OPTIMUM TEACHING PROCEDURES DERIVED FROM
MATHEMATICAL LEARNING MODELS

by
James E. Matheson

Report CCS-2
Prepared under contract with
Westinghouse Research Laboratories

COMPUTER-COORDINATED SYSTEMS
INSTITUTE IN ENGINEERING-ECONOMIC SYSTEMS
STANFORD UNIVERSITY
STANFORD, CALIFORNIA

August 1964
ABSTRACT

This work is an initial investigation of the optimization of teaching procedures through the use of mathematical learning models. The typical situation studied is that of teaching a list of paired-associate items in a fixed number of presentations. At each step in the process an item is selected by the decision procedure and presented to the student. A reward structure is formulated in order to measure the effectiveness of the teaching; and in terms of the reward structure and the learning model, optimum teaching procedures are derived by dynamic programming techniques.

In the usual application of dynamic programming to Markov processes, it is assumed that the state of the Markov process is directly observable at each step in the process. A consequence of this assumption is that all decisions in the process can be based on the state of the Markov process without regard to the past history of the process. However for Markov learning models such as the one-element model, the state of the model is not directly observable. The observations available depend on the state of the model in a probabilistic manner. It is shown that, for the purpose of optimization, an equivalent Markov process can be found in observable states of history. The observable process usually has more states than the original learning model. This new process can then be treated by conventional methods to optimize the process.
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>A. The General Problem</td>
<td>1</td>
</tr>
<tr>
<td>B. Background and Organization of this Work</td>
<td>1</td>
</tr>
<tr>
<td>1. Programmed Instruction</td>
<td>1</td>
</tr>
<tr>
<td>2. Mathematical Learning Models</td>
<td>2</td>
</tr>
<tr>
<td>3. Optimization</td>
<td>4</td>
</tr>
<tr>
<td>II. DYNAMIC PROGRAMMING WITH OBSERVABLE STATES</td>
<td>7</td>
</tr>
<tr>
<td>A. Direct Enumeration</td>
<td>7</td>
</tr>
<tr>
<td>B. Dynamic Programming</td>
<td>8</td>
</tr>
<tr>
<td>C. Dynamic Programming Example</td>
<td>11</td>
</tr>
<tr>
<td>D. Intuitive Suboptimal Policies</td>
<td>14</td>
</tr>
<tr>
<td>III. MOTIVATING EXAMPLES FOR UNOBSERVABLE STATES</td>
<td>17</td>
</tr>
<tr>
<td>A. The Problem</td>
<td>17</td>
</tr>
<tr>
<td>B. The Model</td>
<td>17</td>
</tr>
<tr>
<td>C. The Reward Structure</td>
<td>18</td>
</tr>
<tr>
<td>D. The Observable Markov Process</td>
<td>19</td>
</tr>
<tr>
<td>E. Expected Immediate Reward</td>
<td>21</td>
</tr>
<tr>
<td>F. The Application of Dynamic Programming</td>
<td>21</td>
</tr>
<tr>
<td>G. Maximization of Expected Immediate Reward</td>
<td>22</td>
</tr>
<tr>
<td>IV. DYNAMIC PROGRAMMING WITH UNOBSERVABLE STATES</td>
<td>25</td>
</tr>
<tr>
<td>A. The Teaching Process</td>
<td>25</td>
</tr>
<tr>
<td>B. The Learning Model</td>
<td>26</td>
</tr>
<tr>
<td>C. History Space</td>
<td>27</td>
</tr>
<tr>
<td>D. Measures of Effectiveness</td>
<td>27</td>
</tr>
<tr>
<td>E. Optimization</td>
<td>28</td>
</tr>
<tr>
<td>1. Notation</td>
<td>28</td>
</tr>
<tr>
<td>2. Solution by Enumeration</td>
<td>29</td>
</tr>
<tr>
<td>F. Dynamic Programming Solution</td>
<td>29</td>
</tr>
<tr>
<td>1. Markov Histories</td>
<td>30</td>
</tr>
<tr>
<td>2. Properties of Markov Histories</td>
<td>30</td>
</tr>
<tr>
<td>3. Recursive Equations</td>
<td>33</td>
</tr>
</tbody>
</table>
G. Sufficient Histories ......................................... 35

V. ONE-ELEMENT LEARNING MODEL WITH FIXED PARAMETERS ......................................... 36
   A. The Learning Model ....................................... 36
   B. The Sufficient History ................................... 37
      1. Transition Probabilities ................................. 39
      2. Expected Immediate Reward .............................. 41
   C. Dynamic Programming Solution ........................... 42

VI. ONE-ELEMENT LEARNING MODEL WITH AN UNKNOWN PARAMETER ......................................... 43
   A. Prior Distribution on the Conditioning Probability .... 43
   B. Choice of Parameters in the Prior Density ............... 45
   C. Sufficient History ......................................... 46
   D. Expected Immediate Reward ................................. 47
   E. Transition Probabilities ................................... 49
   F. Dynamic Programming Equations ........................... 50
   G. Maximization of Expected Immediate Reward ............. 51

VII. A STATIC PROBLEM ........................................ 54
   A. The Problem ............................................... 54
   B. The One-Element Model .................................... 54
      1. Dynamic Programming Formulation ....................... 55
      2. Maximization of Expected Immediate Reward .......... 56
   C. Mean-Learning-Curve Formulation ........................ 56
      1. The One-Element Model .................................. 56
      2. The Linear Model ....................................... 58
      3. An Alternative Derivation of the Solution ............ 58
      4. An Approximate Solution ................................ 60

VIII. COMPUTATIONAL RESULTS ................................. 62
   A. Structure of the Optimal Policies ........................ 62
   B. Suboptimal Policies ....................................... 63

IX. CONCLUSIONS ............................................... 66
   A. Learning Models and Optimization ........................ 66
   B. Dynamic Programming ..................................... 67

REFERENCES .................................................. 69
TABLES

<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dynamic programming quantities for the example</td>
<td>13</td>
</tr>
<tr>
<td>2</td>
<td>Dynamic programming calculations for the example with the initial reward structure</td>
<td>13</td>
</tr>
<tr>
<td>3</td>
<td>Dynamic programming calculations for the example with the modified reward structure</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>Probability of the unknown state $x^n$ given the state of sufficient history $z^n$</td>
<td>20</td>
</tr>
<tr>
<td>5</td>
<td>Expected immediate reward for presenting item $i$ given the state of the sufficient history $z^n_1$</td>
<td>21</td>
</tr>
<tr>
<td>6</td>
<td>Markov transition probabilities for the sufficient history $z^n_1$ when item $i$ is presented</td>
<td>40</td>
</tr>
<tr>
<td>7</td>
<td>Markov transition probabilities for the sufficient history $z^n_{i_1} = \left(r^n_{i_1}, b^n_{i_1}\right)$ when item $i$ is presented</td>
<td>50</td>
</tr>
</tbody>
</table>

ILLUSTRATIONS

Figure

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Comparison of optimal and suboptimal policies as a function of $c_2$ with $p_1 = 0.5$, $c_1 = 0.2$, $p_2 = 0.2$</td>
<td>64</td>
</tr>
<tr>
<td>2</td>
<td>Comparison of optimal and suboptimal policies as a function of $c_2$ with $p_1 = 0.5$, $c_1 = 0.2$, $p_2 = 0.5$</td>
<td>65</td>
</tr>
</tbody>
</table>
ACKNOWLEDGMENT

The author thanks Professor William K. Linvill of the Institute in Engineering-Economic Systems and Professor Richard C. Atkinson of the Institute for Mathematical Studies in the Social Sciences, under whose joint direction this research was developed into a doctoral dissertation. Appreciation is expressed for the helpful comments and encouragement of Professors David G. Luenberger and Patrick C. Suppes.

Acknowledgment is also made of the support of the Westinghouse Research Laboratories.
I. INTRODUCTION

A. THE GENERAL PROBLEM

The purpose of this study is to investigate optimum teaching procedures through the use of mathematical models of learning. Since this work is an initial investigation of this problem area, its scope is limited to certain restricted types of teaching situations for which reasonably accurate learning models are available. This study gives some insight into the interrelationship between the optimization procedure and the mathematical model used to describe the learning process. Also, a theory of optimization of Markov processes with unobservable states unifies most of the situations considered. Hopefully, these results have some bearing on the optimization and modeling problems in more general situations.

Two models used extensively in this work are a linear model and a one-element model. The typical teaching situation studied is that of teaching a list of paired-associate items in a fixed number of presentations. A reward structure is formulated to measure the effectiveness of the teaching, and in terms of the reward structure and the learning model, optimum teaching procedures are derived by the application of dynamic programming techniques.

B. BACKGROUND AND ORGANIZATION OF THIS WORK

1. Programmed Instruction

Much previous work has been done on programmed instruction. Much of the significant early work is available in a source book edited by Lumsdaine and Glaser [Ref. 1], which was published in 1960. A relatively complete book surveying the teaching-machine field was published in 1961 by Stolowe [Ref. 2]. A short survey of the teaching-machine literature with an extensive bibliography was written by Silberman [Ref. 3] in 1962. The general approach in the programmed instruction field is to write the programmed materials by more or less heuristic methods. That is, the programmer uses a few techniques, such as
reinforcement, stimulus vanishing, branching, multiple choice response, and constructed response, and combines these techniques with his good judgment and intuition to write his programmed materials. He then tries out his materials on some students, and uses the results to improve his program. Hopefully after several such cycles the program becomes nearly optimum. Unfortunately, the success of this method of programming depends critically on some qualitative skills of the programmer.

2. Mathematical Learning Models

The optimization techniques developed in this work depend on mathematical models describing the learning process of the student. The author does not claim that adequate learning models are currently available for describing complex-learning situations. The approach here is to use simple learning models, which have been shown to describe certain restricted types of learning situations fairly well [Ref. 4], in order to investigate what statements about optimum teaching procedures can be inferred from these models. This study focuses on the problems encountered in mathematically optimizing the teaching process, assuming that the models hold, and points out some problems resulting from characteristics of the models used.

The two models described here have been applied to several learning situations. The learning situation used as an example throughout this work is that of paired-associate learning. The objective of the optimization is to teach the student M paired-associate item pairs in a total of N presentations. In each step in the process any one of the item pairs may be presented. At each step a decision is made, based on the past history of the process, to determine the optimum item pair for presentation during the next step.

In a presentation of an item pair the student is shown a stimulus item and is asked to respond with the correct response item. At the conclusion of the presentation, he is shown the correct response item to reinforce his ability to make the correct choice. Typical paired associates may be foreign words and their translation into English.

The linear model describes the probability of a correct response to a stimulus item as a function of the number of presentations.
Background on linear models appears in Ref. 5, 6, and 7. Let \( P_z \) be the probability of a correct response on presentation \( z+1 \) of an item. Then the linear model is described by the difference equation

\[
P_{z+1} = (1-c)P_z + c, \quad 0 \leq P_0 = p \leq 1, \quad 0 \leq c \leq 1
\]

where \( c \) is the learning parameter and \( p \) is the initial guessing probability. This difference equation has the solution

\[
P_z = 1 - (1-p)(1-c)^z
\]

(1.1)

The above equation is the mean learning curve for the linear model.

The one-element model [see Atkinson and Estes, Ref. 4], is based on the assumption that the student has a binary state of learning for each item which is either conditioned or unconditioned. That is, he either knows the item or he does not know it. The term "one element" comes from the learning-theory assumption that the state of learning corresponds to a single stimulus element within the student. Let \( x = 1 \) denote the conditioned state and \( x = 0 \) denote the unconditioned state. If the item is presented when \( x = 1 \) (conditioned), the correct response is always made; and if the item is presented when \( x = 0 \) (unconditioned), the correct response is made with guessing probability \( p \). If the state before the reinforcement is \( x = 1 \), then the state after the reinforcement is \( x' = 1 \). On the other hand, if the state before the reinforcement is \( x = 0 \), then during the reinforcement the state makes a Markov transition to \( x' = 1 \) with probability \( c \) and to \( x' = 0 \) with probability \( 1-c \). The response probabilities and transition matrix that apply when the item is presented are as follows:

<table>
<thead>
<tr>
<th>Old state ( x )</th>
<th>New state ( x' )</th>
<th>Probability of correct response</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x = 0 )</td>
<td>( x' = 0 )</td>
<td>1</td>
</tr>
<tr>
<td>( x = 1 )</td>
<td>( x' = 1 )</td>
<td>( 1 )</td>
</tr>
<tr>
<td></td>
<td>( x' = 0 )</td>
<td>( c )</td>
</tr>
<tr>
<td></td>
<td>( x' = 1 )</td>
<td>( 1-c )</td>
</tr>
</tbody>
</table>
Methods of obtaining estimates of parameter values for these models are presented in Refs. 8 and 9.

3. Optimization

There has been only a little work done in the optimization of the teaching process using mathematical learning models. Probably the earliest attempt is contained in a paper written by Dear and Atkinson [Ref. 10] in 1962. This paper considered a process containing only two items, each of which essentially follows a one-element model. The items were administered in a probabilistic manner. Values of various quantities were found as a function of the probability of presenting each item, and some statements were made about optimal values for this probability. A later paper by Dear [Ref. 11] applies game theory to a similar two-item learning situation. This time Dear considered a decision structure which used past history to choose an optimum item for each presentation. Dear succeeded in stating the problem in the elaborate notation and terminology of game theory [Ref. 12]. However, all of this sophistication led to little insight into the optimization problems involved.

Smallwood [Ref. 13] took a different approach. He developed intuitive models for the learning process and developed optimization procedures by methods similar to dynamic programming. One of the features of Smallwood's procedure was that his model contained parameters which indicated the degree of the student's intelligence and the difficulty of the questions. The parameters were continually estimated during the course of instruction. One drawback of his work is that the models probably do not describe the learning process accurately, at least not at the item-by-item level. However, Smallwood's type of model and estimation procedure might be more applicable to the optimization of the selection of large blocks of material, such as chapters or complete lessons.

As previously indicated, this study uses the one-element model and the linear model to describe the item-by-item behavior in the learning process. The objective is to use the past history of the process to select the optimum item to be presented at each point in the process.
With the linear model the objective of the optimization is to maximize the number of correct responses on a set of test presentations administered at the end of the process. This same objective can be used with the one-element model. Another appropriate objective, when using the one-element model, is to maximize the number of items in the conditioned state at the end of the process.

A basic property of the linear model is that the response probability progresses as a function of only the number of times the item has been presented, as given by Eq. (1.1). That is, the knowledge of the history of past responses gives no more information about the response probability than the knowledge of only the number of presentations of each item. If this model holds, then there is no reason to use an active machine to observe the responses and decide on an optimum item for each presentation. Thus the presentation order can be completely determined before the process is started.

On the other hand, the response probability of the one-element model depends on the conditioning state of the item. Thus responses can be used to gain information about the underlying state of conditioning. For example, an incorrect response always indicates that the state of the item is unconditioned. Thus at each stage in the process the history of past responses must be used to select the item for presentation during the next frame.

The main technique used to derive optimal teaching procedures is dynamic programming. Although the theory of dynamic programming was initiated and developed principally by Bellman [Refs. 14, 15], the practical application of dynamic programming to Markov processes is probably best presented by Howard [Ref. 16].

The portion of Howard's theory having direct bearing on this work is reviewed here in Chapter II. A fundamental assumption in this theory is that, after the conclusion of a step in the process, the state of the item is directly observable. Chapter III serves to illustrate this problem and offers a solution for a special case. With the example of Chapter III as motivation, a general theory of optimization is developed in Chapter IV for a class of Markov processes with unobservable states.
This theory is applied to the one-element model in Chapter V, and a case of the one-element model with unknown conditioning probabilities is treated in Chapter VI. Chapter VII treats the problem of optimizing when no observations are permitted, and in this case the linear model and the one-element model are shown to be equivalent. A few computational results are presented in Chapter VIII. Finally, some general conclusions are drawn in Chapter IX about both the optimization problem, and problems inherent in the models themselves.
II. DYNAMIC PROGRAMMING WITH OBSERVABLE STATES

One of the main technical problems of this research is the optimization of discrete Markov processes in which the state of the process is not directly observable. This problem is solved in Chapter IV by conceptually finding states of observable history which form a new Markov process in these new observable states. This chapter serves to introduce the optimization of Markov processes with observable states by dynamic programming, in a manner similar to the value iteration procedure of Howard [Ref. 16].

Consider a discrete-time Markov process with a finite-dimensional discrete state space $Z$. At each stage of the process the state $z$ is observed, and based on the observation, a decision $d$ is selected from a finite-dimensional discrete decision space $D$. The process has a total of $N$ stages. With $n$ stages remaining in the process, the state is denoted $z^n$, and the decision is denoted $d^n$. Let $P(z^{n-1}|z^n,d^n)$ denote the probability of a transition to state $z^{n-1}$ from state $z^n$ when decision $d^n$ is used. The Markov property means that this distribution is independent of all past events.

In order to construct a criterion of optimality, consider two types of rewards to be "paid" to the optimizer:

1. Terminal reward, paid at the end of the process for the various terminal states, and represented by the function $\phi(z^0)$.

2. Immediate reward, paid for a specific transition from state $z^n$ to $z^{n-1}$ and depending on the decision $d^n$ used. This reward is considered to be paid at the time the transition occurs, and it is represented by the function $R(z^{n-1},z^n,d^n)$.

At each point in time, the optimum decision is the one which maximizes the total expected reward to be collected in the remaining portion of the process, given the current state and given that all succeeding decisions will be made by the same criterion of optimality.

A. DIRECT ENUMERATION

To illustrate the advantages of dynamic programming, first consider the solution of the problem by direct enumeration. If there are $n$ stages
remaining in the process and the process is in state \( z^n \), a decision \( d^n \) must be made for the next step, based on the value of \( n \) and \( z^n \). A decision policy \( d \) is a function which assigns decision \( d^n = d(n,z^n) \) for each \( n = 1, 2, ..., N \) and each \( z^n \) in the state space \( Z \). Thus \( d \) is a mapping from the direct product space \( \{1, 2, ..., N\} \times Z \) into the decision space \( D \). Since the spaces involved are finite, there are only a finite number of such mappings. Now, given a policy \( d \), the expected total reward for the whole process, using policy \( d \), is

\[
E \left[ \sum_{n=N}^{1} R(z^{n-1},z^n,d^n) + \phi(z^0) \mid d \right]
\]

Thus \( \hat{d} \) is an optimum policy if and only if

\[
E \left[ \sum_{n=N}^{1} R(z^{n-1},z^n,d^n) + \phi(z^0) \mid \hat{d} \right] \geq E \left[ \sum_{n=N}^{1} R(z^{n-1},z^n,d^n) + \phi(z^0) \mid d \right]
\]

(2.1)

for all possible decision policies \( d \). The problem with using this statement of optimality to find a solution is the relatively large number of possible policies to be considered. For example, if at each stage there are two possible decisions and four possible states, then there are \( 2^{4N} \) possible decision policies. If \( N = 10 \), there are \( 2^{40} \) or \( 1,099,511,627,766 \) cases to consider. The dynamic programming formulation reduces the number of cases, in this situation, to \( 2(4N) \) or only 80 cases for \( N = 10 \).

B. DYNAMIC PROGRAMMING

It is convenient to define a function \( r(z^n,d^n) \) as the expected immediate reward for making decision \( d^n \) when the process is in state \( z^n \). That is, \( r(z^n,d^n) \) is the expected reward to be collected during the next step of the process given that the process is in state \( z^n \) and decision \( d^n \) is made. Thus
\[ r(z^n, d^n) = E[R(z^{n-1}, z^n, d^n)|z^n, d^n] \]

\[ = \sum_{z^{n-1} \in Z} R(z^{n-1}, z^n, d^n)p(z^{n-1}|z^n, d^n) \quad (2.2) \]

Dynamic programming is an iterative technique for finding the optimal decision policy. Define \( v(n, z^n) \) as the expected value of the total of immediate and terminal rewards remaining in the process if, when the process is in state \( z^n \) with \( n \) steps remaining, decision \( d^n \) is used followed by optimum decisions thereafter. Define \( v(n, z^n) \) as the same quantity using optimum decisions for all of the \( n \) remaining steps. Then clearly

\[ v(n, z^n) = \max_{d^n} v(n, z^n) \quad (2.3) \]

Also, at the end of the process the final value of \( v \) is

\[ v(0, z^0) = \psi(z^0) \]

Now with one step remaining in the process

\[ v_{d^1}(1, z^1) = E[R(z^0, z^1, d^1) + \phi(z^0)|z^1, d^1] \]

\[ = \sum_{z^0 \in Z} [R(z^0, z^1, d^1) + \phi(z^0)]p(z^0|z^1, d^1) \]

\[ = \sum_{z^0 \in Z} R(z^0, z^1, d^1)p(z^0|z^1, d^1) + \sum_{z^0 \in Z} \phi(z^0)p(z^0|z^1, d^1) \quad (2.4) \]
or, making appropriate identifications,

\[
v_d^1(1, z^1) = r(z^1, d^1) + \sum_{z^0 \in Z} v(0, z^0) p(z^0 | z^1, d^1)
\]  

(2.5)

Thus

\[
v(l, z^l) = \max_{d^l} \left[ r(z^l, d^l) + \sum_{z^0 \in Z} v(0, z^0) p(z^0 | z^l, d^l) \right]
\]  

(2.6)

and \( \hat{d}(l, z^l) \) is equal to a \( d^l \) for which the above maximum occurs.

For general \( n \) the equation is similar, but it requires the Markov property in the derivation. Since the optimum decision policy is calculated recursively, when \( \hat{d}(n, z^n) \) is being calculated \( \hat{d}(j, z^j) \) is known for all \( j \) less than \( n \). Thus

\[
v_{d^n}(n, z^n) = E \left[ R(z^{n-1}, z^n, d^n) + \sum_{j=n-1}^1 R(z^{j-1}, z^j, \hat{d}(j, z^j)) + \phi(z^0) \left| z^n, d^n \right. \right]
\]  

\[
= r(z^n, d^n) + E \left[ \sum_{j=n-1}^1 R(z^{j-1}, z^j, \hat{d}(j, z^j)) + \phi(z^0) \left| z^n, d^n \right. \right]
\]  

(2.7)

Now, by the properties of conditional expectation

\[
E \left[ \sum_{j=n-1}^1 R(z^{j-1}, z^j, \hat{d}(j, z^j)) + \phi(z^0) \left| z^n, d^n \right. \right] = E \left[ E \left[ \sum_{j=n-1}^1 R(z^{j-1}, z^j, \hat{d}(j, z^j)) + \phi(z^0) \left| z^{n-1}, z^n, d^n \right. \right] \left| z^n, d^n \right. \right]
\]  

(2.8)
Since the inner expectation depends only on events occurring after $z^{n-1}$, it is not dependent on $z^n$ or $d^n$; that is, the condition $z^{n-1}, z^n, d^n$ can be reduced to the condition $z^{n-1}$ by the Markov property. Applying this fact and recognizing the term $v(n-1, z^{n-1})$ gives

\[
\mathbb{E} \left[ \sum_{j=n-1}^{\infty} R(z^{j-1}, z^j, \hat{d}(j, z^j)) + s(z^0) \Big| z^{n-1}, z^n, d^n \right] = \mathbb{E} \left[ \sum_{j=n-1}^{1} R(z^{j-1}, z^j, \hat{d}(j, z^j)) + s(z^0) \Big| z^{n-1} \right]
\]

\[
= \mathbb{E}[v(n-1, z^{n-1})|z^{n-1}] = \sum_{z^{n-1} \in Z} v(n-1, z^{n-1}) p(z^{n-1}|z^n, d^n)
\]  

(2.9)

Combining Eqs. (2.7), (2.8), and (2.9) yields

\[
v_d(n, z^n) = r(z^n, d^n) + \sum_{z^{n-1} \in Z} v(n-1, z^{n-1}) p(z^{n-1}|z^n, d^n)
\]  

(2.10)

Thus

\[
v(n, z^n) = \max_{d^n} \left[ r(z^n, d^n) + \sum_{z^{n-1} \in Z} v(n-1, z^{n-1}) p(z^{n-1}|z^n, d^n) \right]
\]  

(2.11)

and $\hat{d}(n, z^n)$ is equal to a $d^n$ for which the above maximum occurs.

Equation (2.11) is the basic recurrence relation of dynamic programming.

C. DYNAMIC PROGRAMMING EXAMPLE

The purpose of the following example is to illustrate the use of the dynamic programming equations. The Markov model to be considered is similar to the learning models used in the sequel, but it is not presented as an empirically adequate model of the learning situation described.
Suppose we have 4 days to teach a student some material. On each of the 4 days he comes to us for a lesson. When he arrives we ask him a few questions, and from his answers we can decide whether he knows the material. We then have the choice of giving him a lesson, which costs us $10.00, or sending him home, which costs us nothing. At the end of 4 days we are paid $45.00 if he knows the material and nothing if he does not know it.

Assume that we can model his learning process with two states of learning:

\[
\begin{align*}
z = 1, & \quad \text{he knows the material} \\
z = 2, & \quad \text{he does not know the material}
\end{align*}
\]

At each step we also have two possible decisions:

\[
\begin{align*}
d = 1, & \quad \text{give him a lesson} \\
d = 2, & \quad \text{send him home}
\end{align*}
\]

If we give him a lesson when he knows the material he retains his knowledge, while if he does not know the material he learns it with probability 0.5. On the other hand, if we send him home when he knows the material he forgets it with probability 0.4, whereas if he does not know the material he will not learn it on his own.

From the reward structure, the expected immediate reward is $-10.00 if we give a lesson and $0.00 if we send him home. However, the terminal reward is $45.00 if he knows the material and $0.00 if he does not know it. The model is summarized in Table 1.

In this example Eq. (2.10) becomes

\[
\begin{align*}
v_1(n,1) &= -10 + v(n-1,1) \\
v_2(n,1) &= 0.6v(n-1,1) + 0.4v(n-1,2) \\
v_1(n,2) &= -10 + 0.5v(n-1,1) + 0.5v(n-1,2) \\
v_2(n,2) &= v(n-1,2)
\end{align*}
\]

(2.12)

The iterative computations are carried out in Table 2.
TABLE 1. DYNAMIC PROGRAMMING QUANTITIES FOR THE EXAMPLE

| Decision | \( p(z^{n-1}|z^n, d^n) \) as a Transition Matrix | \( r(1,1) = -10 \) | \( r(2,1) = -10 \) |
|----------|---------------------------------|----------------|----------------|
| \( d^n = 1 \) | \( z^{n-1} = 1 \) \( z^{n-1} = 2 \) | | |
| \( z^n = 1 \) | \[
\begin{bmatrix}
1 & 0 \\
0.5 & 0.5
\end{bmatrix}
\] | | |
| \( z^n = 2 \) | | | |
| | | | |
| \( d^n = 2 \) | \( z^{n-1} = 1 \) \( z^{n-1} = 2 \) | \( r(1,2) = 0 \) | \( r(2,2) = 0 \) |
| \( z^n = 1 \) | \[
\begin{bmatrix}
0.6 & 0.4 \\
0 & 1
\end{bmatrix}
\] | | |
| \( z^n = 2 \) | | | |

Spaces: \( D = \{1, 2\} \), \( Z = \{1, 2\} \).
Terminal conditions: \( v(0,1) = 45 \), \( v(0,2) = 0 \).

TABLE 2. DYNAMIC PROGRAMMING CALCULATIONS FOR THE EXAMPLE
WITH THE INITIAL REWARD STRUCTURE

<table>
<thead>
<tr>
<th>( n )</th>
<th>( z^n )</th>
<th>( d^n )</th>
<th>( v_{dn}(n, z^n) )</th>
<th>( v(n, z^n) )</th>
<th>( d(n, z^n) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-10 + 45 = 35</td>
<td>35</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0.6(45) + 0.4(0) = 27</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>-10 + 0.5(45) + 0.5(0) = 12.5</td>
<td>12.5</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>-10 + 35 = 25</td>
<td>26</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>2</td>
<td>0.6(35) + 0.4(12.5) = 26</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>-10 + 0.5(35) + 0.5(12.5) = 13.75</td>
<td>13.75</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>12.5</td>
<td>12.5</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>-10 + 26 = 16</td>
<td>21.1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
<td>0.6(26) + 0.4(13.75) = 21.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>-10 + 0.5(26) + 0.5(13.75) = 9.875</td>
<td>13.75</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>2</td>
<td>13.75</td>
<td>13.75</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>-10 + 21.1 = 11.1</td>
<td>18.16</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>0.6(21.1) + 0.4(13.75) = 18.16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>1</td>
<td>-10 + 0.5(21.1) + 0.5(13.75) = 7.425</td>
<td>13.75</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>2</td>
<td>13.75</td>
<td>13.75</td>
<td></td>
</tr>
</tbody>
</table>
The results are that when the student arrives on the first day (4 days to go) we send him home until the third day (2 days to go). When he arrives with 2 days remaining we ask him questions to determine his state. If he knows the material we send him home without giving him a lesson. However, if he does not know the material we give him a lesson. When he arrives on the last day we give him a lesson independent of his state. Given that with \( n \) days remaining he arrives in state \( z^n \), we can read off our expected total income for the remainder of the process from the \( v(n,z^n) \) column of the table.

D. INTUITIVE SUBOPTIMAL POLICIES

The dynamic programming formulation may always be modified so that the terminal rewards are zero. This modification is accomplished by adding the terminal reward, \( \phi(z^n) \), to the expected immediate reward, \( r(z^n,d^n) \), for every transition into state \( z^n \), and by subtracting \( \phi(z^n) \) from the expected immediate reward for every transition out of state \( z^n \). Thus a new expected immediate reward, \( r'(z^n,d^n) \), is formed as

\[
r'(z^n,d^n) = r(z^n,d^n) - \phi(z^n) + \sum_{z^{n-1} \in Z} \phi(z^{n-1}) p(z^{n-1} | z^n,d^n) \]

Conceptually, in the new process the terminal reward is collected ahead of time, when the process enters a given state. However, when the process leaves that state, the terminal reward for that state has to be given back and the terminal reward re-collected for the state entered. At the end of the process, this collecting and giving back of terminal rewards balances out to the proper terminal reward. At any point in the new process the terminal reward for the current state has already been collected. Thus if \( v'(n,z^n) \) is the expected total reward function for the new process, it is related to the old function, \( v(n,z^n) \), by

\[
v(n,z^n) = v'(n,z^n) - \phi(z^n) \]
Since the total reward received during the whole process is the same in either case, the difference being only the conceptual time when the rewards are collected, an optimal decision policy for one reward structure is optimal for the other reward structure.

For the previous example, the expected immediate reward for a new equivalent process having no terminal reward is given by:

\[
\begin{align*}
    r'(1,1) &= -10 - 45 + 1.0(45) + 0.0(0) = -10 \\
    r'(2,1) &= -10 - 0 + 0.5(45) + 0.5(0) = 12.5 \\
    r'(1,2) &= 0 - 45 + 0.6(45) + 0.4(0) = -18 \\
    r'(2,2) &= 0 - 0 + 0.0(45) + 1.0(0) = 0
\end{align*}
\]

The optimization, carried out for the new reward structure in Table 3, is seen to yield the same optimal policy as that for the original reward structure in Table 2.

**TABLE 3. DYNAMIC PROGRAMMING CALCULATIONS FOR THE EXAMPLE WITH THE MODIFIED REWARD STRUCTURE**

<table>
<thead>
<tr>
<th>n</th>
<th>z^n</th>
<th>d^n</th>
<th>( v'_{dn}(n,z^n) )</th>
<th>( v'(n,z^n) )</th>
<th>d(n,z^n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-10 + 0 = -10</td>
<td>-10</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>-18 + 0 = -18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>12.5 + 0 = 12.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0 + 0 = 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>-10 + (-10) = -20</td>
<td></td>
<td>-19</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>2</td>
<td>-18 + 0.6(-10) + 0.4(12.5) = -19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>12.5 + 0.5(-10) + 0.4(12.5) = 13.75</td>
<td>13.75</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0 + 12.5 = 12.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>-10 + (-19) = -29</td>
<td></td>
<td>-23.9</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
<td>-18 + 0.6(-19) + 0.4(13.75) = -23.9</td>
<td>13.75</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>12.5 + 0.5(-19) + 0.4(13.75) = 9.875</td>
<td>13.75</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>2</td>
<td>0 + 13.75 = 13.75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>-10 + (-23.9) = -33.9</td>
<td></td>
<td>-26.84</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>-18 + 0.6(-23.9) + 0.4(13.75) = -26.84</td>
<td>13.75</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>1</td>
<td>12.5 + 0.5(-23.9) + 0.4(13.75) = 7.425</td>
<td>13.75</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>2</td>
<td>0 + 13.75 = 13.75</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
After the terminal reward is adjusted to zero, the quantity $v'(j, z^n)$ may be thought of as the optimized expected total reward, in the new reward structure, to be collected in the next $j$ steps of the process when there are $n$ steps remaining in the process, $j$ necessarily being no greater than $n$. As $j$ is made larger the optimization is carried out farther into the future, until finally, when $j$ equals $n$, the optimization is carried out all the way to the end of the process.

An interesting suboptimal policy is to optimize only $j$ steps into the future, conceptually using the new reward structure. Thus a $j^{th}$ order suboptimal policy, denoted $d_j$, is

$$d_j(n, z^n) = \begin{cases} 
\hat{d}(n, z^n) & \text{for } n = 1, 2, \ldots, j \\
\hat{d}(j, z^n) & \text{for } n = j+1, j+2, \ldots
\end{cases}$$

In cases where there are a large number of states, such a policy may be justified as avoiding the cost of computing the optimal policy for all $n$. Also, such a policy may be used to compute the optimum decision in real time, evaluating $j$ steps ahead over the restricted number of states obtainable in $j$ steps starting from $z^n$, rather than computing the optimal decision policy, $\hat{d}(n, z^n)$, for all possible states, before the process is started.
III. MOTIVATING EXAMPLES FOR UNOBSERVABLE STATES

The dynamic programming formulation of the preceding chapter depends critically on the fact that the state $z$ of the Markov process is directly observable, so that a decision can be made based on that state without regard to the past history of the process. This chapter contains an example of a dynamic programming formulation of a teaching-machine problem with a state that is not directly observable. This situation will be treated with more generality in Chapter IV.

A. THE PROBLEM

Consider a situation where we want to teach a student a list of paired associates. Each item pair forms a frame of the teaching machine. Each pair consists of a stimulus item and a response item. When a frame is presented to the student the following events occur:

1. The stimulus item is presented (shown) to the student and he is asked to make a (correct) response.
2. The student makes a response which is recorded as correct or incorrect.
3. The correct response item is shown to the student to reinforce his knowledge of the item pair.

When no confusion arises the word "item" is used in place of item pair, stimulus item, or response item. Assume that there is a total of $M$ item pairs to be taught, and that there is a total of $N$ steps available in which to teach them. Any particular item may be presented more than once, and it need not be presented at all. The objective is to select the frame to be presented at each step, based on the past history of frames selected and responses made, so that the student learns as many items as possible in the $N$ steps.

B. THE MODEL

The learning process for each item is modeled by a one-element model. Thus item $i$ has a state which is either,
1. $x_i = 0$, the item is conditioned (has been learned), or
2. $x_i = 1$, the item is unconditioned (has not been learned).

The vector $\mathbf{x} = \left(x_1^n, x_2^n, \ldots, x_M^n\right)$ represents the state of the subject's knowledge when there are $n$ steps remaining in the process. When a student makes his response to item $i$, he is assumed to make a correct response if $x_i^n = 0$ (conditioned) and an incorrect response if $x_i^n = 1$ (unconditioned). The learning of any item is assumed to be independent of the state of learning of any other item. Thus if item $i$ is reinforced, when $x_i^n = 0$, then $x_i^{n-1} = 0$ with probability one. If item $i$ is reinforced when $x_i^n = 1$, then $x_i^{n-1} = 0$ with probability $c_i$ and $x_i^{n-1} = 1$ with probability $1 - c_i$. The response probabilities and the transition matrix that apply when item $i$ is presented are given below:

\[
\begin{array}{c|cc}
\text{New state} & x_i^{n-1} = 0 & x_i^{n-1} = 1 \\
\hline
\text{Probability of} & 1 & 0 & \frac{1}{3}
\end{array}
\]

When item $i$ is presented the states of all other items remain unchanged. Initially, all of the items are in the unconditioned state, i.e., $x_i^{(N)} = 1$, $i = 1, 2, \ldots, M$.

C. THE REWARD STRUCTURE

The objective of the decision process is to teach the student as many items as possible during the $N$ frames. To make this objective precise, the optimizer receives one unit of reward when the state of an item makes a transition from the unconditioned ($x_i^n = 1$) to the conditioned ($x_i^{n-1} = 0$) state. Thus, the appropriate immediate reward function is

\[
R(x_i^{n-1}, x_i^n) = \sum_{i=1}^{M} (x_i^n - x_i^{n-1}) \quad (3.2)
\]
D. THE OBSERVABLE MARKOV PROCESS

If the state \( x^n \) were directly observable, the dynamic programming procedure presented in Chapter II would be directly applicable. However, since, after the student makes an incorrect response on an item, the state of that item may change from unconditioned to conditioned as a result of reinforcement, the state of the item is uncertain at the end of the step. That is, at the end of an item \( i \) presentation, during which the student makes an incorrect response, the probability of item \( i \) being in the conditioned state is \( c_i \), and the probability of item \( i \) being in the unconditioned state is \( 1 - c_i \). However if the student makes a correct response on item \( i \), at the end of that step he is in the conditioned state with probability one. If an item has not been presented, by assumption it is in the unconditioned state. Thus although the state of the item cannot be observed directly, there are only three possible states of the past observable history which are relevant to the determination of the unobservable state. Summarizing, the possible relevant states of the past history of item \( i \) are:

\[
\begin{align*}
  z_i^n &= 0, \quad \text{if item } i \text{ has not been presented} \\
  z_i^n &= 1, \quad \text{if item } i \text{ has been presented and received only errors} \\
  z_i^n &= 2, \quad \text{if item } i \text{ has been presented and received a correct response}
\end{align*}
\]

The probabilities of the various states of \( x_i^n \), given the state of history \( z_i^n \), are given in Table 4.

The state \( z_i^n \) is called a sufficient history (or a sufficient statistic of the history) for the state \( x_i^n \). The reason for this name is that the addition of any other past history to \( z_i^n \) does not change the probability distribution of \( x_i^n \).

Now the probability of any future event given the history can be decomposed into the summation, over all possible states of learning, of the product of the probability of the event given the history and given
the state of learning, with the probability of the learning state given the history. The latter probability can be computed from Table 4 and the former probability can be determined from the learning model since, given the state of learning, the past history is irrelevant by the Markov property. The state of the sufficient history $z^{n-1}$ at the end of the next step is a particular future event. By the previous remarks the probability distribution of $z^{n-1}_i$, given the entire past history with $n$ steps remaining, is a function of only $z^n_1$. Thus the transitions of the history $z^n_1$ form a Markov process. Using Table 4, the following transition matrix for an item 1 presentation can be easily constructed:

$$
\begin{bmatrix}
0 & 1 & 0 \\
0 & 1-c_i & c_i \\
0 & 0 & 1
\end{bmatrix}
$$

When item 1 is presented, clearly the states of history for all of the other items remain unchanged. Thus the transition matrix for the
state \( z^n = (z^n_1, z^n_2, \ldots, z^n_m) \) of the entire process is a well-defined function of the item chosen for presentation.

E. EXPECTED IMMEDIATE REWARD

From the previously described reward structure, one unit of reward is received only if an item is in the unconditioned state and then makes the transition to the conditioned state. The probability of this event occurring, given the state of history \( z^n_1 \), is the probability that \( x^n_1 = 1 \) given \( z^n_1 \) (see Table 4) times the probability \( c_i \), that is, 
\[ c_i P(x^n_1 = 1 | z^n_1). \]
Thus the expected immediate reward, \( r(z^n, i) \), for making decision \( i \) when the history state is \( z^n \), is given by Table 5.

<table>
<thead>
<tr>
<th>( z^n_i )</th>
<th>( r(z^n, i) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( c_i )</td>
</tr>
<tr>
<td>1</td>
<td>( c_i (1-c_i) )</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

F. THE APPLICATION OF DYNAMIC PROGRAMMING

All of the quantities needed for the application of dynamic programming to this particular problem have now been given. Section D developed a new Markov process in the (observable) states of the sufficient history \( z^n \), from the unobservable Markov process in the learning states \( x^n \). Section E developed the expression for the expected immediate reward as a function of the state of history and the item selected for presentation. Thus the dynamic programming equations of Chapter II are now applicable. Repeating the equations here, the terminal reward (initial condition for the recursion) is

\[ v(0, z^0) = 0 \]
and the recursive equation for computation of the optimum policy is

\[ v(n, z^n) = \max_{d^n} \left[ r(z^n, d^n) + \sum_{z^{n-1} \in Z} v(n-1, z^{n-1}) p(z^{n-1} | z^n, d^n) \right] \]

where the optimum decision function is \( \hat{d}(n, z^n) \), equal to a \( d^n \) for which the above maximum occurs.

G. MAXIMIZATION OF EXPECTED IMMEDIATE REWARD

For this particular example it is possible to show that an optimum decision policy is to present the item which maximizes the expected immediate reward, without regard to the terms representing longer run rewards. This means that

\[ \hat{d}(n, z^n) = \hat{d}(1, z^n) \quad n = 1, 2, \ldots, N \]

so that only the first step in the recursive equations need be carried out. Thus the optimum policy can be represented by \( 3^M \) numbers corresponding to the optimum decision for each of the \( 3^M \) possible states of sufficient history.

If all of the \( c_i \) are equal to each other, then the following is a description of the optimum procedure to be used until the \( N \) frames are exhausted:

1. Present each item once, in any order.
2. Present any item which has not yet received a correct response. If possible, repeat this step until the end of the process. If all items receive correct responses before the \( N \) frames are exhausted, all items have been learned and nothing is to be gained by any more presentations.

From the terminal condition \( v(0, z^0) = 0 \), it is seen that

\[ v(1, z^1) = \max_{d^1} r(z^1, d^1) \]
Thus for \( n = 1 \) it is clear that maximizing expected immediate reward is an optimal policy. Assume now that maximizing expected immediate reward is an optimal policy for \( n = k \). Suppose that with \((k+1)\) steps remaining the sufficient history is in state \( z^{k+1} \). Suppose that presenting item \( i \) maximizes expected immediate reward. Then

\[
r(z^{k+1}, i) \geq r(z^{k+1}, \ell) \quad \ell = 1, 2, \ldots, M \quad (3.4)
\]

Suppose further that the presentation of item \( j \) is an optimum decision in this situation. If \( j = i \), then maximizing expected immediate reward is optimum. If \( j \neq i \), it will be shown that \( i \) must also be an optimum decision. If item \( j \) is presented, from the transition matrix (3.3) it is seen that

\[
\begin{align*}
z^{k} & \not\supseteq z^{k+1} \\
\text{Also from the expected immediate rewards (Table 5) it is seen that} & \\
r(z^{n}, j) & \text{is a monotonically decreasing function of } z^{n}_{j}. \text{ Thus, if} \\
\text{item } j \text{ is presented} & \\
r(z^{k+1}, j) & \geq r(z^{k}, j)
\end{align*}
\]

From (3.4)

\[
r(z^{k+1}, i) \leq r(z^{k+1}, j)
\]

and combining the last two equations gives

\[
r(z^{k+1}, i) \geq r(z^{k}, j) \quad (3.5)
\]

Now for all \( \ell \neq j \) the state remains unchanged; that is,

\[
\frac{z^{k+1}}{z^{k}_{\ell}} = z^{k}_{\ell} \quad \ell = 1, 2, \ldots, M; \quad \ell \neq j
\]
Thus from (3.4)

$$r(z^k, i) \equiv r(z^k, \ell) \quad \ell = 1, 2, \ldots, M; \quad \ell \neq j$$

and from (3.7)

$$r(z^k, i) \equiv r(z^k, j)$$

Combining the last two equations yields

$$r(z^k, i) \equiv r(z^k, \ell) \quad \ell = 1, 2, \ldots, M \quad (3.6)$$

Equation (3.6) shows that, independently of exactly which events occur upon the presentation of item \( j \), at the next step (\( k \) steps remaining) the presentation of item \( i \) maximizes expected immediate reward. By assumption, maximizing expected immediate reward is optimum with \( k \) steps remaining. Thus if item \( j \) is an optimum decision, then before item \( j \) is presented it is known that an optimum procedure is to present item \( j \) and, regardless of the response, present item \( i \) on the next successive step. Since the items are independent it must also be optimum to present item \( i \) and, regardless of the response, present item \( j \) on the next successive step. Thus, it has now been shown that if \( j \neq i \), then \( i \) must also be an optimum decision, so maximizing expected immediate reward is an optimal policy with \((k+1)\) steps remaining. This completes the proof by induction.
IV. DYNAMIC PROGRAMMING WITH UNOBSERVABLE STATES

The purpose of this chapter is to apply the ideas of Chapter III to a more general situation. A fairly general learning model is formulated, of which many models in the learning-theory literature are specific cases. For this general model an optimum teaching process is defined. After some restrictions are introduced, a dynamic programming solution is developed.

The learning model is essentially a Markov process with states which are not directly observable. It is shown that, for the purpose of optimization, a new Markov process with observable states can be inferred from the original model. Dynamic programming is then applied, using the new states, in a manner similar to that of Chapter II.

In the following chapters several specific examples are put into this framework.

A. THE TEACHING PROCESS

It is assumed that the learning model may be thought of as having a state \( x \) which changes from step to step as the learning process proceeds. All future events, in the learning process, are assumed to depend on past events through the state of the process. That is, given the exact state \( x \) at a certain time, all future events depend only on this state and are independent of past events.

At any given time, there is some recorded history of observable past events upon which decisions must be based. This history is denoted by \( h \).

The material to be taught is organized in frames, which are available for presentation at any step in the process. Each frame consists of some study material and/or some tests to which the student gives some observable response denoted by \( a \). The choice of the frame selected for presentation at each step in the process is the variable used to optimize or control the learning process.

At the beginning of the process, an optimum first frame is picked and presented to the student. The presentation of this frame causes a change in the state \( x \) of the student's learning model, and causes an
observable response. The frame presented at the initial step and the
observable response of the frame are used to select an optimum frame for
the second step. The second frame is then presented to the student.
This procedure is repeated until the end of the teaching process. This
work treats the case of a fixed total number of steps, N. Similar
mathematical techniques apply in other situations.

The following list summarizes the procedure from the conclusion of
one step to the conclusion of the following step:

1. At the conclusion of the previous step, the process was left in
   (unobservable) state $x$, and the updated history was $h$.
2. Decision $d$ is made, based only on information contained in the
   history $h$.
3. The frame indicated by decision $d$ is presented to the student.
   During this presentation the state of learning changes from $x$
   to $x'$ and some response $a$ is observed.
4. The recorded history $h$ is updated by the decision $d$ and the
   response $a$ to form the new history, $h'$.
5. At the conclusion of the step, the learning process is left in
   state $x'$, and the updated history is $h'$.

B. THE LEARNING MODEL

The learning model describes the transition from $x$ to $x'$ and the
observable response $a$ of step 3 above, depending on the decision $d$.
Let $X$ be the set of all possible states of the process, let $A$ be the
set of all possible responses, and let $D$ be the set of all possible
decisions. In this work it is assumed that the set $D$ has a finite
number of elements. For the sets $X$ and $A$, notation for a finite
number of elements is used for simplicity. A Markov learning model is
deﬁned as the joint probability distribution of the possible responses
$a \in A$ and the possible new states $x' \in X$ which may occur during the
next step, given the former state $x \in X$ and the decision $d$ used on
the step. The discrete probability function is denoted $P(x',a|x,d)$.
The fundamental assumption about the learning model is the Markov property
of the state $x$. That is,

$$P(x',a|x,d, \text{ any past events}) = P(x',a|x,d)$$
Typical components of the state \( x \) may be the Markov state of the one-element model, possible values of response probabilities for the linear model, or the values of unknown parameters in either type of model. In general the state is regarded as changing from step to step; however, unknown parameters characterizing the learning model would be fixed throughout the process.

C. HISTORY SPACE

Let \( H \) denote the set of all possible histories \( h \) that may occur. This set is assumed finite. The history \( h \) is only that history which has been recorded for the purpose of making decisions, and it is not necessarily a complete history of all past observable events in the process. The new history \( h' \) is obtained by updating the former history \( h \), using the decision \( d \) and the observed response \( a \). Thus \( h' \) is determined by a mapping from \( a \in A, d \in D, \) and \( h \in H \) into \( h' \in H \).

It should be clear that the learning model \( P(x',a|x,d) \), along with the above mapping, generates a probability distribution \( P(x',h'|x,h,d) \) which is independent of past histories, responses, states, and decisions; i.e.,

\[
P(x',h'|x,h,d, \text{any past events}) = P(x',h'|x,h,d)
\]

D. MEASURES OF EFFECTIVENESS

The effectiveness of the teaching process is measured in terms of units of reward for various events occurring during the teaching process. The reward is divided into additive parts, terminal reward paid to the optimizer at the end of the process, and immediate reward paid during each step in the process.

Terminal reward is some real function of the final state \( x^0 \), denoted by \( \psi(x^0) \). This function may be given directly in the formulation of the problem, or it may be derived as the expected value of a random reward based on the final state of the process, \( x^0 \), and independent of past events. For example, this reward may be the expected score on a test administered after the process, where the expectation is conditioned on the final state of learning, \( x^0 \).
Immediate reward is paid to the optimizer for events which occur during each step. The immediate reward is a real function of the decision d, the initial state x on the step, the final state x’ on the step, and the observable response a. The immediate reward is denoted R(x’,x,a,d). Again, other random events which are independent of the past can be taken into account in this function.

E. OPTIMIZATION

An optimum decision \( \hat{d} \), given history h, is a decision d which maximizes the expected total reward to be collected in the remainder of the process given h and d, under the condition that optimum decisions will be used for the remainder of the process. This criterion of optimality is essentially the same as that of the ordinary dynamic programming in Chapter II.

1. Notation

The various quantities are indexed by the number of steps remaining until the end of the process. If N is the total number of steps in the process, then the first step is indexed by N, the second by N-1, and the last step by 1. The quantities between steps, at the time decisions are being made, are indexed with the same index as the following frame. To clarify the notation, the procedure followed in selecting and presenting the frame for the \( n \)th step from the end of the process is given below:

1. At the conclusion of the previous step, the process was left in state \( x^n \), and the updated history was \( h^n \).
2. Decision \( d^n \) is made, based only on information contained in the history \( h^n \).
3. The frame indicated by decision \( d^n \) is presented to the student. During this presentation the state of learning changes from \( x^n \) to \( x^{n-1} \) and a response \( a^n \) is observed.
4. The recorded history \( h^n \) is updated by the decision \( d^n \) and the response \( a^n \) to form the new history, \( h^{n-1} \).
5. At the conclusion of the step, the learning process is left in state \( x^{n-1} \) and the updated history is \( h^{n-1} \).
2. Solution by Enumeration

As in Chapter II, the solution of this problem can be formulated in terms of a direct enumeration process. Let \( H \) be the set of all possible past histories of decisions and outcomes. A decision policy \( d(n,h^n) \) is a mapping from the product space \( \{1, 2, ..., N\} \times H \) into the decision space \( D \), which indicates that decision \( d(n,h^n) \) is to be made if \( h^n \) is the history with \( n \) frames remaining in the process. Given such a mapping \( d \), the expected total reward \( f(d) \) for the whole process using policy \( d \) is

\[
f(d) = \mathbb{E} \left[ \sum_{n=N}^{1} R(x^{n-1}, x^n, a^n, d^n) + \psi(x^0) \mid d \right]
\]

Thus \( d \) is an optimal policy if and only if

\[
f(\hat{d}) \geq f(d)
\]

for all possible decision policies \( d \). The trouble with using direct enumeration to find the optimal policy is that there are usually very large numbers of possible decision policies.

F. DYNAMIC PROGRAMMING SOLUTION

The dynamic programming solution is approached by first restricting the class of possible recorded histories to a subclass defined as Markov histories. This restriction causes the evolution of successive recorded histories to be a Markov process. The dynamic programming solution is then developed in a manner similar to that of Chapter II. Then, sufficient histories are defined as statistics summarizing the recorded histories. The existence of sufficient histories with a relatively small number of possible values is necessary in order for dynamic programming to offer a great reduction in the number of computations required by direct enumeration. The dimensionality of the sufficient history is a property of the learning model being used.
1. Markov Histories

A history $h^n$ is called a Markov history if the conditional probability of the learning state $x^n$, given $h^n$, is independent of all past histories and all past decisions; that is,

$$P(x^n|h^n, h^{n+1}, \ldots, h^N, d^{n+1}, d^{n+2}, \ldots, d^N) = P(x^n|h^n)$$

If $h^n$ is a Markov history, it contains all of the information about the unknown state $x^n$ that is contained in all of the past histories and all of the past decisions. Note that $h^n$ does not necessarily contain all of the information in the past responses but only that portion contained in the recorded histories, the recording process being selected freely. In an extreme example, the responses may be completely disregarded, so that the recorded history is only the history of past decisions. This is the static case which is described in more detail in Chapter VII.

A Markov history always exists. For example, in the static case, if $h^n$ is the 2-tuple $h^n = (d^{n+1}, h^{n+1})$, then clearly $h^n = (d^{n+1}, d^{n+2}, \ldots, d^N)$ and hence is a Markov history. For a second example, if $h^n$ is the 3-tuple $h^n = (a^{n+1}, d^{n+1}, h^{n+1})$, then $h^n = (a^{n+1}, a^{n+2}, d^{n+2}, \ldots, a^N, d^N)$. Since in the first example the history contains all past decisions and no past responses, all of the decisions may be made before the process is started. In this case an optimum sequence of frames may be put into a linear-program-type teaching-machine book. In the second example the history contains all past responses. Thus the actual decisions must be made dynamically as the student progresses through the process. In this case a branching-type teaching-machine book or some sort of automated teaching machine is required to interact with the student and make the optimum decisions at each step in the process.

2. Properties of Markov Histories

Three key concepts used in the development of the various formulas and properties are:
1. The Markov property of the learning model

2. The Markov property of the history

3. Bayes' formula in the form $P(A|B,C) = \frac{P(A,B|C)}{P(B|C)}$

For a Markov history,

\[
P(x^n, h^n, x^{n+1}|h^{n+1}, h^{n+2}, \ldots, h^N, d^{n+1}, d^{n+2}, \ldots, d^N)
\]
\[
= P(x^n, h^n|x^{n+1}, h^{n+1}, h^{n+2}, \ldots, h^N, d^{n+1}, d^{n+2}, \ldots, d^N)
\]
\[
\times P(x^{n+1}|h^{n+1}, h^{n+2}, \ldots, h^N, d^{n+1}, d^{n+2}, \ldots, d^N)
\]
\[
= P(x^n, h^n|x^{n+1}, h^{n+1}, d^{n+1})P(x^{n+1}|h^{n+1})
\]

Thus, since

\[
P(x^n|h^n) = P(x^n|h^n, x^{n+1}, \ldots, h^N, d^{n+1}, d^{n+2}, \ldots, d^N)
\]
\[
= \frac{P(x^n, h^n|h^{n+1}, \ldots, h^N, d^{n+1}, d^{n+2}, \ldots, d^N)}{P(h^n|h^{n+1}, \ldots, h^N, d^{n+1}, d^{n+2}, \ldots, d^N)}
\]

a recursive formula for $P(x^n|h^n)$ in terms of the learning model is

\[
P(x^n|h^n) = \sum_{x^{n+1}} \frac{P(x^n, h^n|x^{n+1}, h^{n+1}, d^{n+1})P(x^{n+1}|h^{n+1})}{\sum_{x^n} \sum_{x^{n+1}} P(x^n, h^n|x^{n+1}, d^{n+1})P(x^{n+1}|h^{n+1} )}
\]
\[
(4.1)
\]

where $P(x^n|h^n)$ represents the initial distribution of $x^N$.

Also if $h^n$ is a Markov history, then, remembering that $d^n$ is just a function of $h^n$,
\[ P(h^{n-1} | h^n, h^{n+1}, \ldots, h^N, d^n, d^{n+1}, \ldots, d^N) \]

\[ = \sum_{x^{n-1}} \sum_{x^n} P(h^{n-1}, x^{n-1}, x^n | h^n, h^{n+1}, \ldots, h^N, d^n, d^{n+1}, \ldots, d^N) \]

\[ = \sum_{x^{n-1}} \sum_{x^n} P(h^{n-1}, x^{n-1} | x^n, h^n, h^{n+1}, \ldots, h^N, d^n, d^{n+1}, \ldots, d^N) \cdot P(x^n | h^n, h^{n+1}, \ldots, h^N, d^n, d^{n+1}, \ldots, d^N) \]

\[ = \sum_{x^{n-1}} \sum_{x^n} P(h^{n-1}, x^{n-1} | x^n, h^n, d^n) P(x^n | h^n, d^n) = P(h^{n-1} | h^n, d^n) \]

\[(4.2)\]

The preceding calculation shows that the elements of the set, \( H \), of possible histories form the states of a Markov process with transition probabilities determined by the current decision. This is the reason for the name Markov history.

If \( h^n \) is a Markov history, the expected immediate reward under decision \( d^n \) is

\[ E[R(x^{n-1}, x^n, a^n, d^n) | h^n, h^{n+1}, \ldots, h^N, d^n, d^{n+1}, \ldots, d^N] \]

\[ = \sum_{x^{n-1}} \sum_{x^n} \sum_{a^n} R(x^{n-1}, x^n, a^n, d^n) P(x^{n-1}, x^n, a^n | h^n, h^{n+1}, \ldots, h^N, d^n, d^{n+1}, \ldots, d^N) \]

\[(4.3)\]
Now

\[ P(x^{n-1}, x^n, a^n | h^n, h^{n+1}, \ldots, h^N, d^n, d^{n+1}, \ldots, d^N) \]

\[ = P(x^{n-1}, a^n | x^n, h^n, h^{n+1}, \ldots, h^N, d^n, d^{n+1}, \ldots, d^N) \]

\[ \cdot P(x^n | h^n, h^{n+1}, \ldots, h^N, d^n, d^{n+1}, \ldots, d^N) \]

\[ = P(x^{n-1}, a^n | x^n, h^n, d^n) P(x^n | h^n, d^n) = P(x^{n-1}, x^n, a^n | h^n, d^n) \]

Thus, using (4.3) and (4.4),

\[ E[R(x^{n-1}, x^n, a^n, d^n) | h^n, h^{n+1}, \ldots, h^N, d^n, d^{n+1}, \ldots, d^N] \]

\[ = \sum_{x^{n-1}} \sum_{x^n} \sum_{a^n} R(x^{n-1}, x^n, a^n, d^n) P(x^{n-1}, x^n, a^n, h^n, d^n) \]

\[ = E[R(x^{n-1}, x^n, a^n, d^n) | h^n, d^n] = r(h^n, d^n) \tag{4.5} \]

The function \( r(h^n, d^n) \) is referred to as the expected immediate reward for making decision \( d^n \) when the history is \( h^n \). From Eq. (4.5) it is seen that this quantity is independent of the path of Markov histories.

3. Recursive Equations

In deriving the dynamic programming recursive equations, one starts at the end of the process and works back to the beginning. In this formulation, the assumption that \( h^n \) is a Markov history is critical. This development is similar to that of Chapter II.

First, consider the end of the process with history \( h^0 \) and state \( x^0 \). Let \( v(0, h^0) \) be the expected value of the terminal reward, given the history \( h^0 \) at the end of the process. Then

\[ v(0, h^0) = E[\lambda(x^0) | h^0] = \sum_{x} \lambda(x^0) P(x^0 | h^0) \tag{4.6} \]
Since $h_0^n$ is a Markov history, the above expected value is independent of all past histories and all past decisions.

Now suppose there are $n$ steps remaining with history $h^n$. Let $v_{d^n}(n,h^n)$ be the expected value of the total of the immediate and terminal rewards remaining in the process, given $h^n$, if decision $d^n$ is made with $n$ steps remaining, and an optimum decision process is used thereafter. Let $v(n,h^n)$ be the expected value of the total of the immediate and terminal rewards remaining in the process if optimum decisions are used for the current decision and all decisions thereafter. Then clearly

$$v(n,h^n) = \max_{d^n} v_{d^n}(n,h^n) \quad (4.7)$$

Again the use of a Markov history assures that these quantities are independent of past histories and past decisions.

Now, using the Markov property of the history as in Chapter II,

$$v_{d^n}(n,h^n) = r(h^n,d^n) + E[v(n-1,h^{n-1})|h^n,d^n]$$

or

$$v_{d^n}(n,h^n) = r(h^n,d^n) + \sum_{h^{n-1}} v(n-1,h^{n-1})p(h^{n-1}|h^n,d^n)$$

Thus $v(n,h^n)$ must satisfy the recurrence relation

$$v(n,h^n) = \max_{d^n} \left[ r(h^n,d^n) + \sum_{h^{n-1}} v(n-1,h^{n-1})p(h^{n-1}|h^n,d^n) \right] \quad (4.8)$$

and the optimal policy $\hat{d}(n,h^n)$ is found as a $d^n$ for which the above maximum occurs.
G. SUFFICIENT HISTORIES

If $z^n$ is some function of $h^n$, usually a simplification, $z^n$ is called a sufficient history if the conditional probability of $x^n$, given $z^n$, is independent of $h^n$ and of all past histories and all past decisions; that is,

$$p(x^n|z^n, h^n, h^{n+1}, \ldots, h^N, d^{n+1}, \ldots, d^N) = p(x^n|z^n)$$

Clearly if $h^n$ is a Markov history, then $z^n$ is also a Markov history. Also the same optimum decisions will be made based on $z^n$ instead of $h^n$. Thus, the use of a sufficient history allows a simplification of the recorded history without changing the optimization.

If sufficient histories exist, with a relatively small number of possible values or states of history, then dynamic programming offers a reduction of the number of computations required by direct enumeration. In the example of Chapter III, the past history was reduced to three states of a sufficient Markov history for each item. For a process with $M$ items there were $3^M$ states of sufficient history for the learning process. The situation in later chapters, where different learning models are used, is not so simple. In these cases, the dynamic programming formulation is better than direct enumeration, but a large number of states of sufficient history are still required. The implication of these problems for the modeling problem is discussed in the final chapter.

In the application of this dynamic programming formulation, simplifying sufficient Markov histories are usually found at the outset, and all of the calculations are done with the sufficient history directly. Since a sufficient Markov history is a case of a Markov history, all of the equations developed in this chapter are valid with $h^n$ replaced by $z^n$. 

- 35 -
V. ONE-ELEMENT LEARNING MODEL WITH FIXED PARAMETERS

This chapter considers an item allocation problem similar to that of the example in Chapter III. The objective is to teach a student a list of M items. At each step of the process, a block of K different items is presented. Thus, necessarily $M \geq K$. There is a total of N steps in the teaching process.

A. THE LEARNING MODEL

The learning of each item is assumed to follow a one-element learning model with conditioning probability $c_i$, $i = 1, 2, ..., M$, and guessing probability $p_i$, $i = 1, 2, ..., M$. The learning of each item occurs only when the item is presented and is independent of the learning of any other item. In the one-element model, each item is assumed to have an underlying Markov state of conditioning $x^n_i$. If $x^n_i = 1$, item $i$ is in the unconditioned state, and if $x^n_i = 0$, item $i$ is in the conditioned state. If item $i$ is presented when $x^n_i = 1$ (unconditioned), the subject makes the correct response with guessing probability $p_i$; whereas if $x^n_i = 0$ (conditioned), the subject always makes the correct response. On each presentation of an item, the observable response is denoted as correct or incorrect.

After the item is presented and the student makes a response, the student is given a reinforcement for the correct response. For example, the reinforcement may consist of simply showing him the correct response. This reinforcement is modeled by the application of a Markov transition to the state of the learning model. If item $i$ is presented when $x^n_i = 1$ (unconditioned), after the response the state makes a Markov transition to state $x^{n-1}_i = 0$ (conditioned) with conditioning probability $c_i$, and to state $x^{n-1}_i = 0$ (unconditioned) with probability $1-c_i$. On the other hand, if $x^n_i = 0$ (conditioned), the transition is always to $x^{n-1}_i = 0$, and hence the conditioned state is a trapping state. The response probability and transition matrix that apply when item $i$ is presented is given below:

- 36 -
When item \( i \) is presented, the states of all other items remain unchanged. At the beginning of the process, each item is assumed to be in the unconditioned state.

Before each step is presented, a decision \( d^n \) is made, completely determining the items to be presented on the next step, based on the complete observable history \( h^n \). Note that \( h^n \) is a Markov history. Decision \( d^n \) is represented as a \( K \)-tuple, \( d^n = (d^n_1, d^n_2, \ldots, d^n_K) \) where \( d^n_k \) is the item number of the \( k \)th item to be presented in the next frame, \( k = 1, 2, \ldots, K \).

The following list summarizes the procedure from the conclusion of one step to the conclusion of the following step:

1. At the conclusion of the previous step, the learning process was left in state \( x^n \), and the updated history was \( h^n \).
2. Decision \( d^n = (d^n_1, d^n_2, \ldots, d^n_K) \) is made based only on information contained in the history \( h^n \); that is, items \( d^n_1, d^n_2, \ldots, d^n_K \) are picked for presentation.
3. The frame indicated by decision \( d^n \) is presented to the student. During this presentation the state of learning changes from \( x^n \) to \( x^{n-1} \) and a response \( a^n \), consisting of the responses to the \( K \) items, is observed.
4. The recorded history \( h^n \) is updated by the decision \( d^n \) and the response \( a^n \) to form the new history \( h^{n-1} \).
5. At the conclusion of the step, the process is left in state \( x^{n-1} \) and the recorded history is \( h^{n-1} \).

B: THE SUFFICIENT HISTORY

When the student makes a correct response on an item, the state of his learning is in either the conditioned or the unconditioned state.
However, when the student makes an incorrect response, his state of learning must be in the unconditioned state. Thus, if on a particular item the student has a series of correct and incorrect responses, by the Markov property of the state of learning the history previous to the last incorrect response has no effect on the probability of future events and hence it need not be retained. Thus the number of correct responses since the most recent error is a sufficient history.

There is one complicating factor, however. After an incorrect response on item $i$, the student receives a reinforcement for item $i$ before the end of the step. Thus if an incorrect response is made, at the end of the step item $i$ is conditioned with probability $c_i$ and unconditioned with probability $1-c_i$. However, if item $i$ has never been presented, it is unconditioned with probability 1. On first presentation of the item, the response contains no information, and at the end of the presentation the item is conditioned with probability $c_i$ and unconditioned with probability $1-c_i$. Thus the situation after one correct response on the first presentation of an item is the same as after any error on an item. Similarly, the situation after $n$ correct responses on the first $n$ presentations of an item is the same as $(n-1)$ correct responses following an error on that item. These considerations show that the sufficient history covering both cases is the number of item $i$ reinforcements which have been given since item $i$ was known to be in the unconditioned state.

Let $z_i^n$ equal the number of item $i$ reinforcements since the last known unconditioned state. That is, if item $i$ has received only correct responses, then the last known unconditioned state was at the beginning of the process, and hence $z_i^n$ is the number of times item $i$ has been presented. Note that in this case on the first response item $i$ was known to be in the unconditioned state, so the first response contains no information about the current state of item $i$, and only the last $z_i^n$ (correct) responses contain state information. On the other hand, if item $i$ has ever received an incorrect response, then the state was unconditioned at the time of the last error. Thus, if $y_i^n$ is the number of correct responses since the last error, there have been
the last known unconditioned state, and the last \( y_i^n \) (correct) responses were made from unknown states. In this case \( z_i^n = y_i^n + 1 \) and again only the last \( (z_i^n - 1) \) responses contain state information. The previous remarks should have made it clear that \( z_i^n \) is a sufficient history for item \( i \), and that \( z^n = (z_1^n, z_2^n, ..., z_M^n) \) is a sufficient history for the whole process.

1. Transition Probabilities

The following is a computation of the transition probabilities for the sufficient history \( z_i^n \) if item \( i \) is presented. Note that \( z_i^n \) means \( z_i^n \) reinforcements, starting from the unconditioned state, denoted here by "z R's" with a correct response after each reinforcement but the last, denoted here by "z-l C's." Let \( x^n \) denote \( x_i^n \), the state of item \( i \) with \( n \) steps remaining in the process. The subscript \( i \) and the superscript \( n \) are dropped for these computations.

Bayes' formula yields

\[
P(x = 1|z) = P(x = 1|z \ R's, \ z-l \ C's)
\]

\[
= \frac{P(z-l \ C's|z = 1, \ z \ R's)P(x = 1|z \ R's)}{P(z-l \ C's|z = 1, \ z \ R's)P(x = 1|z \ R's) + P(z-l \ C's|z = 0, \ z \ R's)P(x = 0|z \ R's)}
\]

Now

\[P(x = 1|z \ R's) = (1-c)^Z\]

\[P(x = 0|z \ R's) = 1 - P(x = 1|z \ R's) = 1 - (1-c)^Z\]

\[P(z-l \ C's|z = 1, \ z \ R's) = p^{Z-l}\]

\[P(z-l \ C's|z = 0, \ z \ R's) P(x = 0|z \ R's) = P(z-l \ C's, x = 0|z \ R's)\]

\[= c + (1-c)pc + (1-c)^2p^2c + ..., + (1-c)^{Z-l}p^{Z-l-1}c\]

\[= c \sum_{i=0}^{Z-l} \alpha^i = \frac{c(1-\alpha^Z)}{1-\alpha}\]

- 39 -
where $\alpha = (1-c)p$. Thus,

$$P(x = 1 \mid z) = \frac{p^{z-1}(1-c)^z}{p^{z-1}(1-c)^z + \frac{c(1 - \alpha)}{1 - \alpha}} - \frac{p^z(1-c)^z}{p^z(1-c)^z + \frac{pc(1 - \alpha)}{1 - \alpha}}$$

$$= \frac{\alpha^z(1 - \alpha)}{\alpha^z(1 - \alpha) + pc(1 - \alpha^z)} = \frac{\alpha^z(1 - \alpha)}{p - \alpha + (1-p)\alpha^z} \quad (5.1)$$

and

$$P(x = 0 \mid z) = 1 - P(x = 1 \mid z) \quad (5.2)$$

Now if $x = 0$ (conditioned), then on the next trial $z$ goes to $z' = z + 1$; and if $x = 1$ (unconditioned), then on the next trial $z$ goes to $z' = z + 1$ with guessing probability $p$ and to $z' = 1$ with probability $1-p$. These are the only possibilities, and hence the Markov transition probabilities for the sufficient history $z_i^n$ when $i$ is presented, $P(z_i^{n-1} \mid z_i^n, i \text{ presented})$, are given by Table 6.

**TABLE 6. MARKOV TRANSITION PROBABILITIES FOR THE SUFFICIENT HISTORY $z_i^n$ WHEN ITEM $i$ IS PRESENTED**

<table>
<thead>
<tr>
<th>$z_i^{n-1}$</th>
<th>$P(z_i^{n-1} \mid z_i^n, i \text{ presented})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$</td>
<td>$(1-p_i)P(x_i^n = 1 \mid z_i^n)$</td>
</tr>
<tr>
<td>$z_i^{n+1}$</td>
<td>$1 - (1-p_i)P(x_i^n = 1 \mid z_i^n)$</td>
</tr>
<tr>
<td>Otherwise</td>
<td>0</td>
</tr>
</tbody>
</table>

The components of the sufficient history $z_i^n$ for those items not presented clearly remain unchanged. Thus appropriate products of the above transition probabilities under decision $d_i^n$ form the transition probability distribution for the entire state, $P(z_i^{n-1} \mid z_i^n, d_i^n)$. 

- 40 -
2. **Expected Immediate Reward**

The objective of this process is to condition as many items as possible in a total of \( N \) steps, each step containing \( K \) different item presentations. For a slight increase in generality, assume that reward \( w_i \) is received if item \( i \) is conditioned at the end of the process.

One natural way to set up the reward structure is to use the terminal reward function

\[
\psi(x^0) = \sum_{i=1}^{M} w_i (1-x_i)
\]

If \( w_i = 1 \) for each \( i \), then the expected total number of items in the conditioned state at the end of the process is the quantity being maximized. In some situations it might be desirable, instead, to maximize the expected score on a test administered at the conclusion of the teaching process, the test consisting of one presentation of each item. The expected test score, given the final state \( x^0 \), is

\[
\sum_{i=1}^{M} (1-x_i) + p_i x_i = \sum_{i=1}^{M} (1-p_i) (1-x_i) + \sum_{i=1}^{M} p_i
\]

Since the last term on the right is just a constant, the choice of \( w_i = 1 - p_i \) maximizes the expected test score. This choice agrees with intuition, as it gives less weight to those items having a relatively high guessing probability.

Alternatively, since the conditioned state is a trapping state, the reward \( w_i \) can be collected at the time the transition from \( x_i = 1 \) (unconditioned) to \( x_i = 0 \) (conditioned) occurs. Using this equivalent formulation, the expected immediate reward \( r(z^n, d^n) \) for using decision \( d^n \) when the state of the sufficient history is \( z^n \), is
The terms on the right are nonzero only if \( x^n_i = 1 \) and \( x^{n-1}_i = 0 \), and this situation can occur only on items \( d^n_1, d^n_2, \ldots, d^n_K \). Each of these items makes an independent transition from state 1 to 0 with probability \( p(x^{n-1}_i = 0 | x^n_i = 1, i \text{ presented}) = p(x^{n}_i = 1 | z^n_i) = c_i p(x^n_i = 1 | z^n_i) \). The latter probability is given by Eq. (5.1). Thus, the expected immediate reward function is given by

\[
r(z^n, d^n) = \sum_{i = d^n_1, d^n_2, \ldots, d^n_K} w_i c_i p(x^n_i = 1 | z^n_i)
\]

C. DYNAMIC PROGRAMMING SOLUTION

Putting the transition probabilities and expected immediate reward into the dynamic programming equations of Chapter IV gives the expected reward at the end of the process:

\[
v(0, z^n) = 0
\]

The expected reward to be collected in the remainder of the process, given history \( z^n \) when there are \( n \) stages remaining, is given by the recursive formula:

\[
v(n, z^n) = \max_{d^n} \left\{ r(z^n, d^n) + \sum_{z^{n-1}} v(n-1, z^{n-1}) p(z^{n-1} | z^n, d^n) \right\}
\]

Finally, an optimum policy, \( \hat{d}(n, z^n) \) is found from the above recursion as a \( d^n \) for which the above maximum occurs.
VI. ONE-ELEMENT LEARNING MODEL WITH AN UNKNOWN PARAMETER

One advantage of using a teaching machine is the possibility of improving the machine's performance as successive students are taught by the machine. To illustrate such a possibility, this chapter deals with the teaching situation of Chapter V, using the one-element learning model with zero guessing probability \( p_i = 0 \) and unknown conditioning probability \( c_i \). In this case the machine is able to use the histories for past students to obtain better information about the conditioning probabilities, and thus achieve better performance as time progresses.

A. PRIOR DISTRIBUTION ON THE CONDITIONING PROBABILITY

The approach of this chapter is to regard the unknown conditioning parameter as a random variable with a known prior distribution. An appropriate prior distribution for this process is the beta distribution given by the probability density function

\[
f_p(c|s,t) \propto c^{s-1}(1-c)^{t-s-1}, \quad t > s > 0, \quad 0 \leq c \leq 1
\]

The choice of the beta distribution is based on the property that, for this process, the posterior distribution of \( c \) after some history is recorded may be represented as another beta distribution with new \( s \) and \( t \) parameters. The choice of prior distribution is a direct consequence of the process involved. Such distributions are called natural conjugate prior distributions [Ref. 17] or reproducing distributions [Ref. 18]. The process under investigation behaves essentially like a sequence of independent Bernoulli trials. Suppose \( c \) is the success probability in a sequence of such trials. Here \( P \) denotes a discrete density and \( f \) denotes a continuous density. The probability of observing \( s \) successes in \( t \) trials is

\[
f(s,t|c) \propto c^s(1-c)^{t-s}
\]
If \( c \) is assumed to have a prior density

\[
f(c|\tilde{s}, \tilde{t}) = f(\tilde{c}|\tilde{s}, \tilde{t})
\]

then after observing \( s \) successes in \( t \) trials, the posterior density of \( c \) is

\[
f(c|s, t, \tilde{s}, \tilde{t}) = \frac{P(s, t|c, \tilde{s}, \tilde{t})f(c|\tilde{s}, \tilde{t})}{P(s, t|\tilde{s}, \tilde{t})}
\]

Now clearly

\[
P(s, t|c, \tilde{s}, \tilde{t}) = P(s, t|c)
\]

and since \( P(s, t|\tilde{s}, \tilde{t}) \) is not a function of \( c \),

\[
f(c|s, t, \tilde{s}, \tilde{t}) \propto c^{s}(1-c)^{t-s} \tilde{c}^{-l}(1-c)^{\tilde{t}-\tilde{s}-l} = c^{s+\tilde{s}-1}(1-c)^{t+\tilde{t}-s-\tilde{s}-l}
\]

Thus

\[
f(c|s, t, \tilde{s}, \tilde{t}) = f(\tilde{c}|s+\tilde{s}, t+\tilde{t})
\]

If the Bernoulli process is continued, the new posterior distribution is again obtained simply by adding the number of successes to \( \tilde{s}+s \), and the number of trials to \( \tilde{t}+t \). Hence, it is very easy to keep account of the effect of history on the distribution of \( c \).

In the learning process, assume that the conditioning probability of an item has a prior density \( f(\tilde{c}|\tilde{s}, \tilde{t}) \), and that \((t+1)\) presentations of the item have been made. If there have been no correct responses, then the last \( t \) (incorrect) responses indicate \( t \) failures on the first \( t \) conditioning Bernoulli trials. Hence the posterior density is \( f(\tilde{c}|\tilde{s}, \tilde{t}+t) \). If the first correct response was made on presentation \( t+1 \), then \((t-1)\) conditioning failures followed by one success were observed in a sequence of \( t \) Bernoulli trials. Hence, in this case, the posterior density of the conditioning probability is \( f(\tilde{c}|\tilde{s}+1, \tilde{t}+t) \).
B. CHOICE OF PARAMETERS IN THE PRIOR DENSITY

One method of choosing a prior distribution is to use data from past students. For each item assume that the conditioning probability is constant for all students. Suppose that $J$ students have been run on the item, and that student $j$ made a total of $t_j$ incorrect responses on the item. Also if student $j$ eventually made a correct response on the item, set $b_j = 1$, otherwise set $b_j = 0$. Assume that before any students were run, the prior density of $c$ was $f_{\theta}(c|\tilde{s}, \tilde{t})$. If no information is available on the density of $c$ at the beginning of the process, the choice of a uniform distribution, $f_{\theta}(c|1,2)$, may be an appropriate assumption. Let

$$\ell_j = \begin{cases} 0 & \text{if } t_j = 0 \\ t_j & \text{if } t_j > 0 \end{cases} \quad j = 1, 2, \ldots, J$$

Then student $j$ has made $b_j$ successes in $(\ell_j + b_j)$ conditioning Bernoulli trials, with success probability $c$. Let

$$S = \sum_{j=1}^{J} b_j, \quad T = \sum_{j=1}^{J} (\ell_j + b_j)$$

so that all of the past students together have made $S$ successes in $T$ conditioning Bernoulli trials, with success probability $c$. Thus the posterior density for the past students is

$$f(c|S,T,\tilde{s},\tilde{t}) = f_{\theta}(c|S+\tilde{s},T+\tilde{t})$$

This density is the prior density on the conditioning probability for the next student.

If the assumption that all students have the same parameter values is not valid, or if some other information about the distribution of $c$ is available, this information may be used to pick some reasonable values.
of the parameters for the prior distribution. One way to use this information is to generate some hypothetical data that agree with it and obtain values of \( S \) and \( T \) to set the parameters of the prior distribution. Another method is to pick parameters to set the mean and variance of the prior distribution. The mean and variance of the beta density, \( f_p(c|s,t) \), are

\[
\bar{c} = E[c|s,t] = \frac{s}{t}, \quad \text{var} \ c = E[(c-\bar{c})^2|s,t] = \frac{s(t-s)}{t(t+1)} = \frac{c(1-c)}{t+1}
\]

C. SUFFICIENT HISTORY

The state of the Markov learning model for item \( j \) with \( n \) steps remaining in the process is \( x^n_1 = 1 \) if item \( i \) is unconditioned, or \( x^n_1 = 0 \) if item \( i \) is conditioned. The unknown value of the conditioning probability \( c_1 \) is also an unknown state of the process. Of course \( c_1 \) does not change from presentation to presentation, but the posterior distribution of \( c_1 \) does change. Thus, the unknown Markov state of the whole process is the sequence of numbers, \( \{ (x^n_1, x^n_2, \ldots, x^n_M), (c_1, c_2, \ldots, c_M) \} \).

Since the items are assumed to be independent, one item may be considered at a time in deriving probability distributions. Let

\[
t^n_1 = \text{the number of trials in which item } i \text{ has received an incorrect response}
\]

\[
b^n_1 = \begin{cases} 1 & \text{if item } i \text{ has received a correct response} \\ 0 & \text{if item } i \text{ has received only incorrect responses} \end{cases}
\]

Then from Secs. A and B, clearly the pair \( (t^n_1, b^n_1) \) form a sufficient history for item \( i \).

The basic quantity, useful in computing the dynamic programming equations, is the probability of the unknown state \( (x^n_1, c_1) \), given the state of the sufficient history \( (t^n_1, b^n_1) \) for each item. This joint density is represented as the product of a discrete probability density in \( x^n_1 \) and a continuous probability density in \( c_1 \). Using the prior density \( f_p(c_1|s_1, t_1) \),
\[ f(c_i | t^n_1, b^n_1 = 0, \tilde{s}_i, \tilde{t}_i) = \begin{cases} f_{\beta}(c_i | \tilde{s}_i, \tilde{t}_i) & \text{if } t^n_1 = 0 \\ f_{\beta}(c_i | \tilde{s}_i, \tilde{t}_i + t^n_1 - 1) & \text{if } t^n_1 > 0 \end{cases} \]

If \( b^n_1 = 1 \) item \( i \) is conditioned, so the density is not needed. Also

\[ p(x^n_1 = 1 | c_i, t^n_1, b^n_1 = 0, \tilde{s}_i, \tilde{t}_i) = \begin{cases} 1 & \text{if } t^n_1 = 0 \\ (1-c_i) & \text{if } t^n_1 > 0 \end{cases} \]

\[ p(x^n_1 = 0 | c_i, t^n_1, b^n_1 = 0, \tilde{s}_i, \tilde{t}_i) = \begin{cases} 0 & \text{if } t^n_1 = 0 \\ c_i & \text{if } t^n_1 > 0 \end{cases} \]

Thus, the joint distribution of \( x^n_1 \) and \( c_i \) may be expressed as

\[ p(x^n_1 | c_i, t^n_1, b^n_1 = 0, \tilde{s}_i, \tilde{t}_i) f(c_i | t^n_1, b^n_1 = 0, \tilde{s}_i, \tilde{t}_i) \]

\[ = \begin{cases} f_{\beta}(c_i | \tilde{s}_i, \tilde{t}_i) & \text{if } t^n_1 = 0 \\ (1-c_i) f_{\beta}(c_i | \tilde{s}_i, \tilde{t}_i + t^n_1 - 1) & \text{if } t^n_1 > 0 \end{cases} \]

\[ = \begin{cases} 0 & \text{if } t^n_1 = 0 \\ c_i f_{\beta}(c_i | \tilde{s}_i, \tilde{t}_i + t^n_1 - 1) & \text{if } t^n_1 > 0 \end{cases} \]

D. EXPECTED IMMEDIATE REWARD

As in Chapter V, the reward \( w_i \) is collected only if item \( i \) is in the unconditioned state and makes a transition to the conditioned state.
Let \( r_i(t_i^n, b_i^n) \) be the expected immediate reward if item \( i \) is presented given its sufficient history \( (t_i^n, b_i^n) \). Then

\[
r_i(t_i^n, b_i^n = 0) = w_i P(x_i^{n-1} = 0, x_i^n = 1 | t_i^n, b_i^n = 0, s_i, t_i) = w_i \int_0^1 P(x_i^{n-1} = 0 | x_i^n = 1, c_i, t_i^n, b_i^n = 0, s_i, t_i) \cdot P(x_i^n = 1 | c_i, t_i^n, b_i^n = 0, s_i, t_i) f(c_i | t_i^n, b_i^n = 0, s_i, t_i) dc_i
\]

\[
= \begin{cases} 
  w_i \int_0^1 c_i \frac{f(c_i | s_i, t_i)}{dc_i} & \text{if } t_i^n = 0 \\
  w_i \int_0^1 c_i (1-c_i) \frac{f(c_i | s_i, t_i + t_i^n - 1)}{dc_i} & \text{if } t_i^n > 0
\end{cases}
\]

The above integrals are easily evaluated using moments of the beta density function, yielding the expression

\[
r_i(t_i^n, b_i^n = 0) = \begin{cases} 
  w_i \frac{s_i}{\tilde{t}_i} & \text{if } t_i^n = 0 \\
  w_i \frac{s_i (\tilde{t}_i + t_i^n - 1)}{(\tilde{t}_i + t_i^n)(\tilde{t}_i + t_i^n - 1)} & \text{if } t_i^n > 0
\end{cases}
\]  

(6.1)

Also clearly,

\[
r_i(t_i^n, b_i^n = 1) = 0
\]  

(6.2)
Let \( z^n_i = (t^n_i, b^n_i) \) denote the sufficient history for item \( i \), and let 
\[
z^n = (z^n_1, z^n_2, \ldots, z^n_M)
\]
denote the sufficient history for the whole process.
Then the expression for the expected immediate reward under decision 
\[
d^n = (d^n_1, d^n_2, \ldots, d^n_K)
\]
given the sufficient history \( z^n \) is

\[
r(z^n, d^n) = \sum_{i = d^n_1, d^n_2, \ldots, d^n_K} r(z^n_i, d^n_i)
\]

E. TRANSITION PROBABILITIES

If item \( i \) is presented, the response will be incorrect if and only if \( x^n_i = 1 \). Thus

\[
p(t^n_{i-1}, b^n_{i-1} | t^n_i, b^n_i, i \text{ presented}) = p(x^n_i = 1 | t^n_i, b^n_i = 0, \tilde{s}_i, \tilde{t}_i)
\]

if \( t^n_{i-1} = t^n_{i-1}, b^n_{i-1} = b^n_i = 0 \). Now

\[
p(x^n_i = 1 | t^n_i, b^n_i = 0, \tilde{s}_i, \tilde{t}_i) = \int_0^1 p(x^n_i = 1 | c_i, t^n_i, b^n_i = 0, \tilde{s}_i, \tilde{t}_i) f(c_i | t^n_i, b^n_i = 0, \tilde{s}_i, \tilde{t}_i, dc_i)
\]

\[
= \begin{cases} 
\int_0^1 f(c_i | \tilde{s}_i, \tilde{t}_i) dc_i & \text{if } t^n_i = 0 \\
\int_0^1 (1-c_i) f(c_i | \tilde{s}_i, \tilde{t}_i, t^n_{i-1}) dc_i & \text{if } t^n_i > 0 
\end{cases}
\]

\[
= \begin{cases} 
1 & \text{if } t^n_i = 0 \\
1 - \frac{\tilde{s}_i}{\tilde{t}_i + t^n_{i-1}} & \text{if } t^n_i > 0 
\end{cases}
\]
When $b_i^n = 0$ the other possibility is that of a correct response on the next trial, yielding $b_i^{n-1} = 1$ and $t_i^{n-1} = t_i^n$. This event occurs with 1 minus the above probability. When $b_i^n = 1$ the sufficient history for item 1 does not change from trial to trial. These results are summarized in Table 7.

**TABLE 7. MARKOV TRANSITION PROBABILITIES FOR THE SUFFICIENT HISTORY $z_i^n = (t_i^n, b_i^n)$ WHEN ITEM 1 IS PRESENTED**

| $t_i^n$ | $b_i^n$ | $t_i^{n-1}$ | $b_i^{n-1}$ | $P(z_i^{n-1} | z_i^n, i$ presented) |
|--------|--------|-------------|-------------|----------------------------------|
| 0      | 0      | 1           | 0           | 1                                |
| $t_i^n > 0$ | 0      | $t_{i+1}^n$ | 0           | $1 - \frac{\tilde{s}_i}{t_i^n + t_i^{n-1}}$ |
| $t_i^n > 0$ | 0      | $t_i^n$    | 1           | $\frac{\tilde{s}_i}{t_i^n + t_i^{n-1}}$ |
| $t_i^n$ | 1      | $t_i^n$    | 1           | 1                                |

For items not presented, the sufficient histories remain unchanged. The Markov transition matrix $P(z_i^{n-1} | z_i^n, d^n)$ for the sufficient history $z_i^n$, under decision $d^n$, is formed by taking an appropriate product of these probabilities.

F. DYNAMIC PROGRAMMING EQUATIONS

The dynamic programming equations are now given by simply putting the expected immediate reward function of Sec. D and the transition probabilities of Sec. E into the dynamic programming formulation of Chapter IV. Since there is no terminal reward in this process,

$$v(0, z^0) = 0$$
The expected reward to be collected in the remainder of the process, given the state of history \( z^n \), when there are \( n \) steps remaining is given by the recursive formula

\[
v(v,z^n) = \max_{d^n} \left[ r(z^n,d^n) + \sum_{z^{n-1}} v(n-1,z^{n-1}) p(z^{n-1}|z^n,d^n) \right]
\]

An optimum decision policy \( d(n,z^n) \) is found as a \( d^n \) for which the above maximum occurs.

G. MAXIMIZATION OF EXPECTED IMMEDIATE REWARD

In Chapter III, Sec. G, it was shown that for the example under consideration the policy of presenting the item which maximized the expected immediate reward, without regard to the longer run terms, was an optimum policy. The proof was based on the fact that the items were independent and on the property that, no matter what events occurred on a presentation, the expected immediate reward for presenting an item at the next step in the process was always less than or equal to the reward for presenting the item at the current step. Under certain conditions, maximization of expected immediate reward will be shown to be optimum for the processes considered in this chapter. This will be accomplished by showing that the expected immediate rewards for presenting a given item are a non-increasing function of time. The rest of the proof is similar to the one in Chapter III and will not be presented here.

Dropping the subscripts and superscripts from the variables in Eqs. (6.1) and (6.2) yields

\[
r(t,b=0) = \begin{cases} 
\frac{w_s}{t} & \text{if } t = 0 \\
\frac{w_s(t+t-1-s)}{(t+t)(t+t-1)} & \text{if } t > 0
\end{cases}
\]

(6.3)
and

\[ r(t, b = 1) = 0 \] \hspace{1cm} (6.4)

While incorrect responses are being made on an item, \( b = 0 \) and \( t \) keeps increasing. When the first correct response is made, \( b = 1 \) and both \( b \) and \( t \) remain fixed for the remainder of the process. It is clear that

\[ \frac{ws}{\tilde{t}} \frac{ws(\tilde{t}+1-\tilde{s})}{(\tilde{t}+t)(\tilde{t}+t-1)} \geq 0 \hspace{1cm} \text{for} \hspace{0.5cm} t > 0 \]

or

\[ r(0, b = 0) \geq r(t, b = 0) \geq r(t, b = 1) \hspace{1cm} \text{for} \hspace{0.5cm} t > 0 \]

Thus to show that the expected immediate reward is a nonincreasing function of time, it must be shown that \( r(t, b = 0) \) is a nonincreasing function of \( t \) for \( t > 0 \).

Consider the difference \( \Delta r(t) \) defined as

\[ \Delta r(t) = r(t+1, b = 0) - r(t, b = 0) = \frac{ws(\tilde{t}-t+1+2\tilde{\tau})}{(\tilde{t}+t)(\tilde{t}+t+1)(\tilde{t}+t-1)} \hspace{1cm} t = 1, 2, \ldots \]

The difference \( \Delta r(t) \) is nonpositive if

\[ -\tilde{t}+t+1+2\tilde{\tau} \leq 0 \hspace{1cm} t = 1, 2, \ldots \]

The left side of the above inequality is maximum when \( t = 1 \). Thus \( \Delta r(t) \) is nonpositive if

\[ -\tilde{\tau}+2\tilde{\tau} \leq 0 \hspace{1cm} \text{or} \hspace{1cm} \frac{s}{\tilde{t}} \leq \frac{1}{2} \]

The quantity \( \frac{s}{\tilde{t}} \) is just the mean of the prior beta density. Thus, if the prior densities on all items have means no greater than one-half, an
optimum procedure is to always present the items which maximize the expected immediate reward. That is

\[ \hat{d}(n, z^n) = \hat{d}(1, z^n) \quad n = 1, 2, \ldots \]

If all items have exactly the same prior distribution on their conditioning probabilities with mean no greater than one-half and equal reward \( w_i \), an optimum procedure is to present each item in succession, cycling through the list of items over and over, dropping items out of the list when a correct response is received, until the end of the process is reached.

If the items have different prior beta distributions, and all their means are no greater than one-half, the optimum procedure is similar, but the cyclic order is not preserved. In this case a table of expected immediate reward as a function of \( t_i \) may be constructed for each item, and the items having maximum expected immediate reward may be presented.
VII. A STATIC PROBLEM

If the history of the process contains only the history of past decisions and no information about past responses, then all decisions for the whole process can be made before the process is started. Thus, since no dynamic interaction is needed with the student during the teaching process, this situation is called the static case. For example, the static case may represent a linear-program-type teaching-machine book, where the book is completely written before any students are taught and where all students receive exactly the same course of instruction.

A. THE PROBLEM

The problem considered here is essentially that of Chapter III, except that no response information is recorded in the history of the process. Thus we want to teach a student M items in a total of N steps. Each step consists of a presentation of a single item pair. In Sec. B the solution is obtained by applying dynamic programming to the one-element model. In Sec. C the problem is reformulated in terms of the mean learning curve, and the same result is obtained for a wider class of models.

B. THE ONE-ELEMENT MODEL

The learning of each item i is assumed to be governed by a one-element model with learning state $x_i^n$. The transition matrix, which is applied each time item i is presented, is

$$
\begin{align*}
\text{Old state} & \quad \begin{cases} x_i^n = 0 \\ x_i^n = 1 \end{cases} \\
\text{New state} & \quad \begin{cases} x_i^{n-1} = 0 \\ x_i^{n-1} = 1 \end{cases} \\
& \begin{bmatrix} 1 & 0 \\ c_i & 1-c_i \end{bmatrix}
\end{align*}
$$
1. Dynamic Programming Formulation

Because no response information is available, after \( z_i^n \) presentations of item 1 the probability distribution of the unknown state \( x_i^n \) is given by

\[
P(x_i^n = 1 | z_i^n) = (1 - c_i)^{z_i^n} \\
P(x_i^n = 0 | z_i^n) = 1 - (1 - c_i)^{z_i^n}
\] (7.1)

Since, given the number of times each item has been presented, the order of their presentation does not affect the resulting state probabilities, \( z^n = (z_1^n, z_2^n, \ldots, z_M^n) \) is a sufficient history for this process.

The following list summarizes the procedure from the conclusion of one step to the conclusion of the following step:

1. At the conclusion of the previous step the process was left in state \( x^n \), and the updated sufficient history was \( z^n \).
2. Based on the sufficient history \( z^n \), item \( d^n \) is chosen for presentation.
3. Item \( d^n \) is presented to the student, causing a transition from \( x^n \) to \( x^{n-1} \).
4. The sufficient history is updated by the decision \( d^n \) to form the new sufficient history \( z^{n-1} \).
5. At the conclusion of the step the process is left in state \( x^{n-1} \), and the new state of the sufficient history is \( z^{n-1} \).

If item \( d^n \) is presented with \( n \) steps remaining, then \( z^n = (z_1^n, z_2^n, \ldots, z_M^n) \) is changed to \( z^{n-1} = (z_1^n, z_2^n, \ldots, z_{d^n-1}^n, z_{d^n}^n, z_{d^n+1}^n, \ldots, z_M^n) \) with probability 1. This is the (deterministic) Markov transition probability distribution for the sufficient history.

Assume reward \( w_i \) is received when item \( i \) makes a transition from the unconditioned state \( x_i^n = 1 \) to the conditioned state \( x_i^{n-1} = 0 \). If \( x_i^n = 1 \), this transition occurs with probability \( c_i \), and it cannot occur otherwise. Hence the expected immediate reward \( r(z^n, d^n) \), under decision \( d^n \) given history \( z^n \), is
\[ r(z^n, d^n) = w_n c_n (1-c_n)^{z^n_n} d^n 
\]

2. Maximization of Expected Immediate Reward

Clearly \( z^n_i \) is a non-decreasing function of time, and the expected immediate reward for a presentation of item \( i \) is a non-increasing function of \( z^n_i \). Thus, for a given item, the expected immediate reward at some future time cannot be greater than the expected immediate reward at the current time. Also the items are assumed to be independent. These are the properties used in Chapter III to show that presenting the item which maximizes expected immediate reward is an optimum decision policy. Using this policy, at each stage the item \( d^n \) is presented when it is a \( z^n \) for which \( r(z^n, d^n) \) is maximized.

Since the items are independent, only \( z^0 = (z^0_1, z^0_2, ..., z^0_M) \), the total number of times to present each item, is needed to construct an optimum sequence of frames. The following algorithm implements the procedure of maximization of expected immediate reward to yield \( z^0 \).

1. Set the variables \( z^0_1, z^0_2, ..., z^0_M \) all to zero.
2. Inspect the quantity \( w_i (1-c_i)^{z^0_i} \) over all \( i = 1, 2, ..., M \) to determine a maximum. Denote by \( j \) an item for which this maximum occurs.
3. Increase the value of \( z^0_j \) by 1.
4. Go back to step 2 and continue until steps 2 and 3 have been executed \( N \) times.
5. Set \( z^0_i = z^0_i \) for \( i = 1, 2, ..., M \).

C. MEAN-LEARNING-CURVE FORMULATION

1. The One-Element Model

For the one-element model, the probability an item is unconditioned given that it has been presented \( z_i \) times is \( (1-c_i)^{z_i} \). On the following presentation of item \( i \), an incorrect response is made with probability
$1 - p_i$ if item $i$ is in the unconditioned state, and an incorrect response cannot be made if item $i$ is in the conditioned state. Thus the probability of a correct response on presentation $z_i$ of item $i$, given $z_i$, is

$$P_{i,z_i} = P(\text{correct} \mid z_i, \text{item } i \text{ presented}) = 1 - (1-p_i)(1-c_i)^{z_i}$$

This correct response probability as a function of $z_i$ is called the mean learning curve for item $i$.

Assume reward $w_i$ is received if item $i$ is in the conditioned state at the end of the process. If item $i$ is presented a total of $z_i^0$ times during the process, then the expected reward for item $i$ is

$$r_i(z_i^0) = w_i \left[ 1 - (1-c_i)^{z_i^0} \right] = w_i - w_i (1-c_i)^{z_i^0} \quad (7.2)$$

An alternative reward structure is to receive reward $R_i$ for a correct response on an item $i$ test trial administered at the end of the process. In this case the expected reward for item $i$ is

$$r_i(z_i^0) = R_i \left[ 1 - (1-p_i)(1-c_i)^{z_i^0} \right] = R_i - [R_i(1-p_i)](1-c_i)^{z_i^0} \quad (7.3)$$

The terms $w_i$ in Eq. (7.2) and $R_i$ in Eq. (7.3) are constant terms collected independently of the number of times the items are presented. Thus the terms being maximized are $-w_i (1-c_i)^{z_i^0}$ in Eq. (7.2) and $-[R_i(1-p_i)](1-c_i)^{z_i^0}$ in Eq. (7.3). Thus, if $w_i$ or $R_i$ is adjusted so that $w_i = [R_i(1-p_i)]$, the reward structures are identical. The latter reward structure is the one appropriate for the linear learning model.
2. **The Linear Model**

The linear model, which was presented in Chapter I, is given by the difference equation

\[ P_{i,z_{i+1}} = (1-c_i)P_{i,z_i} + c_i, \quad P_{i,0} = P_i \]

where \( P_{i,z_i} \) is the probability of a correct response on presentation \( z_{i+1} \), and \( P_i \) is the probability of a correct response on the first presentation. This difference equation has the solution for the mean learning curve

\[ P_{i,z_i} = 1 - (1-P_i)(1-c_i)^z_i \]

This mean learning curve is identical to the one already derived for the one-element model. If reward \( R_i \) is received for a correct response on an item \( i \) test presentation at the end of the process, the expected reward for item \( i \) is again given by Eq. (7.3). Thus, by the previous remarks, if \( w_i \) is set to \( R_i(1-P_i) \) the algorithm of Sec. B gives an optimum solution.

3. **An Alternative Derivation of the Solution**

The preceding solution of this problem was based on the use of dynamic programming. The alternative derivation in this section is based directly on the mean learning curve and is useful in obtaining the approximate solution in Sec. C4.

In each of the cases presented in Sec. C2, a set of weights \( w_i \) and a constant \( w_0 \) can be found so that the problem is to pick \( z^0 \) to maximize the functional \( F(z^0) \) given by

\[ F(z^0) = \sum_{i=1}^{M} w_i \left[ 1 - (1-c_i)^z_i \right] + w_0 \]
Since $w_0$ is a constant, an equivalent problem is to maximize the functional

$$G(z^0) = F(z^0) - w_0 = \sum_{i=1}^{M} w_i \left[ 1 - (1-c_i)^{z_i^0} \right] \quad (7.4)$$

Since there is a total of $N$ steps in the process, $z_i^0$ must also satisfy the constraint

$$H(z^0) = \sum_{i=1}^{M} z_i^0 - N = 0 \quad (7.5)$$

Now, applying the identity

$$1 - (1-c)^Z = c[(1-c)^0 + (1-c)^1 + \ldots + (1-c)^{z-1}]$$

to Eq. (7.4) yields

$$G(z^0) = \sum_{i=1}^{M} \left[ w_i c_i (1-c_i)^0 + w_i c_i (1-c_i)^1 + \ldots + w_i c_i (1-c_i)^{z_i^0-1} \right]$$

Since each sequence $w_i c_i (1-c_i)^0, w_i c_i (1-c_i)^1, \ldots$ is a positive monotonically decreasing sequence, the maximization reduces to finding the $N$ largest elements of the set

$$\left\{ w_i c_i (1-c_i)^j : j = 0, 1, \ldots, N-1; i = 1, 2, \ldots, M \right\}$$

The set of the $N$ largest elements contains the first $z_i^0$ elements of the item $i$ sequence, and hence $z_i^0$ is an optimum solution. Clearly this procedure is implemented by the algorithm of Sec. B2.
4. An Approximate Solution

By allowing the integer-valued variables, $z_i$, to take on real values, a relatively simple solution for the optimum values is found. This real-valued solution may be used as a starting point for determining the integer solution, and it also may be used to determine an upper bound for the maximum of $G$ achieved by the integer solution.

The problem [see Eqs. (7.4) and (7.5)] is to maximize the functional

$$G(z_1, z_2, \ldots, z_M) = \sum_{j=1}^{M} w_j \left[ 1 - (1-c_j) z_j \right]$$

subject to the constraint

$$H(z_1, z_2, \ldots, z_M) = \sum_{j=1}^{M} z_j - N = 0$$

Using the technique of Lagrange multipliers, set

$$-\frac{\partial G}{\partial z_j} + \lambda \frac{\partial H}{\partial z_j} = 0, \quad j = 1, 2, \ldots, M$$

Eliminating $\lambda$ between equations i and j yields

$$-w_j (1-c_j) z_j^0 \ln (1-c_j) = -w_i (1-c_i) z_i^0 \ln (1-c_i)$$

Taking logarithms and solving for $z_i^0$ yields

$$z_i^0 = \frac{z_j^0 \ln (1-c_j) + \ln [-w_j \ln (1-c_j)] - \ln [-w_i \ln (1-c_i)]}{\ln (1-c_i)}$$
Substituting the above equation into the constraint equation gives

\[
z_j^0 = \frac{N - \alpha \ln \left[-w_j \ln \left(1-c_j \right)\right] + \beta}{\alpha \ln \left(1-c_j \right)}
\]

where

\[
\alpha = \sum_{i=1}^{M} \frac{1}{\ln \left(1-c_i \right)}, \quad \beta = \sum_{i=1}^{M} \frac{\ln \left[-w_i \ln \left(1-c_i \right)\right]}{\ln \left(1-c_i \right)}
\]

In this solution, some of the \( z_j^0 \) may turn out to be negative. This possibility results from the inclusion of items which should not be used at all. When this situation occurs, the item corresponding to the most negative \( z_j^0 \) should be eliminated from consideration and the calculations should then be repeated.
A. STRUCTURE OF THE OPTIMAL POLICIES

Several times in this work the policy of maximization of expected immediate reward has appeared as an optimal policy. For the fixed-parameter problem of Chapter V this result could not be shown. Hence, a computer was programmed to evaluate the dynamic programming solution in order to investigate the structure of the optimal policies. Two-item problems with ten steps and three-item problems with six steps were both investigated.

It was found in every case where the $c_i$ values were equal on all items and the $p_i$ values were equal on all items that the policy of maximizing expected immediate reward was an optimal policy. Under this policy the item having the lowest value of $z_i$ is always presented. The following procedure is a more detailed description of this policy to be used until the $N$ steps are exhausted:

1. Set the level variable $l$ to $l = 0$.
2. Present any item with $z'_i = l$. Suppose item $j$ is presented.
3. If in step 2 an incorrect response causes $z_j$ to return to a value less than $l$, repeat item $j$ presentations until the $z_j$ value is increased back to $l$.
4. If all $z_i$ values are equal to $l+1$, increment the level $l$ by 1.
5. Go back to step 2.

The above procedure tries to simultaneously increase the $z_i$ values of all items by cycling through the items and increasing the $z_i$ value of each item by 1. However, if an item receives an error, its $z_i$ value returns to $z_i = 1$. The optimum procedure is to then repeat presentations of this item until its $z_i$ value "catches up" with that of the other items.

When the items are not homogeneous, that is when the $c_i$ or $p_i$ values are not equal across items, maximization of expected immediate reward is usually not the optimal policy. In this case the policy depends not only on the state $x$ of the items, but also on the number of frames remaining in the process. In general this procedure is similar
to the previous one. The $z_i$ values of items are increased somewhat together, but they are not locked as tightly in step. Usually when an incorrect response occurs on an item, taking it to a $z_i$ value of one, that item is repeated until its $z_i$ value is increased to its former value. The exceptions to this rule occur near the termination of the process, where the time-varying nature of the policy tends to become more pronounced.

B. SUBOPTIMAL POLICIES

Since the maximization of expected immediate reward (that is, the expected one-step reward) was optimum in the homogeneous case, it seems reasonable that the maximization of the expected $j$-step reward, as suggested in Chapter II, Sec. D, might be at least nearly optimum for $j$ greater than 1 but less than $N$. The policy which maximizes the expected $j$-step reward is denoted $d_j$ and is defined by

$$d_j(n, z^n) = \begin{cases} 
\hat{d}(n, z^n) & \text{for } n = 1, 2, \ldots, j \\
\hat{d}(j, z^n) & \text{for } n = j + 1, j + 2, \ldots 
\end{cases}$$

For a two-item process, the value of being in the initial state, $z^n = (0,0)$, for an $(N=10)$-step process was computed for optimal and suboptimal policies for many choices of parameter values. The optimum value was also computed for the static case, in which no observations are permitted, as in Chapter VII. The differences between the value of the optimal and suboptimal policies, expressed as percentages of the optimum values, are plotted in Figs. 1 and 2. Notice that the curve for the optimal static solution varies smoothly as the parameters change. This smoothness can be explained by the fact that the static solution, being an optimal policy under the no-observation restriction, switches smoothly between 10-step value curves for various policies as the parameter varies. On the other hand, the $j$-step suboptimal policies are switching smoothly between $j$-step value curves for various policies as the parameter varies. This switching has no direct relation to the 10-step value curves and
hence causes the loss over the 10-step optimal policy to be discontinuous. One general conclusion that can be drawn from these graphs is that although the optimal static policy is usually 1 to 7 percent worse than the true optimal policy, the speed of convergence of the j-step suboptimal policy to the optimal policy is somewhat erratic as the parameter value changes.

![Graph](image)

**FIG. 1. COMPARISON OF OPTIMAL AND SUBOPTIMAL POLICIES AS A FUNCTION OF c_2 WITH**

\[ p_1 = 0.5, \ c_1 = 0.2, \ p_2 = 0.2. \]
FIG. 2. COMPARISON OF OPTIMAL AND SUBOPTIMAL POLICIES AS A FUNCTION OF $c_2$ WITH $p_1 = 0.5$, $c_1 = 0.2$, $p_2 = 0.5$. 
A. LEARNING MODELS AND OPTIMIZATION

In this work, optimum teaching procedures have been found for several teaching situations. The learning process was described by mathematical learning models, and the optimum teaching procedures were derived by the application of dynamic programming. From the study of these situations, the following conclusions are drawn concerning the use of mathematical models in the optimization of teaching machines.

The main requirement for the dynamic programming equations to provide a practical method for computing the optimum teaching procedure is that the number of possible states of sufficient history for the whole process be relatively small. This number is a property of the learning model being used. For the learning model of Chapter III, there are three states of sufficient history for each item, making $3^M$ possible states of sufficient history for the whole process. For the learning models used in Chapters V-VII, there are $N$ states of sufficient history for each item, where $N$ is the total number of steps in the process. Also the sum of the state values across items can never be greater than $N$, so that there are $\binom{N+M}{M}$ possible states of sufficient history which may occur during the process. This number behaves as $N^M$ for large $N$. With modern computer capabilities, optimization of processes with as many as 10,000 states may be reasonable.

One reason that the number of states of sufficient history becomes so large is that the state of the learning model is not directly observable. Thus, if the model is made at all complicated, most of the history will be relevant to the estimation of the unknown state. A suggestion for alleviating this situation is that learning models be formulated as Markov processes with directly observable states. To justify these models, it has been shown that for any Markov learning model with unobservable states, a Markov process in states of observable history can be inferred. However, this process usually has a large number of states of sufficient history. This new Markov process, with the possible values of sufficient history as its states, may be thought of as the learning model for the
purpose of optimization. For a given teaching situation, it may be possible to reduce the number of states of sufficient history by grouping states which are not significantly different within the limitations of the model, thus forming a new model with a smaller number of states. For example, one may simplify the model of Chapter V by assuming an item is in the conditioned state once I consecutive correct responses have occurred, that is, once $z$ reaches the value of $1 + 1$. Then there are only $(I+1)^{M}$ states of sufficient history in the whole process.

Another possible use of optimization techniques is comparing different learning models. Learning models are usually compared by seeing which model predicts some chosen statistic of experimental data better than the other models. If a model predicts all statistics accurately, then the optimization based on that model will almost surely give good results. However, there are usually several feasible models for a given situation, each predicting some statistics more accurately than the other models. Unfortunately, it is not apparent which statistical predictions are important in performing a good optimization. Thus it seems that an appropriate way to compare models, at least for teaching purposes, is to (1) find optimum teaching procedures based on each of the models; (2) teach students by each method; and (3) test the student to find which model led to the best learning.

In many of the situations considered, an optimum procedure is always to make the decision which maximizes the expected one-step reward. In other situations, a practical suboptimal procedure is always to make the decision which maximizes the expected $j$-step reward, where $j$ is greater than or equal to 1 but much less than the total number of steps in the process. Such procedures may be a practical way to approach the machine teaching of paired-associate material, such as vocabulary or spelling.

B. DYNAMIC PROGRAMMING

The main contribution of this research to the theory of dynamic programming is developed in Chapter IV, "Dynamic Programming with Unobservable States." For the purpose of optimization, it was shown
that for any Markov process with unobservable states, an equivalent
Markov process can be found in observable history. The number of states
in this new process is determined by the number of states of sufficient
history required to statistically summarize the information in the
history relevant to the unobservable state of the original process. It
is possible that these results may also have application to problems
outside the teaching-machine field.

Several times in this work, it was found that an optimum procedure
was to maximize the expected immediate reward. That is, at each step
in the process the decision is made which maximizes the expected reward
to be collected during the next step in the process, without regard to
any future rewards. Since, for a given basic reward structure, there
may be several ways to choose the immediate reward function by trading
off reward between immediate and terminal reward, an interesting, out-
standing question is, What are necessary and sufficient conditions on
the structure of the problem under which maximization of expected
immediate reward is an optimum procedure?
REFERENCES


Division of Research, Graduate School of Business Administration,
Harvard University, Boston, Mass., 1961.

18. J. D. Spragins, Jr., "Reproducing Distributions for Machine
Learning," Rept SEL-63-099 (TR No. 6103-7), Stanford Electronics
Laboratories, Stanford, Calif., Nov 1963. AD 431994.