulted in a saving of almost $400 per year. The decrease of period cost with iteration is shown in Figure 4. The gain approaches the optimal value roughly exponentially. Notice that the gains for the last three iterations are so close that for all practical purposes the corresponding policies may be considered to be equivalent. The fact that a 3-year-old car is the best buy is discovered as early as iteration 4. The model year discontinuity occurring at three years is no doubt responsible for this particular selection.

The replacement problem described above is typical of a large class of industrial replacement problems. Placing these problems in the framework of the policy iteration method requires only a thorough understanding of their peculiarities and some foresight in selecting a suitable formulation.

The Iteration Cycle as a Control System

It is possible to view the entire iteration process as a unique type of control system. The system might be diagrammed as follows:



Here the box labelled "Improve" is the policy improvement routine, while the box labelled "Current Policy" is the value determination operation. The

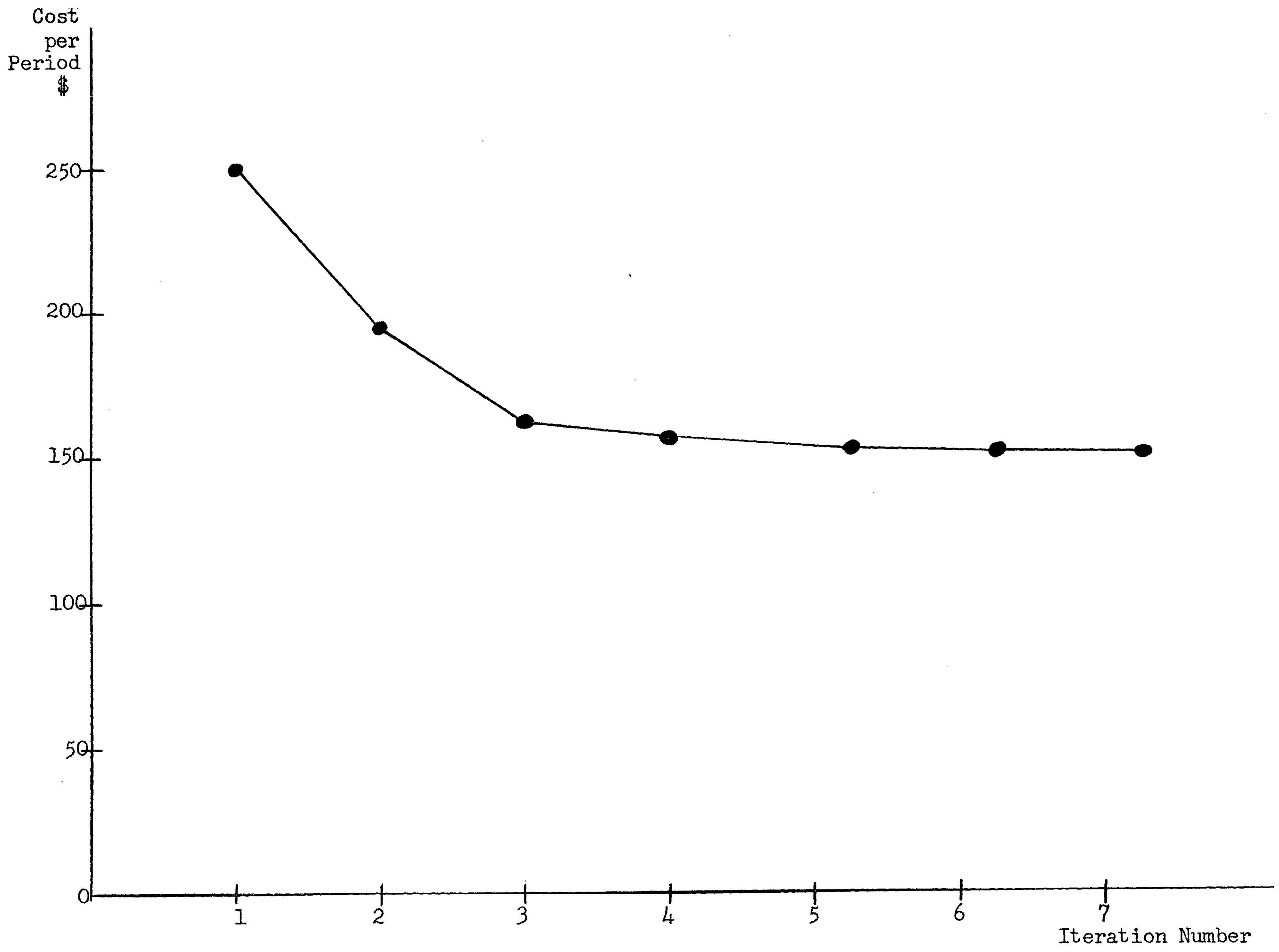
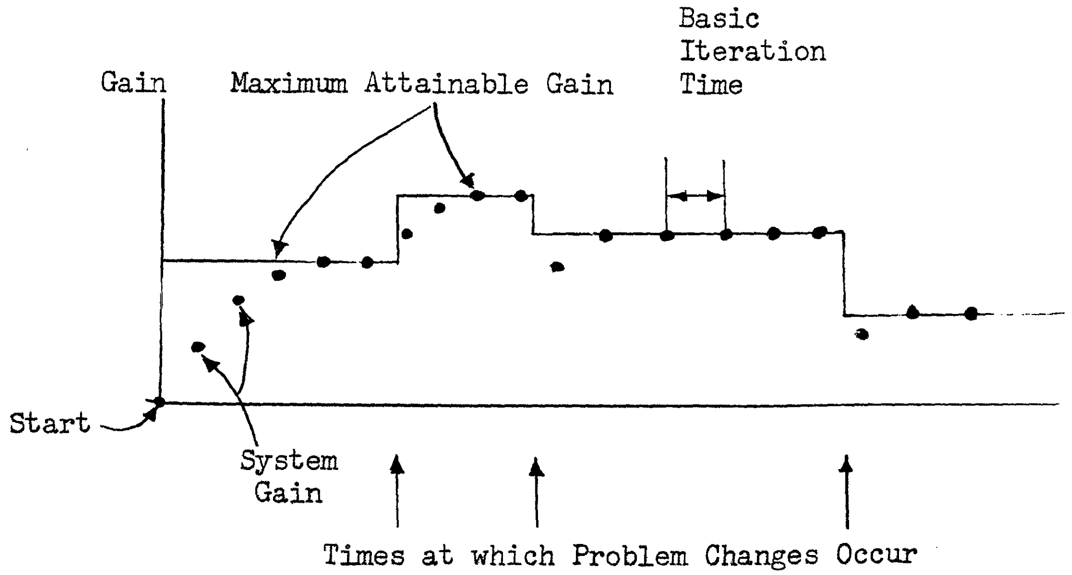


Figure 4

AUTOMOBILE REPLACEMENT SUMMARY

system works as follows. When a start order is provided, the improve box improves on the current policy by picking a new policy from among the alternatives available and by using as part of its improvement procedure the optimizing parameters v_i of the current policy. The feedback of the optimizing parameters v_i is what makes the system unique. Normally feedback control systems are designed to improve themselves based on their actual output (g in this case). However, this system generates a special set of optimizing parameters (the v_i) in order to improve its performance. If the system is used on a fixed problem it will iterate until it has achieved the maximum g and then it may stop because no further improvement is possible. A more interesting case is the one in which the problem specifications are changed at times (so that the maximum gain changes) while the system is continually iterating. In this situation the system exhibits very interesting behavior of the type sketched below:



The system is attempting to follow changes in the problem so that the largest return (perhaps minimum error) will be achieved in the long run.

The point of this whole discussion is that the dynamic programming scheme outlined above is a type of adaptive system. It should be possible to build high-level servomechanisms using the ideas that have been developed.

Summary

A policy iteration method for solving sequential decision processes of long duration has been presented. The general iteration cycle of this method is shown in Figure 5. Note that this diagram includes the case where the process may have ~~many~~ independent gains.

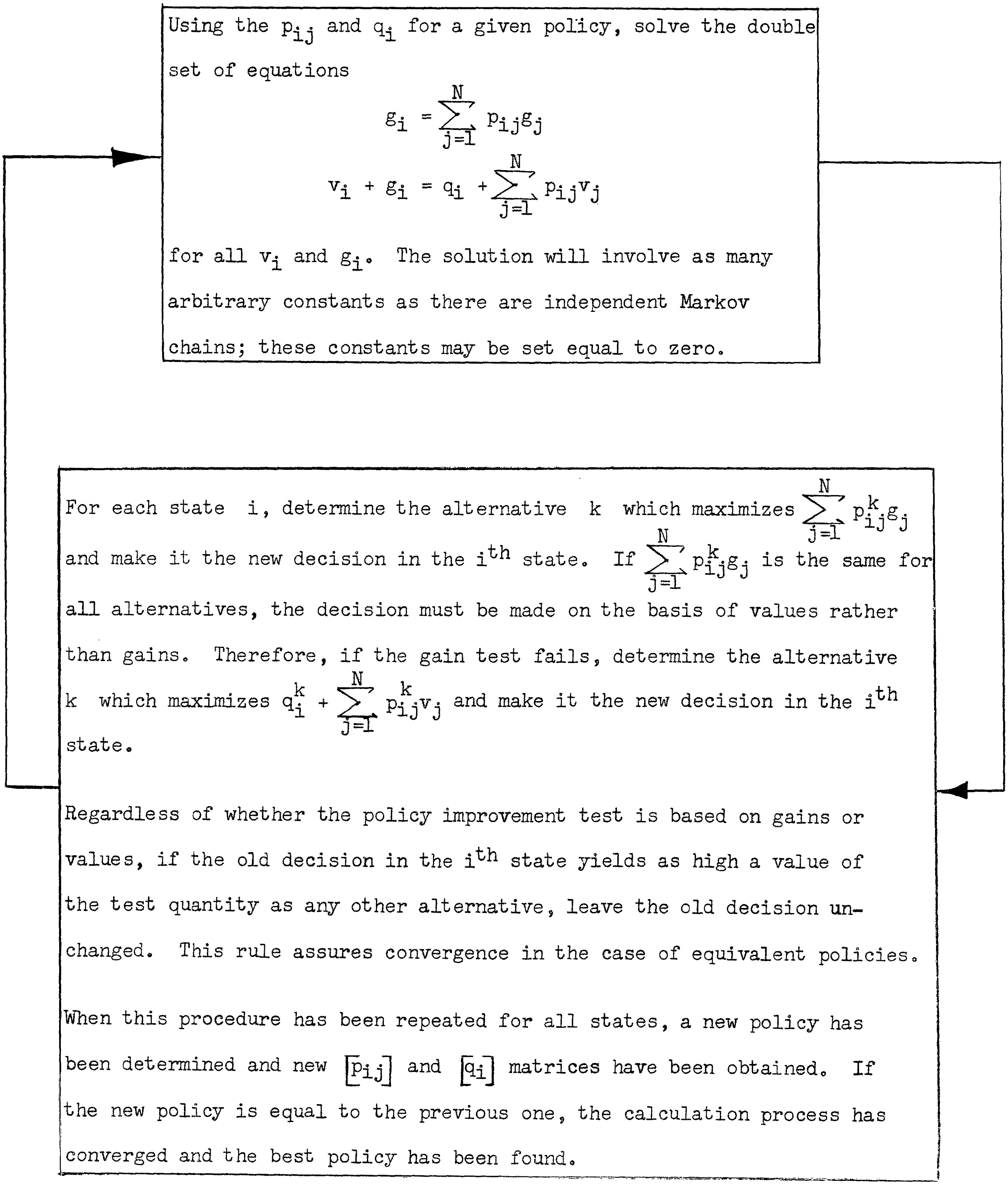
The properties of the process that have been derived are the following:

1. The total expected return in n moves starting from state i, V_i^n, can be represented in the form $v_i + ng_i$ for very large n. The transient values v_i and the gains g_i depend only on the starting state i. In most practical cases, g_i is independent of i and may be given the symbol g. The quantity g is called the gain of the system and is the average return per transition as the number of moves becomes very large.

2. The value determination operation is computationally equivalent to the solution of the Markov process for its absolute probabilities. The values and gains of the process can be found either deterministically or by a Monte Carlo process. The deterministic method requires only the solution of linear simultaneous equations and enjoys the advantage of relatively high speed. The simulation method yields more physical insight into the process and avoids some of the accuracy problems that may be encountered in solving large sets of simultaneous equations.

Figure 5

DIAGRAM OF GENERAL ITERATION CYCLE



Using the p_{ij} and q_i for a given policy, solve the double set of equations

$$g_i = \sum_{j=1}^{N} p_{ij} g_j$$

$$v_i + g_i = q_i + \sum_{j=1}^{N} p_{ij} v_j$$

for all v_i and g_i. The solution will involve as many arbitrary constants as there are independent Markov chains; these constants may be set equal to zero.

For each state i, determine the alternative k which maximizes $\sum_{j=1}^{N} p_{ij}^{k} g_j$ and make it the new decision in the i^{th} state. If $\sum_{j=1}^{N} p_{ij}^{k} g_j$ is the same for all alternatives, the decision must be made on the basis of values rather than gains. Therefore, if the gain test fails, determine the alternative k which maximizes $q_i^k + \sum_{j=1}^{N} p_{ij}^{k} v_j$ and make it the new decision in the i^{th} state.

Regardless of whether the policy improvement test is based on gains or values, if the old decision in the i^{th} state yields as high a value of the test quantity as any other alternative, leave the old decision unchanged. This rule assures convergence in the case of equivalent policies.

When this procedure has been repeated for all states, a new policy has been determined and new $\left[p_{ij}\right]$ and $\left[q_i\right]$ matrices have been obtained. If the new policy is equal to the previous one, the calculation process has converged and the best policy has been found.

3. The policy improvement routine will find a better policy than the present one if such a policy exists; in other words, this routine must make a finite increase in the gain if such an increase is possible. Furthermore, the policy improvement routine will find that policy which has the highest attainable gain; it cannot be misled by relative maxima. If the policy improvement routine cannot increase the gain, then it must increase the transient values.

The main advantage of the policy iteration approach is that it finds the long-run behavior of the decision process directly rather than asymptotically. Since it avoids the Principle of Optimality and the consequent computational difficulties, it becomes feasible to work small problems with a pencil and a piece of paper. This advantage is not trivial from an expository point of view.

The transient values, v_i, have an important use in determining how much one should be willing to pay to make an instantaneous change of state; they are indispensable in making decisions regarding "special offers."

The limitations of the method are clear from its assumptions. First, one must be willing to continue the process for a large number of moves; the results are inapplicable to decisions which are to be made only a few times. Second, the transition probabilities and rewards for each alternative must be known. The game to be played is one of complete information.

Much additional theoretical work remains to be done. One could consider the extension of the above methods to cases of incomplete information and policy restrictions. The question of how to define states originally is still open. Problems involved in the introduction of discounting of future returns have yet to be investigated.

In the realm of application there is a variety of interesting and important problems. Problems in inventory control and industrial replacement which can be formulated in discrete terms are almost without number. One of the most interesting areas of application could be in the area of hydroelectric power generation. If the state of the system is defined by the head of the dam in feet, then we may speak of probabilistic state transitions caused by rainfall and customer demand and of alternatives such as producing electric power from coal. The complexity of the problem considered would depend only on the size of the computational equipment available.

There are problems in many areas of system analysis that may be formulated as Markovian decision processes. The policy iteration method should provide a convenient approach to their solution.

BIOGRAPHICAL NOTE

Ronald Arthur Howard was born in New York, New York, on August 27, 1934. He entered Massachusetts Institute of Technology in 1951 on a Grumman Aircraft Engineering Corporation scholarship, and in 1955 received a Bachelor of Science degree in Electrical Engineering and a Bachelor of Science degree in Economics and Engineering. From September 1955 until he received a Master of Science degree in Electrical Engineering in June 1956, his education was financed by a Schlumberger Fellowship. He received the Electrical Engineer degree in June 1957 as an International Business Machines Corporation Fellow.

During the summer of 1955 he performed research on transistor and magnetic amplifier devices at the Raytheon Manufacturing Company. He was associated with the Operations Research Group at Arthur D. Little, Inc., first as an employee during the summer of 1956 and then as a consultant during the school years of 1956-1957 and 1957-1958. He was employed by the Management Services Department of the Ramo-Wooldridge Corporation in Los Angeles during the summer of 1957. As a Ramo-Wooldridge Fellow at Massachusetts Institute of Technology for the academic year 1957-1858, he pursued his doctoral program and taught a graduate course in discrete systems analysis during the spring term.

He is a member of Eta Kappa Nu, Tau Beta Pi, Sigma Xi, the American Institute of Electrical Engineers, the Institute of Radio Engineers, and the Operations Research Society of America.

BIBLIOGRAPHY

1. Beckenbach, Modern Mathematics for the Engineer, McGraw Hill, 1956, p.28.

2. Bellman, R., Dynamic Programming, Princeton University Press, 1957.

3. Doob, J. L., Stochastic Processes, Wiley, New York, 1953, p. 175 ff.

4. Friedman, Bernard, Principles and Techniques of Applied Mathematics, Wiley, New York, 1956, p. 67.

5. Sittler, R. W., Systems Analysis of Discrete Markov Processes, IRE Transactions on Circuit Theory, Volume CT-3, Number 1, December 1956, pp. 257-266.

6. Feller, W., An Introduction to Probability Theory and Its Applications, Volume I, Second Edition, Wiley, New York, 1957.

7. Huggins, W. H., Signal-Flow Graphs and Random Signals, Proc. IRE, Volume 45, pp. 74-86, January 1957.

8. Bellman, R., A Markovian Decision Process, Journal of Mathematics and Mechanics, Volume 6, Number 5 (1957).

9. Sarymsakov, T. A., Osnovy Teorii Protšessov Markova, State Publishers, Moscow, 1949.

10. Romanovskii, V. I., Diskretnye Tšepi Markova, State Publishers, Moscow, 1949.

11. Elving, Gustav, Zur Theorie der Markoffschen Ketten, Acta Societatis Scientarum Fennicae, Volume 2, Number 8, 1937.

12. Kimball, G. E., Monte Carlo Methods, Technical Memorandum, Arthur D. Little, Inc., 1957.