# Mutually Dependent Decision Processes Models

Bull. Kyushu Inst. Tech.
Pure Appl. Math. No. 63, 2016, pp. 15–26

# MUTUALLY DEPENDENT DECISION PROCESSES MODELS

Toshiharu FUJITA

**Abstract**

We introduce a new framework for dynamic programming called mutually dependent decision processes (MDDPs). Each MDDPs model is constructed from two or more finite-stage deterministic decision processes. At each stage, the reward in one process depends on the optimal values of the other processes, whose initial state is determined by the current state and decision of the original process. We formulate the MDDPs models and derive their mutually dependent recursive equations by dynamic programming.

## 1. Introduction

Dynamic programming [1] is a powerful tool for solving various problems. However, it cannot be denied that plenty of problems which cannot be handled by dynamic programming still exist. Here, we propose a novel framework of dynamic programming theory that extends the applicability of dynamic programming methods. In this framework, called mutually dependent decision processes (MDDPs), each MDDPs model is constructed from at least two finite-stage deterministic decision processes. At each stage, the reward in one process depends on the optimal values of the other processes. The initial state is determined by the current state and the decision of the original process. To some extent, the transition structure yielded by our models can be regarded as a nonserial system [10, 2]. However, the emergence of a mutually dependent structure through reward functions is an entirely novel concept. Here, the MDDPs are newly constructed on a nonserial transition system.

Section 2 introduces our basic model involving two decision processes. Each decision process is an ordinary additive process. We also introduce mutual dependency and derive the mutually dependent recursive equations. Section 3 discusses the associative reward systems, whose recursive equations are derived by an invariant embedding technique. Section 4 formulates our general model. Specifically, we state the recursive equations of the MDDPs with more than two processes and generalize the model criteria.

Our models enable easier treatment of some classes of complex multi-stage decision processes.

## 2. Basic model

In this section, we formulate our basic model of MDDPs. This model comprises two additive decision processes; the main-process and sub-process. The main-process

$\mathrm{P}(x_0)$ is formulated as follows:

$$\begin{aligned}
&\text{Maximize} && r(x_0,u_0)+r(x_1,u_1)+\cdots+r(x_{N-1},u_{N-1})+r_G(x_N)\\
&\text{subject to} && x_{n+1}=f_{XX}(x_n,u_n)\quad n=0,1,\ldots,N-1\\
&&& u_n\in U(x_n)\quad n=0,1,\ldots,N-1\\
&&& (N=N(x_0,u_0,x_1,u_1,\ldots)=\max\{n: x_n\notin T_X\}+1),
\end{aligned}$$

where

1. $X$ is a nonempty finite set called the state space and $T_X\subset X$ denotes the terminal state set. The transition is terminated if $x_n\in T_X$. $x_n$ $(\in X)$ represents the state of the process at time $n$, with $n=0,1,\ldots,N$. The initial state $x_0\in X\backslash T_X$ is specified at the beginning of the process.
2. $U$ is a nonempty finite set called the decision space. $u_n$ $(\in U)$ represents the selected action at time $n$, with $n=0,1,\ldots,N-1$. The power set of $U$ is denoted by $2^U$:
$$2^U=\{A:\text{a set}\,|\,A\subset U\}.$$
Furthermore, we denote a point-to-set valued mapping from $X\backslash T_X$ to $2^U\backslash\{\phi\}$ by $U$. $U(x)$, called the feasible decision space, represents the set of all feasible actions in state $x$. Let $G_r(U)$ denote the graph of $U(\cdot)$:
$$G_r(U)=\{(x,u)\,|\,u\in U(x),x\in X\backslash T_X\}.$$
3. $r: G_r(U)\to\mathbf{R}$ is the reward function, where $\mathbf{R}=(-\infty,\infty)$. At each stage, an action $u$ selected in state $x$ confers a reward $r(x,u)$. The function $r_G: X\to\mathbf{R}$ is the terminal reward function.
4. $f_{XX}: X\times U\to X$ is a deterministic transition law. If a process in state $x$ selects action $u$, it deterministically proceeds to the next state $f_{XX}(x,u)$.

Similarly, the sub-process $\mathrm{Q}(y_0)$ is formulated as follows:

$$\begin{aligned}
&\text{Maximize} && q(y_0,v_0)+q(y_1,v_1)+\cdots+q(y_{N-1},v_{N-1})+q_G(y_N)\\
&\text{subject to} && y_{n+1}\sim f_{YY}(y_n,v_n)\quad n=0,1,\ldots,N-1\\
&&& v_n\in V(y_n)\quad n=0,1,\ldots,N-1\\
&&& (N=N(y_0,v_0,y_1,v_1,\ldots)=\max\{n: y_n\notin T_Y\}+1),
\end{aligned}$$

where

1′. $Y$ is a nonempty finite set called the state space, and $T_Y\subset Y$ denotes the terminal state set. The transition is terminated if $y_n\in T_Y$. $y_n$ $(\in Y)$ represents the state of the process at time $n$, with $n=0,1,\ldots,N$. The initial state $y_0\in Y\backslash T_Y$ is specified at the beginning of the process.

$2'$. $V$ is a nonempty finite set called the decision space. $v_n$ $(\in V)$ represents the action chosen at time $n$, $n = 0, 1, \ldots, N-1$. Furthermore, we denote a point-to-set valued mapping from $Y \backslash T_Y$ to $2^V \backslash \{\phi\}$ by $V$. $V(y)$, called the feasible decision space, represents the set of all feasible actions in state $y$.

$3'$. $q : G_r(V) \to \mathbf{R}$ is the reward function. At each stage, an action $v$ selected in state $y$ confers a reward $q(y, v)$. The function $q_G : Y \to \mathbf{R}$ is the terminal reward function.

$4'$. $f_{YY} : Y \times V \to Y$ is a deterministic transition law.

We now introduce two transition laws that connect the state spaces $X$ and $Y$:

$$f_{XY} : X \times U \to Y, \qquad f_{YX} : Y \times V \to X.$$

The initial state of a sub-process problem is given by the transition $f_{XY}$, which depends on the state $x_n$ and decision $u_n$ of the main-process at time $n$. Conversely, the initial state of a main-process problem is given by the transition $f_{YX}$, which depends on the state $y_n$ and decision $v_n$ of the sub-process at time $n$.

The rewards $r$ and $q$ are then defined as follows:

$$r(x,u) = \begin{cases} q_G(y_0) & y_0 = f_{XY}(x,u) \in T_Y, \\ \max\limits_{\substack{y_{n+1}=f_{YY}(y_n,v_n) \\ v_n \in V(y_n) \\ n=0,1,\ldots,N-1}} [q(y_0,v_0) + \cdots + q(y_{N-1},v_{N-1}) + q_G(y_N)] & y_0 = f_{XY}(x,u) \notin T_Y, \end{cases}$$

$$q(y,v) = \begin{cases} r_G(x_0) & x_0 = f_{YX}(y,v) \in T_X, \\ \max\limits_{\substack{x_{n+1}=f_{XX}(x_n,u_n) \\ u_n \in U(x_n) \\ n=0,1,\ldots,N-1}} [r(x_0,u_0) + \cdots + r(x_{N-1},u_{N-1}) + r_G(x_N)] & x_0 = f_{YX}(y,v) \notin T_X. \end{cases}$$

In this fromulation, $r(x,u)$ is the maximum value of the sub-process problem with the corresponding initial state $y_0 = f_{XY}(x,u)$. In particular, when $y_0$ is a terminal state, $r(x,u)$ equals the terminal reward $q_G(y_0)$. Similarly, $q(y,v)$ is the maximum value of the main-process problem with initial state $x_0 = f_{YX}(y,v)$. We assume a finite maximum length of all state sequences along the alternating processes.

The goal is to get the maximum value of the main-process problem $\mathrm{P}(\bar{x}_0)$, where $\bar{x}_0 \in X \backslash T_X$ is a given initial state. This problem is denoted by $(\mathrm{P}, \mathrm{Q}, \bar{x}_0)$.

We now give recursive equations for $(\mathrm{P}, \mathrm{Q}, \bar{x}_0)$. Each process is an ordinary additive decision process. Therefore, both processes are treated in the standard way [3]. The optimal values $v(x_0)$ of the main-process with initial state $x_0 \in X$ are given by

$$v(x_0) = r_G(x_0) \qquad x_0 \in T_X,$$

$$v(x_0) = \max_{\substack{x_{n+1}=f_{XX}(x_n,u_n) \\ u_n \in U(x_n) \\ n=0,1,\ldots,N-1}} [r(x_0,u_0) + \cdots + r(x_{N-1},u_{N-1}) + r_G(x_N)] \qquad x_0 \notin T_X,$$

and the recursive equation is obtained as follows:

**Recursive equation (main-process)**

$$v(x) = r_G(x) \qquad x \in T_X,$$

$$v(x) = \max_{u \in U(x)} [r(x,u) + v(f_{XX}(x,u))] \qquad x \notin T_X.$$

Similarly, the optimal value $w(y_0)$ of the sub-process with initial states $y_0 \in Y$ is computed as:

$$w(y_0) = q_G(y_0) \qquad y_0 \in T_Y,$$

$$w(y_0) = \max_{\substack{y_{n+1}=f_{YY}(y_n,v_n) \\ v_n \in V(y_n) \\ n=0,1,\ldots,N-1}} [q(y_0,v_0) + \cdots + q(y_{N-1},v_{N-1}) + q_G(y_N)] \qquad y_0 \notin T_Y.$$

and the recursive equation is given as follows:

**Recursive equation (sub-process)**

$$w(y) = q_G(y) \qquad y \in T_Y,$$

$$w(y) = \max_{v \in V(y)} [q(y,v) + w(f_{YY}(y,v))] \qquad y \notin T_Y.$$

Furthermore, specifying the reward functions $r$ and $q$ as $w$ and $v$ respectively, we obtain

$$r(x,u) = w(f_{XY}(x,u)), \qquad q(y,v) = v(f_{YX}(y,v)).$$

These formulations are collected into the following theorem.

THEOREM 2.1. *We have the following mutually dependent recursive equations.*

$$v(x) = r_G(x) \qquad x \in T_X,$$

$$v(x) = \max_{u \in U(x)} [w(f_{XY}(x,u)) + v(f_{XX}(x,u))] \qquad x \notin T_X,$$

$$w(y) = q_G(y) \qquad y \in T_Y,$$

$$w(y) = \max_{v \in V(y)} [v(f_{YX}(y,v)) + w(f_{YY}(y,v))] \qquad y \notin T_Y.$$

*The above recursive equations yield the desired optimal value* $v(\bar{x}_0)$ *of* $(\mathrm{P}, \mathrm{Q}, \bar{x}_0)$.

Then, letting

$$\pi_X^*(x) \in \operatorname*{argmax}_{u \in U(x)} [w(f_{XY}(x,u)) + v(f_{XX}(x,u))] \qquad x \notin T_X,$$

$$\pi_Y^*(y) \in \operatorname*{argmax}_{v \in V(y)} [v(f_{YX}(y,v)) + w(f_{YY}(y,v))] \qquad y \notin T_Y,$$

we obtain a pair of optimal Markov policies $(\pi_X^*, \pi_Y^*)$ for the target MDDPs problem $(\mathrm{P}, \mathrm{Q}, \bar{x}_0)$.

## 3. Associative reward

We now introduce associative reward systems into the MDDPs. The notations are those of the preceding section, and sets $C$ and $D$ are subsets of $\mathbf{R}$. We note that decision process with additive reward system has an optimal solution in Markov policy class [3]. However, a decision process with an associative reward system might yield no optimal solution in the Markov policy class. By generalizing the policy class, we can guarantee an optimal policy in any associative reward system [4, 5, 7, 8].

In the main-process, $\mathrm{P_A}(x_0)$ and $\mathrm{Q_A}(y_0)$ are respectively given as follows:

$$\begin{aligned} \mathrm{P_A}(x_0) \qquad & \text{Maximize} \quad r(x_0,u_0)\circ r(x_1,u_1)\circ\cdots\circ r(x_{N-1},u_{N-1})\circ r_G(x_N) \\ & \text{subject to} \quad x_{n+1}=f_{XX}(x_n,u_n) \quad n=0,1,\ldots,N-1 \\ & \qquad\qquad \sigma=(\sigma_0,\sigma_1,\ldots,\sigma_{N-1})\in\Sigma \\ & \qquad\qquad (N=N(x_0,u_0,x_1,u_1,\ldots)=\max\{n: x_n\notin T_X\}+1), \end{aligned}$$

where

5. $\circ : C\times C\to C$ is a binary operator that satisfies the associative law:

$$a\circ(b\circ c)=(a\circ b)\circ c \qquad a,b,c\in C.$$

We assume that there exists a left identity element $\tilde{\lambda}$ in $C$:

$$\tilde{\lambda}\circ a=a \qquad a\in D.$$

6. $\sigma_n : X^{n+1}\to U$, $n=0,1,\ldots,N-1$ is an $n$th general decision function that gives decision $u_n=\sigma_n(x_0,x_1,\ldots,x_n)\in U(x_n)$ at time $n$. Therefore, each decision at time $n$ depends on the sequence of states up to time $n$. The sequence $\sigma=(\sigma_0,\sigma_1,\ldots,\sigma_{N-1})$ is called a general policy and the set of all general policies is denoted by $\Sigma$.

$$\begin{aligned} \mathrm{Q_A}(y_0) \qquad & \text{Maximize} \quad q(y_0,v_0)\bullet q(y_1,v_1)\bullet\cdots\bullet q(y_{N-1},v_{N-1})\bullet q_G(y_N) \\ & \text{subject to} \quad y_{n+1}\sim f_{YY}(y_n,v_n) \quad n=0,1,\ldots,N-1 \\ & \qquad\qquad \gamma=(\gamma_0,\gamma_1,\ldots,\gamma_{N-1})\in\Gamma \\ & \qquad\qquad (N=N(y_0,v_0,y_1,v_1,\ldots)=\max\{n: y_n\notin T_Y\}+1), \end{aligned}$$

where

5′. $\bullet : D\times D\to D$ is a binary operator satisfying the associative law. We assume that $D$ contains a left identity element $\tilde{\mu}$.

6′. $\gamma_n : Y^{n+1} \to U$, $n = 0, 1, \ldots, N-1$ is an $n$th general decision function that gives decision $v_n = \gamma_n(y_0, y_1, \ldots, y_n) \in V(y_n)$ at stage $n$. The set of all general policies is denoted by $\Gamma$.

The reward functions are given as follows:

$$r(x,u) = \begin{cases} q_G(y_0) & y_0 = f_{XY}(x,u) \in T_Y, \\ \max\limits_{\substack{\gamma \in \Gamma, y_{n+1} = f_{YY}(y_n, v_n) \\ n=0,1,\ldots,N-1}} [q(y_0, v_0) \bullet \cdots \bullet q(y_{N-1}, v_{N-1}) \bullet q_G(y_N)] & y_0 = f_{XY}(x,u) \notin T_Y, \end{cases}$$

$$q(y,v) = \begin{cases} r_G(x_0) & x_0 = f_{YX}(y,v) \in T_X, \\ \max\limits_{\substack{\sigma \in \Sigma, x_{n+1} = f_{XX}(x_n, u_n) \\ n=0,1,\ldots,N-1}} [r(x_0, u_0) \circ \cdots \circ r(x_{N-1}, u_{N-1}) \circ r_G(x_N)] & x_0 = f_{YX}(y,v) \notin T_X. \end{cases}$$

We now introduce the recursive equations of $(\mathrm{P_A}, \mathrm{Q_A}, \bar{x}_0)$. First we imbed the main-process problem $\mathrm{P_A}(x_0)$ into the following problem with a parameter $\lambda \in C$ and its optimal value is denoted by $V(x_0, \lambda)$ as follows:

$$V(x_0, \lambda) = \lambda \circ r_G(x_0) \qquad x_0 \in T_X,\ \lambda \in C,$$

$$V(x_0, \lambda) = \max_{\substack{\sigma \in \Sigma, x_{n+1} = f_{XX}(x_n, u_n) \\ n=0,1,\ldots,N-1}} [\lambda \circ r(x_0, u_0) \circ \cdots \circ r(x_{N-1}, u_{N-1}) \circ r_G(x_N)] \qquad x_0 \notin T_X,\ \lambda \in C.$$

Similarly, we consider the following imbedded problem with a parameter $\mu \in D$ for the sub-process with initial state $y_0 \in Y$ and optimal value function $W$.

$$W(y_0, \mu) = \mu \bullet q_G(y_0) \qquad y_0 \in T_Y,\ \mu \in D,$$

$$W(y_0, \mu) = \max_{\substack{\gamma \in \Gamma, y_{n+1} = f_{YY}(y_n, v_n) \\ n=0,1,\ldots,N-1}} [\mu \bullet q(y_0, v_0) \bullet \cdots \bullet q(y_{N-1}, v_{N-1}) \bullet q_G(y_N)] \qquad y_0 \notin T_Y,\ \mu \in D.$$

The recursive equations of the imbedded problems are obtained as follows [6, 8]:

**Recursive Equation (Imbedded Main-process)**

$$V(x, \lambda) = \lambda \circ r_G(x) \qquad x \in T_X,\ \lambda \in C,$$

$$V(x, \lambda) = \max_{u \in U(x)} [V(f_{XX}(x,u), \lambda \circ r(x,u))] \qquad x \notin T_X,\ \lambda \in C. \tag{1}$$

**Recursive Equation (Imbedded Sub-process)**

$$W(y, \mu) = \mu \bullet q_G(y) \qquad y \in T_Y,\ \mu \in D,$$

$$W(y, \mu) = \max_{v \in V(y)} [W(f_{YY}(y,v), \mu \bullet q(y,v))] \qquad y \notin T_Y,\ \mu \in D. \tag{2}$$

These formulations are collected into the following theorem.

THEOREM 3.1. *The mutually dependent recursive equations for* $(P_A, Q_A, \bar{x}_0)$ *are given by*

$$V(x,\lambda) = \lambda \circ r_G(x) \qquad x \in T_X,\ \lambda \in C,$$

$$V(x,\lambda) = \max_{u \in U(x)} [V(f_{XX}(x,u), \lambda \circ W(f_{XY}(x,u), \tilde{\mu}))] \qquad x \notin T_X,\ \lambda \in C, \tag{3}$$

$$W(y,\mu) = \mu \bullet q_G(y) \qquad y \in T_Y,\ \mu \in D,$$

$$W(y,\mu) = \max_{v \in V(y)} [W(f_{YY}(y,v), \mu \bullet V(f_{YX}(y,v), \tilde{\lambda}))] \qquad y \notin T_Y,\ \mu \in D. \tag{4}$$

*The optimal value of* $(P_A, Q_A, \bar{x}_0)$ *is given by* $V(\bar{x}_0, \tilde{\lambda})$.

PROOF. It is easy to show that $V(x_0, \tilde{\lambda})$ and $W(y_0, \tilde{\mu})$ are the optimal values of the original main-process problem $P_A(x_0)$ and original sub-process problem $Q_A(y_0)$, respectively. Thus, from the definition of the reward functions, we have $r(x,y) = W(f_{XY}(x,u), \tilde{\mu})$ and $q(y,v) = V(f_{YX}(y,v), \tilde{\lambda})$. Therefore, Eqs. (1) and (2) are obviously equivalent to Eqs. (3) and (4), respectively. □

Moreover, letting

$$\pi_X^*(x,\lambda) \in \operatorname*{argmax}_{u \in U(x)} [V(f_{XX}(x,u), \lambda \circ W(f_{XY}(x,u), \tilde{\mu}))] \qquad x \notin T_X,\ \lambda \in C$$

and

$$\pi_Y^*(y,\mu) \in \operatorname*{argmax}_{v \in V(y)} [W(f_{YY}(y,v), \mu \bullet V(f_{YX}(y,v), \tilde{\lambda}))] \qquad y \notin T_Y,\ \mu \in D,$$

we obtain a pair of parameterized optimal Markov policies $(\pi_X^*, \pi_Y^*)$ for the imbedded MDDPs.

Optimal general policies $\sigma^* = \{\sigma_0^*, \sigma_1^*, \ldots, \sigma_{N-1}^*\}$ and $\gamma^* = \{\gamma_0^*, \gamma_1^*, \ldots, \gamma_{N-1}^*\}$ for $(P_A, Q_A, \bar{x}_0)$ are then constructed from $(\pi_X^*, \pi_Y^*)$ by the following procedures.

PROCEDURE A. Let $x_0 = \bar{x}_0$ and execute Procedure B with $x_0$.

PROCEDURE B (Input: initial state $x_0$). Let $\lambda_0 = \tilde{\lambda}$ and put

$$\sigma_0^*(x_0) = \pi_X^*(x_0, \lambda_0).$$

Next, let $x_1 = f_{XX}(x_0, \sigma_0^*(x_0))$, $\lambda_1 = \lambda_0 \circ r(x_0, \sigma_0^*(x_0))$ and put

$$\sigma_1^*(x_0, x_1) = \pi_X^*(x_1, \lambda_1).$$

Generally, for $n = 1, 2, 3, \ldots$ such that $(x_0, x_1, \ldots, x_{n-1}) \in (X \backslash T_X)^n$,

$$x_n = f_{XX}(x_{n-1}, \sigma^*_{n-1}(x_0, x_1, \ldots, x_{n-1})),$$
$$\lambda_n = \lambda_{n-1} \circ r(x_{n-1}, \sigma^*_{n-1}(x_0, x_1, \ldots, x_{n-1})),$$
$$\sigma^*_n(x_0, x_1, \ldots, x_n) = \pi^*_X(x_n, \lambda_n).$$

For each state sequence $x_0, x_1, \ldots, x_m$ $(m = 0, 1, \ldots)$ generated in this procedure, if $y_0 = f_{XY}(x_m, \sigma^*_m(x_0, x_1, \ldots, x_m))$ is not a terminal state, then execute Procedure C with $y_0$.

PROCEDURE C (Input: initial state $y_0$). Let $\mu_0 = \tilde{\mu}$ and put

$$\gamma^*_0(y_0) = \pi^*_Y(y_0, \mu_0).$$

For $n = 1, 2, 3, \ldots$ such that $(y_0, y_1, \ldots, y_{n-1}) \in (Y \backslash T_Y)^n$,

$$y_n = f_{YY}(y_{n-1}, \gamma^*_{n-1}(y_0, y_1, \ldots, y_{n-1})),$$
$$\mu_n = \mu_{n-1} \bullet q(y_{n-1}, \gamma^*_{n-1}(y_0, y_1, \ldots, y_{n-1})),$$
$$\gamma^*_n(y_0, y_1, \ldots, y_n) = \pi^*_Y(y_n, \mu_n).$$

For each state sequence $y_0, y_1, \ldots, y_m$ $(m = 0, 1, \ldots)$ generated in this procedure, if $x_0 = f_{YX}(y_m, \gamma^*_m(y_0, y_1, \ldots, y_m))$ is not a terminal state, then execute Procedure B with $x_0$.

Note that Procedures B and C are recursively executed until the terminal state is reached.

## 4. General model

In this section, we generalize the above model to more than two decision processes. Furthermore, all objective functions are functions of an associative reward, and each reward function of one process is a function of the optimal values of the other decision processes. The first process $\mathrm{P}_1(x_0)$ $(x_0 \in X_1 \backslash T_1)$ and the $i$th process $\mathrm{P}_i(x_0, c)$ $(x_0 \in X_i \backslash T_i,\ c \in D_i,\ i = 2, 3, \ldots, m)$ are respectively given by

$\mathrm{P}_1(x_0)$ Maximize $g_1(r_1(x_0, u_0) \circ_1 r_1(x_1, u_1) \circ_1 \cdots \circ_1 r_1(x_{N-1}, u_{N-1}) \circ_1 k_1(x_N))$

subject to $x_{n+1} = f_{11}(x_n, u_n) \quad n = 0, 1, \ldots, N-1$

$\sigma_1 = (\sigma_{10}, \sigma_{11}, \ldots, \sigma_{1(N-1)}) \in \Sigma_1$

$(N = N(x_0, u_0, x_1, u_1, \ldots) = \max\{n : x_n \notin T_1\} + 1),$

$$\begin{aligned} \mathrm{P}_i(x_0, c) \qquad & \text{Maximize} \quad g_i(c, r_i(x_0, u_0) \circ_i r_i(x_1, u_1) \circ_i \cdots \circ_i r_i(x_{N-1}, u_{N-1}) \circ_i k_i(x_N)) \\ & \text{subject to} \quad x_{n+1} = f_{ii}(x_n, u_n) \quad n = 0, 1, \ldots, N-1 \\ & \qquad\qquad \sigma_i = (\sigma_{i0}, \sigma_{i1}, \ldots, \sigma_{i(N-1)}) \in \Sigma_i \\ & \qquad\qquad (N = N(x_0, u_0, x_1, u_1, \ldots) = \max\{n : x_n \notin T_i\} + 1), \end{aligned}$$

where the components of the $i$th process $(i = 1, 2, \ldots, m)$ are defined as follows:

i. $X_i$ (a nonempty finite set) is the state space and $T_i \subset X_i$ denotes the terminal state set.

ii. $U_i$ (a nonempty finite set) is the decision space. $u_n$ $(\in U_i)$ represents the selected action at time $n$, with $n = 0, 1, \ldots, N-1$. Furthermore, we denote a point-to-set valued mapping from $X_i \backslash T_i$ to $2^{U_i} \backslash \{\phi\}$ by $U_i$. $U_i(x)$ is called the feasible decision space.

iii. The functions $r_i : G_r(U_i) \to D_i$ and $k_i : X_i \to D_i$ are the reward and terminal reward functions, respectively, where $D_i \subset \mathbf{R}$.

iv. $f_{ii} : X_i \times U_i \to X_i$ is a deterministic transition law.

v. The operator $\circ_i : D_i \times D_i \to D_i$ is an associative binary operator:

$$a \circ_i (b \circ_i c) = (a \circ_i b) \circ_i c \qquad a, b, c \in D_i.$$

The binary operator is assumed to have a left identity element $e_i$:

$$e_i \circ_i a = a \qquad a \in D_i.$$

The function $g_i : D_i \to \mathbf{R}$ is a utility function.

vi. $\sigma_{in} : X_i^{n+1} \to U_i$, $n = 0, 1, \ldots, N-1$ is the $n$th general decision function. The sequence $\sigma_i = (\sigma_{i0}, \sigma_{i1}, \ldots, \sigma_{i(N-1)})$ is called a general policy and the set of all general policies is denoted by $\Sigma_i$.

The optimal values of $\mathrm{P}_1(x_0)$ $(x_0 \in X_1 \backslash T_1)$ and $\mathrm{P}_i(x_0, c)$ $(x_0 \in X_i \backslash T_i,\ c \in D_i,\ i = 2, 3, \ldots, m)$ are denoted by $V_1(x_0)$ and $V_i(x_0, c)$, respectively. Especially, if $x_0$ is a terminal state of the $i$th process, we have

$$V_1(x_0) = g_1(k_1(x_0)), \qquad V_i(x_0, c) = g_i(c, k_i(x_0)) \qquad i = 2, 3, \ldots, m.$$

The following transition laws connect the state space $X_i$ to state space $X_j$:

$$f_{ij} : X_i \times U_i \to X_j \qquad i, j = 1, 2, \ldots, m;\ i \neq j.$$

Then, the reward function for the $i$th process is given by

$$\begin{aligned} r_i(x, u) = R_i(x, u, V_1(f_{i1}(x, u)), V_2(f_{i2}(x, u), c_{i2}(x, u)), \ldots, \\ V_{i-1}(f_{i(i-1)}(x, u), c_{i(i-1)}(x, u)), V_{i+1}(f_{i(i+1)}(x, u), c_{i(i+1)}(x, u)), \ldots, \\ V_m(f_{im}(x, u), c_{im}(x, u))) \qquad i = 1, 2, \ldots, m, \end{aligned}$$

where

$$R_i : G_r(U_i) \times \mathbf{R}^{m-1} \to D_i \qquad i = 1, 2, \ldots, m,$$

$$c_{ij} : G_r(U_i) \to D_j \qquad i = 1, 2, \ldots, m;\ j = 2, 3, \ldots, m;\ i \neq j.$$

We assume a finite maximum length of all state sequences along the recursive processes. The aim is to get the maximum value of $\mathrm{P}_1(\bar{x}_0)$, where $\bar{x}_0 \in X_1 \backslash T_1$ is a given initial state. This problem is denoted by $(\{\mathrm{P}_i\}_{i=1}^m, \bar{x}_0)$.

The recursive equations are induced by the following imbedded problems, where $W_i$ denotes the optimal value function of the imbedded $i$th process problem. For $i = 1$, $W_i$ is defined as follows:

$$W_1(x_0, \lambda) = g_1(\lambda \circ_1 k_1(x_0)) \qquad x_0 \in T_1,\ \lambda \in D_1,$$

$$W_1(x_0, \lambda) = \max_{\substack{\sigma_1 \in \Sigma_1,\, x_{n+1} = f_{11}(x_n, u_n) \\ n=0,1,\ldots,N-1}} [g_1(\lambda \circ_1 r_1(x_0, u_0) \circ_1 \cdots \circ_1 r_1(x_{N-1}, u_{N-1}) \circ_1 k_1(x_N))] \qquad x_0 \notin T_1,\ \lambda \in D_1,$$

and for $i = 2, 3, \ldots, m$,

$$W_i(x_0, c, \lambda) = g_i(c, \lambda \circ_i k_i(x_0)) \qquad x_0 \in T_i,\ \lambda \in D_i,$$

$$W_i(x_0, c, \lambda) = \max_{\substack{\sigma_i \in \Sigma_i,\, x_{n+1} = f_{ii}(x_n, u_n) \\ n=0,1,\ldots,N-1}} [g_i(c, \lambda \circ_i r_i(x_0, u_0) \circ_i \cdots \circ_i r_i(x_{N-1}, u_{N-1}) \circ_i k_i(x_N))] \qquad x_0 \notin T_i,\ \lambda \in D_i.$$

LEMMA 4.1. *The imbedded ith process problems are solved by the following recursive equations. For $i = 1$, we have*

$$W_1(x, \lambda) = g_1(\lambda \circ_1 k_1(x)) \qquad x \in T_1,\ \lambda \in D_1,$$

$$W_1(x, \lambda) = \max_{u \in U_1(x)} [W_1(f_{11}(x, u), \lambda \circ_1 r_1(x, u))] \qquad x \notin T_1,\ \lambda \in D_1. \tag{5}$$

*and for $i = 2, 3, \ldots, m$,*

$$W_i(x, c, \lambda) = g_i(c, \lambda \circ_i k_i(x)) \qquad x \in T_i,\ \lambda \in D_i,$$

$$W_i(x, c, \lambda) = \max_{u \in U_i(x)} [W_i(f_{ii}(x, u), c, \lambda \circ_i r_i(x, u))] \qquad x \notin T_i,\ \lambda \in D_i. \tag{6}$$

PROOF. Given that $c$ is constant through all stages in the $i$th process $\mathrm{P}_i(x_0, c)$ $(i = 2, 3, \ldots, m)$, Eqs. (5) and (6) are essentially equivalent. Therefore, it is sufficient to show that Eq. (5) holds. Furthermore, although $\mathrm{P}_1(x_0)$ appears superficially different from $\mathrm{P}_\mathrm{A}(x_0)$ (or $\mathrm{Q}_\mathrm{A}(y_0)$) in the previous section, Eqs. (1) and (5) are of the same form. Therefore, the truth of Eq. (5) can be demonstrated as described in [6]. □

THEOREM 4.1. *We have the following mutually dependent recursive equations.*

$$W_1(x,\lambda) = g_1(\lambda \circ_1 k_1(x)) \qquad x \in T_1,\ \lambda \in D_1,$$

$$W_1(x,\lambda) = \max_{u \in U_1(x)} [W_1(f_{11}(x,u), \lambda \circ_1 R_1(x,u,W_2,W_3,\ldots,W_m))] \qquad x \notin T_1,\ \lambda \in D_1,$$

$$W_i(x,c,\lambda) = g_i(c, \lambda \circ_i k_i(x)) \qquad x \in T_i,\ \mu \in D_i,\ i = 2,3,\ldots,m,$$

$$W_i(x,c,\lambda) = \max_{u \in U_i(x)} [W_i(f_{ii}(x,u), \lambda \circ_i R_i(x,u,W_1,W_2,\ldots, W_{i-1}, W_{i+1},\ldots,W_m))] \qquad x \notin T_i,\ \lambda \in D_i,\ i = 2,3,\ldots,m,$$

*where*

$$W_1 = W_1(f_{i1}(x,u), e_1), \qquad W_j = W_j(f_{ij}(x,u), c_{ij}(x,u), e_j) \qquad j = 2,3,\ldots,m.$$

*The optimal value of* $(\{P_i\}_{i=1}^m, \bar{x}_0)$ *is given by* $W_1(\bar{x}_0, e_1)$.

PROOF. Since

$$V_1(f_{i1}(x,u)) = W_1(f_{i1}(x,u), e_1)$$

and

$$V_j(f_{ij}(x,u), c_{ij}(x,u)) = W_j(f_{ij}(x,u), c_{ij}(x,u), e_j) \qquad j = 2,3,\ldots,m$$

are hold, we have

$$r_1(x,u) = R_1(x,u,W_2,W_3,\ldots,W_m),$$

$$r_i(x,u) = R_i(x,u,W_1,W_2,\ldots,W_{i-1},W_{i+1},\ldots,W_m) \qquad i = 2,3,\ldots,m.$$

Thus, the result follows directly from Lemma 4.1. □

We remark that the optimal general policy for $(\{P_i\}_{i=1}^m, \bar{x}_0)$ is similarly constructed to $\sigma^*$ and $\gamma^*$ in Section 3.

**Acknowledgements.** This work was supported by JSPS KAKENHI Grant Number 23654038, 15K05004.

## References

[1] R. E. Bellman, Dynamic Programming, NJ: Princeton Univ. Press, 1957.
[2] U. Bertelé and F. Brioschi, Nonserial Dynamic Programming, Academic Press, New York, 1972.
[3] T. Fujita, Re-examination of Markov Policies for Additive Decision Process, Bulletin of Informatics and Cybernetics, **29** (1997), 51–65.

[4] T. Fujita and K. Tsurusaki, Stochastic Optimization of Multiplicative Functions with Negative Value, Journal of Operations Research Society of Japan, **41** (1998), 351–373.
[5] T. Fujita, On policy classes in dynamic programming theory, Proceedings of the 9th Bellman Continuum International Workshop on Uncertain System and Soft Computing, Series of Information & Management Sciences, **2** (2002), 39–43.
[6] S. Iwamoto, Associative dynamic programs, Journal of Mathematical Analysis and Applications, **201** (1996), 195–211.
[7] S. Iwamoto, K. Tsurusaki and T. Fujita, On Markov policies for minimax decision processes, Journal of Mathematical Analysis and Applications, **253** (2001), 58–78.
[8] S. Iwamoto, T. Ueno and T. Fujita, Controlled Markov Chains with Utility Functions, Eds. H. Zhenting, J. A. Filar and A. Chen, Markov Processes and Controlled Markov Chains, Chap. 8, 135–148, Kluwer, 2002.
[9] A. Kira, T. Ueno and T. Fujita, Threshold probability of non-terminal type in finite horizon Markov decision processes, Journal of Mathematical Analysis and Applications, **386** (2012), 461–472.
[10] G. L. Nemhauser, Introduction to Dynamic Programming, Wiley, New York, 1966.
[11] M. Sniedovich, Dynamic Programming, Marcel Dekker, Inc. NY, 1992.

*Toshiharu Fujita*
*Graduate School of Engineering*
*Kyushu Institute of Technology*
*Tobata, Kitakyushu* 804-8550, *Japan*
*E-mail: fujita@mns.kyutech.ac.jp*