A dynamic programming algorithm for optimizing baseball strategies

Kira, Akifumi
Institute of Mathematics for Industry, Kyushu University

Inakawa, Keisuke
Faculty of Systems Science and Technology, Akita Prefectural University

Fujita, Toshiharu
Graduate School of Engineering, Kyushu Institute of Technology

Ohori, Kotaro
Knowledge Information Processing Laboratory, Fujitsu Laboratories Ltd.

https://hdl.handle.net/2324/1547352

A dynamic programming algorithm for optimizing baseball strategies

Akifumi Kira, Keisuke Inakawa, Toshiharu Fujita & Kotaro Ohori

A dynamic programming algorithm
for optimizing baseball strategies

Akifumi KIRA
Institute of Mathematics for Industory
Kyushu University
kira@imi.kyushu-u.ac.jp

Keisuke INAKAWA
Faculty of Systems Science and Technology
Akita Prefectural University
inakawa@akita-pu.ac.jp

Toshiharu FUJITA
Graduate School of Engineering
Kyushu Institute of Technology
fujita@mns.kyutech.ac.jp

Kotaro OHORI
Knowledge Information Processing Laboratory
Fujitsu Laboratories Ltd.
ohori.kotaro@jp.fujitsu.com

Abstract

In this paper, baseball is formulated as a finite Markov game with approximately 6.45 million states. We give an effective dynamic programming algorithm which computes Markov perfect equilibria and the value functions of the game for both teams in 2 second per game. Optimal decision making can be found depending on the situation—for example, for the batting team, whether batting for a hit, stealing a base or sacrifice bunting will maximize their win percentage, or for the fielding team, whether to pitch to or intentionally walk a batter, yields optimal results. In addition, our algorithm makes it possible to compute the optimal batting order, in consideration of strategy optimization such as a sacrifice bunt or a stolen base. The authors believe that this baseball model is also useful as a benchmark instance for evaluating the performances of (multi-agent) Reinforcement Learning methods.

Keywords Markov game, Markov perfect equilibrium, dynamic programming, intentional walk, advantage of the last-batting team, optimal lineup.

1 Introduction

In the field of mathematical science, the first research paper evaluating batting in baseball is said to be by Lindsey [14]. In his paper, baseball is analyzed by a statistical method. A dynamic programming (DP) approach to baseball is the main theme for this paper, and we first see a prototype of this idea in Howard's famous book [8]. Howard set maximization of the expected number of runs scored for one inning as a criterion, formulating baseball as a Markov decision

process with 25 states. Orders from the manager, such as base stealing, sacrifice bunting, and batting for a hit, were also taken into consideration. Howard's work is based on the assumption that all nine batters on the team have equal abilities. The transition probabilities (the success rate of sacrificing, etc.) were artificially set and the optimal strategies for a manager were determined using a computer of that time. Bellman [1] proposes a more detailed formulation. His model analyzes not only the batter-by-batter level, but also the pitch-by-pitch level. He provided a subtle insight into strategies through a discussion based on the two criteria of maximizing the expected number of runs scored and the threshold probability of scoring at least k runs in one inning. However, due to the shortage of computing capacity at the time, Bellman's approach was not implemented.

On the other hand, at the same time as the work of Bellman, in 1977, two papers were published that use a Markov chain approach with matrix analysis. D'Esopo and Lefkowitz [6] propose a scoring index (SI) as an evaluation index for the expected number of runs scored in one inning, assuming that the same player steps up to the plate repeatedly. Under the same assumption, Cover and Keilers [5] propose a similar index, the OERA (Offensive Earned-Run Average) value, as an index to evaluate the expected number of runs scored in a single game. In the OERA model, the baseball rules are simplified to apply the absorbing Markov chain model in the calculation of the expected runs scored. While Howard and Bellman focused on strategy optimization, the goal of the matrix analysis approach is to express the contributions of each individual player in numerical form. From this point on, this method became popular. Bukiet et al. [4] in 1997 take the batting order into consideration. Their algorithm can calculate the expected runs scored in one inning, assuming the nine players constituting the team step up to the plate in a given order. However, the expected runs scored in a game can not be obtained by a simple multiplication.[1] Therefore, they propose a heuristic method.

Now, half a century from Howard's proposal, the ability of computers has rapidly developed and an approach using DP has become possible. In 2008, Turocy [19] produces a model to also couple the strategies of the opposing team with DP approach by using a Markov game, which is a multi-agent extension of MDPs (e.g. see Shapley [17] and Zachrisson [24]). Here, the manager of each opposing team is the game-theoretic player who maximizes the probability of their team winning. As an order from the manager, intentional walk is also taken into consideration. Since the model adopts the MLB rule to play extra innings until a winner is determined (i.e., the length of the game is finite with probability 1), the states for an extra inning can be identified as being the same as for the ninth inning. In addition, by establishing an upper limit of 30 runs for the run difference (mercy-rule), his stochastic game has a finite number of states and a finite number of actions, so a Markov perfect equilibrium (MPE) exists. The total number of states in this case is approximately 2.13 million. Turocy performed numerical experiments using backward induction from the start of the game up to the completion of the eighth inning, and using a fixed-point approximation by a value-function iteration for the ninth inning. The details of the recursive formula and the algorithm are omitted in the paper, but he states that the values of the game (the equilibrium winning percentages for both teams) could be solved with high accuracy in less than a minute.

In our previous paper [10, 11], we formulate baseball as a finite Markov game with approximately 3.5 million states. We also suppose that the manager of each team maximizes the probability of their team winning. The principal advantages compared to Turocy's model are: (a) to consider the success rate with stolen bases in more detail, the identity of the runner on first base is stored in the state, and (b) the rule from Japanese professional baseball that extra innings are restricted to a maximum of three is included, so the game may end in a draw. Because of the mercy-rule and the finiteness of the number of extra innings, the length of the

---

[1]If the same player steps up to the plate repeatedly, nine times the expected runs scored in one inning equals that scored in a game.

game also becomes finite. Hence, at least one pure-strategy MPE does exist in this model. We derive a recursive formula that is satisfied by the MPEs and the value functions of the game. By solving this, we realize to calculate the value functions of the game and a MPE in approximately 1 second per game. However, the details of the algorithm is omitted in our previous paper. In addition, another disadvantage is that intentional walk is not taken into consideration.

In this paper, we take intentional walk into consideration and reformulate baseball as a finite Markov game with approximately 6.45 million states. When a batter steps up to the plate, we always assume that the team in defence chooses their action whether to pitch to or to intentionally walk the batter before the team in offence chooses their action whether batting for a hit, stealing a base or sacrifice bunting. Hence our model is still a sequential game and there exists at least one pure-strategy MPE. We describe a depth-first search DP algorithm for effectively computing the value functions of the game for both teams and pure-strategy MPEs in detail.

In the theory of finite MDPs, the usual optimization criterion is to maximize the expected value of the total (discounted) sum of stage-wise rewards. Is is well known that this criterion can be solved in polynomial time. On the other hand, the threshold probability problem, which attempts to maximize the probability that the total sum of stage-wise rewards exceeds a specified value, has been extensively studied by many researchers (Boda and Filar [2], Bouakiz and Kebir [3], Kira et al. [12], Ohtsubo and Toyonaga [16], Sobel [18], White [20, 21], Wu and Lin [22], and others), and this problem has been proved to be $\mathcal{NP}$-hard (Xu and Mannor [23]). So, handling with probability criteria is more difficult than that with the usual expectation criterion in general. The same applies to Markov games. However, in baseball, our algorithm runs in less than 2 second per game. It will be clearly understood, through this paper, that baseball possesses some properties quite suitable for DP computation. In addition, although our baseball model is a large-scale Markov game, the optimal value functions can be found. Therefore, the authors believe that our model is also useful as a benchmark for (multi-agent) Reinforcement Learning methods.

In Section 2, we provide a finite Markov game as a formulation for baseball. States, actions, state transitions with simplifying rules, and payoff functions are all defined. In Section 3, we define MPEs and the value functions of the game, and derive the recursive formula that is satisfied by them. By solving this, the value functions of the game and MPEs are obtained. In Section 4, a depth-first search DP algorithm for effectively solving the recursive formula is described in detail. Section 5 discusses whether the last-batting team has an advantage. Section 6 shows our results for computing the optimal lineup and the worst lineup for the Fukuoka Softbank Hawks. So far as we know, there has been no study that has tried to optimize batting order, in consideration of strategy optimization such as a sacrifice bunt or a stolen base. Our effort reducing the computational time per game makes it possible. Concluding remarks and future direction are discussed in Section 7.

2   Formulation as a Markov game

Most decision problems in the real world require multi-stage decisions, where successive decisions need to be made while taking into account changes in the situation arising from the results of previous decisions, rather than where decision making takes place just once. Dynamic programming (DP) is an optimization method to efficiently solve these kinds of problems. However, in order to solve them using DP, the problem needs to be properly formulated as a mathematical model. One such mathematical model is the Markov game. In a Markov game, the following three elements are considered: (i) successive decision making conducted over time; (ii) uncertainty of changes in the situation; (iii) multiple decision makers competing against one another. The word "uncertainty" needs to be treated with caution as it can mean completely different

things depending on the community; however, here it refers to a situation whereby future events can be probabilistically estimated from past data.

In this section, baseball is formulated as a finite Markov game. For convenience, we call the first-batting team and the last-batting team "team 0" and "team 1" respectively.

2.1 States

Let $\mathcal{S}$ be the state space. A state $s = (\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) \in \mathcal{S}$ is made up of 7 components. Each component is defined as follows:

1. $\iota \in \{1, 2, \ldots, 12\}$ represents the current inning. $\iota = 9$ is the final inning, and for a tie, extra innings are played up to a maximum of $\iota = 12$.

2. $\tau \in \{0, 1\}$ represents offense in the top half of the inning ($\tau = 0$) or offense in the bottom half of the inning ($\tau = 1$).

3. $\omega \in \{0, 1, 2, 3\}$ represents the current number of outs.

4. λ is the current run difference and represents the value found by subtracting the runs of team 0 from the runs of team 1. For the purpose of determining the final winner, we store not the runs scored by each team but the current run difference. This is a state aggregation technique (e.g. see Sniedovich [15, Chap. 11]).

5. $\boldsymbol{r} = (r_3, r_2, r_1)$ represents the state of the runners.

   - $r_3 \in \{0, 1\}$ takes a value 0 if there is no runner on third base, and a value 1 if a runner is present.
   - $r_2 \in \{0, 1\}$ takes a value 0 if there is no runner on second base, and a value 1 if a runner is present.
   - $r_1 \in \{0, 1, \ldots, 9\}$ takes a value 0 if there is no runner on first base, and the same value as the batting order of the runner if a runner is present.

   Only r_1 distinguishes between runners, to take into account the success rate which is dependent on the runner when performing a stolen base from first to second base. In this paper, neither a stolen base from second to third base, nor a stolen base from third to home base, are considered.

6. $\boldsymbol{b} = (b_0, b_1)$ is made up of 2 components. $b_i \in \{1, 2, \ldots, 9\}$ indicates to which batter the batting order of team i rotates ($i = 0, 1$). It represents, when in offense ($\tau = i$), that the b_i-th batter steps up to the plate. When in defense ($\tau \neq i$), it means that the leadoff hitter in the next inning is the b_i-th batter.

7. $m \in \{0, 1\}$ is the index of the team which is on move. In this state, team m can choose their actions.

By the above definition, the initial state s_0 at the start time of the game is as follows:

$$s_0 = (\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m)_0 = (1, 0, 0, 0, (0, 0, 0), (1, 1), 1).$$

$\mathcal{S}_Q$ denotes the total states (absorbing states) at the end of the game:

$$\mathcal{S}_Q := \mathcal{S}_Q^0 \cup \mathcal{S}_Q^1 \cup \mathcal{S}_Q^{\text{tie}} \cup \mathcal{S}_Q^{\text{m}}(0) \cup \mathcal{S}_Q^{\text{m}}(1) \subset \mathcal{S},$$

where

$$\mathcal{S}_Q^0 = \{(\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) \in \mathcal{S} \mid \iota \geq 9,\ \tau = 1,\ \omega = 3,\ \lambda > 0\},$$

$$\mathcal{S}_Q^1 = \{(\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) \in \mathcal{S} \mid \iota = 9,\ \tau = 0,\ \omega = 3,\ \lambda < 0\}$$
$$\cup \{(\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) \in \mathcal{S} \mid \iota \geq 9,\ \tau = 1,\ \lambda < 0\},$$

$$\mathcal{S}_Q^{\mathrm{tie}} = \{(\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) \in \mathcal{S} \mid \iota = 12,\ \tau = 1,\ \omega = 3,\ \lambda = 0\},$$

$$\mathcal{S}_Q^{\mathrm{m}}(0) = \{(\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) \in \mathcal{S} \mid \lambda \geq 30\}, \quad \text{(mercy-rule)}$$

$$\mathcal{S}_Q^{\mathrm{m}}(1) = \{(\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) \in \mathcal{S} \mid \lambda \leq -30\}. \quad \text{(mercy-rule)}$$

$\mathcal{S}_Q^0$ and $\mathcal{S}_Q^1$ correspond to a victory for team 0 and a victory for team 1, respectively. $\mathcal{S}_Q^{\mathrm{tie}}$ corresponds to a tie in the 12th inning after playing extra innings. We note that the mercy-rule refers to the establishment of a called game during the inning. By adopting both the Japanese professional baseball rules (i.e., the number of extra innings is finite) and the above mercy-rule, we get a finite Markov game. In addition, we equate the state s, such that $\omega = 3$ and $s \notin \mathcal{S}_Q$, with the corresponding state after the inning is over.

2.2 Actions

The manager of each team is the game-theoretic player maximizing the probability of their team winning. Here we let $\mathcal{S}_i$ be the set of states of moves for team i. Namely,

$$S_i = \{(\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) \in \mathcal{S} \setminus \mathcal{S}_Q \mid m = i\}, \quad i = 0, 1.$$

In $\mathcal{S}_0 \cup \mathcal{S}_1$, there are a lot of states that are not reachable from the initial state s_0. For example, $(\tau, r_1, b_0) = (0, 1, 2)$ is feasible, but $(0, 2, 1)$ is infeasible. We find out that the number of reachable states in $\mathcal{S}_0 \cup \mathcal{S}_1$ is $6{,}454{,}296$ by coding a computer program for counting them. In this paper, the action space is defined as

$$\mathcal{A} = \mathcal{A}^{\mathrm{defence}} \cup \mathcal{A}^{\mathrm{offence}},$$

where

$$\mathcal{A}^{\mathrm{defence}} = \{\text{pitching, intentional walk}\},$$
$$\mathcal{A}^{\mathrm{offence}} = \{\text{batting, stolen base, sacrifice bunt}\}.$$

Let us consider a point-to-set valued mapping $\mathcal{A} : \mathcal{S} \setminus \mathcal{S}_Q \to 2^{\mathcal{A}} \setminus \{\phi\}$. $\mathcal{A}(s)$, called the feasible action space, represents the set of all actions in state s. In this paper, we define $\mathcal{A}(s)$, for any $s = (\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) \in \mathcal{S} \setminus \mathcal{S}_Q$ in the following manner:

$$\text{pitching} \in \mathcal{A}(s) \iff \tau \neq m,$$
$$\text{intentional walk} \in \mathcal{A}(s) \iff \tau \neq m,$$
$$\text{batting} \in \mathcal{A}(s) \iff \tau = m,$$
$$\text{stolen base} \in \mathcal{A}(s) \iff \tau = m, \quad r_2 = 0, \quad r_1 \geq 1,$$
$$\text{sacrifice bunt} \in \mathcal{A}(s) \iff \tau = m, \quad \omega \leq 1, \quad r_3 + r_2 + r_1 \geq 1.$$

Hence, stolen bases are feasible if and only if there is no second base runner and there is a runner present on first base, and sacrifice hits are feasible if and only if a runner is present with 0 or 1 outs.

2.3 State transitions

For any $a \in \mathcal{A}$, let us define $\mathcal{X}(a)$, the set of all results that can occur stochastically when the action a is chosen, as follows:

$$\mathcal{X}(a) = \begin{cases} \{\text{game}\} & \text{if } a = \text{pitching}, \\ \{\text{walk}\} & \text{if } a = \text{intentional walk}, \\ \{\text{out, single, double, triple, home run, walk}\} & \text{if } a = \text{batting}, \\ \{\text{success, fail}\} & \text{otherwise}. \end{cases}$$

We denote the graph of $\mathcal{A}(\cdot)$ by $G_r(\mathcal{A})$. Namely,

$$G_r(\mathcal{A}) = \{(s, a) \mid a \in \mathcal{A}(s), \ s \in \mathcal{S}\}.$$

For any $(s, a) \in G_r(\mathcal{A})$ and any $x \in \mathcal{X}(a)$, $p(x \mid s, a)$ represents the conditional probability with which the result x occurs, given that action a is chosen in state s.

$$p(\cdot \mid s, a) : \mathcal{X}(a) \to [0, 1], \quad \forall (s, a) \in G_r(\mathcal{A}),$$
$$\sum_{x \in \mathcal{X}(a)} p(x \mid s, a) = 1, \quad \forall (s, a) \in G_r(\mathcal{A}).$$

With our definition, the state $s = (\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m)$ includes information about the inning number, whether it is the top or bottom half, the out count, the run difference, the batting order of the batters, whether there are runners present or not, and also, the batting order of the runner if one is present on first base. Therefore, the transition probability generally depends on all of these states. However, in the numerical experiments carried out in Section 5 and Section 6, we assume that the transition probability depends only on the players that make a hit, or perform sacrifice hits or stolen bases, and does not depend on those other components which comprise the state. Table 1 shows the probability parameters for the starting order of the Fukuoka Softbank Hawks, and was compiled based on the values achieved in Japan's professional baseball 2014 season [25, 26].

Table 1: Probability parameters

Name	AVG	Hitting						Stolen Base	Sacrifice Hit
		Out	Single	Double	Triple	HR	Walk	Success	Success
1 Y. Honda	.291	.648	.217	.032	.016	.000	.087	.793	.941
2 A. Nakamura	.308	.627	.231	.035	.006	.006	.095	.833	.800 *
3 Y. Yanagita	.317	.593	.211	.029	.007	.025	.136	.846	.000
4 S. Uchikawa	.307	.653	.199	.049	.002	.034	.063	.000	.000
5 Lee Dae-Ho	.300	.637	.195	.048	.000	.031	.090	.000	.000
6 Y. Hasegawa	.300	.624	.193	.056	.006	.011	.110	.500	.000
7 N. Matsuda	.301	.655	.185	.048	.007	.043	.062	.667	.500
8 S. Tsuruoka	.216	.750	.167	.024	.018	.000	.042	.000	.944
9 K. Imamiya	.240	.698	.174	.044	.002	.005	.077	.667	.873

* The minimum of 0.8 and the actual value was adopted for the sacrifice hit success rate for players with extremely small numbers of attempted sacrifice hits (less than 4) over the year.

This paper simplifies baseball in a similar manner to previous research. The simplifying rules used in this paper are as below.

"Simplifying rules"

1. With a mishit (an out), neither a batter nor a runner can advance bases.

2. A single advances a runner on first base to third base, and runners on second and third base reach the home plate.

3. A double and a triple allows all runners to reach the home plate.

4. It is assumed that there are no double plays.

5. For a successful stolen base, the runner on first base advances to second base.

6. For an unsuccessful stolen base, the runner on first base is out.

7. For a successful sacrifice hit, the runners advance one base forward, and the batter performing the sacrifice hit is out.

8. For an unsuccessful sacrifice hit, the runner closest to the home plate is out, the other runners advance one base forward, and the batter is then the runner on first base.

If these simplifying rules are followed, then the next state s' is determined uniquely when in state s, action a is chosen, and result x occurs. We denote this next state by

$$s' = t(s, a, x).$$

Figure 1 illustrates the two-step transitions from a state with a runner on first with one out.
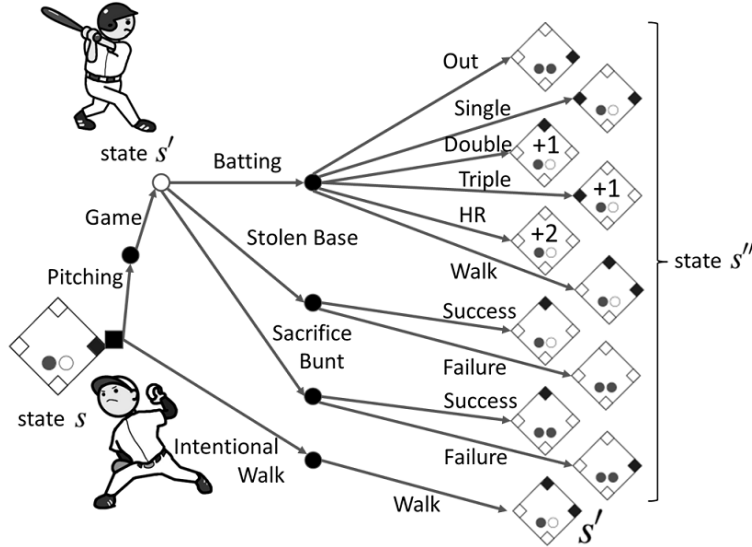
Figure 1: A part of the game tree

2.4 Markov policies

As a class of allowable policies, we consider the following class of Markov policies. The reason why we restrict our attention to this class will be stated in Section 3.

Definition 2.1 (Markov policy). *A mapping $\pi_i : \mathcal{S}_i \to \mathcal{A}$ is called a* (deterministic) *Markov policy for team i if $\pi_i(s) \in \mathcal{A}(s)$ for all $s \in \mathcal{S}_i$ ($i = 0, 1$). We denote the set of all deterministic Markov policies for team i by Π_i ($i = 0, 1$).*

Suppose that Markov policy π_i is employed by team i. In this case, the Markov game commencing from each state s can be regarded as a Markov chain. In other words, if we let X_n be the state after n step transition from the initial state s_0, then $\{X_n\}$ is the Markov chain satisfying

$$\mathrm{P}^{\pi_0,\pi_1}(X_{n+1} = s' \mid X_n = s) = \begin{cases} p(x \mid s, \pi_i(s)) & \text{if } s \in \mathcal{S}_i, \ s' = t(s, \pi_i(s), x), \\ 1 & \text{if } s \in \mathcal{S}_Q, \ s' = s, \\ 0 & \text{otherwise,} \end{cases}$$

where P^{π_0,π_1} represents the conditional probability given that the policy π_i is employed by team i with $i = 0, 1$. Let T be the arrival time of $\{X_n\}$ to $\mathcal{S}_Q$. Namely,

$$T := \min\{n \mid X_n \in \mathcal{S}_Q\} < \infty.$$

We denote the probabilities of team i winning by $v_i(s; \pi_0, \pi_1)$.

$$v_i(s; \pi_0, \pi_1) := \mathrm{P}^{\pi_0,\pi_1}\left(X_T \in \mathcal{S}_Q^i \cup \mathcal{S}_Q^{\mathrm{m}}(i) \mid X_0 = s\right), \quad s \in \mathcal{S}, \quad (\pi_0, \pi_1) \in \Pi_0 \times \Pi_1, \quad i = 0, 1.$$

2.5 Payoff functions

For any Borel set $\mathcal{B}$ and any random variable X, we know the relation

$$\Pr(X \text{ is in } \mathcal{B}) = \mathrm{E}[\mathbf{1}_{\mathcal{B}}(X)], \tag{1}$$

where

$$\mathbf{1}_{\mathcal{B}}(X) = \begin{cases} 1 & \text{if } X \text{ is in } \mathcal{B}, \\ 0 & \text{otherwise}. \end{cases}$$

Thus, let us define team i's terminal payoff function $\psi_i : \mathcal{S}_Q \to \{0, 1\}$ as follows:

$$\psi_i(s) = \begin{cases} 1 & (-1)^i \lambda > 0, \\ 0 & \text{othewise}, \end{cases} \quad s = (\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) \in \mathcal{S}_Q, \quad i = 0, 1.$$

If the game is won, a payoff of 1 is acquired, whereas a loss or a tie is a payoff of 0. This value depends only on the current run difference λ. Now, we can rewrite $v_i(s; \pi_0, \pi_1)$ as follows:

$$v_i(s; \pi_0, \pi_1) = \mathrm{E}^{\pi_0,\pi_1}\left[\psi_i(X_T) \mid X_0 = s\right], \quad s \in \mathcal{S}, \quad (\pi_0, \pi_1) \in \Pi_0 \times \Pi_1, \quad i = 0, 1,$$

where E^{π_0,π_1} represents the conditional expectation given that the policy π_i is employed by team i with $i = 0, 1$. The approach of using the relation (1) to reduce a probability criterion to the usual expectation criterion is often used in the field of MDPs (see Kira et al. [12]).

From the viewpoint of the theoretical framework of Markov games, the runs of team i may be treated as a reward system. In our model, to express the problem in the form of the usual expectation criterion, we store information about the runs scored as a component of the states. In the field of MDPs, such the state space $\mathcal{S}$ is called the augmented state space. In the most general context of Markov games, the sum of rewards, for any state, earned up to that point depends on the history of the process. This indicates that the cardinality of the augmented state space increases exponantially with the length of the game. However, in our baseball model, it is sufficient and efficient to sotre the run difference λ and it only takes small integer values. This property is quite suitable for DP computation.

3 Markov perfect equilibria and recursive formula

In this section, we define an MPE and the value functions of the game, and derive the recursive formula for effectively computing them.

An MPE is a profile of Markov policies that yields a Nash equilibrium in every proper subgame.

Definition 3.1 (Markov perfect equilibrium, MPE). *A profile of* (deterministic) *Markov policies* (π_0^*, π_1^*) *is called a* (pure-strategy) *MPE if it is a subgame perfect equilibrium. Namely, it satisfies*

$$v_0(s; \pi_0, \pi_1^*) \leq v_0(s; \pi_0^*, \pi_1^*), \quad \forall s \in \mathcal{S}, \quad \forall \pi_0 \in \Pi_0,$$
$$v_1(s; \pi_0^*, \pi_1) \leq v_1(s; \pi_0^*, \pi_1^*), \quad \forall s \in \mathcal{S}, \quad \forall \pi_1 \in \Pi_1.$$

Remark 3.1 (existence of the equilibria). *In the most general context, the action chosen by a policy in each state may be randomized. However, it is well-known that at least one pure-strategy MPE exists for a finite Markov game with perfect information (e.g. see Fundenberg and Tirole [7], Chap.13, p.516)[2]. We thus restrict our attention to the class of pure-strategy MPEs.*

Definition 3.2 (the value function of the game). *Let (π_0^*, π_1^*) be a MPE, and for any state $s \in \mathcal{S}$, let $V_i(s)$ be the probability of team i winning when in state s. That is,*

$$V_i(s) = v_i(s; \pi_0^*, \pi_1^*), \quad s \in \mathcal{S}.$$

Then the function V_i is called the value function of the game for team i.

Remark 3.2 (uniqueness of the value functions). *Every MPE is a Nash equilibrium. It follows from Kuhn's theorem [13] that the value function of the game for each team is unique. In other words, all equilibria must result in the same probability of the team winning.*

Theorem 3.1 (Bellman equation). *The value functions and any MPE (π_0^*, π_1^*) satisfy the following recursive formula.*

$$V_i(s) = \begin{cases} \psi_i(s) & s \in \mathcal{S}_Q, \\ \displaystyle\operatorname*{Max}_{a \in \mathcal{A}(s)} \sum_{x \in \mathcal{X}(a)} V_i(t(s,a,x)) p(x|s,a) & s \in \mathcal{S}_i, \\ \displaystyle\sum_{x \in \mathcal{X}(\pi_j^*(s))} V_i(t(s, \pi_j^*(s), x)) p(x|s, \pi_j^*(s)) & s \in \mathcal{S}_j. \end{cases}$$

$$\pi_i^*(s) \in \operatorname*{arg\,max}_{a \in \mathcal{A}(s)} \sum_{x \in \mathcal{X}(a)} V_i(t(s,a,x)) p(x|s,a), \quad s \in \mathcal{S}_i,$$

where $(i,j) = (0,1), (1,0)$.

Proof. As the initial condition for backward induction, we have

$$V_i(s) = \psi_i(s), \quad s \in \mathcal{S}_Q, \quad i = 0, 1.$$

Suppose that we are now in position to evaluate $V_0(s)$ and $V_1(s)$ for some state $s \in \mathcal{S}_0 \cup \mathcal{S}_1$, and suppose that we have evaluated $V_0(\cdot)$ and $V_1(\cdot)$ for all accessible states in one-step transition

---

[2]This fact is an immediate consequence of the well-known Kuhn's theorem [13]. Kawasaki et al. [9] give another proof using a discrete fixed point theorem.

from s. If the team on move in the state s choses an action $a \in \mathcal{A}(s)$, and if each team does their best in the subsequent subgame, then the winning percentages of both teams are

$$\sum_{x \in \mathcal{X}(a)} V_i(t(s, a, x)) p(x|s, a), \quad i = 0, 1.$$

Therefore, any MPE (π_0^*, π_1^*) must satisfy

$$\pi_i^*(s) \in \arg\max_{a \in \mathcal{A}(s)} \sum_{x \in \mathcal{X}(a)} V_i(t(s, a, x)) p(x|s, a),$$

where i is such that $s \in \mathcal{S}_i$. We thus obtain the result by backward induction. $\qquad\square$

4 Dynamic programming algorithm

In the previous section, we have obtained the recursive formula satisfied by the optimal value functions and any MPEs. By solving this, the optimal equilibrium strategies for each state, such as a sacrifice bunt or a stolen base, are obtained.

In the theory of finite Markov games (including finite MDPs), any general-purpose algorithm takes quadratic time with respect to the size of the state space in worst case. This time complexity is required because all the states must be evaluated and all the states may be accessible from all the states in one-step transition. However, in our baseball model, the number of all accessible states in one-step transition from any state s can be counted on both hands (See Figure 1). This indicates that we can construct a specialized algorithm which takes linear time with the size of the state space. We realize it by the use of memoized recursion. The algorithm can be implemented as described in Algorithm 1.

Data: an instance of the transition probabilities $p(\cdot \,|\, s, a) : \mathcal{X}(a) \to [0, 1], \ (s, a) \in G_r(\mathcal{A})$.
Result: the optimal value functions V_0 and V_1 and an pure-strategy MPE $\pi^* = (\pi_0^*, \pi_1^*)$
Initialize $V_0(s)$ and $V_1(s)$ to -1 for all $s \in \mathcal{S}$;
$s_0 = (\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m)_0 \leftarrow (1, 0, 0, 0, (0, 0, 0), (1, 1), 1)$;
Call Evaluate(arguments: s_0);

Algorithm 1: A dynamic programming algorithm for solving the baseball game

The memoized recurisive function Evaluate(parameters: $s = (\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) \in \mathcal{S}$), which evaluates $V_i(s)$ with $i = 0, 1$, can be implemented as follows:

```
if s ∈ S_Q then
 | V_i(s) ← ψ_i(s) for i = 0, 1;
else
    Declare local variables: an integer j, a floating-point variable temp, and a state s′;
    if m = 0 then
     | j ← 1;
    else
     | j ← 0;

    forall the a ∈ A(s) do
        temp ← 0;
        forall the x ∈ X(a) such that p(x|s, a) > 0 do
         | s′ ← Call Transition(arguments: s, a, x);
         | if V_m(s′) = −1 then
         |  | Call Evaluate(arguments: s′);
         | temp ← temp + V_m(s′)p(x|s, a);
        if V_m(s) < temp then
         | V_m(s) ← temp,    π*_m(s) ← a;

    V_j(s) ← 0;
    forall the x ∈ X(π*_m(s)) such that p(x|s, π*_m(s)) > 0 do
     | s′ ← Call Transition(arguments: s, π*_m(s), x);
     | V_j(s) ← V_j(s) + V_j(s′)p(x|s, π*_m(s));
```

Function Evaluate(parameters: $s = (\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) \in \mathcal{S}$)

This memoized recursion solves the recursive formula just for reachable states from the initial state by implementing the depth-first search of the game tree. The function Transition(parameters: s, a, x) returns the next state $s′ = t(s, a, x)$, and can be implemented as follows:

```
switch the value of a do
    case pitching
     | (Do nothing)
    case intentional walk or batting
     | s ← Call TransBatting(arguments: s, x);
    case stolen base
     | s ← Call TransStolenBase(arguments: s, x);
    case sacrifice bunt
     | s ← Call TransSacrificeBunt(arguments: s, x);

if ω = 3, s ∉ S_Q then
 | s ← Call InningIsOver(arguments: s);
else if a ≠ intentional walk then
 | m ← (m + 1)  mod 2;
return s;
```

Function Transition(parameters: $s = (\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) \in \mathcal{S}, a \in \mathcal{A}(s), x \in \mathcal{X}(a)$)

```
switch the value of x do
    case out
    |   ω ← ω + 1;
    case single
    |   λ ← λ + (−1)^τ(r₃ + r₂),    r₃ ← 𝟙_{>0}(r₁),    r₂ ← 0,    r₁ ← b_τ;
    case double
    |   λ ← λ + (−1)^τ(r₃ + r₂ + 𝟙_{>0}(r₁)),    r₃ ← 0,    r₂ ← 1,    r₁ ← 0;
    case triple
    |   λ ← λ + (−1)^τ(r₃ + r₂ + 𝟙_{>0}(r₁)),    r₃ ← 1,    r₂ ← 0,    r₁ ← 0;
    case home run
    |   λ ← λ + (−1)^τ(r₃ + r₂ + 𝟙_{>0}(r₁) + 1),    r₃ ← 0,    r₂ ← 0,    r₁ ← 0;
    case walk
        if r₃ = r₂ = 𝟙_{>0}(r₁) = 1 then
        |   λ ← λ + (−1)^τ;
        else
            Declare a local integer variable n;
            n ← min{j ∈ {1, 2, 3} | r_j = 0};
            if n ≥ 2 then
            |   r_n ← 1;
        r₁ ← b_τ;
b_τ ← b_τ  mod 9 + 1;
return s;
```
$$\lambda \leftarrow \lambda + (-1)^{\tau}(r_3 + r_2), \quad r_3 \leftarrow \mathbf{1}_{>0}(r_1), \quad r_2 \leftarrow 0, \quad r_1 \leftarrow b_\tau;$$
$$\lambda \leftarrow \lambda + (-1)^{\tau}(r_3 + r_2 + \mathbf{1}_{>0}(r_1)), \quad r_3 \leftarrow 0, \quad r_2 \leftarrow 1, \quad r_1 \leftarrow 0;$$
$$\lambda \leftarrow \lambda + (-1)^{\tau}(r_3 + r_2 + \mathbf{1}_{>0}(r_1)), \quad r_3 \leftarrow 1, \quad r_2 \leftarrow 0, \quad r_1 \leftarrow 0;$$
$$\lambda \leftarrow \lambda + (-1)^{\tau}(r_3 + r_2 + \mathbf{1}_{>0}(r_1) + 1), \quad r_3 \leftarrow 0, \quad r_2 \leftarrow 0, \quad r_1 \leftarrow 0;$$

Function TransBatting(parameters: $s = (\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) \in \mathcal{S}$, $x \in \mathcal{X}(\text{batting})$)

```
switch the value of x do
    case success
    |   r₂ ← 1;
    case fail
    |   ω ← ω + 1;
r₁ ← 0;
return s;
```

Function TransStolenBase(parameters: $s \in \mathcal{S}$, $x \in \mathcal{X}(\text{stolen base})$)

```
switch the value of x do
    case success
    |    λ ← λ + (−1)^τ r_3,    r_3 ← r_2,    r_2 ← 1_{>0}(r_1),    r_1 ← 0;
    case fail
    |    Declare a local integer n and set n ← max{i | r_i > 0};
    |    if n = 3 then
    |    |    r_3 ← r_2,    r_2 ← 1_{>0}(r_1);
    |    else if n = 2 then
    |    |    r_2 ← 1_{>0}(r_1);
    |    r_1 ← b_τ;
ω ← ω + 1,    b_τ ← b_τ  mod 9 + 1;
return s;
```

Function TransSacrificeBunt(parameters: $s \in \mathcal{S}$, $x \in \mathcal{X}$(sacrifice bunt))

```
if τ = 0 then
|    τ ← 1;
else
|    ι ← ι + 1,    τ ← 0;
ω ← 0,    r_3 ← 0,    r_2 ← 0,    r_1 ← 0;
return s;
```

Function InningIsOver(parameters: $s = (\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) \in \mathcal{S}$)

5   Advantage of the last-batting team

In baseball, there is often talk of whether the last-batting team has an advantage. In Japanese professional baseball games, the visiting team bats first and the home team bats second. The wins and losses for home and visiting teams in the 2014 season are shown in Table 2.

Table 2: Win-loss records by home/road (2014 season)

TEAM	G	HOME				ROAD			
		W	L	D	PCT	W	L	D	PCT
Giants	144	44	27	1	.611	38	34	0	.528
Tigars	144	41	30	1	.569	34	38	0	.472
Carp	144	42	29	1	.583	32	39	1	.444
Dragons	144	35	35	2	.486	32	38	2	.444
Baystars	144	34	37	1	.472	33	38	1	.458
Swallows	144	32	39	1	.444	28	42	2	.389
Hawks	144	45	24	3	.625	33	36	3	.458
Buffaloes	144	46	26	0	.639	34	36	2	.472
Fighters	144	43	29	0	.597	30	39	3	.417
Marines	144	37	33	2	.514	29	43	0	.403
Lions	144	31	38	3	.431	32	39	1	.444
Eagles	144	30	42	0	.417	34	38	0	.472
Total		460	389	15	.532	389	460	15	.450

PCT := $W/(W + L + D)$

Source: Nippon Professional Baseball Official Website [25]

In total, 864 games were played in the Central and Pacific leagues combined. The winning percentage for teams batting first was .450 and the winning percentage for teams batting last

was .532, approximately 8 % higher. There are various advantages to being able to play a game at home, such as support from the home crowd. However, is there an advantage caused strictly by baseball rules?

Turocy [19] argued for the advantage of batting last by calculating the value of the game for the hypothetical situation where the same team plays itself. When doing this, the strategies that can be chosen by the manager are base stealing, sacrifice bunting, and intentionally walking a batter. These strategies can be turned "ON" or "OFF," and the value of the game is compared over a total of 8 different situations. In our paper, we have the Fukuoka Softbank Hawks (shown as in Table 1) play against themselves. We also switched each manager plan ON and OFF, both for the team batting first and the team batting last, and evaluated the value of the game in a total of 64 situations. We show the results in Table 3. We implemented our DP

Table 3: Values of the games at the point of the game starting, and the effects of the strategies

Batting-last Batting-first	−, −, −	−, −, W	−, B, −	−, B, W	S, −, −	S, −, W	S, B, −	S, B, W
−, −, −	.4932 .4932	.4933 .4930	.5180 .4699	.5181 .4698	.5190 .4690	.5191 .4689	.5437 .4458	.5438 .4456
−, −, W	.4929 .4935	.4930 .4933	.5169 .4713	.5170 .4712	.5186 .4694	.5187 .4692	.5420 .4475	.5421 .4473
−, B, −	.4710 .5167	.4715 .5160	.4957 .4933	.4962 .4926	.4966 .4925	.4971 .4918	.5213 .4689	.5219 .4682
−, B, W	.4707 .5170	.4712 .5163	.4945 .4947	.4950 .4940	.4962 .4929	.4967 .4922	.5197 .4707	.5202 .4700
S, −, −	.4700 .5177	.4701 .5176	.4949 .4942	.4950 .4941	.4957 .4934	.4958 .4933	.5206 .4698	.5207 .4697
S, −, W	.4697 .5180	.4698 .5179	.4937 .4956	.4938 .4955	.4953 .4938	.4954 .4937	.5189 .4715	.5190 .4714
S, B, −	.4480 .5409	.4485 .5402	.4727 .5174	.4732 .5167	.4734 .5167	.4739 .5160	.4983 .4929	.4988 .4922
S, B, W	.4477 .5412	.4482 .5405	.4715 .5188	.4719 .5181	.4731 .5171	.4735 .5164	.4966 .4947	.4970 .4940

- The upper row (right) displays the value of the game for the last-batting team, the lower row (left) for the first-batting team
- S = Base stealing is "ON", B = Sacrifice bunting is "ON", W = Intentional Walk is "ON"

algorithm using C++ Language and executed it on a desktop PC with Intel® Core™ i7-3770K processor and 16GB memory installed. Computational time is longest when all strategies for both teams are ON. In this case, the calculation for pure-strategy MPE and value of the game was completed in 1.61 second per game. Figure 2 illustrates the computational result of $V_i(s)$ for $s = (\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) = (7, 0, 1, 0, (0, 0, 1), (2, 1), 1)$.

For the condition of only intentional walks being ON for both teams, the team batting first had a higher winning percentage. On the other hand, for the condition of only base stealing being ON for both teams and for the condition of only sacrifices being ON for both teams, the winning percentage of the team batting last was higher. Thus, intentional walks are most advantageous to the team batting first, while stolen bases and sacrifice bunting are most advantageous to the team batting last. Therefore, when the strategies of both teams are all switched ON, whether the winning percentage of the team batting first or of the team batting last is higher depends on the transition probabilities of chance moves. However, as Table 3 shows, the influence of walks is less than that of base stealing and sacrifice bunting. Thus, it seems safe to say that, normally, the winning percentage of the team batting last would be higher. These results correspond to the facts outlined by Turocy [19]. However, Turocy does state in his paper that "the disparity is slight, and does not make a big difference." Although this is true, when considering the fact that a .007 increase in winning percentage would amount to 1 additional win in a 144 game

Figure 2: Computational result: $V_i(s)$ for $s = (\iota, \tau, \omega, \lambda, \boldsymbol{r}, \boldsymbol{b}, m) = (7, 0, 1, 0, (0, 0, 1), (2, 1), 1)$

season, this difference should not be ignored.

We note that optimal decision also depends on the current run difference. Take a state with runner on first base with no outs in the bottom half of the final inning, for instance. Sacrifice bunt will probably maximize team 1's win percentage when the score is tied. However, in the case of 3 runs behind, sacrifice bunt should not be executed. Hence, the run difference between the opposing teams must be considered to win a game. In other words, the number of runs scored by the opposing team is an important piece of information. When the first-batting team makes decisions on batting for a hit, stealing a base, or sacrifice bunting (i.e., when in the top half of some inning), the manager can not know the the runs scored in the bottom half of the inning by the last-batting team. However, when the last-batting team makes such decisions (i.e., when in the bottom half of some inning), the manager can know the runs scored in the top half of the inning by the first-batting team. Therefore, the last-batting team has an advantage of a half inning more observation. On the other hand, when making decisions on intentional walks, the stands of the first-batting team and the last-batting team is reversed. We believe this asymmetry due to the rules of baseball is the reason for the slight difference in winning percentage between the team batting first and that batting last.

6   Optimal lineup

Since the paper by Bukiet et al. [4] was published, the hottest topic within research on Markov chain approaches with matrix analysis has been the calculation of optimal batting order. This topic has been addressed in many previous studies. In a DP approach, considerable computa-

15

tional time is required for the optimization of strategy itself; thus, the computational cost of completing an exhaustive search of batting lineup to find the optimal one is very high. However, in actuality, it is enough to search 8! permutations. A memoized value of a subgame can be reused for another, so the time taken to evaluate a single lineup $\sigma = (1, 2, 3, 4, 5, 6, 7, 8, 9)$ is nearly the same as the time taken to evaluate all 9 lineups that can be obtained by rotations (e.g. $\sigma' = (2, 3, 4, 5, 6, 7, 8, 9, 1)$, $\sigma'' = (3, 4, 5, 6, 7, 8, 9, 1, 2)$).

Tables 4 and 5 show our results for computing the optimal lineup for the Fukuoka Softbank Hawks. In this case, we also created a hypothetical game where the first-batting team and the last-batting team are the exact same team. The batting lineup for the first team was fixed as the default batting order, shown in Table 1. An exhaustive search was then conducted for the batting order of the last-batting team, and we found the optimal lineup and the worst lineup which maximizes and minimize the winning percentage of the last-batting team, respectively. Moreover, we changed the run difference established in a called game from 30 runs to 20 runs, because it was sufficient to be able to precisely calculate the values of the games at the start of the game. Since we conducted a simple exhaustive search on a single thread, the computation took approximately half a day.

Table 4: Optimal lineup

Default Lineup	Worst Lineup	Optimal Lineup
1 Y. Honda	1 S. Tsuruoka	1 A. Nakamura
2 A. Nakamura	2 Y. Hasegawa	2 Y. Yanagita
3 Y. Yanagita	3 K. Imamiya	3 S. Uchikawa
4 S. Uchikawa	4 Y. Honda	4 Lee Dae-Ho
5 Lee Dae-Ho	5 N. Matsuda	5 Y. Hasegawa
6 Y. Hasegawa	6 S. Uchikawa	6 N. Matsuda
7 N. Matsuda	7 Lee Dae-Ho	7 Y. Honda
8 S. Tsuruoka	8 A. Nakamura	8 S. Tsuruoka
9 K. Imamiya	9 Y. Yanagita	9 K. Imamiya

Table 5: Probability of the last-batting team winning

Batting-first \ Batting-last	Default Lineup	Worst Lineup	Optimal Lineup
Default Lineup	.4970	.4768	.5002
	.4940	.5140	.4909

- The upper row (right) displays the value of the game for the last-batting team, the lower row (left) for the first-batting team

With the assumption that both teams make the best choices in terms of their game decision making on batting for a hit, stealing a base, or sacrifice bunting, the winning percentage difference between the optimal and worst batting order was only 2.34 %. However, in the context of a 144 game regular season, this would amount to a difference of 3.374 wins. Considering that more wins equate to more losses for other teams, the game difference with the other teams would be even greater.

7 Summary and Future Challenges

In this paper, baseball has been formulated as a finite Markov game with approximately 6.45 million states. We demonstrated that the value functions of the games and MPEs, where both teams' managers maximize the probabilities of their respective team winning, can be computed in 2 second per game. Based on this computational improvement, we have successfully computed

16

the optimal batting order, in consideration of strategy optimization such as a sacrifice bunt or a stolen base.

The authors have been also interested in applying the model description capability of the Markov game and the calculation power of DP to decision making problems in social fields such as marketing. More specifically, problems such as production planning for a producer, price and advertising strategies for a retailer, and purchasing strategies of consumers are to be formulated as large-scale, rich Markov games. We would like to solve them and use results to sicial system design as challenges for the future.

References

[1] R. Bellman, Dynamic Programming and Markovian Decision Processes, with Application to Baseball in *Optimal Strategies in Sports*, S.P. Ladany and R.E. Macol(eds.), Elsevier-North Holland, New York, (1977), 77–85.

[2] K. Boda and J.A. Filar, Time consistent dynamic risk measures, *Mathematical Methods of Operations Research*, **63**, (2006), 169-186.

[3] M. Bouakiz and Y. Kebir, Target-level criterion in Markov decision processes, *Journal of Optimization Theory and Applications*, **86**, (1995), 1-15.

[4] B. Bukiet, E.R. Harold, and J.L. Palacios, A Markov Chain Approach to Baseball, *Operations Research*, **45**(1), (1997), 14–23.

[5] T.M. Cover and C.W. Keilers, An Offensive Earned-Run Average for Baseball, , *Operations Research*, **25**(5), (1977), 729–740.

[6] D.A. D'Esopo and B. Lefkowitz, The Distribution of Runs in the Game of Baseball in *Optimal Strategies in Sports*, S.P. Ladany and R.E. Macol(eds.) , Elsevier North-Holland, (1977), 55–62.

[7] D. Fudenberg and J. Tirole, *Game Theory*, MIT Press, Cambridge MA, 1991.

[8] R.A. Howard, *Dynamic Programming and Markov Processes* , M.I.T. Technology Press and Wiley, Cambridge, Mass, (1960).

[9] H. Kawasaki, A. Kira, and S. Kira, An application of a discrete fixed point theorem to a game in expansive form, *Asia-Pacific Journal of Operational Research*, vol. **30**, No. 3, (2013).

[10] A. Kira, and K. Inakawa, On Markov perfect equilibria in baseball, *TMARG Discussion Papers*, no. **115**, Graduate School of Economics and Management, Tohoku University.

[11] A. Kira, and K. Inakawa, On Markov perfect equilibria in baseball, *Bulletin of Informatics and Cybernetics*, to appear.

[12] 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*, vol. **386**, (2012), 461–472.

[13] H. Kuhn, Extensive games and the problem of information, *Annals of mathematics studies, no.* **28**, Princeton University Press. Princeton, 1953.

[14] G.R. Lindsey, Statistical Data Useful for the Operation of a Baseball Team, *Operations Reserach*, **7**(2), (1959), 197–207.

[15] M. Sniedovich, *Dynamic Programming: Foundations and Principles*, 2nd edn. CRC Press, (2010).

[16] Y. Ohtsubo and K. Toyonaga, Optimal policy for minimizing risk models in Markov decision processes. *Journal of Mathematical Analysis and Applications*, **271**(2002), 66-81.

[17] L.S. Shapley, Stochastic games, *Proceedings of the National Academy of Sciences of the United States of America*, **39**, (1953), 1095-1100.

[18] M.J. Sobel, The variance of discounted Markov decision processes, *Journal of Applied Probability*, **19**, (1982), 794-802.

[19] T.L. Turocy, In Search of the "Last-Ups" Advantage in Baseball: A Game-Theoretic Approach, *Journal of Quantitative Analysis in Sports*, **4**(2) , (2008), Article 5.

[20] D.J. White, Mean, variance, and probabilistic criterion in finite Markov decision processes: A review, **56**, (1988), 1-29.

[21] D.J. White, Minimizing a threshold probability in discounted Markov decision processes, *Journal of Mathematical Analysis and Applications*, **173**, (1993), 634-646.

[22] C. Wu and Y. Lin, Minimizing risk models in Markov decision processed with policies depending on target values, *Journal of Mathematical Analysis and Applications*, **231**, (1999), 47-67.

[23] H. Xu, and S. Mannor, Probabilistic goal Markov decision processes, In: T. Walsh (ed.) *Proceedings of the 22nd International Joint Conference on Artificial Intelligence* (IJCAI-11), 2011, 2046-2052.

[24] L.E. Zachrisson, Markov games. *Annals Math. Studies*, **52**. *Advances in Game Theory*, M. Drescher, L.S. Shapley and A.W. Tucker(eds.), Princeton University Press, Princeton, 1964.

[25] Nippon Professional Baseball Official Website, ⟨http://www.npb.or.jp/eng/⟩, Accessed 2015 Nov 9.

[26] Let's enjoy baseball data! (in Japanese), ⟨http://baseballdata.jp/⟩, Accessed 2015 Nov 9.

List of MI Preprint Series, Kyushu University

The Global COE Program
Math-for-Industry Education & Research Hub

MI

MI2008-1   Takahiro ITO, Shuichi INOKUCHI & Yoshihiro MIZOGUCHI
Abstract collision systems simulated by cellular automata

MI2008-2   Eiji ONODERA
The intial value problem for a third-order dispersive flow into compact almost Hermitian manifolds

MI2008-3   Hiroaki KIDO
On isosceles sets in the 4-dimensional Euclidean space

MI2008-4   Hirofumi NOTSU
Numerical computations of cavity flow problems by a pressure stabilized characteristic-curve finite element scheme

MI2008-5   Yoshiyasu OZEKI
Torsion points of abelian varieties with values in nfinite extensions over a p-adic field

MI2008-6   Yoshiyuki TOMIYAMA
Lifting Galois representations over arbitrary number fields

MI2008-7   Takehiro HIROTSU & Setsuo TANIGUCHI
The random walk model revisited

MI2008-8   Silvia GANDY, Masaaki KANNO, Hirokazu ANAI & Kazuhiro YOKOYAMA
Optimizing a particular real root of a polynomial by a special cylindrical algebraic decomposition

MI2008-9   Kazufumi KIMOTO, Sho MATSUMOTO & Masato WAKAYAMA
Alpha-determinant cyclic modules and Jacobi polynomials

MI2008-10   Sangyeol LEE & Hiroki MASUDA
Jarque-Bera Normality Test for the Driving Lévy Process of a Discretely Observed Univariate SDE

MI2008-11   Hiroyuki CHIHARA & Eiji ONODERA
A third order dispersive flow for closed curves into almost Hermitian manifolds

MI2008-12   Takehiko KINOSHITA, Kouji HASHIMOTO and Mitsuhiro T. NAKAO
On the L^2 a priori error estimates to the finite element solution of elliptic problems with singular adjoint operator

MI2008-13   Jacques FARAUT and Masato WAKAYAMA
Hermitian symmetric spaces of tube type and multivariate Meixner-Pollaczek polynomials

MI2008-14 Takashi NAKAMURA
Riemann zeta-values, Euler polynomials and the best constant of Sobolev inequality

MI2008-15 Takashi NAKAMURA
Some topics related to Hurwitz-Lerch zeta functions

MI2009-1 Yasuhide FUKUMOTO
Global time evolution of viscous vortex rings

MI2009-2 Hidetoshi MATSUI & Sadanori KONISHI
Regularized functional regression modeling for functional response and predictors

MI2009-3 Hidetoshi MATSUI & Sadanori KONISHI
Variable selection for functional regression model via the L_1 regularization

MI2009-4 Shuichi KAWANO & Sadanori KONISHI
Nonlinear logistic discrimination via regularized Gaussian basis expansions

MI2009-5 Toshiro HIRANOUCHI & Yuichiro TAGUCHII
Flat modules and Groebner bases over truncated discrete valuation rings

MI2009-6 Kenji KAJIWARA & Yasuhiro OHTA
Bilinearization and Casorati determinant solutions to non-autonomous 1+1 dimensional discrete soliton equations

MI2009-7 Yoshiyuki KAGEI
Asymptotic behavior of solutions of the compressible Navier-Stokes equation around the plane Couette flow

MI2009-8 Shohei TATEISHI, Hidetoshi MATSUI & Sadanori KONISHI
Nonlinear regression modeling via the lasso-type regularization

MI2009-9 Takeshi TAKAISHI & Masato KIMURA
Phase field model for mode III crack growth in two dimensional elasticity

MI2009-10 Shingo SAITO
Generalisation of Mack's formula for claims reserving with arbitrary exponents for the variance assumption

MI2009-11 Kenji KAJIWARA, Masanobu KANEKO, Atsushi NOBE & Teruhisa TSUDA
Ultradiscretization of a solvable two-dimensional chaotic map associated with the Hesse cubic curve

MI2009-12 Tetsu MASUDA
Hypergeometric τ-functions of the q-Painlevé system of type $E_8^{(1)}$

MI2009-13 Hidenao IWANE, Hitoshi YANAMI, Hirokazu ANAI & Kazuhiro YOKOYAMA
A Practical Implementation of a Symbolic-Numeric Cylindrical Algebraic Decomposition for Quantifier Elimination

MI2009-14 Yasunori MAEKAWA
On Gaussian decay estimates of solutions to some linear elliptic equations and its applications

MI2009-15 Yuya ISHIHARA & Yoshiyuki KAGEI
Large time behavior of the semigroup on L^p spaces associated with the linearized compressible Navier-Stokes equation in a cylindrical domain

MI2009-16 Chikashi ARITA, Atsuo KUNIBA, Kazumitsu SAKAI & Tsuyoshi SAWABE
Spectrum in multi-species asymmetric simple exclusion process on a ring

MI2009-17 Masato WAKAYAMA & Keitaro YAMAMOTO
Non-linear algebraic differential equations satisfied by certain family of elliptic functions

MI2009-18 Me Me NAING & Yasuhide FUKUMOTO
Local Instability of an Elliptical Flow Subjected to a Coriolis Force

MI2009-19 Mitsunori KAYANO & Sadanori KONISHI
Sparse functional principal component analysis via regularized basis expansions and its application

MI2009-20 Shuichi KAWANO & Sadanori KONISHI
Semi-supervised logistic discrimination via regularized Gaussian basis expansions

MI2009-21 Hiroshi YOSHIDA, Yoshihiro MIWA & Masanobu KANEKO
Elliptic curves and Fibonacci numbers arising from Lindenmayer system with symbolic computations

MI2009-22 Eiji ONODERA
A remark on the global existence of a third order dispersive flow into locally Hermitian symmetric spaces

MI2009-23 Stjepan LUGOMER & Yasuhide FUKUMOTO
Generation of ribbons, helicoids and complex scherk surface in laser-matter Interactions

MI2009-24 Yu KAWAKAMI
Recent progress in value distribution of the hyperbolic Gauss map

MI2009-25 Takehiko KINOSHITA & Mitsuhiro T. NAKAO
On very accurate enclosure of the optimal constant in the a priori error estimates for H_0^2-projection

MI2009-26 Manabu YOSHIDA
Ramification of local fields and Fontaine's property (Pm)

MI2009-27 Yu KAWAKAMI
Value distribution of the hyperbolic Gauss maps for flat fronts in hyperbolic three-space

MI2009-28 Masahisa TABATA
Numerical simulation of fluid movement in an hourglass by an energy-stable finite element scheme

MI2009-29 Yoshiyuki KAGEI & Yasunori MAEKAWA
Asymptotic behaviors of solutions to evolution equations in the presence of translation and scaling invariance

MI2012-3 Hiroshi INOUE
A Weak RIP of theory of compressed sensing and LASSO

MI2012-4 Yasuhide FUKUMOTO & Youich MIE
Hamiltonian bifurcation theory for a rotating flow subject to elliptic straining field

MI2012-5 Yu KAWAKAMI
On the maximal number of exceptional values of Gauss maps for various classes of surfaces

MI2012-6 Marcio GAMEIRO, Yasuaki HIRAOKA, Shunsuke IZUMI, Miroslav KRAMAR, Konstantin MISCHAIKOW & Vidit NANDA
Topological Measurement of Protein Compressibility via Persistence Diagrams

MI2012-7 Nobutaka NAKAZONO & Seiji NISHIOKA
Solutions to a q-analog of Painlevé III equation of type $D_7^{(1)}$

MI2012-8 Naoyuki KAMIYAMA
A new approach to the Pareto stable matching problem

MI2012-9 Jan BREZINA & Yoshiyuki KAGEI
Spectral properties of the linearized compressible Navier-Stokes equation around time-periodic parallel flow

MI2012-10 Jan BREZINA
Asymptotic behavior of solutions to the compressible Navier-Stokes equation around a time-periodic parallel flow

MI2012-11 Daeju KIM, Shuichi KAWANO & Yoshiyuki NINOMIYA
Adaptive basis expansion via the extended fused lasso

MI2012-12 Masato WAKAYAMA
On simplicity of the lowest eigenvalue of non-commutative harmonic oscillators

MI2012-13 Masatoshi OKITA
On the convergence rates for the compressible
Navier- Stokes equations with potential force

MI2013-1 Abuduwaili PAERHATI & Yasuhide FUKUMOTO
A Counter-example to Thomson-Tait-Chetayev's Theorem

MI2013-2 Yasuhide FUKUMOTO & Hirofumi SAKUMA
A unified view of topological invariants of barotropic and baroclinic fluids and their application to formal stability analysis of three-dimensional ideal gas flows

MI2013-3 Hiroki MASUDA
Asymptotics for functionals of self-normalized residuals of discretely observed stochastic processes

MI2013-4 Naoyuki KAMIYAMA
On Counting Output Patterns of Logic Circuits

MI2013-5 Hiroshi INOUE
RIPless Theory for Compressed Sensing