PRIMITIVE DYNAMIC PROGRAMMING

Iwamoto, Seiichi
Departmetn of Economic Engineering, Graduate School of Economics, Kyushu University

https://doi.org/10.5109/12585

出版情報：Bulletin of informatics and cybernetics. 36, pp.163-172, 2004-12. Research Association of Statistical Sciences
バージョン：
権利関係：

PRIMITIVE DYNAMIC PROGRAMMING

by

Seiichi Iwamoto

Reprinted from the Bulletin of Informatics and Cybernetics
Research Association of Statistical Sciences, Vol.36

FUKUOKA, JAPAN
2004

Bulletin of Informatics and Cybernetics, Vol. 36, 2004

PRIMITIVE DYNAMIC PROGRAMMING

By

Seiichi Iwamoto*

Abstract

[1] What is dynamic programming? This paper is concerned with the class of *all* dynamic programmings — with or without optimization. Where is an origin in the class? What is it? The paper is a travel which leads to the origin. We present a primitive form of dynamic programming. It is a non-deterministic dynamic programming, which generates stochastic dynamic programming, which in turn reduces to deterministic dynamic programming. Thus we propose the non-deterministic dynamic programming as a primitive dynamic programming.

Key Words and Phrases: dynamic programming, nondeterministic, stochastic, deterministic

1. INTRODUCTION

Since R. Bellman (1957) has originated dynamic programming (DP), there has been a controversy on "What is DP?" by Aris (1964), Bellman (1968, 1981, 1984, 1986), Denardo (1982), Iwamoto(1987), Nemhauser (1966), Sniedovich (1992) and others. It has at least state, decision, objective function and dynamics. The basic principle is Principle of Optimality (POP). There has been many discussions on POP by Denardo (1982), Nemhauser (1966) and Sniedovich (2002). In general, DP has two types of dynamics: *deterministic* (Denardo (1982), Iwamoto(1987) and Nemhauser (1966)) and *stochastic* (Bellman (1958), Howard (1960), Hinderer (1970) and Puterman (1994)).

On the other hand, the conventional DP method, which is strongly related to invariant imbedding method, can also solve a class of non-optimization problems in Bellman (1968, 1986), Bellman and Denman (1971) and Iwamoto (1991).

In this paper, we climb a mountain — DP — at the base of which both stochastic DP and deterministic DP lie. The moutain commands a view of optimization and non-optimization problems here and there. We see at the top a *primitive* DP without dynamics, which yields both stochastic and deterministic DPs. The primitive DP is *non-deterministic*. It is neither stochastic nor deterministic. However, it generates both. The primitive DP also solves not only the optimization problems but also the non-optimization problems.

In Section 2, we present a non-deterministic DP. It consists of four components: horizon, state, decision and objective function. It has no dynamics. A recursive equation for multiple sums is derived. Two universal rules are derived from the recursive equation for a two-stage DP. One is on optimization. The other is on summation.

* Department of Economic Engineering, Graduate School of Economics, Kyushu University, Fukuoka 812-8581, Japan. tel +81–92–642–2488 iwamoto@en.kyushu-u.ac.jp

[1] This paper was presented at Euro/Informs 2003 (Istanbul).

In Section 3, we introduce a stochastic transition law into the non-derministic DP, which generates a stochastic DP. By incorporating the probability function into the objective function, we derive a recursive equation for expected values. In Section 4, a deterministic DP is derived as a "degenerate" stochastic DP. Thus the deterministic DP is a special form of primitive DP.

We state the contents within a simple framework — disctere and finite state, decision and horizon. However, the objective function is one of the most general functions. It depends on the total history. It has neither separability nor monotonicity.

2. NON-DETERMINISTIC DYNAMIC PROGRAMMING

We consider a primitive form of DP. A *non-deterministic DP* is specified by four tuple (N, X, U, g) :

$$\begin{aligned} &N \geq 2 \text{ is an integer; the } \textit{total number of stages} \\ &X = \{s_1, s_2, \ldots, s_l\} \text{ is a finite } \textit{state space} \\ &U = \{a_1, a_2, \ldots, a_k\} \text{ is a finite } \textit{decision space} \\ &g : H_N \to R^1 \text{ is an } \textit{objective function} \end{aligned}$$

where

$$H_n := X \times U \times X \times U \times \cdots \times U \times X \quad (2n+1)\text{-factors}$$

is a *subhistory space up to n-th stage*. Its element

$$h_n = (x_0, u_0, x_1, u_1, \ldots, u_{n-1}, x_n) \in H_n$$

is a subhistory up to there. Further, for each $n = -1, 0, 1, \ldots, N-1$, the direct product space

$$X^{N-n} := X \times X \times \cdots \times X \quad (N-n)\text{-factors}$$

denotes the set of all *state sequences up to the final N-th stage with length* $(N-n)$:

$$(x_{n+1}, x_{n+2}, \ldots, x_N) \in X^{N-n}.$$

That is,

$$\begin{aligned} X^{N-n} \quad &= \quad X_{n+1} \times X_{n+2} \times \cdots \times X_N \qquad\qquad -1 \leq n \leq N \\ &\qquad\qquad \text{where} \quad X_m := X. \end{aligned}$$

We denote the non-deterministic DP by $\mathcal{N}(N, X, U, g)$ or simply by $\mathcal{N}$.

Now let us introduce a large class of policies for the non-deterministic DP $\mathcal{N}$, which depend on history up to date. A mapping $\nu_n : H_n \to U$ is called *n-th primitive decision function*, whose sequence $\nu = \{\nu_0, \nu_1, \ldots, \nu_{N-1}\}$ is a *primitive policy*. The set of all primitive policies Π_p is called *primitive class*.

Now we see that any primitive policy $\nu \in \Pi_p$ defines a function

$$g^\nu : X^{N+1} \to R^1$$

through

$$g^\nu(x_0, x_1, \ldots, x_N) \quad := \quad g(x_0, u_0, x_1, u_1, \ldots, u_{N-1}, x_N) \tag{1}$$

where the sequence of decisions $u_0, u_1, \ldots, u_{N-1}$ is substituted successively and repeatedly in (1) :

$$\begin{aligned} u_0 = \nu_0(x_0),\ u_1 = \nu_1(x_0, u_0, x_1), \\ \ldots,\ u_{N-1} = \nu_{N-1}(x_0, u_0, x_1, u_1, \ldots, x_{N-1}). \end{aligned} \tag{2}$$

For instance, the cases $N = 2$ and $N = 3$ yield

$$g^{\nu}(x_0, x_1, x_2) \quad = \quad g(x_0, \nu_0(x_0), x_1, \nu_1(x_0, \nu_0(x_0), x_1), x_2)$$

and

$$\begin{aligned} g^{\nu}(x_0, x_1, x_2, x_3) \quad = \quad & g(\, x_0, \nu_0(x_0), x_1, \nu_1(x_0, \nu_0(x_0), x_1), x_2, \\ & \nu_2(x_0, \nu_0(x_0), x_1, \nu_1(x_0, \nu_0(x_0), x_1), x_2)\, , x_3), \end{aligned}$$

respectively.

We evaluate any primitive policy $\nu \in \Pi_p$ at an initial state $x_0 \in X$ by the *multiple sum* of the function $g^{\nu} : X^{N+1} \to R^1$ over the direct product space X^N of the last N state spaces :

$$M^{\nu}_{x_0}[g] \quad := \quad \sum_{(x_1, x_2, \ldots, x_N) \in X^N} \sum \cdots \sum g^{\nu}(x_0, x_1, x_2, \ldots, x_N). \tag{3}$$

Thus the non-deterministic DP $\mathcal{N}(N, X, U, g)$ generates an optimization problem on primitive class. This problem is expressed as follows :

$$\mathrm{P}_0(x_0) \qquad \text{Maximize} \quad M^{\nu}_{x_0}[g] \quad \text{subject to} \quad \nu \in \Pi_p.$$

Which policy is optimal at *any* initial state $x_0 \in X$?

Let $v^0(x_0)$ be the *maximum value* of $\mathrm{P}_0(x_0)$. Then our problem is to find the maximum value function $v^0 : X \to R^1$ and an *optimal policy* $\nu^* \in \Pi_p$:

$$M^{\nu^*}_{x_0}[g] \quad = \quad \operatorname*{Max}_{\nu \in \Pi_p} M^{\nu}_{x_0}[g] \qquad x_0 \in X. \tag{4}$$

2.1. Imvariant Imbedding

Now let us solve the problem through invariant imbedding. Our aim is to find the maximum value funxtion v^0 and an optimal policy ν^* in primitive class Π_p. First for any primitive policy $\nu = \{\nu_0, \nu_1, \ldots, \nu_{N-1}\} \in \Pi_p$, we let $\nu(n) := \{\nu_n, \nu_{n+1}, \ldots, \nu_{N-1}\}$. It is called a *primitive policy from n-th stage on.* The set of all such policies $\Pi_p(n)$ is called *primitive class from there.* Needless to say, $\nu(0) = \nu$ and $\Pi_p(0) = \Pi_p$.

Second we see that any primitive policy from n-th stage on
$\nu = \{\nu_n, \nu_{n+1}, \ldots, \nu_{N-1}\} \in \Pi_p(n)$ defines

$$g^{\nu} : H_n \times X^{N-n} \to R^1$$

through

$$\begin{aligned} & g^{\nu}(h_n, x_{n+1}, x_{n+2}, \ldots, x_N) \\ := \quad & g(h_n, u_n, x_{n+1}, u_{n+1}, \ldots, u_{N-1}, x_N) \end{aligned} \tag{5}$$

where $u_n, u_{n+1}, \ldots, u_{N-1}$ is substituted one by one in (5) :

$$\begin{aligned} u_n = \nu_n(h_n),\ u_{n+1} &= \nu_{n+1}(h_n, u_n, x_{n+1}), \\ &\ldots,\ u_{N-1} = \nu_{N-1}(h_n, u_n, x_{n+1}, u_{n+1}, \ldots, x_{N-1}). \end{aligned} \tag{6}$$

We evaluate any primitive policy $\nu \in \Pi_p(n)$ at subhistory $h_n \in H_n$ by the multiple sum of the function $g^\nu : H_n \times X^{N-n} \to R^1$ over the direct product space X^{N-n} of the remaining $(N-n)$ spaces :

$$M^\nu_{h_n}[g] \ := \sum_{(x_{n+1}, x_{n+2}, \ldots, x_N) \in X^{N-n}} \sum \cdots \sum g^\nu(h_n, x_{n+1}, x_{n+2}, \ldots, x_N). \tag{7}$$

The non-deterministic DP $\mathcal{N}(N, X, U, g)$ generates the family of optimization problems :

$$\mathrm{P}_n(h_n) \qquad \text{Maximize} \quad M^\nu_{h_n}[g] \quad \text{subject to} \quad \nu \in \Pi_p(n)$$
$$h_n \in H_n, \quad 0 \le n \le N.$$

We note that the case $h_0 = x_0 \in X,\ n = 0$ yields the original $\mathrm{P}_0(x_0)$. This is the first step of imbedding.

The second is to derive a relation among adjacent problems. Let $v^n(h_n)$ be the *maximum value* of $\mathrm{P}_n(h_n)$:

$$v^n(h_n) \ = \ \operatorname*{Max}_{\nu \in \Pi_p(n)} M^\nu_{h_n}[g] \qquad h_n \in H_n \tag{8}$$

where

$$v^N(h_N) := g(h_N) \qquad h_N \in H_N.$$

Then we have a recursive formula between a current value $v^n(h_n)$ and the next value function $v^{n+1} = v^{n+1}(h_{n+1})$:

THEOREM 2.1. (Primitive Recursive Equation)

$$v^n(h) = \operatorname*{Max}_{u \in U} \sum_{y \in X} v^{n+1}(h, u, y) \qquad h \in H_n, \quad 0 \le n \le N\text{–}1 \tag{9}$$

$$v^N(h) = g(h) \qquad h \in H_N. \tag{10}$$

Further, let $\nu^*_n(h)$ denote a maximizer. Then $\nu^* = \{\nu^*_0, \nu^*_1, \ldots, \nu^*_{N-1}\}$ is an optimal policy in primitive class Π_p.

PROOF. It is straightforward.

2.2. Non-optimization dynamic programming

Let us take any *decision-free* objective function

$$g : X^N \to R^1.$$

Then Theorem 2.1 claims that

$$\sum_{(x_1, x_2, \cdots, x_N) \in X^N} \sum \cdots \sum g(x_1, x_2, \cdots, x_N) = \sum_{x_1 \in X} \sum_{x_2 \in X} \cdots \sum_{x_N \in X} g(x_1, x_2, \cdots, x_N). \tag{11}$$

2.3. Two universal rules

First let us consider a two-stage model $N = 2$ for non-deterministic DP $\mathcal{N}(N, X, U, g)$. Then the objective function is

$$g : X{\times}U{\times}X{\times}U{\times}X \to R^1.$$

For any primitive policy $\nu = \{\nu_0, \nu_1\} \in \Pi_p$ we see that

$$g^\nu(x, y, z) \quad = \quad g(x, u, y, v, z) \quad \text{where} \quad u = \nu_0(x), \quad v = \nu_1(x, u, y),$$

namely,

$$g^\nu(x, y, z) \quad = \quad g(x, \nu_0(x), y, \nu_1(x, \nu_0(x), y), z).$$

Then Theorem 2.1 claims that

$$\operatorname*{Max}_{\nu} \sum_{(y,z)\in X\times X} \sum g^\nu(x, y, z) \quad = \quad \operatorname*{Max}_{\nu_0} \sum_{y\in X} \left[\operatorname*{Max}_{\nu_1} \sum_{z\in X} g^\nu(x, y, z) \right]. \tag{12}$$

Now let us show that Eq.(12) yields two universal rules. By reducing the decision space, we have a decision-free function of (y, z)

$$g : X{\times}X \to R^1.$$

Then Eq.(12) claims one universal rule

$$\sum_{(y,z)\in X\times X} \sum g(y, z) \quad = \quad \sum_{y\in X} \sum_{z\in X} g(y, z). \tag{13}$$

On the other hand, let us take a *state-free* function $g = g(u, v) : U{\times}U \to R^1$. Then Eq.(12) becomes

$$\operatorname*{Max}_{(u,\nu)} g(u, \nu(u)) \quad = \quad \operatorname*{Max}_{u} \left[\operatorname*{Max}_{\nu} g(u, \nu(u)) \right]$$

where $\nu : U \to U$. Here we note that

$$\begin{aligned} \operatorname*{Max}_{\nu} g(u, \nu(u)) \quad &= \quad \operatorname*{Max}_{v} g(u, v) \qquad \forall\ u \in U \\ \operatorname*{Max}_{(u,v)} g(u, v) \quad &= \quad \operatorname*{Max}_{(u,\nu)} g(u, \nu(u)). \end{aligned}$$

Thus we have another universal rule

$$\operatorname*{Max}_{(u,v)} g(u, v) \quad = \quad \operatorname*{Max}_{u} \left[\operatorname*{Max}_{v} g(u, v) \right]. \tag{14}$$

Finally we comment on optimization and summation separately. Both operations for any real-valued function yield real values, respectively. Each operation has two-types : in one and one by one. As for optimization, simultaneous versus sequential. Summation is multiple versus iterative. We remark that a simultaneous/multiple operation is equivalent to sequential/iterative operation. Both are universal rules (13),(14). Eq.(14) claims that the simultaneous optimization reduces to the repeated optimization in Hardy, Littlewood and Pólya (1952). Two more sophisticated optimization variants are Maximax Theorem in Iwamoto (1985), Bimax Theorem in Iwamoto (1993, 1994) and Principle of Conditional Optimization in Sniedovich (1992). Eq.(13) claims that the multiple sum reduces to the iterative sum (Iwamoto (1991)).

3. STOCHASTIC DYNAMIC PROGRAMMING

Throughout this section, we impose an additional data, stochastic dynamics, on the primitive DP $\mathcal{N}(N, X, U, g)$. Let $p = \{p_n\}_0^{N-1}$ be a *stochastic dynamics*. Here $p_n = p_n(\cdot|\cdot, \cdot)$ is an *n-th stochastic transition law* :

$$p_n(y|h, u) \geq 0, \qquad \forall (h, u, y) \in H_n \times U \times X,$$

$$\sum_{y \in X} p_n(y|h, u) = 1, \qquad \forall (h, u) \in H_n \times U.$$

Hereafter we use the notation $y \sim p_n(\,\cdot\,|h, u)$, which denotes that a next state y conditioned on a history h up to n-th stage and a decision u on n-th stage appears with probability $p_n(y|h, u)$.

Let an N-stage controlled stochastic process $\{(X_n, U_n)\}$ on the finite state space X and the finite decision space U be under the stochastic transition law p. Then *stochastic DP* on the controlled stochastic process is specified by the five components N, X, U, g, p. It is denoted by $\mathcal{S}(N, X, U, g, p)$ or $\mathcal{S}$.

Now we consider the expected value of the random variable

$$g = g(X_0, U_0, X_1, U_1, \ldots, X_{N-1}, U_{N-1}, X_N)$$

over the stochastic process governed by primitive policy $\nu \in \Pi_p$. First we remark that the probability $P(h_N)$ that the process experiences a history h_N is

$$P(h_N) \quad = \quad p_0(x_1|x_0, u_0)p_1(x_2|h_1, u_1) \cdots p_{N-1}(x_N|h_{N-1}, u_{N-1}). \tag{15}$$

Thus we have the *probability function* over the history space

$$P : H_N \to [0,\, 1]. \tag{16}$$

Second we take the *multiplicative function* $g \cdot P$:

$$g \cdot P : H_N \to R^1. \tag{17}$$

Third we see that the expected value at initial state $x_0 \in X$, $E_{x_0}^{\nu}[g]$, under the stochastic process governed by primitive policy $\nu \in \Pi_p$ is the multiple sum of the multiplicative function $g \cdot P$ over the direct product space X^N :

$$E_{x_0}^{\nu}[g] \quad := \quad M_{x_0}^{\nu}[g \cdot P]. \tag{18}$$

Then the stochastic DP $\mathcal{S}(N, X, U, g, p)$ expresses the optimization problem :

$$\mathrm{S}_0(x_0) \qquad \text{Maximize} \quad E_{x_0}^{\nu}[g] \quad \text{subject to} \quad \nu \in \Pi_p.$$

Let $v_0(x_0)$ denote the maximum value. Now let us find the maximum value function v_0 and an optimal policy ν^* in primitive class Π_p.

3.1. Two Imbeddings

Now we propose two types of invariant imbedding. First we take the primitive DP $\mathcal{N}(N, X, U, g \cdot P)$. Then Theorem 2.1 reduces to the backward relation :

COROLLARY 3.1. (Primitive Recursive Equation)

$$v^n(h) = \underset{u\in U}{\mathrm{Max}} \sum_{y\in X} v^{n+1}(h,u,y), \qquad h \in H_n, \quad 0 \le n \le N\text{–}1, \tag{19}$$

$$v^N(h) = g(h)P(h), \qquad h \in H_N. \tag{20}$$

Let $\nu_n^*(h)$ denote a maximizer in (19). Then the policy $\nu^* = \{\nu_0^*, \nu_1^*, \ldots, \nu_{N-1}^*\}$ is optimal in Π_p.

The second is through the stochastic DP $\mathcal{S}(N, X, U, g, p)$. It becomes as follows. First we define

$$\begin{aligned} & P_n(h_n, u_n, x_{n+1}, u_{n+1}, \ldots, u_{N-1}, x_N) \\ := \quad & p_n(x_{n+1}|h_n, u_n)p_{n+1}(x_{n+2}|h_{n+1}, u_{n+1}) \cdots p_{N-1}(x_N|h_{N-1}, u_{N-1}) \\ & \qquad 0 \le n \le N-1. \end{aligned} \tag{21}$$

We note that $P_0 = P$. Further, let a subhistory up to n-th stage

$$h_n = (x_0, u_0, x_1, u_1, \ldots, u_{n-1}, x_n) \in H_n$$

be given. Then we consider the conditional expected value, $E_{h_n}^{\nu}[g]$, of

$$g = g(X_0, U_0, X_1, U_1, \ldots, X_{N-1}, U_{N-1}, X_N)$$

over the stochastic process governed by primitive policy $\nu \in \Pi_p(n)$. We see that it is the multiple sum of the multiplicative function $g \cdot P_n$ over the direct product space X^{N-n} :

$$E_{h_n}^{\nu}[g] \quad := \quad M_{h_n}^{\nu}[g \cdot P_n]. \tag{22}$$

Now let us imbed the $\mathrm{S}_0(x_0)$ into the family of subproblems :

$$\mathrm{S}_n(h_n) \qquad \text{Maximize} \quad E_{h_n}^{\nu}[g] \quad \text{subject to} \quad \nu \in \Pi_p(n)$$
$$h_n \in H_n, \quad 0 \le n \le N.$$

Let $v_n(h_n)$ be the maximum value, where

$$v_N(h_N) := g(h_N).$$

Then we have the backward formula :

THEOREM 3.2. (Stochastic Recursive Equation)

$$v_n(h) = \underset{u\in U}{\mathrm{Max}} \sum_{y\in X} v_{n+1}(h,u,y)p_n(y|h,u), \qquad h \in H_n, \ 0 \le n \le N\text{–}1, \tag{23}$$

$$v_N(h) = g(h), \qquad h \in H_N. \tag{24}$$

Let $\widetilde{\nu}_n(h)$ denote a maximizer in (23). Then we have an optimal policy $\widetilde{\nu} = \{\widetilde{\nu}_0, \widetilde{\nu}_1, \ldots, \widetilde{\nu}_{N-1}\}$ in primitive class Π_p.

THEOREM 3.3.

$$v^n(h_n) \quad = \quad v_n(h_n)P^n(h_n), \qquad h_n \in H_n,\ 0 \le n \le N \tag{25}$$

where

$$P^n(h_n) \quad = \quad p_0(x_1|x_0,u_0)p_1(x_2|h_1,u_1)\cdots p_{n-1}(x_n|h_{n-1},u_{n-1}), \tag{26}$$
$$P^0(h_0) \quad = \quad 1. \tag{27}$$

Thus we see that both the primitive DP $\mathcal{N}(N,X,U,g\cdot P)$ and the stochastic DP $\mathcal{S}(N,X,U,g,p)$ yields the common optimal value function

$$v^0(x_0) = v_0(x_0), \qquad x_0 \in X.$$

4. DETERMINISTIC DYNAMIC PROGRAMMING

In this section, we take a deterministic dynamics $f = \{f_n\}_0^{N-1}$ in stead of the stochastic transition law $p = \{p_n\}_0^{N-1}$ in the stochastic DP $\mathcal{S}(N,X,U,g,p)$. Here $f_n : H_n{\times}U \to X$ is an *n-th deterministic transition function.*

Given the determinstic dynamics f, we take the the stochastic transition law p as follows :

$$p_n(y|h,u) := \delta_{f_n(h,u)}(y) \qquad \forall (h,u,y) \in H_n{\times}U{\times}X$$

where $\delta_a(\cdot)$ is the Dirac measure :

$$\delta_a(y) = \begin{cases} 1 & y = a \\ 0 & \text{otherwise.} \end{cases}$$

Then a *deterministic DP* with dynamics f is specified by the five components N,X,U,g,f. It is denoted by $\mathcal{D}(N,X,U,g,f)$ or $\mathcal{D}$. It represents the optimization problem :

$$\mathrm{D}_0(x_0) \qquad \text{Maximize} \quad E^\nu_{x_0}[g] \quad \text{subject to} \quad \nu \in \Pi_p.$$

where the objective value is an *expected value* in (22). Since the Markov transition law p degenerates into the determistic dynamics f, we see that the expected value

$$E^\nu_{x_0}[g] \quad = \quad g(x_0,u_0,x_1,u_1,\ldots,u_{N-1},x_N) \tag{28}$$

is uniquely determined through an initial state x_0, a primitive policy ν and the dynamics f. We note that the triplet (x_0,ν,f) determines uniquely a history $h_N = (x_0,u_0,x_1,u_1,\ldots,u_{N-1},x_N)$. In fact, the sequence of decisions $u_0,u_1,\ldots,u_{N-1}$ and the sequence of states $x_1,x_2,\ldots,x_N$ are generated altenatingly as follows :

$$\begin{aligned} &u_0 := \nu_0(x_0), \quad x_1 := f_0(x_0,u_0), \quad u_1 := \nu_1(h_1), \quad x_2 := f_1(h_1,u_1), \\ &u_2 := \nu_2(h_2), \quad x_3 := f_2(h_2,u_2), \quad u_3 := \nu_3(h_3), \quad x_4 := f_3(h_3,u_3), \\ &\qquad\qquad\qquad\qquad \vdots \\ &u_{N-1} := \nu_{N-1}(h_{N-1}), \quad x_N := f_{N-1}(h_{N-1},u_{N-1}). \end{aligned} \tag{29}$$

Now let us imbed the $\mathrm{D}_0(x_0)$ into the family of subproblems :

$$\mathrm{D}_n(h_n) \qquad \text{Maximize} \quad E^{\nu}_{h_n}[g] \quad \text{subject to} \quad \nu \in \Pi_p(n)$$
$$h_n \in H_n, \quad 0 \le n \le N.$$

where the objective value

$$E^{\nu}_{h_n}[g] \quad = \quad g(h_n, u_n, x_{n+1}, u_{n+1}, \ldots, u_{N-1}, x_N) \tag{30}$$

is uniquely determined through the triplet (h_n, ν, f) as follows :

$$\begin{aligned} &u_n := \nu_n(h_n), \quad x_{n+1} := f_n(h_n, u_n), \\ &u_{n+1} := \nu_{n+1}(h_{n+1}), \quad x_{n+2} := f_{n+1}(h_{n+1}, u_{n+1}), \\ &\qquad\qquad \vdots \\ &u_{N-1} := \nu_{N-1}(h_{N-1}), \quad x_N := f_{N-1}(h_{N-1}, u_{N-1}). \end{aligned} \tag{31}$$

Let $w_n(h_n)$ be the maximum value of $\mathrm{D}_n(h_n)$, where

$$w_N(h_N) := g(h_N).$$

Then we have the backward formula :

Corollary 4.1. (Deterministic Recursive Equation)

$$w_n(h) = \operatorname*{Max}_{u \in U} w_{n+1}(h, u, f_n(h, u)), \qquad h \in H_n, \ 0 \le n \le N\text{–}1 \tag{32}$$

$$w_N(h) = g(h), \qquad h \in H_N. \tag{33}$$

Let $\nu^*_n(h)$ denote a maximizer in (32). Then we have an optimal policy $\nu^* = \{\nu^*_0, \nu^*_1, \ldots, \nu^*_{N-1}\}$ in Π_p.

Acknowledgement

The author would like to thank anonymous referee for careful reading and some useful comments.

References

Aris, R. (1964). *Discrete Dynamic Programming*, Braisdell, New York.

Bellman, R.E. (1957). *Dynamic Programming*, Princeton Univ. Press, New Jersey.

Bellman, R.E. (1958). Dynamic programming and stochastic control processes, *Inf. Control* **1**, 228–239.

Bellman, R.E. (1968). *Some Vistas of Modern Mathematics*, University of Kentucky Press, Lexington, KY.

Bellman, R.E. (1981). List of Publications: Richard Bellman, *IEEE Transactions on Automatic Control* **AC-26**, No.5(Oct.), 1213–1223.

Bellman, R.E. (1984). *Eye of the Hurricane: an Autobiography*, World Scientific, Singapore.

Bellman, R.E. (1986). *The Bellman Continuum: A Collection of the Works of Richard E. Bellman* (Roth, R.S. Ed.), World Scientific, Singapore.

Bellman, R.E. and Denman, E.D. (1971). Invariant Imbedding, *Lecture Notes in Operation Research and Mathematical Systems* **52**, Springer-Verlag, Berlin.

Denardo, E.V. (1982). *Dynamic Programming: Models and Applications*, Prentice-Hall, New Jersey.

Hardy, G.H., Littlewood, J.E. and Pólya, G. (1952). *Inequalities, 2nd ed.*, Cambridge Univ. Press.

Hinderer, K. (1970). Foundations of Non-Stationary Dynamic Programming with Discrete Time Parameter, *Lecture Notes in Operation Research and Mathematical Systems* **33**, Springer-Verlag, Berlin.

Howard, R.A. (1960). *Dynamic Programming and Markov Processes*, MIT Press, Cambridge, Mass.

Iwamoto, S. (1985). Sequential minimaximization under dynamic programming structure, *J. Math. Anal. Appl.* **108**, 267–282.

Iwamoto, S. (1987). *Theory of Dynamic Program*, Kyushu Univ. Press, Fukuoka, (*in Japanese*).

Iwamoto, S. (1991). Iterative integral versus dynamic programming, *in "Proceedings of The Fourth BELLMAN Continuum Workshop"*, Kansas State University, 1990, (E. S. Lee, Ed.), *Computers Math. Applic.* **21**, 23–39.

Iwamoto, S. (1993). From dynamic programming to bynamic programming, *J. Math. Anal. Appl.* **177**, 56-74.

Iwamoto, S. (1994). On bidecision processes, *J. Math. Anal. Appl.* **187**, 676-699.

Nemhauser, G.L. (1966). *Introduction to Dynamic Programming*, Wiley, New York.

Puterman, M.L. (1994). *Markov Decision Processes : discrete stochastic dynamic programming*, Wiley & Sons, New York.

Sniedovich, M. (1992). *Dynamic Programming*, Marcel Dekker, New York.

Sniedovich, M. (2002). Eureka! Bellman's principle of optimality is valid! (M. Dror, P. L'Ecuyer and F. Szidarovszky, Ed.) *Modeling Uncertainty, An Examination of Stochastic Theory, Methods, and Applications*, Kluwer, 735–749.

Received October 6, 2003
Revised June 5, 2004