Operations Research

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Composition Principles for Synthesis of Optimal Multistage Processes

L. G. Mitten,

To cite this article:
L. G. Mitten, (1964) Composition Principles for Synthesis of Optimal Multistage Processes. Operations Research 12(4):610-619.
https://doi.org/10.1287/opre.12.4.610

COMPOSITION PRINCIPLES FOR SYNTHESIS OF OPTIMAL MULTISTAGE PROCESSES

L. G. Mitten

Department of Industrial Engineering and Management Sciences, The Technological Institute, Northwestern University, Evanston, Illinois

(Received March 20, 1964)

Any multistage process may be constructed by a sequence of serial and/or parallel compositions. With certain restrictions on the form of the return composition function, optimal processes may be constructed in an efficient manner from optimal subprocesses. A number of simple sufficient conditions on the return functions are given and several approximation procedures are outlined.

THE CONSTRUCTION of optimal multistage processes is a problem which increases in difficulty with the number of stages considered. Computationally feasible methods are obtained only by postulating certain exploitable properties, such as linearity, continuity, etc. In this paper we propose to exploit certain recursive properties of optimal return functions to obtain computationally efficient methods of solution. The work represents an extension of BELLMAN's developments in dynamic programming[1, 4] and makes use of KARLIN's formulation[2] of such problems in terms of decision functions.[3]

The general approach and results reported in this paper have also been influenced by informal exchanges with G. NEMHAUSER, R. ARIS, and members of the Dynamic Programming Seminar at Northwestern University. In particular, E. DENARDO of the latter group has made valuable suggestions concerning notation and the weakening of the lemma to its present form. A remark by Aris that the dynamic programming 'principle of optimality' should perhaps be recast to read "any process operating between fixed endpoints must be operated optimally" is very much in the spirit of this paper. The author, however, assumes full responsibility for this exposition and any errors there-in.

In the following sections, we define a process and show how multistage processes may be synthesized by two types of composition operations. Recursive methods for the construction of optimal processes are considered and a method of reducing the dimensionality of the required state-space is given. The paper concludes with some sufficient conditions on the optimal return composition function and a brief development of several approximation methods.

To keep the technical apparatus as simple as possible, it will be assumed when speaking of an optimum that a unique maximum is actually achieved over the specified region. The assumption has no significant effect on the results, other than to simplify the notation.

DEFINITION OF A PROCESS

THE BASIC elements of a process are *inputs*, *outputs*, *decisions*, and *returns*. The decisions (or policies) transform inputs into outputs and give rise to returns that measure the effectiveness of the process. Thus, a process is described by an input set, an output set, a set of decision functions, and a return function (*see* Fig. 1).

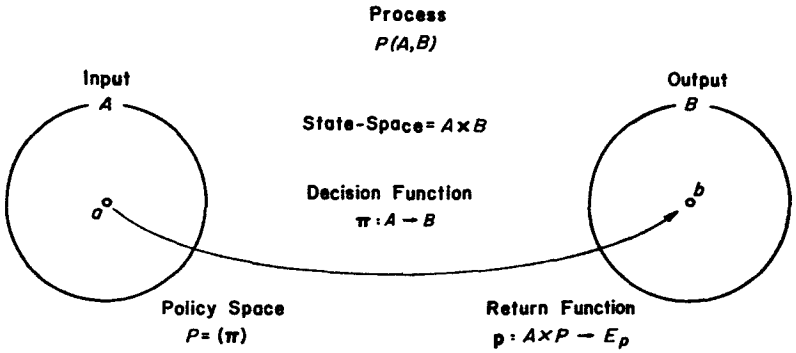

Figure 1

Let $A=(a)$ be an arbitrary set whose elements are called inputs, and let $B=(b)$ be an arbitrary set whose elements are called outputs. The Cartesian product $A\times B$ is called the *state-space* for the process. A *decision function* π: $A\rightarrow B$ is a single valued transformation $\pi(a)=b$ of each input $a\epsilon A$ into a unique output $b\epsilon B$. *Policy space*, $P=(\pi)$, is the set of all admissible decision functions π from A to B. A policy space P, along with its associated state-space $A\times B$, is a process $P(A,B)$.

Solely as a matter of notational convenience, we assume that every $\pi\epsilon P$ has domain A and range B, and we further assume that for every $(a,b)\epsilon A\times B$ there exists at least one $\pi\epsilon P$ such that $\pi(a)=b$. These requirements may always be satisfied by adjoining special elements to P and/or $A\times B$ and assigning very disadvantageous returns to them.

Associated with each process $P(A,B)$ is a single-valued *return function* $\boldsymbol{p}:A\times P\rightarrow E_P$ where E_P is Euclidian n-space with $n(<\infty)$ depending on P. The return $\boldsymbol{p}(a,\pi)\epsilon E_P$ measures the utility of using the transformation $\pi\epsilon P$ on the input $a\epsilon A$. If E_P is the real line, then the *optimal return function* $\boldsymbol{p}^*:A\times B\rightarrow E_P$ is a single-valued mapping from the state-space $A\times B$

to E_P defined by

$$\boldsymbol{p}^*(a,b)=\max[\boldsymbol{p}(a,\pi)|\pi(a)=b]. \tag{1}$$

We call p^*(a,b) the *optimal return from a to b*. The *optimal process from a to b* is the set $P^*(a,b)$ defined by

$$P^*(a,b)=\{\pi|[\pi(a)=b][\boldsymbol{p}(a,\pi)=\boldsymbol{p}^*(a,b)]\}. \tag{2}$$

In our definition of a process, the input and output sets are entirely arbitrary and the decision and return functions are arbitrary single-valued mappings. Thus, the formulation is quite general and should include most situations of practical or theoretical interest, except possibly some stochastic processes requiring multivalued maps (these will be investigated in a separate report).

CONSTRUCTION OF PROCESSES BY RECURSIVE COMPOSITION

THERE ARE two basic methods of conjoining component processes to form composite processes: *serial composition* and *parallel composition*. In serial composition, the output of one process becomes the input to the next process, in parallel composition, inputs are associated with inputs and outputs are associated with outputs.

The parallel composition of two processes $R(A',B')$ and $S(A'',B'')$ yields the parallel process $P(A,B)=R(A',B')//S(A'',B'')$ where $P=RXS$ and $A=A'\times A''$ and $B=B'\times B''$. Thus, the parallel transformation $\pi(a)=b$ is exactly equivalent to the pair of independent transformations $\varrho(a')=b'$ and $\sigma(a'')=b''$, provided both the latter are defined. The return associated with the parallel transformation is the vector $\boldsymbol{p}(a,\pi)=[\boldsymbol{r}(a',\varrho),\boldsymbol{s}(a'',\sigma)]\epsilon E_P=E_R\times E_S$.

Any two processes may be joined by parallel composition, but only certain processes (termed *conformable*) may be joined by serial composition. Two processes $R(A,C)$ and $S(D,B)$ are said to be conformable (in the given order) if and only if $C=D\neq\emptyset$. The serial composition of two conformable processes $R(A,C)$ and $S(C,B)$ yields the serial process $P(A,B)=R(A,C)\cdot S(C,B)$ with the serial transformation $\pi=\sigma\cdot\varrho$ defined by $\pi(a)=\sigma\cdot\varrho(a)=\sigma[\varrho(a)]$ for each $(\varrho,\sigma)\epsilon R\times S=P$. In general, the return $p(a,\pi)$ associated with the serial transformation $\pi(a)=\sigma\cdot\varrho(a)=b$ will be a function, say $\boldsymbol{h}_P(\pi,\sigma,\varrho,a,b)$ but in order to obtain useful results we shall have to make stronger assumptions in subsequent sections.

Recursive application of serial and/or parallel composition enables us to construct processes of any size and complexity. However, the dimensionality of the policy space for such composite processes increases directly with the number of stages in the process. Thus, the practicality of using recursive composition as a means of constructing optimal processes depends

on being able to limit the search for an optimal process to a small region in policy space. We propose to limit the region to be searched by using optimal component processes to construct optimal composite processes. Various construction procedures will be outlined and the conditions under which each is valid will be delineated.

RECURSIVE COMPOSITION OF OPTIMAL SERIAL PROCESSES

THE POSSIBILITY of constructing optimal serial processes from optimal component processes depends on the existence of recursively exploitable properties of optimal return functions. For the serial composition $P(A,B) = R(A,C) \cdot S(C,B)$, three obvious assumptions suggest themselves: the optimal composite return $\boldsymbol{p}^*(a,b)$ may depend on one or the other or both of the optimal component returns $\boldsymbol{r}^*(a,c)$ and $\boldsymbol{s}^*(c,b)$—i.e., either

$$\boldsymbol{p}^*(a,b) = \max_c \boldsymbol{h}_P[\boldsymbol{r}^*(a,c), \boldsymbol{s}^*(c,b)], \tag{3}$$

or

$$\boldsymbol{p}^*(a,b) = \max_{\varrho} \boldsymbol{h}_P[\varrho, a, c, \boldsymbol{s}^*(c,b)], \tag{4}$$

or

$$\boldsymbol{p}^*(a,b) = \max_{\vartheta} \boldsymbol{h}_P[\boldsymbol{r}^*(a,c), \vartheta, c, b], \tag{5}$$

where $\boldsymbol{h}_P$ is the return composition function by which the return $\boldsymbol{p}(a,\pi)$ from the composite process $P(A,B)$ is computed using information about the component processes $R(A,C)$ and $S(C,B)$. Note that $\boldsymbol{p}^*(a,b)$ is *defined* in (1). Thus, (3), (4), and (5) are formally stated postulates about *properties* of the return composition functions $\boldsymbol{h}_P$. When $\boldsymbol{h}_P$ possesses the postulated property, then the corresponding equation—(3), (4), or (5)—may be used as a recursion relation for the construction of optimal serial processes.

The differences in (3), (4), and (5) rest largely in the regions which must be searched to locate the optimal return $\boldsymbol{p}^*(a,b)$. Equation (3) assumes that the optimal serial process may be constructed using optimal processes for both components. All that is required to find the optimal return $\boldsymbol{p}^*(a,b)$ are the two sets of optimal returns, $[\boldsymbol{r}^*(a,c) | c \epsilon C]$ and $[\boldsymbol{s}^*(c,b) | c \epsilon C]$, with the maximization over the region of all $c \epsilon C$.

The somewhat weaker requirements (4) and (5) employ one optimal component process, with the other component process not necessarily optimal. In (4), for each decision function $\varrho \epsilon R$ the output $c = \varrho(a)$ is computed, and it is then assumed that an optimal process $S^*(c,b)$ will be used from c to b. Thus, maximization is over all $\varrho \epsilon R$. In equation (5), only optimal processes $R^*(a,c)$ are investigated from a to each $c \epsilon C$; then, however, all decision functions for which $\vartheta(c) = b$ must be explored to locate the optimal composite return and process.

In all cases, the optimal return $\boldsymbol{p}^*(a,b)$ and the associated optimal process $P^*(a,b)$ will be found for all $(a,b) \epsilon A \times B$. These optimal returns

and processes may then be further employed in recursion relations such as (4), (5), and (6) to obtain new optimal returns and processes. The great advantage of this procedure, of course, is that the size of the region which must be searched to find the optimal process does not increase with the addition of more stages to the process.

RECURSIVE COMPOSITION OF OPTIMAL PARALLEL PROCESSES

IN PARALLEL composition, as in serial composition, the dimensionality of the policy space increases directly with the number of component processes. Unfortunately, however, the dimensionality of the state-space also increases (and quite rapidly!) with the number of parallel component processes. Thus, special and quite strong assumptions must be made to limit the extent of the regions that must be searched to locate an optimal parallel process. Our general strategy will be to postulate the existence of (possibly hypothetical) processes that will map the input and output of the parallel process onto smaller sets.

Let $S(B,C)$ be a process with composite input $B=B'\times B''$ and assume there exists a process $R(A,B)$ with dimensionality of A less than B. Then the serial composition $P(A,C)=R(A,B)\cdot S(B,C)$ effectively reduces the dimensionality of the input, provided of course that the process $P(A,C)$ continues to provide a reasonable representation for the situation under study. Examples of processes $R(A,B)$ might be the separation of an input A into its constituents B' and B'' or the allocation of a resource $A=B'+B''$ to competing activities. Optimal processes $P^*(a,c)$ may be developed for all $(a,c)\epsilon A\times C$ using the methods of optimal serial composition detailed in the preceding section.

In similar fashion, a process $R(A,B)$ with composite output $B=B'\times B''$ may be combined with a process $S(B,C)$ to form the serial process $P(A,C)=R(A,B)\cdot S(B,C)$ to reduce the dimensionality of the output (assuming C to be smaller than B). Assembly activities are examples of the process $S(B,C)$. Again, the methods of optimal serial composition may be used to construct optimal processes $P^*(a,c)$.

The above devices may be applied in an obvious way to a parallel process, say $R(B,C)$ with $B=B'\times B''$ and $C=C'\times C''$. First the process $P(A,C)=S(A,B)\cdot R(B,C)$ is formed and the optimal processes $P^*(a,c)$ constructed; then the process $W(A,D)=P(A,C)\cdot T(C,D)$ is formed and the optimal processes $W^*(a,d)$ constructed. The procedure may be employed recursively to reduce a multibranch parallel process to a much simpler optimal process. Although not all parallel processes fall within the scope of this method, it provides a very powerful tool for optimal synthesis when applicable.

REDUCING THE DIMENSIONALITY OF STATE-SPACE

THROUGHOUT the preceding, optimal processes have been defined from input elements to output elements over the input-output product space. The dimensionality of the required state-space may be reduced (at the expense of flexibility) by defining optimal processes from and/or to *sets* rather than elements. Thus, for real-valued returns we may define the following optimal returns:

$$\boldsymbol{p}^*(A,b)=\max[\boldsymbol{p}^*(a,b)|a\epsilon A], \tag{6}$$

$$\boldsymbol{p}^*(a,B)=\max[\boldsymbol{p}^*(a,b)|b\epsilon B], \tag{7}$$

$$\boldsymbol{p}^*(A,B)=\begin{cases}\max[\boldsymbol{p}^*(a,b)|(a,b)\epsilon A\times B], & (8)\\ \max[\boldsymbol{p}^*(A,b)|b\epsilon B], & (9)\\ \max[\boldsymbol{p}^*(a,B)|a\epsilon A], & (10)\end{cases}$$

with the equivalence of (8), (9), and (10) readily established.

The corresponding optimal processes $P^*(A,b)$, $P^*(a,B)$, and $P^*(A,B)$ are defined in the same fashion as (2)—i.e., $P^*(A,B)=[\pi|\boldsymbol{p}(a,\pi)=\boldsymbol{p}^*(A,B)]$. For given A and B, $\boldsymbol{p}^*(A,b)$ is a function only of b and $\boldsymbol{p}^*(a,B)$ is a function only of a, with $\boldsymbol{p}^*(A,B)$ a fixed quantity; similar remarks apply to the corresponding optimal processes. Thus, the size of the state-space over which the optimal returns and processes are described has been significantly reduced.

Using the above definitions, a number of additional recursive properties may be postulated for the serial return composition function $\boldsymbol{h}_P$ associated with the serial composition $P(A,B)=R(A,C)\cdot S(C,B)$:

$$\boldsymbol{p}^*(A,b)=\max_c\boldsymbol{h}_P[\boldsymbol{r}^*(A,c),\boldsymbol{s}^*(c,b)], \tag{11}$$

or

$$\boldsymbol{p}^*(A,b)=\max_{\mathfrak{d}}\boldsymbol{h}_P[\boldsymbol{r}^*(A,c),\mathfrak{d},c,b], \tag{12}$$

$$\boldsymbol{p}^*(a,B)=\max_c\boldsymbol{h}_P[\boldsymbol{r}^*(a,c),\boldsymbol{s}^*(c,B)], \tag{13}$$

or

$$\boldsymbol{p}^*(a,B)=\max_{\varrho}\boldsymbol{h}_P[\varrho,a,c,\boldsymbol{s}^*(c,B)], \tag{14}$$

$$\boldsymbol{p}^*(A,B)=\max_c\boldsymbol{h}_P[\boldsymbol{r}^*(A,c),\boldsymbol{s}^*(c,B)]. \tag{15}$$

Remarks similar to those following (3), (4), and (5) also apply here, with obvious modifications. Some important differences may be noted, however. Since $\boldsymbol{p}^*(A,b)$ is a function only of b, recursive optimization methods cannot generally be used for any serial composition $T(D,A)\cdot P(A,B)$. Similarly, since $\boldsymbol{p}^*(a,B)$ is a function only of a, the process $P(A,B)$ cannot generally enter into a serial composition of the form $P(A,B)\cdot T(B,D)$. Also, (15) is really a terminal composition rather than a recursion relation since the optimal process $P^*(A,B)$ will generally not be conformable with any other process of interest.

It may be noted that (11) through (14) are precisely the properties postulated in dynamic programming. Indeed, (11) through (14) are simply transcriptions of the functional equation of dynamic programming into our somewhat more general notation. The recursive procedure is, of course, the same under either interpretation.

In application, (11) and (12) permit the recursive construction of a optimal process by starting at the *initial* input to the process and successively adding stages to the output of the preceding stage. On the other hand, (13) and (14) may be used when starting construction of the optimal process with the *final* output of the process and successively adding stages to the input of the preceding stage. Equation (15) may be used to join two process segments, one constructed using (11) or (12) and the other using (13) or (14); no further additions can normally be made to the resulting process, however, since both the input and the output are fixed.

In all cases, the reduction in the dimensionality of the state-space makes relations (11) through (15) much preferred over (3), (4), and (5), when the former are applicable. However, (3), (4), or (5) must be used in most cases involving processes with several loops and/or parallel branches.

SOME SUFFICIENT CONDITIONS FOR OPTIMAL COMPOSITION

We now present several results that are useful in determining whether a given return function h_P meets the conditions imposed by the recursion formulas (3) through (15).

Lemma. *Let $X=(x)$ be an arbitrary set and let $Y_i(x)=(y_i)$, $i=1, \cdots, n$, be a finite sequence of arbitrary sets whose definitions may depend on $x\epsilon X$. Let $g_i(x,y_i)$, $i=1, \cdots, n$, be a sequence of real-valued functions for which $g_i^*(x)=\max[g_i(x,y_i)|y_i\epsilon Y_i(x)]$ exists for $i=1, \cdots, n$ and all $x\epsilon X$. Let $f(x)$ be an arbitrary function and let $G(x,y_1, \cdots, y_n)=F[f(x),g_1(x,y_1), \cdots, g_n(x,y_n)]$ be a real-valued function for which $G^*=\max[G(x,y_1, \cdots, y_n)|(x,y_1, \cdots, y_n)\epsilon\bigcup_{x\epsilon X}x\times Y_1(x)\times\cdots\times Y_n(x)]$ exists and $F^*(x)=F[f(x),g_1^*(x), \cdots, g_n^*(x)]$ exists for all $x\epsilon X$. If $F^*(x)\geqq G(x,y_1, \cdots, y_n)$ for every $(x, y_1, \cdots, y_n)\epsilon\bigcup_{x\epsilon X}x\times Y_1(x)\times\cdots\times Y_n(x)$, then $G^*=\max[F^*(x)|x\epsilon X]\equiv F^*$.*

Proof. Assume the contrary—i.e., suppose there exist $x'\epsilon X$ and $y_i'\epsilon Y_i(x)$, $i=1, \cdots, n$, such that $G(x',y_1', \cdots, y_n')>F^*$. But $F^*\geqq F^*(x')$ by definition and $F^*(x')\geqq G(x',y_1', \cdots, y_n')$ by the assumed property of $F^*(x)$. Thus, the contradiction $F^*\geqq G(x',y_1', \cdots, y_n')$ establishes the lemma.

Theorem 1. *If $G(x,y_1, \cdots, y_n)$ is monotonic nondecreasing in each of the $g_i(x,y_i)$, $i=1, \cdots, n$, then $F^*=G^*$.*

The monotonicity postulate implies the property

$$F^*(x)\geqq G(x,y_1, \cdots, y_n)$$

required by the lemma.

THEOREM 2. *If* $\oplus$ *is a composition operator such that* $(u \geqq x)(v \geqq y) \Rightarrow (u \oplus v \geqq x \oplus y)$ *and if* $G(x,y_1, \cdots, y_n) = g_1(x,y_1) \oplus g_2(x,y_2) \oplus \cdots \oplus g_n(x,y_n)$, *then* $F^* = G^*$.

The property of the composition operator $\oplus$ implies the monotonicity condition required by Theorem 1. A number of theorems are obvious consequences of the lemma and theorems 1 and 2. The general form of these theorems is illustrated by

THEOREM 3. *Each of the following is a sufficient condition for* (3) *to hold*:

(*a*) $\boldsymbol{h}_P[\boldsymbol{r}^*(a,c),\boldsymbol{s}^*(c,b)] \geqq \boldsymbol{h}_P[\boldsymbol{r}(a,\varrho),\boldsymbol{s}(c,\delta)]$ *for all* ϱ *and* δ *such that* $\varrho(a) = c$ *and* $\delta(c) = b$;

(*b*) $\boldsymbol{h}_P[\boldsymbol{r}(a,\varrho),\boldsymbol{s}(c,\delta)]$ *is monotonic nondecreasing in* $\boldsymbol{r}(a,\varrho)$ *and* $\boldsymbol{s}(c,\delta)$ *for all* ϱ *and* δ *such that* $\varrho(a) = c$ *and* $\delta(c) = b$;

(*c*) $\boldsymbol{h}_P[\boldsymbol{r}(a,\varrho),\boldsymbol{s}(c,\delta)] = \boldsymbol{r}(a,\varrho) \oplus \boldsymbol{s}(c,\delta)$ *with* $\oplus$ *having the properties required in Theorem* 2.

Conditions of this form will frequently provide a simple test to determine whether the return composition function $\boldsymbol{h}_P$ meets the conditions imposed by a specific recursion relation.

APPROXIMATIONS IN STATE-SPACE

WE NOW undertake a brief development of some methods of successive approximation that may prove useful in cases where the process state-space is of such high dimensionality as to preclude the direct application of the methods of the preceding sections. The first two methods have been previously described in the context of dynamic programming in reference 4.

Let $P(A,B)$ be the process under consideration and assume that we seek the optimal process $P^*(A,B)$ from the set A to the set B. The approximation procedures described below construct sequences of subsets $A_i \subset A$ and $B_i \subset B$ $(i = 0, 1, \cdots)$ such that $\boldsymbol{p}^*(A_i,B_i) \leqq \boldsymbol{p}^*(A_{i+1},B_{i+1})$. The subsets A_i and B_i are chosen with two considerations in mind. First, they must be small enough to permit the computation of the optimal returns $\boldsymbol{p}^*(A_i,B_i)$ and the associated optimal processes $P^*(A_i,B_i)$. Second, the sequence of optimal returns $\boldsymbol{p}^*(A_i,B_i)$ and processes $P^*(A_i,B_i)$ should converge to something hopefully approximating the desired optimal return $\boldsymbol{p}^*(A,B)$ and process $P^*(A,B)$. Several intuitively appealing procedures are presented below; we do not, however, attempt the difficult task of precisely delineating the regularity conditions required for the convergence of the procedures to the optimal policies and returns.

The 'coarse grid' procedure utilizes successively finer meshes over successively smaller regions of state-space. Let $A^i \times B^i$ be the region of state-space under exploration at the ith iteration $(A^0 \times B^0 = A \times B)$. The sets A^i and B^i are each partitioned into subsets and a 'representative point' chosen from each subset. The collections of representative points, say A_i and B_i, form the state-space for the optimization computations of the

ith iteration. The optimal input-output pair, say (a_i^*, b_i^*), associated with the optimal return $p^*(A_i, B_i)$ then designates the subsets of A^i and B^i to be explored at the next iteration (i.e., $a_i^* \epsilon A^{i+1} \subset A^i$ and $b_i^* \epsilon B^{i+1} \subset B^i$).

The above method has the advantage of rather rapid convergence, but what it converges to depends very heavily on the fineness of the grid (i.e., on the number of subsets in each partition). Also, each representative point in the A_i and B_i has the same dimensionality as A and B, so that the fineness of the grid is directly dependent on the dimensionality of the state-space $A \times B$—a difficulty which the following procedures circumvent, but at the expense of slower convergence.

One scheme for dealing with state-spaces of high dimensionality is to confine the search to one dimension at a time. Let $A = A^1 \times \cdots \times A^m$ and $B = B^1 \times \cdots \times B^n$ be input and output product spaces. At the ith iteration an input $a_i = (a_i^1, \cdots, a_i^m)$ and output $b_i = (b_i^1, \cdots, b_i^n)$ are given (a_0 and b_0 are arbitrary initial 'guesses'), and a subspace A^J of A and a subspace B^K of B are chosen. The optimal return $p^*(a_i^1, \cdots, a_i^{J-1}, A^J, a_i^{J+1}, \cdots, a_i^m; b_i^1, \cdots, b_i^{K-1}, B^K, b_i^{K+1}, \cdots, b_i^n)$ is computed and the associated optimal input-output pair, say (a^J, b^K), provide the basis for the next iteration—i.e., $a_{i+1}^J = a^J$ and $a_{i+1}^j = a_i^j$ for all $j \neq J$ and similarly $b_{i+1}^K = b^K$ and $b_{i+1}^k = b_i^k$ for all $k \neq K$. A variety of rules have been proposed for selecting the subspace to be explored at each iteration—e.g., in a fixed sequence, at random, etc.,—but in all cases the number of iterations required goes up (roughly) exponentially with the dimensionality of the state-space $A \times B$.

The following procedure, which is related to the gradient methods frequently employed in optimization problems, attempts to increase the rate of convergence by permitting movement in any direction in state-space, but restricts the size of the step that may be made at each iteration. At the ith iteration, let $p^*(a_i, b_i)$ be the optimal return from a specified input a_i to a specified output b_i (a_0 and b_0 are arbitrary initial 'guesses'), and let A_i and B_i be (small) subsets of A and B containing selected sets of 'representative points' in the vicinity of a_i and b_i. The optimal input-output pair, say (a_i^*, b_i^*), associated with the optimal return $p^*(A_i, B_i)$ designates the starting points for the next iteration—i.e., $a_{i+1} = a_i^*$ and $b_{i+1} = b_i^*$. The maximum size of the step at any iteration depends on the size of the 'vicinity' of the given points a_i and b_i, which will in turn depend on the dimensionality of the state-space $A \times B$.

The last method is particularly well suited for use in adaptive systems. The required information concerning decision and return functions in the vicinities A_i and B_i may be generated by perturbing the system about an operating point (a_i, b_i). For real operating systems, the necessary data may thus be developed empirically, or by any desired combination of empirical and theoretical means, in the spirit of the concept of 'evolutionary

operations.' The procedure also gives a method for computing the maximum gradient in systems that can be analyzed as multistage processes. In both of the last two methods, however, the rate of convergence and what the procedure converges to depend directly on the initial 'guess' (a_0,b_0).

REFERENCES

1. R. BELLMAN, *Dynamic Programming*, Princeton University Press, Princeton, New Jersey, 1957.
2. S. KARLIN, "The Structure of Dynamic Programming Models," *Naval Res. Log. Quart.* **2**, 285–294 (1955).
3. A. WALD, *Statistical Decision Functions*, Wiley, New York, 1950.
4. R. BELLMAN AND S. DREYFUS, *Applied Dynamic Programming*, Princeton University Press, Princeton, New Jersey, 1962.