数理解析研究所講究録
1488 巻 2006 年 15-24

On Nondeterministic Dynamic Programming

九州工業大学・工学部　藤田 敏治 (Toshiharu Fujita)
Faculty of Engineering,
Kyushu Institute of Technology

1 Introduction

This paper considers a dynamic programming model with nondeterministic system. Dynamic programming has been developed and applied by many authors([1], [2], [4], [8], [9]). Dynamic programming models are classified under three transition systems. They are deterministic system, stochastic system ([8]) and fuzzy system ([2], [5]). In this paper nondeterministic system is introduced as a transition system of dynamic programming. Under nondeterministic system, next state is not unique, that is, a single state yields more than one state simultaneously in the next stage. We introduce this nondeterministic system and study on related optimization problems. Nondeterministic dynamic programming covers traditional ones and has a strong possibility for applying the idea of dynamic programming to more various problems.

2 Finite Stage Model

2.1 Notations and Definitions

A finite nondeterministic dynamic programming is defined by five-tuple:

$$\mathcal{N} = (N,\ X,\ \{U, U(\cdot)\},\ T,\ \{r, k, \beta\}),$$

where the definitions of each component are as follows.

1. $N(\geq 2)$ is an integer which means the *total number of stage.* The subscript n specifies the current number of stage.

2. X is a nonempty finite set which denotes a *state space.* Its elements $x_n \in X$ are called nth states. x_0 is an initial state and x_N is a terminal state.

3. U is a nonempty finite set which denotes an *action space.* Furthermore we also denote by U a mapping from X to 2^U and $U(x)$ is the set of all feasible actions for a state $x \in X$, where 2^Y denotes the following power set:

$$2^Y = \{A | A \subset Y,\ A \neq \emptyset\}.$$

After this, let $G_r(U)$ denote the graph of a mapping $U(\cdot)$:

$$G_r(U) := \{(x, u) \mid u \in U(x),\ x \in X\} \subset X \times U.$$

4. $T : G_r(U) \to 2^X$ is a *nondeterministic transition law.* For each pair of a state and an action $(x,u) \in G_r(U)$, $T(x,u)$ means the set of all states appeared in the next stage. If an action u_n is chosen for a current state x_n, each $x_{n+1} \in T(x,u)$ will become a next state.

5. $r : G_r(U) \to R^1$ is a *reward function,* $k : X \to R^1$ is a *terminal reward function* and $\beta : G_r(T) \to [0,\infty)$ is a *weight function.* If an action u_n is chosen for a current state x_n, we get a reward $r(x_n,u_n)$ and each next state x_{n+1} will be appeared with a corresponding weight $\beta(x_n,u_n,x_{n+1})$ (≥ 0). For a terminal state x_N we get a terminal reward $k(x_N)$.

A mapping $f : X \to U$ is called *decision function* if $f(x) \in U(x)$ for any $x \in X$. A sequence of decision functions:

$$\pi = \{f_0, f_1, \ldots f_{N-1}\}$$

is called a *Markov policy.* Let $\Pi(0)$ denotes the set of all Markov policies, which is called *Markov policy class.* If a decision-maker takes a Markov policy $\pi = \{f_0, f_1, \ldots f_{N-1}\}$, he chooses $f_n(x_n)\,(\in U)$ for state x_n at nth stage.

2.2 Formulation

For an initial state $x_0 \in X$ and Markov Policy $\pi \in \Pi(0)$, we introduce *total weighted value* is given by

$$\begin{aligned} V(x_0;\pi) \;&:=\; r_0 + \sum_{x_1 \in X(1)} \beta_0 r_1 + \sum\sum_{(x_1,x_2)\in X(2)} \beta_0\beta_1 r_2 + \\ &\cdots + \sum\sum\cdots\sum_{(x_1,\ldots,x_{N-1})\in X(N-1)} \beta_0\beta_1\cdots\beta_{N-1} r_{N-1} + \sum\sum\cdots\sum_{(x_1,\ldots,x_N)\in X(N)} \beta_0\beta_1\cdots\beta_{N-1} k \\ &\qquad\qquad x_0 \in X,\ \pi = \{f_0, f_1, \ldots f_{N-1}\} \in \Pi(0) \end{aligned}$$

where

$$r_n = r(x_n, f_n(x_n)), \quad k = k(x_N), \quad \beta_n = \beta(x_n, f_n(x_n), x_{n+1}),$$

$$X(m) = \{(x_1,\ldots,x_m) \in X \times \cdots \times X \mid x_{l+1} \in T(x_l, f_l(x_l))\ \ 0 \leq l \leq m-1\}.$$

Thus the *nondeterministic dynamic programming problem* is formulated as a maximization problem :

$$\mathrm{P}_0(x_0) \qquad \text{Maximize} \quad V(x_0;\pi) \quad \text{subject to} \quad \pi \in \Pi(0).$$

The problem $\mathrm{P}_0(x_0)$ means an N-stage decision process starting at 0th stage with an initial state x_0.

A policy π^* is called *optimal* if

$$V(x_0;\pi^*) \geq V(x_0;\pi) \qquad \forall \pi \in \Pi(0),\ \forall x_0 \in X.$$

2.3 Recursive Equation

Let $v_0(x_0)$ be the maximum value of $\mathrm{P}_0(x_0)$. Similarly, we consider the $(N-n)$-stage process with a starting state $x_n(\in X)$ on nth stage. The Markov policy class for this process is

$$\Pi(n) = \{\pi = \{f_n, f_{n+1}, \cdots f_{N-1}\} \mid f_l : X \to U,\ f_l(x) \in U(x),\quad n \le l \le N-1\}.$$

Thus weighted value is given by

$$\begin{aligned} V_n(x_n;\pi) \ :=\ & r_n + \sum_{x_n \in X(n)} \beta_n r_{n+1} + \sum\sum_{(x_n, x_{n+1}) \in X(n+1)} \beta_n \beta_{n+1} r_{n+1} + \cdots \\ & + \sum\sum\cdots\sum_{(x_n,\ldots,x_N)\in X(N)} \beta_n\beta_{n+1}\cdots\beta_{N-1}k, \qquad x_n \in X,\ \pi \in \Pi(n) \end{aligned}$$

where

$$X(m) \ =\ \{(x_n, \ldots, x_m) \in X \times \cdots \times X \mid x_{l+1} \in T(x_l, f_l(x_l)),\ n \le l \le m-1\}.$$

Then for $n = 1, 2, \ldots, N-1$ the *imbedded problem* is defined by

$$\mathrm{P}_n(x_n) \quad \text{Maximize} \quad V(x_n;\pi) \quad \text{subject to} \quad \pi \in \Pi(n),$$

and let $v_n(x_n)$ be the maximum value of $\mathrm{P}_n(x_n)$. For $n = N$ let $v_N(x_N) := k(x_N)$.

Then we have the following recursive equation.

Theorem 2.1 (nondeterministic)

$$\begin{aligned} v_N(x) &= k(x) \qquad x \in X \\ v_n(x) &= \max_{u\in U(x)} \left[r(x,u) + \sum_{y\in T(x,u)} \beta(x,u,y) v_{n+1}(y) \right] \quad x \in X,\ 0 \le n \le N-1. \end{aligned}$$

Let $f_n^*(x) \in U(x)$ be a point which attains $v_n(x)$. Then we get the optimal Markov policy $\pi^* = \{f_0^*, f_1^*, \cdots f_{N-1}^*\}$ in Markov class $\Pi(0)$.

The following results are for other transition systems.

Collorary 2.1 (stochastic) *In case* $\beta(x,u,y) = \beta \cdot p(y|x,u)$, $\beta \ge 0$ *and* $p = p(y|x,u)$ *is a Markov transition law,* $\mathrm{P}_0(x_0)$ *is a stochastic dynamic programming problem. Then we have the following recursive equation:*

$$\begin{aligned} v_N(x) &= k(x) \qquad x \in X \\ v_n(x) &= \max_{u\in U(x)} \left[r(x,u) + \beta \sum_{y\in T(x,u)} v_{n+1}(y) p(y|x,u) \right] \quad x \in X,\ 0 \le n \le N-1. \end{aligned}$$

Collorary 2.2 (deterministic) *In case* $T(x,u)$ *is a singleton,* $\mathrm{P}_0(x_0)$ *is a deterministic dynamic programming problem. Then we have the following recursive equation:*

$$\begin{aligned} v_N(x) &= k(x) \qquad x \in X \\ v_n(x) &= \max_{u\in U(x)} [r(x,u) + \beta(x,u,T(x,u)) v_{n+1}(T(x,u))] \quad x \in X,\ 0 \le n \le N-1. \end{aligned}$$

where $\beta(x,u,\{y\}), v_n(\{y\})$ *are equated with* $\beta(x,u,y), v_n(y)$*, respectively.*

3 Chained Matrix Products Problem

We consider the problem on chained matrix products (see tutOR, http://www.tutor.ms.unimelb.edu.au/). When we compute the product of three matrices A, B and C, the result is independent of the product order, that is $A(BC) = (AB)C$. On the other hand the number of scalar products required for computing the product depends on the product order. The purpose is to minimize the number of scalar products. We call this problem the chained matrix products problem.

Suppose that we have M matrices $A_1, A_2, \ldots, A_M$ to multiply and each matrix A_i has m_i rows and m_{i+1} columns. Then chained matrix products problem is formulated as the following nondeterministic dynamic programming problem :

$$\mathcal{N} \;=\; (M-1,\ X,\ \{U, U(\cdot)\},\ T,\ \{r, k, \beta\})$$

where

$$\begin{aligned}
X &= \{\{i,\, i+1,\, \ldots,\, j\} \mid 1 \le i < j \le M+1\} \\
U &= \{2, 3, \ldots, M\} \\
U(x) &= \{i+1,\, i+2,\, \ldots,\, j-1\}, \quad x = \{i,\, i+1,\, \ldots,\, j\} \in X \\
T(x,u) &= \{\{i,\, \ldots,\, u\}, \{u,\, \ldots,\, j\}\}, \quad x = \{i,\, i+1,\, \ldots,\, j\} \in X,\ u \in U(x) \\
\beta(x,u,y) &= \begin{cases} 0 & x = \{i,\, i+1\} \\ 1 & \text{otherwise} \end{cases}, \qquad (x,u,y) \in Gr(T) \\
r(x,u) &= \begin{cases} 0 & i+1 = j \\ m_i m_u m_j & i+1 < j \end{cases}, \qquad (x,u) = (\{i,\ldots,j\}, u) \in Gr(U) \\
k(x) &= 0, \qquad x = \{i, i+1\} \in X,
\end{aligned}$$

and the problem we must solve is the minimizing problem for the initial state $x_0 = \{1, 2, \ldots, M+1\}$.

In this case, we need not differentiate among value functions v_n. Therfore we have the following recursive equation by Theprem 2.1.

$$\begin{aligned}
v(x) &= 0 \qquad x = \{i,\ i+1\} \in X \\
v(x) &= \min_{u \in U(x)} \left[m_i m_u m_j + \sum_{y \in T(x,u)} v(y) \right] \quad x = \{i, \ldots, j\} \in X \quad (i+1 < j),
\end{aligned}$$

where we suppose that for $U(x) = \phi$ the result of minimizig on $U(x)$ is equal to 0.

Numerical Example

Let $M = 4, m_1 = 3, m_2 = 10, m_3 = 5, m_4 = 4$ and $m_5 = 16$. Then we find the optimal product order for chained matrix products:

$$A_1 A_2 A_3 A_4.$$

To start with

$$v(x) = 0, \quad x = \{i, i+1\} \in X.$$

Then we get

$$
\begin{aligned}
v(\{1,2,3\}) &= r(\{1,2,3\},2) + (v(\{1,2\}) + v(\{2,3\})) \\
&= m_1m_2m_3 + (0+0) = 150, \qquad f^*(\{1,2,3\}) = 2,
\end{aligned}
$$

$$
\begin{aligned}
v(\{2,3,4\}) &= r(\{2,3,4\},3) + (v(\{2,3\}) + v(\{3,4\})) \\
&= m_2m_3m_4 + (0+0) = 200, \qquad f^*(\{2,3,4\}) = 3,
\end{aligned}
$$

$$
\begin{aligned}
v(\{3,4,5\}) &= r(\{3,4,5\},4) + (v(\{3,4\}) + v(\{4,5\})) \\
&= m_3m_4m_5 + (0+0) = 320, \qquad f^*(\{3,4,5\}) = 4.
\end{aligned}
$$

Similarly,

$$
\begin{aligned}
v(\{1,2,3,4\}) &= \min\{r(\{1,2,3,4\},2) + (v(\{1,2\}) + v(\{2,3,4\})), \\
&\qquad\quad r(\{1,2,3,4\},3) + (v(\{1,2,3\}) + v(\{3,4\}))\} \\
&= \min\{m_1m_2m_4 + (0+200), m_1m_3m_4 + (150+0)\} \\
&= \min\{120+200, 60+150\} = \min\{320, 210\} \\
&= 210, \qquad f^*(\{1,2,3,4\}) = 3,
\end{aligned}
$$

$$
\begin{aligned}
v(\{2,3,4,5\}) &= \min\{r(\{2,3,4,5\},3) + (v(\{2,3\}) + v(\{3,4,5\})), \\
&\qquad\quad r(\{2,3,4,5\},4) + (v(\{2,3,4\}) + v(\{4,5\}))\} \\
&= \min\{1120, 840\} = 840, \qquad f^*(\{2,3,4,5\}) = 4.
\end{aligned}
$$

Finally, for $x_0 = \{1,2,3,4,5\}$,

$$
\begin{aligned}
v(\{1,2,3,4,5\}) &= \min\{r(\{1,2,3,4,5\},2) + (v(\{1,2\}) + v(\{2,3,4,5\})), \\
&\qquad\quad r(\{1,2,3,4,5\},3) + (v(\{1,2,3\}) + v(\{3,4,5\})), \\
&\qquad\quad r(\{1,2,3,4,5\},4) + (v(\{1,2,3,4\}) + v(\{4,5\}))\} \\
&= \min\{1320, 710, 402\} = 402, \qquad f^*(\{1,2,3,4,5\}) = 4.
\end{aligned}
$$

As a result, the minimum of the number of scalar products is

$$v(\{1,2,3,4,5\}) = 402,$$

and the optimal decision sequence $\{u_1^*, u_2^*, u_3^*\}$ is given by

$$u_1^* = f^*(\{1,2,3,4,5\}) = 4, \; u_2^* = f^*(\{1,2,3,4\}) = 3, \; u_3^* = f^*(\{1,2,3\}) = 2.$$

This means that $((A_1A_2)A_3)A_4$ is the optimal product order .

4 Infinite Stage Model

An infinite nondeterministic dynamic programming is defined by four-tuple:

$$\mathcal{N}^\infty = (X, \; \{U, U(\cdot)\}, \; T, \; \{r, \beta\}),$$

where definition of each component is given in section 2.

We note that an infinite sequence of decision functions:

$$\pi = \{f_0, f_1, \cdots f_n, \cdots\}$$

is called a *Markov policy* and let Π denotes the set of all Markov policies defined above.

In this case, total weighted value is given by

$$\begin{aligned} V(x_0;\pi) \quad := \quad & r_0 + \sum_{x_1 \in X(1)} \beta_0 r_1 + \sum\sum_{(x_1,x_2)\in X(2)} \beta_0\beta_1 r_2 + \\ & \cdots + \sum\sum\cdots\sum_{(x_1,\ldots,x_n)\in X(n)} \beta_0\beta_1\cdots\beta_{n-1} r_n + \cdots, \qquad x_0 \in X,\ \pi \in \Pi, \end{aligned}$$

where

$$r_n = r(x_n, f_n(x_n)), \quad \beta_n = \beta(x_n, f_n(x_n), x_{n+1})$$

$$X(n) = \{(x_1, \ldots, x_n) \in X \times \cdots \times X \mid x_{m+1} \in T(x_m, f_m(x_m)) \ 0 \le m \le n-1\}.$$

Thus the *infinite nondeterministic dynamic programming problem* is formulated as

$$\mathrm{P}(x_0) \qquad \text{Maximize} \quad V(x_0;\pi) \quad \text{subject to} \quad \pi \in \Pi$$

Let $v(x_0)$ be the maximum value of $\mathrm{P}(x_0)$ and the norm of β is defind by

$$\beta_1 := ||\beta||_1 = \max_{(x,u)\in G_r(U)} \sum_{y \in T(x,u)} |\beta(x,u,y)|.$$

Then we have the following result.

Theorem 4.1 *Under the assumption*

$$\beta_1 < 1,$$

value function $v(\cdot)$ satisfies the following optimal equation :

$$v(x) \quad = \quad \max_{u \in U(x)} \left[r(x,u) + \sum_{y \in T(x,u)} \beta(x,u,y) v(y) \right] \qquad x \in X.$$

Note that the solution of this equation is unique.

Let $f^*(x) \in U(x)$ be a point which attains $v(x)$. Then we get the optimal stationaly Markov policy $\pi^* = \{f^*, f^*, \ldots, f^*, \ldots\} \in \Pi$.

5 Maximum Linear Equations

In this section, we use the following notations. For two real values a, b, their maxima and minima are denoted by

$$a \vee b = \max\{a, b\}, \qquad a \wedge b = \min\{a, b\},$$

respectively, and for the set of real values $\{a_1, a_2, \dots, a_n\}$, their maxima and minima by

$$\bigvee_{i=1}^{n} a_i = \max\{a_1, a_2, \dots, a_n\},$$

$$\bigwedge_{i=1}^{n} a_i = \min\{a_1, a_2, \dots, a_n\}.$$

For the set $A = \{a_{ij}^k \in \mathbf{R} \mid 1 \le k \le K_i,\ 1 \le i, j \le N\}$, we use

$$\|A\| = \max_{1 \le k \le K_i, 1 \le i \le N} \sum_{j=1}^{N} |a_{ij}^k|,$$

$$A \ge O \iff a_{ij}^k \ge 0 \quad \text{for} \quad 1 \le k \le K_i,\ 1 \le i, j \le N$$

Then let us consider the system of maximized linear equations,

$$x_i = \bigvee_{k=1}^{K_i} \left(\sum_{j=1}^{N} a_{ij}^k x_j + b_i^k \right) \qquad i = 1, 2, \dots, N, \tag{1}$$

where $b_i^k \in \mathbf{R}$ $(1 \le k \le K_i,\ 1 \le i \le N)$. We call the system (1) *maximum linear equation.*

The maximum linear equation is equivalent to the optimal equation for the following infinite nondeterministic dynamic programming problem :

$$\mathcal{N}^\infty \;=\; (X,\ \{U, U(\cdot)\},\ T,\ \{r, \beta\})$$

where

$$\begin{aligned}
X &= \{1, 2, \dots, N\} \\
U &= \left\{1, 2, \dots, \bigvee_{x \in X} K_x \right\} \\
U(x) &= \{1,\ 2,\ \dots,\ K_x\}, \qquad x \in X \\
T(x, u) &= X, \quad (x, u) \in Gr(U) \\
r(x, u) &= b_x^u, \qquad (x, u) \in G_r(U) \\
\beta(x, u, y) &= a_{xy}^u, \qquad (x, u, y) \in G_r(T).
\end{aligned}$$

In fact, for the optimal equation :

$$v(x) \;=\; \max_{u \in U(x)} \left[r(x, u) + \sum_{y \in T(x,u)} \beta(x, u, y) v(y) \right] \qquad x \in X,$$

let $T(x, u) = X$, $r(x, u) = b_x^u$ and $\beta(x, u, y) = a_{xy}^u$, then

$$v(x) \;=\; \max_{u \in U(x)} \left[b_x^u + \sum_{y \in X} a_{xy}^u v(y) \right] \qquad x \in X.$$

Since $X = \{1, 2, \dots, N\}$, $U(x) = \{1,\ 2,\ \dots,\ K_x\}$,

$$v(x) \;=\; \bigvee_{u=1}^{K_x} \left[\sum_{y=1}^{N} a_{xy}^u v(y) + b_x^u \right] \qquad x = 1, 2, \dots, N.$$

This is the maximum linear equation (1).

Theorem 5.1 (existence, uniqueness) Under the assumption

$$\|A\| < 1$$

there exists a unique solution of Eq.(1).

Further under the additional assumption

$$A \geq O$$

we have the following algorithm for finding the unique solution.

Algorithm

Step 1 (initial selection)
Let $n = 0$. Take any feasible selection (decision function) f_0.

Step 2 (value determination)
Calculate $x^n = x(f_n) = (x_1(f_n), x_2(f_n), \ldots, x_N(f_n))$ satisfying

$$x_i^n = \sum_{j=1}^{N} a_{ij}^{f_n(i)} x_j^n + b_i^{f_n(i)} \qquad i = 1, 2, \ldots, N.$$

Step 3 (optimality test)
If x_n satisfies

$$x_i^n = \bigvee_{k=1}^{K_i} \left(\sum_{j=1}^{N} a_{ij}^k x_j^n + b_i^k \right) \qquad i = 1, 2, \ldots, N,$$

then go to step 6. Otherwise, go to step 4.

Step 4 (selection improvement)
Choose a feasible selection f_{n+1} satisfying

$$\bigvee_{k=1}^{K_i} (\sum_{j=1}^{N} a_{ij}^k x_j^n + b_i^k) = \sum_{j=1}^{N} a_{ij}^{f_{n+1}(i)} x_j^n + b_i^{f_{n+1}(i)} \qquad i = 1, 2, \ldots, N.$$

Step 5 (next step)
Let $n = n + 1$. Go to step 2.

Step 6 (optimal solution)
The selection f_n is optimal and x^n is the desired solution.

Numerical Example
We consider the following maximum linear equation

$$\begin{array}{rcl} x & = & \left(\frac{1}{3}x + \frac{1}{2}y - 12\right) \vee \left(\frac{1}{4}x + \frac{2}{3}y + 24\right) \vee \left(\frac{3}{4}x + \frac{1}{5}y - 20\right) \\ y & = & \left(\frac{2}{3}x + \frac{1}{5}y - 15\right) \vee \left(\frac{1}{2}x + \frac{1}{3}y + 12\right) \vee \left(\frac{1}{2}x + \frac{2}{5}y + 10\right) \\ & & (\qquad k = 1 \qquad\qquad k = 2 \qquad\qquad k = 3 \qquad) \end{array}$$

Algorithm solves the equation as follows ($x_1 := x,\ x_2 := y$).

1. step 1 $\quad n = 0, \quad f_0 = (1,1)$.
   ($f_n = (i,j)$ means $f_n(1) = i$ and $f_n(2) = j$.)

2. step 2 The linear equation

$$\begin{cases} x = & \frac{1}{3}x + \frac{1}{2}y - 12 \\ y = & \frac{2}{3}x + \frac{1}{5}y - 15 \end{cases}$$

has the solution $(x^0, y^0) = \left(-\frac{171}{2}, -90\right)$.

3. step 3

$$\begin{cases} x^0 & \neq & \left(\frac{1}{3}x^0 + \frac{1}{2}y^0 - 12\right) & \vee & \left(\frac{1}{4}x^0 + \frac{2}{3}y^0 + 24\right) & \vee & \left(\frac{3}{4}x^0 + \frac{1}{5}y^0 - 20\right) \\ y^0 & \neq & \left(\frac{2}{3}x^0 + \frac{1}{5}y^0 - 15\right) & \vee & \left(\frac{1}{2}x^0 + \frac{1}{3}y^0 + 12\right) & \vee & \left(\frac{1}{2}x^0 + \frac{2}{5}y^0 + 10\right) \end{cases}$$

$\Rightarrow$ step 4.

4. step 4 $\quad f_1 = (2,2)$.

5. step 5 $\Rightarrow$ step 2.

6. step 2 The linear equation

$$\begin{cases} x = & \frac{1}{4}x + \frac{2}{3}y + 24 \\ y = & \frac{1}{2}x + \frac{1}{3}y + 12 \end{cases}$$

has the solution $(x^1, y^1) = (144, 126)$.

7. step 3

$$\begin{cases} x^1 & \neq & \left(\frac{1}{3}x^1 + \frac{1}{2}y^1 - 12\right) & \vee & \left(\frac{1}{4}x^1 + \frac{2}{3}y^1 + 24\right) & \vee & \left(\frac{3}{4}x^1 + \frac{1}{5}y^1 - 20\right) \\ y^1 & \neq & \left(\frac{2}{3}x^1 + \frac{1}{5}y^1 - 15\right) & \vee & \left(\frac{1}{2}x^1 + \frac{1}{3}y^1 + 12\right) & \vee & \left(\frac{1}{2}x^1 + \frac{2}{5}y^1 + 10\right) \end{cases}$$

$\Rightarrow$ step 4.

8. step 4 $\quad f_2 = (2,3)$.

9. step 5 $\Rightarrow$ step 2.

10. step 2 The linear equation

$$\begin{cases} x = & \frac{1}{4}x + \frac{2}{3}y + 24 \\ y = & \frac{1}{2}x + \frac{2}{5}y + 10 \end{cases}$$

has the solution $(x^2, y^2) = \left(\frac{1264}{7}, \frac{1164}{7}\right)$.

11. step 3 This solution (x^2, y^2) satisfies the original equation. $\Rightarrow$ step 6.

12. step 6 Thus $f_2 = (2,3)$ is the optimal selection, and $(x^*, y^*) = (x^2, y^2) = \left(\frac{1264}{7}, \frac{1164}{7}\right)$ is the desired unique solution.

References

[1] R.E. Bellman, *Dynamic Programming*, NJ: Princeton Univ. Press, 1957.

[2] R.E. Bellman and L.A. Zadeh, Decision-making in a fuzzy environment, Management Science, **17**(1970), B141-B164.

[3] T. Fujita and K. Tsurusaki, Stochastic optimization of multiplicative functions with negative value, J. Oper. Res. Soc. Japan, **41**(1998), 351-373.

[4] R. A. Howard, *Dynamic Programming and Markov Processes*, MITPress, Cambridge, Mass., 1960.

[5] S. Iwamoto, A class of dual fuzzy dynamic programs, Appl. Math. Comput. **120** (2001), 91-108.

[6] S. Iwamoto and T. Fujita, Stochastic decision-making in a fuzzy environment, J. Oper. Res. Soc. Japan, **38**(1995), 467-482.

[7] S. Iwamoto, K. Tsurusaki and T. Fujita, On Markov Policies for Minimax Decision Processes, J. Math. Anal. Appl., **253**(2001), 58-78.

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

[9] M. Sniedovich, *Dynamic Programming*, Marcel Dekker, Inc. NY, 1992.