Journal of the Operations Research Society of Japan
Vol. 60, No. 2, April 2017, pp. 110–121

SURE WAY TO WIN A GAME USING A MUTUALLY DEPENDENT DECISION PROCESS MODEL

Toshiharu Fujita
Kyushu Institute of Technology

(Received March 14, 2016; Revised October 29, 2016)

Abstract The purpose of this study is to consider the problem of finding a guaranteed way of winning a certain two-player combinatorial game of perfect knowledge from the standpoint of mutually dependent decision processes (MDDPs). Our MDDP model comprises two one-stage deterministic decision processes. Each decision process expresses every turn of a player. We analyze a MDDP problem in which the length of turns taken by a player is minimized, allowing him to win regardless of the decisions made by his opponent. The model provides a formulation for finding the shortest guaranteed strategy. Although computational complexity remains, the concept introduced in this paper can also be applied to other two-player combinatorial games of perfect knowledge.

Keywords: Dynamic programming, decision process, combinatorial game, winning strategy

1. Introduction

The concept of mutually dependent decision processes (MDDPs) was introduced in [6]. An MDDP comprises a recursive combination of decision processes using reward functions at each stage and transitions between state spaces. The transition structure of an MDDP can be regarded as a nonserial system [2, 9]. The MDDP model applied in this paper is structured upon two types of finite-stage deterministic decision processes: main processes and subprocesses. Each process consists of states, decisions, reward functions, and a deterministic transition law and operates by an ordinary decision process. The flow of the model is implemented as follows. First, the initial state in the main process is given. At each stage, the current state is observed and a decision is made. The next state is determined by the transition law, and this flow, constituting the main process, is an ordinary deterministic decision process. However, the reward at each stage is given by the optimal value (in general, the function of the optimal value) of the alternative decision process (the subprocess), which also operates as an ordinary deterministic decision process. The initial state in the subprocess is determined by the current state and decision in the main process. Similarly, the reward at each stage in the subprocess is given by the optimal value of the main process whose initial state is determined by the current state and decision in the subprocess. This interaction is recursively repeated in all state sequences along alternating processes. However, the terminal rewards in both processes are assigned in the usual manner and are independent of the other decision process. This is called an MDDP.

The MDDP model enables easier treatment of some classes of complex multi-stage decision processes. We actually apply a deterministic MDDP model with associative reward systems to a problem on convex polyhedra constructed by paper units [8]. Moreover, under stochastic environment, the MDDP framework includes Markov game models as MDDP

models with one-stage Markov decision processes.

In this paper, we consider an additive MDDP model consisting of two one-stage deterministic decision processes. It can be applied to the problem of finding the shortest guaranteed strategy for winning a type of two-player game known as the pyramid game, which is one of the impartial combinatorial games of perfect knowledge [1]. We formulate the problem and solve it using mutually dependent recursive equations introduced by dynamic programming. The concept underlying this research can be applied to other two-player combinatorial games of perfect knowledge, including partizan games.

2. Mutually Dependent Decision Processes

Typical deterministic MDDP models are discussed in [3, 4, 6]. In this section, we describe a deterministic MDDP model with two one-stage decision processes and the associated mutually dependent recursive equations. The model has an additive criterion and both processes have a common state space and a common decision space.

The main process $\mathrm{P}(x_0)$ and the subprocess $\mathrm{Q}(x_0)$ are formulated as follows:

$$\begin{aligned} \mathrm{P}(x_0) \quad & \text{minimize } r(x_0, u_0) + r_G(x_1) \\ & \text{subject to } \ x_1 = f(x_0, u_0), \\ & \qquad\qquad\ u_0 \in U(x_0), \end{aligned}$$

$$\begin{aligned} \mathrm{Q}(x_0) \quad & \text{maximize } q(x_0, u_0) + q_G(x_1) \\ & \text{subject to } \ x_1 = f(x_0, u_0), \\ & \qquad\qquad\ u_0 \in U(x_0), \end{aligned}$$

where

1. X, a nonempty finite set, is the state space. $T \subset X$ denotes the terminal state set. The transition is terminated if a terminal state appears by transition law. The initial state x_0 $(\in X \setminus T)$ is specified at the beginning of the process.
2. U, a nonempty finite set, is the action space. We denote the power set of U by 2^U:

   $$2^U = \{A : \text{a set} \,|\, A \subset U\}.$$

   Furthermore, by U, we denote a point-to-set valued mapping from $X \setminus T$ to $2^U \setminus \{\phi\}$. $U(x)$, called the feasible action space, represents the set of all feasible actions in state x. Let $G_r(U)$ denote the graph of $U(\cdot)$:

   $$G_r(U) = \{(x, u) \,|\, u \in U(x),\ x \in X \setminus T\}.$$

3. $r : G_r(U) \to \mathbf{R}$ and $q : G_r(U) \to \mathbf{R}$ are the reward functions in the main process and the subprocess, respectively, where $\mathbf{R} = (-\infty, \infty)$. The functions $r_G : T \to \mathbf{R}$ and $q_G : T \to \mathbf{R}$ are the terminal reward functions in the main process and the subprocess, respectively.
4. $f : G_r(U) \to T$ is a deterministic transition law. Whenever an action is chosen for the initial state, the process progresses to a terminal state. The transition is terminated at time 1.

We now introduce another transition law that connects the main process and the subprocess:

$$g : G_r(U) \to X.$$

The initial state of a subprocess problem is given by this transition g, which depends on the state x_0 and decision u_0 at time 0 in the main process. Conversely, the initial state of a main process problem is given by g, which depends on the state x_0 and decision u_0 at time 0 in the subprocess.

The rewards r and q are then defined as follows:

$$r(x,u) = \begin{cases} q_G(x_0) & x_0 = g(x,u) \in T, \\ \max\limits_{\substack{x_1=f(x_0,u_0) \\ u_0 \in U(x_0)}} \left[q(x_0,u_0) + q_G(x_1)\right] & x_0 = g(x,u) \notin T, \end{cases} \tag{2.1}$$

$$q(x,u) = \begin{cases} r_G(x_0) & x_0 = g(x,u) \in T, \\ \min\limits_{\substack{x_1=f(x_0,u_0) \\ u_0 \in U(x_0)}} \left[r(x_0,u_0) + r_G(x_1)\right] & x_0 = g(x,u) \notin T. \end{cases} \tag{2.2}$$

Here $r(x,u)$ is the maximum value of the subprocess problem with the corresponding initial state $x_0 = g(x,u)$. When x_0 is a terminal state, $r(x,u)$ is the terminal reward $q_G(x_0)$. Similarly, $q(x,u)$ is the minimum value of the main process problem initiated with $x_0 = g(x,u)$. We assume a finite maximum length of all state sequences along the alternating processes. Our target problem then becomes the main process problem $\mathrm{P}(\bar{x}_0)$, where $\bar{x}_0 \in X \setminus T$ is a given initial state.

Next, we consider the following subproblems for the main process with initial states $x_0 \in X$ and optimal values which are denoted by $w(x_0)$.

$$\begin{aligned} w(x_0) &= r_G(x_0) && x_0 \in T, \\ w(x_0) &= \min_{\substack{x_1=f(x_0,u_0) \\ u_0 \in U(x_0)}} [r(x_0,u_0) + r_G(x_1)] && x_0 \notin T. \end{aligned}$$

Similarly, we consider the following subproblems in the subprocess with initial states $x_0 \in X$ and optimal values $z(x_0)$.

$$\begin{aligned} z(x_0) &= q_G(x_0) && x_0 \in T, \\ z(x_0) &= \max_{\substack{x_1=f(x_0,u_0) \\ u_0 \in U(x_0)}} [q(x_0,u_0) + q_G(x_1)] && x_0 \notin T. \end{aligned}$$

Then, the mutually dependent recursive equations for MDDP in [6] reduce to

$$w(x) = r_G(x) \qquad x \in T, \tag{2.3}$$

$$w(x) = \min_{u \in U(x)} [z(g(x,u)) + r_G(f(x,u))] \qquad x \notin T. \tag{2.4}$$

$$z(x) = q_G(x) \qquad x \in T, \tag{2.5}$$

$$z(x) = \max_{u \in U(x)} [w(g(x,u)) + q_G(f(x,u))] \qquad x \notin T. \tag{2.6}$$

3. Guaranteed Way to Win a Game

3.1. Pyramid game

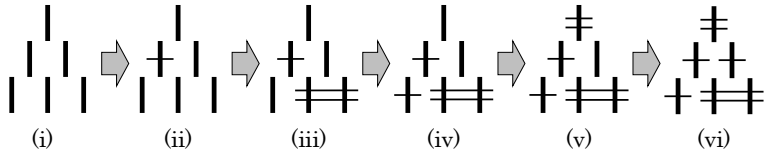

Figure 2: Game flow

The two-player game we consider in this research is the same as that investigated in [5]. Initially, a group of vertical bars is arranged, as shown in Figure 1. The bars are divided into several tiers with n bars in the nth tier. In this analysis, p ($p \geq 2$) is a given positive integer and we consider a p-tier game.

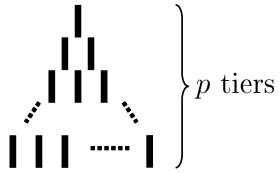

Figure 1: Starting position

Two players mark the bars in turns. In each turn, each player must mark at least one bar. If a player marks more than two bars, those bars must be consecutive in the same tier. The player who marks the last bar loses. The number of tiers is chosen arbitrarily at the beginning of the game. This game, which is known as the pyramid game, is played in some parts of Japan.

An example of the game flow is illustrated in Figure 2. The first player marks the left bar in the second tier (Figure 2 (ii)). The second player then marks two bars from right side in the third tier (Figure 2 (iii)). In the second move, the first player marks the left bar in the third tier (Figure 2 (iv)) and the second player marks the bar in the first tier (Figure 2 (v)). Finally, the first player must mark the last bar (Figure 2 (vi)) and the second player wins.

Our interest is in determining whether a guaranteed winning strategy exists. This is a strategy by which a player can win irrespective of the moves his opponent makes.

3.2. Formulation

The purpose of this study is to find a guaranteed winning strategy using an MDDP approach. Applying the framework introduced in Section 2, the main process and the subprocess correspond to the first and second player, respectively. Both objective functions express the number of turns taken by the first player from the initial state to a win. The first player wishes to win and minimize the objective function. On the other hand, in addition to win, even if the second player cannot win, he wishes to maximize the objective function which is the number of turns taken by the first player.

Consider a general p-tier pyramid game, as shown in Figure 1. In our formulation, a state and an action denote the current game position and the decision of each player. First, the state and state space are defined using the same notations of state and decision described in [5]. Note that equivalent positions need not be differentiated because they lead to the same result (Figure 3). In fact, only two numbers are significant: the number of bars in the consecutive bar series and the number of consecutive bar series of the same length. The location of the bars is irrelevant. In the plays shown in Figure 3, there is one independent bar, one series of two consecutive bars, and no series of three consecutive bars. Therefore, the state is defined as a p-dimensional vector whose nth element denotes the number of n consecutive bars. Formally, the state space is defined as follows:

$$X \;=\; \left\{ (x_1, x_2, \ldots, x_p) \,|\; x_i = 0, 1, 2, \ldots, \quad i = 1, 2, \ldots, p \;\right\}.$$

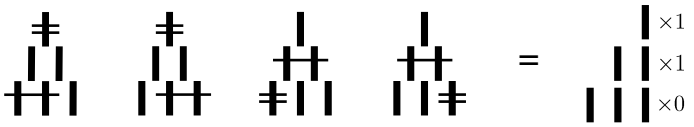

Figure 3: Equivalent positions

The initial state is given by

$$x_0 \;=\; (1,1,\ldots,1) \;\in\; X.$$

The state $(1,0,0,\ldots,0)$ denotes a loss for a player, because he has no option but to mark the last bar. This terminates the game. For convenience, we add the dummy state $(0,0,\ldots,0)$ as a terminal state. The state $(0,0,\ldots,0)$ is used as the terminal state, x_1, of the main process and subprocess. Note that an ordinary state transition connecting both processes (defined below) cannot yield $(0,0,\ldots,0)$; therefore, the terminal state set is given by

$$T \;=\; \{(1,0,0,\ldots,0),\ (0,0,\ldots,0)\} \;\subset\; X.$$

Next, we define the action and the action space. Each player's decision is characterized by three numbers: u_1, u_2, and u_3. The number u_1 specifies the target series of consecutive bars, while u_2 and u_3 specify the range of bars marked by the player. For example, if a player marks the 2nd, 3rd, and 4th bars in a series of six consecutive bars (Figure 4), his action is characterized by $u_1 = 6$, $u_2 = 2$, and $u_3 = 4$. Thus, the action is defined as a three-dimensional vector and the action space is defined as follows:

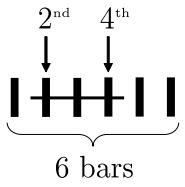

Figure 4: Action $(6,2,4)$

$$U \;=\; \bigcup_{n=1}^{p} U_n,$$

where

$$U_n \;=\; \left\{ (u_1,u_2,u_3) \;\middle|\; \begin{array}{l} u_1 = n \\ u_2, u_3 \in \{1,2,\ldots,n\} \\ u_2 < \dfrac{n}{2} + 1 \\ u_2 \le u_3 \le u_1 - u_2 + 1 \end{array} \right\} \quad n = 1,2,\ldots,p.$$

Each U_n is the set of all actions for n consecutive bars. The inequalities $u_2 < \dfrac{n}{2} + 1$ and $u_2 \le u_3 \le u_1 - u_2 + 1$ exclude redundant actions. For example, the following two cases are equivalent. In the first case, a player marks two bars from the left side of a consecutive bar series, whereas in the second case, he marks two bars from the right side of the same bar series. Both cases lead to the same outcome and do not need to be individually considered. We define $U(x)$, the set of all feasible actions for $x \in X$, as follows:

$$U(x) \;=\; \bigcup_{n;x_n \ge 1} U_n, \qquad x = (x_1, x_2, \ldots, x_p) \in X.$$

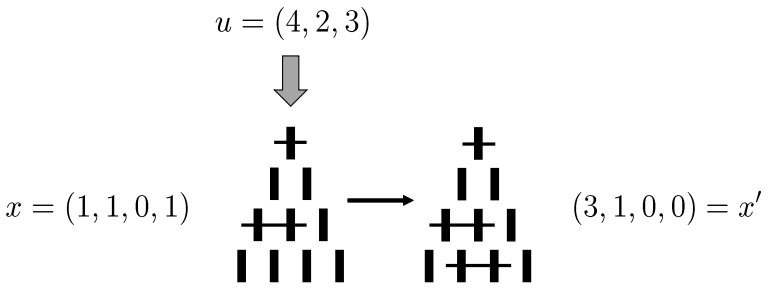

Figure 5: Transition $g((1,1,0,1),(4,2,3))$

$U(x)$ is assumed to exclude actions by which a player deliberately loses the game when other options are available. Specifically, if a state $x = (x_1, x_2, \ldots, x_p)$ satisfies $x_1 = 0$ and $\sum_{i=2}^{p} x_i = 1$, then, for n such that $x_n = 1$, we redefine U_n as follows:

$$U_n \quad \longleftarrow \quad U_n \setminus \{(n,1,n)\}.$$

Example 3.1. For $x = (2,0,1),\ y = (0,0,1) \in X$, we have

$$\begin{aligned} U(x) &= U_1 \cup U_3 = \{(1,1,1)\} \cup \{(3,1,1),\ (3,1,2),\ (3,1,3),\ (3,2,2)\} \\ &= \{(1,1,1),\ (3,1,1),\ (3,1,2),\ (3,1,3),\ (3,2,2)\}, \\ U(y) &= U_3 = \{(3,1,1),\ (3,1,2),\ (3,2,2)\}. \end{aligned}$$

□

Next, we consider the transition system. The transition $f : G_r(U) \to T$ of each process is defined as follows:

$$f(x,u) = (0,0,\ldots,0) \qquad (x,u) \in G_r(U).$$

As noted above, the terminal state $(0,0,\ldots,0)$ is a dummy state. Taking this into account, to define

$$r_G((0,0,\ldots,0)) = 1, \tag{3.1}$$

in the main process, we increase the number of turns taken by the first player by one. The main process and the subprocess are connected by the transition $g : G_r(U) \to X$, as follows:

$$g((x_1,x_2,\ldots,x_p),(u_1,u_2,u_3)) = (x_1',x_2',\ldots,x_p') \qquad ((x_1,x_2,\ldots,x_p),(u_1,u_2,u_3)) \in G_r(U),$$

where

$$x_i' = \begin{cases} x_i - 1 & (\ i = u_1\ ), \\ x_i + 1 & \left(\begin{array}{l} i = u_2 - 1 \neq u_1 - u_3,\ u_2 - 1 > 0 \\ \quad \text{or} \\ i = u_1 - u_3 \neq u_2 - 1,\ u_1 - u_3 > 0 \end{array} \right), \\ x_i + 2 & (\ i = u_2 - 1 = u_1 - u_3\ ), \\ x_i & (\ \text{otherwise}\ ). \end{cases}$$

For example, if a player takes the action $u = (4,2,3)$ for the state $x = (1,1,0,1)$, then the opponent faces position $x' = (3,1,0,0)$ (see Figure 5).

Finally, we specify

$$L = \left[\frac{1+2+\cdots+p}{2}\right],$$

where

$$[x] = \max\{y \mid y \le x,\ y : \text{integer}\},$$

and define the terminal rewards $r_G : T \to \mathbf{R}$ and $q_G : T \to \mathbf{R}$ by

$$\begin{aligned} r_G(x) &= L \qquad x = (1, 0, \ldots, 0), \\ q_G(x) &= 0 \qquad x \in T \end{aligned}$$

and Equation (3.1). As per the definition of the reward functions (Equations (2.1) and (2.2)), the first player needs $r(x, u)$ turns in the worst case until reaching a win for position $g(x, u)$. In this case, when the game ends, the terminal cost L means that the first player is the loser. Note that, before the terminal state $(1, 0, \ldots, 0)$ appears, the number of turns taken by each player is less than or equal to L. For this reason, to ensure that the first player wins, the optimal value must be less than or equal to L. If the minimum of $\mathrm{P}((1, 1, \ldots, 1))$ is less than or equal to L, a guaranteed winning strategy exists and the optimal decision function gives the shortest strategy. Otherwise, no guaranteed winning strategy exists for the first player.

3.3. Recursive equations

In our formulation, the mutually dependent recursive equations are given by the following:

$$W(x) = L \qquad x = (1, 0, \ldots, 0), \tag{3.2}$$

$$W(x) = \min_{u \in U(x)} [Z(g(x, u)) + 1] \qquad x \in X \setminus T. \tag{3.3}$$

$$Z(x) = 0 \qquad x = (1, 0, \ldots, 0), \tag{3.4}$$

$$Z(x) = \max_{u \in U(x)} [W(g(x, u))] \qquad x \in X \setminus T. \tag{3.5}$$

These are immediately derived from Equations (2.3) – (2.6). By computing $W((1, 1, \ldots, 1))$, we can derive the guaranteed winning strategy for the first player as the optimal decision function $\pi^* : X \setminus T \to U$, given by

$$\pi^*(x) \in \operatorname*{argmin}_{u \in U(x)} [Z(g(x, u)) + 1] \qquad x \in X \setminus T.$$

Note that we need not record the optimal decision(s) for the latter player:

$$\operatorname*{argmax}_{u \in U(x)} [W(g(x, u))] \qquad x \in X \setminus T.$$

3.4. Surefire winning strategies for the second player

With some modifications, our formulation can also be used to find a guaranteed winning strategy for the second player. By exchanging the roles of the main process and the subprocess, the problem can be reinterpreted as minimizing the number of turns taken by the second player until he wins. Thus, computing $Z((1, 1, \ldots, 1))$ with Equations (3.2) – (3.5) gives a guaranteed winning strategy for the second player $\pi^* : X \setminus T \to U$ constructed by

$$\pi^*(x) \in \operatorname*{argmin}_{u \in U(x)} [Z(g(x, u)) + 1].$$

This is demonstrated in Example 4.3.

4. Numerical Examples and Computational Results

Example 4.1 (Two-tier Game)**.** Consider a two-tier pyramid game (i.e., $p = 2$). Then

$$L \;=\; \left[\frac{1+2}{2}\right] \;=\; 1, \quad x_0 \;=\; (1,1).$$

First, for the terminal state $(1,0)$,

$$W((1,0)) \;=\; L \;=\; 1, \;\; Z((1,0)) \;=\; 0.$$

Next, since $U((1,1)) \;=\; \big\{(1,1,1),\ (2,1,1),\ (2,1,2)\big\}$,

$$\begin{aligned} W((1,1)) &= \min_{u\in U(1,1)} \big[Z(g((1,1),u)) + 1\big] \\ &= \min\Big[\big(Z(g((1,1),(1,1,1))) + 1\big),\ \big(Z(g((1,1),(2,1,1))) + 1\big),\ \big(Z(g((1,1),(2,1,2))) + 1\big)\Big] \\ &= \min\Big[\big(Z((0,1)) + 1\big),\ \big(Z((2,0)) + 1\big),\ \big(Z((1,0)) + 1\big)\Big]. \end{aligned}$$

Moreover, $U((0,1)) = \big\{(2,1,1)\big\}$ and $U((2,0)) = \big\{(1,1,1)\big\}$ lead to the following:

$$\begin{aligned} Z((0,1)) \;&=\; \max_{u\in U((0,1))} \big[W(g((0,1),u))\big] \;=\; W((1,0)) \;=\; L \;=\; 1, \\ Z((2,0)) \;&=\; \max_{u\in U((2,0))} \big[Z(g((2,0),u))\big] \;=\; W((1,0)) \;=\; L \;=\; 1, \end{aligned}$$

respectively. Thus

$$W((1,1)) \;=\; \min\big[(1+1),\ (1+1),\ (0+1)\big] \;=\; 1, \qquad \pi^*((1,1)) \;=\; (2,1,2).$$

Because $W((1,1)) \le L$, there exists a strategy for winning the game and the optimal decision is $\pi^*((1,1)) = (2,1,2)$. In the first turn, the first player should mark both bars in the second tier. □

We now give some results as the corollary of the recursive equations (3.2) – (3.5). It is easily shown that

$$W((2k,0,0,\ldots,0)) \;=\; k, \qquad k = 1,2,\ldots \tag{4.1}$$

$$\pi^*((2k,0,0,\ldots,0)) \;=\; (1,1,1), \qquad k = 1,2,\ldots$$

$$W((2k+1,0,0,\ldots,0)) \;=\; L + k \;(>\; L), \qquad k = 1,2,\ldots \tag{4.2}$$

$$W((2k,\ldots,0,\overset{l\text{th}}{\underset{\downarrow}{1}},0,\ldots,0)) \;=\; k+1, \quad k = 0,1,\ldots\ ;\ \ l = 2,3,\ldots,p \tag{4.3}$$

$$\pi^*((2k,\ldots,0,\overset{l\text{th}}{\underset{\downarrow}{1}},0,\ldots,0)) \;=\; (l,1,l-1), \quad k = 0,1,\ldots\ ;\ \ l = 2,3,\ldots,p$$

$$W((2k+1,0,\ldots,0,\overset{l\text{th}}{\underset{\downarrow}{1}},0,\ldots,0)) \;=\; k+1, \quad k = 0,1,\ldots\ ;\ \ l = 2,3,\ldots,p \tag{4.4}$$

$$\pi^*((2k+1,0,\ldots,0,\overset{l\text{th}}{\underset{\downarrow}{1}},0,\ldots,0)) \;=\; (l,1,l), \quad k = 0,1,\ldots\ ;\ \ l = 2,3,\ldots,p$$

and

$$Z((2k+1,0,0,\ldots,0)) \;=\; k, \qquad k = 1,2,\ldots \tag{4.5}$$

$$Z((2k,0,0,\ldots,0)) \;=\; L + k - 1, \qquad k = 1,2,\ldots \tag{4.6}$$

$$Z((2k,0,\ldots,0,\overset{l\text{th}}{\underset{\downarrow}{1}},0,\ldots,0)) \;=\; L + k, \quad k = 0,1,\ldots\ ;\ \ l = 2,3,\ldots,p \tag{4.7}$$

$$Z((2k+1,0,\ldots,0,\overset{l\text{th}}{\underset{\downarrow}{1}},0,\ldots,0)) \;=\; L + k, \quad k = 0,1,\ldots\ ;\ \ l = 2,3,\ldots,p \tag{4.8}$$

hold.

Example 4.2 (Three-tier Game)**.** Consider a three-tier pyramid game (i.e., $p = 3$). Then

$$L \;=\; \left[\frac{1+2+3}{2}\right] \;=\; 3, \quad x_0 \;=\; (1,1,1).$$

First, for the terminal state $(1,0,0)$,

$$W((1,0,0)) \;=\; L \;=\; 3, \quad Z((1,0,0)) \;=\; 0.$$

Next, since $U((1,1,1)) \;=\; \big\{(1,1,1),\ (2,1,1),\ (2,1,2),\ (3,1,1),\ (3,1,2),\ (3,1,3),\ (3,2,2)\big\}$, we get

$$\begin{aligned} W((1,1,1)) \;&=\; \min_{u\in U((1,1,1))} \big[Z(g((1,1,1),u)) + 1\big] \\ &=\; \min\Big[\big(Z((0,1,1))+1\big),\ \big(Z((2,0,1))+1\big),\ \big(Z((1,0,1))+1\big), \\ &\quad \big(Z((1,2,0))+1\big),\ \big(Z((2,1,0))+1\big),\ \big(Z((1,1,0))+1\big),\ \big(Z((3,1,0))+1\big)\Big]. \end{aligned}$$

Equations (4.7) and (4.8) give

$$Z((1,0,1)) \;=\; Z((1,1,0)) \;=\; 3, \quad Z((2,0,1)) \;=\; Z((2,1,0)) \;=\; Z((3,1,0)) \;=\; 4,$$

and, by $U((0,1,1)) \;=\; \big\{(2,1,1),\ (2,1,2),\ (3,1,1),\ (3,1,2),\ (3,1,3),\ (3,2,2)\big\}$, we have

$$\begin{aligned} Z((0,1,1)) \;&=\; \max_{u\in U((0,1,1))} \big[W(g((0,1,1),u))\big] \\ &=\; \max\big[W((1,0,1)),\ W((0,0,1)),\ W((0,2,0)),\ W((1,1,0)),\ W((0,1,0)),\ W((2,1,0))\big]. \end{aligned}$$

From Equations (4.1) and (4.2),

$$W((1,0,1)) \;=\; W((0,0,1)) \;=\; W((1,1,0)) \;=\; W((0,1,0)) \;=\; 1, \quad W((2,1,0)) \;=\; 2.$$

Furthermore, $U((0,2,0)) \;=\; \big\{(2,1,1),\ (2,1,2)\big\}$, Equations (4.7) and (4.8) lead to

$$\begin{aligned} W((0,2,0)) \;&=\; \min_{u\in U((0,2,0))} \big[Z(g((0,2,0),u)) + 1\big] \\ &=\; \min\Big[\big(Z((1,1,0))+1\big),\ \big(Z((0,1,0))+1\big)\Big] \\ &=\; \min\big[(3+1),\ (3+1)\big] \;=\; 4\ (\;>\; L). \end{aligned}$$

Thus

$$Z((0,1,1)) \;=\; \max\big[1,\ 1,\ 4,\ 1,\ 1,\ 2\big] \;=\; 4.$$

Similarly,

$$U((1,2,0)) \;=\; \big\{(1,1,1),\ (2,1,1),\ (2,1,2)\big\},$$

$$\begin{aligned} Z((1,2,0)) \;&=\; \max_{u\in U((1,2,0))} \big[W(g((1,2,0),u))\big] \\ &=\; \max\big[W((0,2,0)),\ W((2,1,0)),\ W((1,1,0))\big] \\ &=\; \max\big[4,\ 2,\ 1\big] \;=\; 4. \end{aligned}$$

Therefore

$$W((1,1,1)) \;=\; \min\big[(4+1),\ (4+1),\ (3+1),\ (4+1),\ (4+1),\ (3+1),\ (4+1)\big] \;=\; 4.$$

Since the optimal value $W((1,1,1))$ exceeds $L=3$, the first player does not have a guaranteed strategy for winning the game. □

In the case that $p=3$, whatever decision is made by the first player, the second player has a counteracting winning decision. The next example demonstrates a guaranteed winning strategy for the second player.

Example 4.3 (Three-tier Game, Guaranteed Winning Strategy for the Second Player)**.** From Example 4.2, we have

$$L \;=\; 3, \quad x_0 \;=\; (1,1,1), \quad W((1,0,0)) \;=\; 3, \quad Z((1,0,0)) \;=\; 0,$$

and

$$\begin{aligned} Z((1,1,1)) \;&=\; \max_{u\in U((1,1,1))}\big[W(g((1,1,1),u))\big] \\ &=\; \max\big[W((0,1,1)),\ W((2,0,1)),\ W((1,0,1)),\ W((1,2,0)), \\ &\qquad\qquad W((2,1,0)),\ W((1,1,0)),\ W((3,1,0))\big]. \end{aligned}$$

By Equations (4.3) and (4.4),

$$W((1,0,1)) \;=\; W((1,1,0)) \;=\; 1, \quad W((2,0,1)) \;=\; W((2,1,0)) \;=\; W((3,1,0)) \;=\; 2.$$

Furthermore,

$$\begin{aligned} W((0,1,1)) \;&=\; \min_{u\in U((0,1,1))}\big[Z(g((0,1,1),u))+1\big] \\ &=\; \min\Big[\big(Z((1,0,1))+1\big),\ \big(Z((0,0,1))+1\big),\ \big(Z((0,2,0))+1\big), \\ &\qquad\qquad \big(Z((1,1,0))+1\big),\ \big(Z((0,1,0))+1\big),\ \big(Z((2,1,0))+1\big)\Big]. \end{aligned}$$

Equations (4.7) and (4.8) give

$$Z((1,0,1)) \;=\; Z((0,0,1)) \;=\; Z((1,1,0)) \;=\; Z((0,1,0)) \;=\; 3, \quad Z((2,1,0)) \;=\; 4,$$

and we have

$$\begin{aligned} Z((0,2,0)) \;&=\; \max_{u\in U((0,2,0))}\big[W(g((0,2,0),u))\big] \;=\; \max\big[W((1,1,0)),\ W((0,1,0))\big] \\ &=\; \max\big[1,\ 1\big] \;=\; 1. \end{aligned}$$

Then

$$\begin{aligned} W((0,1,1)) \;&=\; \min\big[(3+1),\ (3+1),\ (1+1),\ (3+1),\ (3+1),\ (4+1)\big] \;=\; 2, \\ \pi^*((0,1,1)) \;&=\; (3,1,1). \end{aligned}$$

Next

$$\begin{aligned} W((1,2,0)) \;&=\; \min_{u\in U((1,2,0))}\big[Z(g((1,2,0),u))+1\big] \\ &=\; \min\Big[\big(Z((0,2,0))+1\big),\ \big(Z((2,1,0))+1\big),\ \big(Z((1,1,0))+1\big)\Big] \\ &=\; \min\big[(1+1),\ (4+1),\ (3+1)\big] \;=\; 2, \\ \pi^*((1,2,0)) \;&=\; (1,1,1). \end{aligned}$$

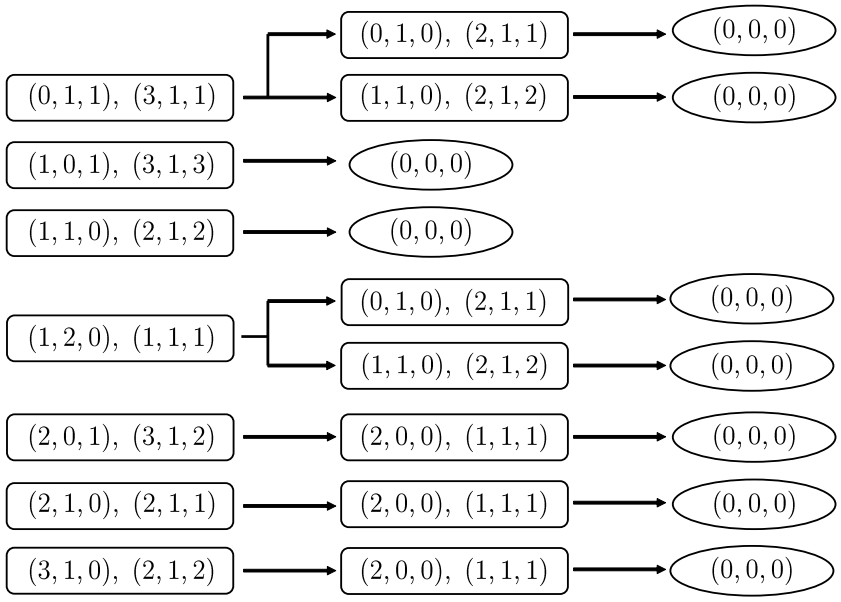

Figure 6: Optimal decision trees for the second player ($p = 3$)

Thus, we have

$$Z((1,1,1)) \;=\; \max\left[2,\ 2,\ 1,\ 2,\ 2,\ 1,\ 2\right] \;=\; 2 \;\le\; L.$$

The optimal decision trees for the second player are shown in Figure 6. In this figure, each state and its corresponding optimal decision are shown within a rectangular block and the final state within an oval block. □

Example 4.4 (Four-tier Game)**.** We now consider a four-tier pyramid game ($p = 4$). Then

$$L \;=\; \left[\frac{1+2+3+4}{2}\right] \;=\; 5, \quad x_0 \;=\; (1,1,1,1).$$

Similarly, using mutually dependent recursive equations (3.2) – (3.5), we have

$$W(x_0) \;=\; 3 \;\;(\;\le\; L), \qquad \pi^*(x_0) \;=\; (4,1,4).$$

Therefore, there exists a strategy in which the first player is guaranteed to win within three turns. The optimal decision in the first turn is to mark all bars in the fourth tier. His position becomes the same as that of the second player in the three-tier game and he should follow the strategy in Example 4.3. □

The turns taken by the first player until he wins are given in Table 1. Comparing each optimal value $W(x_0)$ and L, it can be seen that no guaranteed winning strategy exists for $p = 3, 7$, or 11. In these cases, the turns taken by the second player until he wins are given in parentheses.

5. Conclusions

In this paper, we introduced a model capable of finding a guaranteed way of winning a two-player combinatorial game of perfect knowledge using the concept of mutually dependent

Table 1: Optimal value

p	2	3	4	5	6	7	8	9	10	11	12	13
$W(x_0)(Z(x_0))$	1	5(2)	3	7	8	30(13)	14	22	23	67(32)	33	45
L	1	3	5	7	10	14	18	22	27	33	39	45

one-stage decision processes. This problem has previously been addressed in the framework of nondeterministic dynamic programming ([3, 5]). However, the recursive equations (3.2) – (3.5) are simpler than those presented in [5]. The MDDP model, in addition, executes optimizing operation for both players alternately. Therefore, it has advantages in computing. Another advantage is that our model can also find a guaranteed winning strategy for the second player.

Acknowledgements

The author would like to thank two anonymous referees for their careful reading and useful comments on the original manuscript. This research was supported by JSPS KAKENHI Grant Number 15K05004.

References

[1] E.R. Berlekamp, J.H. Conway, and R. K. Guy: *Winning Ways for Your Mathematical Plays Volume 1* (A K Peters, Ltd., 2001).

[2] U. Bertelé and F. Brioschi: *Nonserial Dynamic Programming* (Academic Press, New York, 1972).

[3] T. Fujita: On nondeterministic dynamic programming. *Mathematical Analysis in Economics (Kyoto, 2005) RIMS Kokyuroku*, **1488** (2006), 15–24.

[4] T. Fujita: Mutually dependent markov decision processes. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, **18** (2014), 992–998.

[5] T. Fujita: Sure ways to win a game — nondeterministic dynamic programming approach —. *Bulletin of the Kyushu Institute of Technology*, **62** (2015), 1–14.

[6] T. Fujita: Mutually dependent decision processes models. *Bulletin of the Kyushu Institute of Technology*, **63** (2016), 15–26.

[7] T. Fujita and A. Kira: Associative criteria in mutually dependent markov decision processes. *Proceedings of IIAI International Conference on Advanced Applied Informatics*, (2014), 147–150.

[8] T. Fujita and K. Nagatomo: On convex polyhedra constructed by paper units – an application of mutually decision processes –. *RIMS Kokyuroku*, **1912** (2014), 17–25 (in Japanese).

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

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