Available online at www.sciencedirect.com

SCIENCE DIRECT®

BioSystems 80 (2005) 219–231

BioSystems

www.elsevier.com/locate/biosystems

A new formalization of a meta-game using the lambda calculus

Gen Masumoto[a,*], Takashi Ikegami[b]

[a] The Faculty of Economics, Kyoto Sangyo University, Kita-ku, Kyoto 603-8555, Japan

[b] The Graduate School of Arts and Sciences, The University of Tokyo, 3-8-1 Komaba Meguro-ku, Tokyo 153-8902, Japan

Received 5 July 2004; received in revised form 19 November 2004; accepted 20 November 2004

Abstract

This paper presents a new game system formalism. The system describes both strategies and a game master (who computes scores in a given game system) in terms of λ-calculus. This formalism revisits the prisoner's dilemma game, to discuss how meta-strategies emerge in this classical game, even without repetition. We have also examined the evolution of meta-strategies in λ formalism.

© 2004 Elsevier Ireland Ltd. All rights reserved.

Keywords: Meta-strategy; λ-Calculus; Prisoner's dilemma; Open-ended evolution

1. Introduction

The relationship between a game and its meta-game is our main concern. While Nash equilibrium with rational players is considered in game theory, we often observe irrational behavior and sometimes unexpected behavior in practical situations (e.g., changing the rules of a game during play). We consider this discrepancy as a difference between a game and "play". Infants play, but seldom show game behaviors; a rule of a game may be violated, so that a game theoretical solution such as Nash equilibrium is only an artifact in play situations.

Bacharach (1997) pondered what kinds of epistemic structure are required in game players to access the Nash equilibrium state. It was shown by Anderlini (1990) that players simulated by Turing machines are insufficient to achieve a Nash equilibrium state. The main drawback is that it is impossible for players to decide whether other players are rational or not. Not even logical omniscience is sufficient. In addition to the usual logical operators, players must possess hyperoperations such as common knowledge operations (Kaneko and Nagashima, 1996).

In artificial life studies (e.g., Artificial Life II meeting, Langton, 1992), an open-ended evolution concept has been proposed and has been taken as a key concept in understanding evolutionary dynamics. Rather than studying a closed static system, open-ended evolution treats an open system. A good example can be found in Lindgren's simulation studies (Lindgren, 1992). He

* Corresponding author. Tel.: +81 757051452; fax: +81 757051949.
E-mail addresses: gen@cc.kyoto-su.ac.jp (G. Masumoto), ikeg@sacral.c.u-tokyo.ac.jp (T. Ikegami).

0303-2647/$ – see front matter © 2004 Elsevier Ireland Ltd. All rights reserved.
doi:10.1016/j.biosystems.2004.11.010

was the first to study the (noisy) iterated prisoner's dilemma game by introducing the open strategy space. Later, Ikegami (1994) showed that complex and longer-memory strategies successively evolve without stacking on some small memory strategies. They studied the iterated prisoner's dilemma game without fixing the number of strategies in advance.

Prediction capability is another way to improve players' abilities. Predictions of the other players' future actions and associated outcomes can embody a player's possible rationalities. Since Binmore (1987), the advantages of such prediction-capable players and their potential complexity and paradox have been identified. Recent simulation studies with some recurrent neural systems have revealed such complexity (Ikegami and Taiji, 1998, 1999; Taiji and Ikegami, 1999).

On the other hand, Howard (1966a,b, 1971) pondered the paradox of rationality and developed a meta-game theory. Hofstadter (1985) introduced "Nomic" game (which was originally invented by Suber, 1990), where players can update the rules of the game that provide how to change themselves. This is motivated by the paradox of self-amendment, that a legal rule for change may apply to itself and authorize its own amendment. Vreeswijk (1995a,b) performed some experiments using self-modifying protocol games to model such a situation. These studies gave us new perspectives of open-endedness in game situations.

The significant feature of these studies is the "openness" of the rules. In these models, the players can interpret or translate the rules. In Nomic, the players can change the rules that regulate their own behaviors. In Howard's meta-game theory, the meta-level players can expand the payoff matrix at the stage of their inference processes. This can yield meta-games with payoff matrices that are different from the original matrix. The meta-level players and the prediction-capable players are similar in the sense that both can infer an opponent's moves.

We propose here a new approach based on the λ-calculus (Masumoto and Ikegami, 2001). An advantage of λ formalism is that each λ-term can be either a function or a variable. We use the formalism here in terms of game strategies. Instead of maintaining the game theoretical concepts (i.e., rational players and Nash equilibrium), we hope to provide a model that bridges game and "play" behaviors. Key concepts are meta-strategies and interventions in game rules.

2. Howard's meta-game theory and the prisoner's dilemma game

Howard (1966a,b, 1971) pondered the paradox of rationality and developed a meta-game theory. We here consider meta-game treatments of the prisoner's dilemma (PD) game. The PD game has only two actions (cooperation (C) and defection (D)) and the payoff matrix has a unique Nash equilibrium, which is mutual defection (Table 1). Because the PD game illustrates the minimal negotiation situation, ways of achieving a good solution (i.e., mutual cooperation) have received much attention. One remedy is to repeat the game infinitely many times. Axelrod (1984) investigated the repeated PD game by conducting the computer program tournaments twice. A famous entry was Tit for Tat, which won the tournaments twice simply by mimicking the other player's previous action.

Howard investigated the game in a radically different fashion. Even without repeating the game, it was shown that mutual cooperation can be expected. A clue is to assume meta-level players.

A meta-level player is defined as a player who infers or predicts other players' behaviors, and can use the inferences to choose moves. Consider a PD game in which player 2's future action is represented by a function of player 1's current action. Because each player has two moves {C, D}, there are four possible functions, f_1, f_2, f_3, and f_4, mapping from {C, D} to {C, D}.

If a meta-level player is introduced, one can define a meta-game, in which player 1 can choose f_i $(1 \leq i \leq 4)$ as the next move. An outcome of the match in this meta-game is computed by the outcome of $(S, f_i(S))$, where S is taken as C or D. Table 2 shows the payoff matrix of the meta-game derived from PD, and the

Table 1
The payoff matrix of PD

Player 1	Player 2	
	C	D
C	3, 3	0, 5
D	5, 0	1, 1

Table 2
The payoff matrix of PD extended with meta-strategies

Player 1	Player 2			
	f_1	f_2	f_3	f_4
C	3, 3	0, 5	3, 3	0, 5
D	5, 0	1, 1	1, 1	5, 0

meta-strategies of player 2 are labeled f_1, f_2, f_3, and f_4, where f_1 stands for the reaction "always choose C", f_2 for the reaction "always choose D", f_3 for "choose C if the opponent chooses C, D if the opponent chooses D" that is, "choose the same as player 1", and f_4 stands for "choose D if the opponent chooses C, C if the opponent chooses D" that is, "choose the opposite to player 1". These meta-strategies correspond to the well-known AllC, AllD, TfT, and a-TfT strategies in the repeated PD game, respectively.

Furthermore, a higher meta-level player can be defined in the same manner. Player 1's meta-strategy against player 2 is all functions from f_1, f_2, f_3, f_4 to $\{C, D\}$. For example, if player 2 plays f_1 then player 1 chooses C, and if player 2 plays f_2 then player 1 chooses D, and so on. There are 16 of these functions, say g_1, ..., g_{16}. This yields the second meta-game. When f and g are chosen by the players, then $(g(f), f(g(f)))$ will be the resulting outcome of the original PD game. Howard found that one of Nash equilibria in the second meta-game yields the outcomes of (C, C), so that cooperation is accomplished.

The remarkable point of the meta-game theory is the concept that players can have a function as their strategy even in a *one-shot* game. This concept is quite common in repeated game theory. In repeated games, the strategy of players is often represented by a function relating the history of the players' previous moves to their current move. In contrast, Howard's approach applies the concept to a one-shot game.

However, when players at the same meta-level come to play each other, they cannot stop guessing what other players will do, and an infinite regression of guessing emerges. With a prediction-capable players model, Taiji and Ikegami (1999) have shown that mutual defection is an almost unique attractor when this type of infinite regression occurs. Howard avoids this situation and discusses only a game between players at different meta-levels. Players on the same meta-level are not allowed to play against each other.

The following presents one possible model system that realizes Howard's meta-game situations. A benefit of our model is that we allow games between players at the same meta-level. Our new model is fully described by the λ-calculus, and we thus introduce the λ-game and present the simulation results.

3. λ-Game system

3.1. λ-Calculus representation

Fontana and Buss (1994a,b) introduced a λ-calculus formalism into chemical reaction and evolution. Any λ-term is a combination of certain atom sentences, which potentially can mimic any computable function. An advantage of λ formalism is that each λ-term can be either a function or a variable. Taking each λ-term as a chemical substance and the concatenation of two λ-terms as a chemical reaction, Fontana simulated λ chemical reactions to see what kind of functions and algebraic structures would appear. The artificial chemistry approach is remarkable as it provides a first step towards understanding evolution as an algebraic object.

The λ-calculus formalism has also been applied to game theory. Because any λ function can take any λ-term as its variable, we can simulate a match between any meta-strategy and action for a λ-formalized game system. An essential part is the design of elementary game actions and a game master in terms of λ-terms. As the first example, we translate the prisoner's dilemma game into the λ formalism in the next section.

In the λ-calculus, one can define natural numbers, Boolean truth values, and a conditional, which are required to express the payoff matrix of the game. Recall that there is no syntactical distinction between operator and operand (functions and arguments) in the λ-calculus; both can be expressed as arbitrary λ-terms.

3.2. Boolean truth values and numerals in the λ-calculus

In the λ-calculus, Boolean truth values and a conditional can be represented as follows:

$$\mathbf{T} \equiv \lambda xy.x, \tag{1}$$

$$\mathbf{F} \equiv \lambda xy.y. \tag{2}$$

If B only takes Boolean terms (i.e., either $\mathbf{T}$ or $\mathbf{F}$) and P and Q take arbitrary λ-terms, then the sentence "if B then P else Q" is represented by the triplet "$B\ P\ Q$". Therefore, a conditional

"if x then P else Q"

is represented by a specific λ-term

$$\lambda x.xPQ. \tag{3}$$

These terms actually function as follows:

$$(\lambda x.xPQ)\mathbf{T} = \mathbf{T}PQ = (\lambda xy.x)PQ = (\lambda y.P)Q = P, \tag{4}$$

$$(\lambda x.xPQ)\mathbf{F} = \mathbf{F}PQ = (\lambda xy.y)PQ = (\lambda y.y)Q = Q. \tag{5}$$

One can define natural numbers in a variety of ways in the λ-calculus. Here, we will use Barendregt numerals (Barendregt, 1981) and henceforth refer to them as numerals $\lceil n \rceil$. Barendregt numerals are defined inductively in the following way:

$$\lceil 0 \rceil = \lambda x.x, \tag{6}$$

$$\lceil n+1 \rceil = \lambda x.x\mathbf{F}\lceil n \rceil. \tag{7}$$

3.3. λ-Game system

The prisoner's dilemma game consists of two elementary actions, cooperation (C) and defection (D). We express C by $\mathbf{T}(=\lambda x.x)$ and D by $\mathbf{F}(=\lambda xy.y)$. The correspondence has no rigid ground, but taking these formulas as the elementary strategies, we construct a hierarchy on top of them. In Howard's meta-game hierarchy, elementary strategies are defined as non-reacting or non-inferring strategies. In the λ-game system, strategies become reactive to the opponent's strategy via a reduction process with the game master.

Simple truth values, such as $\mathbf{T}$ and $\mathbf{F}$, are elementary strategies, and always return the former part or the latter part, respectively, of a given conditional. We call them elementary because they cannot react to the opponent's strategy. In contrast, some λ-terms infer the opponent's behavior in the reduction process to the normal form. They can be reactive if they utilize recursive conditionals provided by a game master. We therefore call these strategies meta-strategies. Thus, the hierarchical order of the meta-game is controlled by synthesized recursive conditionals on an opponent's strategy with a game master. A simple but clear evolution of meta-strategies will be shown in Section 6. In any sense, the kind of meta-strategies that will appear is strongly dependent on how we construct the game master. Below, we will show an example of a game master consisting of conditionals.

We will construct a game master to enable us to interpret the values true and false as C and D actions. There are several realizations of a game master in terms of λ-expressions. A requirement for the game master is to simulate the prisoner's dilemma game with the payoff matrix given in Table 1. Given two strategies in λ-terms, a game master computes a game match between the two strategies and returns the outcomes corresponding to each strategy.

Within this framework, we construct the "game master" $\mathbf{G}$ as a function that treats the game score as Barendregt numerals. In practice, it can take the following form:

$$\begin{aligned}\mathbf{G} &\equiv \lambda x.x(\lambda y.y\lceil 3\rceil\lceil 5\rceil)(\lambda y.y\lceil 0\rceil\lceil 1\rceil)\\ &= \lambda x.x(\lambda y.y(\lambda z.z\mathbf{F}(\lambda z.z\mathbf{F}(\lambda z.z\mathbf{FI})))\\ &\quad(\lambda z.z\mathbf{F}(\lambda z.z\mathbf{F}(\lambda z.z\mathbf{F}(\lambda z.z\mathbf{F}(\lambda z.z\mathbf{FI})))))\\ &\quad(\lambda y.y\mathbf{I}(\lambda z.z\mathbf{FI})).\end{aligned} \tag{8}$$

Given two strategies S_{self} and S_{opp} in λ-terms, the reward of each strategy is calculated by entering the two strategies successively into the game master as follows:

$$\mathbf{G}S_{\text{opp}}S_{\text{self}} \rightarrow \text{Reward}_{\text{self}}, \tag{9}$$

$$\mathbf{G}S_{\text{self}}S_{\text{opp}} \rightarrow \text{Reward}_{\text{opp}}, \tag{10}$$

where $\text{Reward}_{\text{self}}$ and $\text{Reward}_{\text{opp}}$ denote the reward of each strategy, respectively.

We can define ordered pairing in the λ-calculus, and thus we can construct a game master that gives a pair of Barendregt numerals by using such a pairing. However, it should be noted that the original PD matrix is symmetrical. Therefore, we employ this form of the game master to retain the symmetry of the PD game.

A necessary condition on the game master is to return the correct PD scores for inputs C and D. However, this constraint is no longer sufficient for any set of strategies. Some pairs of strategies may return numerical values that are not expected in the PD game.

This rather strange behavior occurs when players seriously interfere with the game master program itself. We can always improve the game master to suppress such strange situations. However, this cycle never comes to an end. Players who deceive a game master to make better scores can never be eliminated completely. Instead of seeking for the best game master, we define the game master to analyze possible meta-strategies and to realize Howard's meta-strategy situation.

4. Numerical experiments

As noted above, there is no syntactical distinction between strategy and score in the λ-game system. Therefore, we can apply **G** to a λ-term that represents a function, giving a strategy instead of a mere Boolean value. Moreover, when we apply **G** to a λ-term that has a non-Boolean value, it sometimes yields a Barendregt numeral via an unexpected reduction process. Thus, we generate random λ-terms, and compute the round-robin tournament of the population of meta-strategies.

4.1. Model

We require that all λ-terms in the system be closed terms, i.e., λ-terms without free variables. Such λ-terms are also known as combinators. Because the application of the game master to non-closed λ-terms never yields Barendregt numerals, we here consider only the closed terms as the strategies.

Random λ-terms are produced by the recursive generation of random sub-terms. First, a λ-term L_0 is chosen randomly from the set M of atomic elements consisting of some combinators and a new variable, i.e., $M = \{\mathbf{T}, \mathbf{F}, \mathbf{I}, \mathbf{S}, \mathbf{V}, x_0, x_1, \ldots\}$, where $\mathbf{T} = \lambda xy.x$, $\mathbf{F} = \lambda xy.y$, $\mathbf{I} = \lambda x.x$, $\mathbf{S} = \lambda xyz.xz(yz)$ and $\mathbf{V} = \lambda xyz.zxy$. Then, a new λ-term L_1 is generated by an application or an abstraction, which both occur with probability 1/2. L_{i+1} is composed from L_i in the same way, and this process is repeated n times (in this experiment, we set $1 \leq n \leq 20$).

The procedure for application is as follows. Given L_i, a new atomic element is chosen randomly from M, and defined as an operator or an operand in the application to be created. Then, the new λ-term L_{i+1} is produced by the application of L_i and the new element in the given order. For abstraction, a new abstraction is inserted at the head of the λ-term L_i. If L_i has free variables, the abstraction of one of the variables should be inserted, otherwise the abstraction of a new variable is inserted.

After generating the random λ-term L_n by the above procedure, we transform L_n into a normal form L before we use it as a game strategy. Then, the rewards of the strategy against **T**, **F**, and itself are computed, i.e., $(\mathbf{G}L\mathbf{T})$, $(\mathbf{GT}L)$, $(\mathbf{G}L\mathbf{F})$, $(\mathbf{GF}L)$, and $(\mathbf{G}LL)$. Some matches between two strategies cannot return any numerical values. When that occurs, those strategies are eliminated as syntactically incorrect. In addition, λ-terms that are not converted into a normal form within a preset limit (here 1000) are removed from meta-game strategies. The present meta-game system, therefore, has no halting problem.

After the population of meta-strategies is produced and outcomes against **T**, **F**, and itself are examined, the round-robin tournament of meta-strategies is computed. In the tournament, every strategy plays against every other strategy. We also employ the game master **G** defined in Eq. (8) in the round-robin tournament, and the score of each strategy is computed by entering two strategies successively into the game master.

4.2. Simulation results

4.2.1. Randomly generated strategies

While examining randomly generated strategies, we noticed that many syntactically correct strategies exist. A certain set of strategies can give and take rewards that are not expected in the original payoff. We have generated randomly 10^7 λ-terms through the procedure described above, of which 10^6 λ-terms are allowed as strategies. These 10^6 λ-terms are reduced to 4407 distinct normal forms. Terms that are equivalent with respect to renaming of variables are identified. Such syntactically correct strategies are exemplified in Table 3.

Table 3 indicates that some strategies acquire unexpected values in the original PD game matrix. There are good reasons to call these strategies *meta-strategies*, as they do not merely follow the PD rule but *interfere* with the game master. In the usual game theory, players cannot change the game rules or the payoff matrix. Here rules, actions, and the payoff matrix are described in the same level, so that players can interfere with the game master. In other words, some players can deceive the

Table 3
Examples of λ-strategies

No.	Examples of λ-terms	Reward vs.			Average score
		T	**F**	Self	
	$\lambda xy.x(=\mathbf{T})$	3, 3	0, 5	3	0.70
	$\lambda xy.y(=\mathbf{F})$	5, 0	1, 1	1	1.77
1	$\lambda x_1.x_1(\lambda x_2.\mathbf{S})\mathbf{VFVF}$	6, 0	2, 1	2	2.57
2	$\lambda x_1.x_1(\lambda x_2x_3x_4x_5x_6.\mathbf{F}(x_4x_5)\mathbf{SSI}$	5, 0	2, 1	2	2.37
3	$\lambda x_1.x_1(\lambda x_2x_3.x_3\mathbf{F}x_2)\mathbf{F}(\lambda x_4.x_1)$	5, 3	0, 5	5	1.15
4	$\lambda x_1x_2.x_2(\lambda x_3x_4.x_3x_4(\lambda x_5x_6.x_6\mathbf{F}x_5))x_1$	4, 3	1, 5	4	1.54
5	$\lambda x_1x_2.x_2(\lambda x_3.x_3(\lambda x_4x_5x_6.x_6(\lambda x_7.\mathbf{V})x_5)\mathbf{I})x_1$	1, 5	0, 1	5	0.27
6	$\lambda x_1x_2.x_2\mathbf{TF}$	0, 0	0, 0	0	0.02
7	$\lambda x_1x_2x_3.x_1\mathbf{STTF}x_3(x_2x_3)$	5, 0	1, 0	0	1.76
8	$\lambda x_1x_2.x_2(\lambda x_3x_4.x_4\mathbf{FT}(\lambda x_9.x_4))x_1$	3, 1	0, 1	1	0.70
9	$\lambda x_1x_2x_3.x_1(x_2\mathbf{IT}x_3\mathbf{T})$	3, 3	0, 3	3	0.70
10	$\lambda x_1x_2.x_2(\lambda x_3.x_3(\lambda x_4x_5.x_1\mathbf{F}))$	4, 5	0, 5	5	0.91
11	$\lambda x_1x_2.x_1(x_2\mathbf{IF})\mathbf{S}x_2x_1$	3, 3	0, 0	3	0.70
12	$\lambda x_1x_2.x_2(\lambda x_3.x_3(\lambda x_4x_5x_6x_7x_8.x_8\mathbf{F}(x_6x_7))\mathbf{T})x_1$	4, 4	0, 6	5	0.90
13	$\lambda x_1x_2.x_1(\lambda x_3.\mathbf{F})\mathbf{VI}x_1\mathbf{FS}$	2, 0	0, 2	0	0.48
14	$\lambda x_1x_2x_3.x_2(x_2(\lambda x_4.x_2)x_3)$	5, 1	1, 1	1	1.75
15	$\lambda x_1.x_1\mathbf{ST}(\lambda x_7x_8.x_1x_8(x_7x_8))$	5, 0	1, 4	4	1.75

game master to obtain the impossible reward values. We will later classify those meta-strategies into three types and discuss their behavior in detail.

4.2.2. *Round-robin tournament*

We have considered outcomes only against **T**, **F**, and itself above. The round-robin tournament of meta-strategies is now examined. In the tournament, every strategy plays against every other strategy. A histogram of average scores is shown in Fig. 1(a). The horizontal axis corresponds to the average reward of each strategy in the round-robin tournament, while the vertical axis corresponds to the number of players having the same averaged score. Fig. 1(b) shows a histogram of a pair of actual rewards of all matches in the round-robin tournament. Each axis indicates a reward of a player, and the frequency of the occurrence of the reward pair is expressed with gray-scale. Darker represents more matches corresponding to each reward pair.

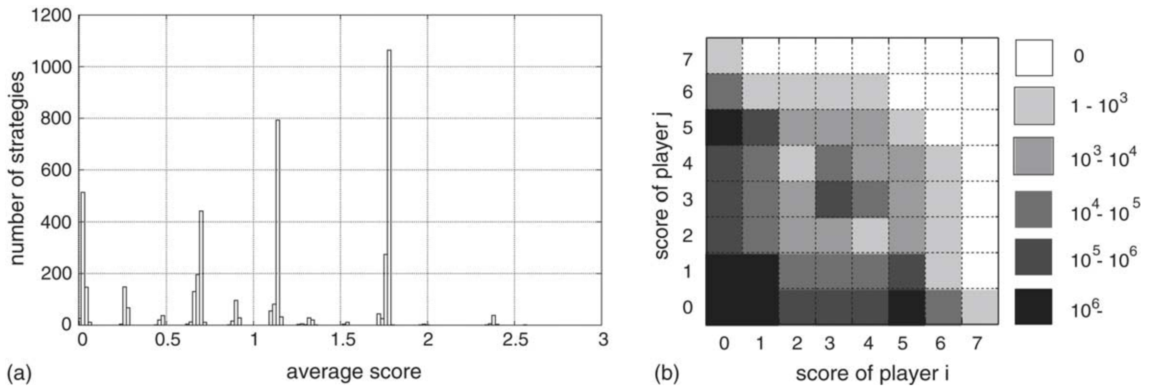

Fig. 1. (a) Histogram of the average score of each player. The horizontal axis corresponds to the average reward of each strategy, and the vertical axis corresponds to the number of players. (b) Histogram of pairs of rewards of all matches in the round-robin tournament. Each axis indicates a reward of a player.

We see a large variety of rewards that are not given in the original PD matrix from Fig. 1(b). As is expected, it shows more diversity of scores than Table 3. In addition, almost all pairs of rewards in the area from zero points to six points of each axis occur in the round-robin tournament. These values are obtained through the interference reduction processes with the game master. It should also be noted that several sharp peaks are shown in Fig. 1(a). They suggest that many matches give the same pair of rewards in the round-robin tournament, indicating that each player acquires the same reward repeatedly. Actually, some peaks are also seen in Fig. 1(b). We will discuss this issue in the following section.

5. Analysis of meta-strategies

To characterize the effects of interference with the game master, we must look into the actual reduction processes of the strategies when they are applied to the game master. First, we will classify the meta-strategies into three types. Second, we will investigate the actual process of computing the reward, and discuss how good strategies use the dynamics.

5.1. Intermediate form $\mathbf{G}'$

Rewards for a pair of strategies are computed by entering the two strategies successively into the game master:

$$\mathbf{G} S_{\text{opp}} S_{\text{self}} \to \mathbf{G}'_{\text{opp}} S_{\text{self}} \to \text{Reward}_{\text{self}} \tag{11}$$

$$\boldsymbol{G} S_{\text{self}} S_{\text{opp}} \to \boldsymbol{G}'_{\text{self}} S_{\text{opp}} \to \text{Reward}_{\text{opp}} \tag{12}$$

A basic requirement for the game master is to return the correct PD rewards with **T**(C) and **F**(D) (for example, $\mathbf{GTF} \to \mathbf{G}'\mathbf{F} \to \lceil 5 \rceil$). A simple construction of such an intermediate $\mathbf{G}'$ is to use a conditional. The basic strategy **T** constructs $\lambda x.x\lceil 3 \rceil\lceil 5 \rceil$ as its $\mathbf{G}'$, and **F** likewise constructs $\lambda x.x\lceil 0 \rceil\lceil 1 \rceil$. These conditionals mean "if x then $\lceil 3 \rceil$ else $\lceil 5 \rceil$" and "if x then $\lceil 0 \rceil$ else $\lceil 1 \rceil$", respectively. These λ-terms of conditional forms are the subterms in the game master program, and these satisfy the original PD payoff table. Many other **T**-like and **F**-like strategies are processed as well to give $\lambda x.x\lceil 3 \rceil\lceil 5 \rceil$ and $\lambda x.x\lceil 0 \rceil\lceil 1 \rceil$, respectively.

All the meta-strategies obtained in our simulation can be classified into three types. First, we roughly categorize $\mathbf{G}'$s with respect to whether they are reactive to the input strategy or not. Then, we categorize the reactive $\mathbf{G}'$s into conditionals and others according to their forms. Hence, there are three types with respect to the intermediate $\mathbf{G}'$ forms.

(1) Type I (conditional form)

$\mathbf{G}'$ is computed as

$$\mathbf{G}' = \lambda x.x\,\lceil m \rceil\,\lceil n \rceil\,, \tag{13}$$

where $\lceil m \rceil$ and $\lceil n \rceil$ are Barendregt numerals. This λ-term has the conditional structure shown in Eq. (3) and is reactive to the input strategy. In particular, when $\lceil m \rceil = \lceil 0 \rceil$ and $\lceil n \rceil = \lceil 1 \rceil$, or $\lceil m \rceil = \lceil 3 \rceil$ and $\lceil n \rceil = \lceil 5 \rceil$, it is exactly the same conditional as is found in the game master program (see Eq. (8)). Strategies of type I that construct $\lambda x.x\lceil 0 \rceil\lceil 1 \rceil$ behave as "**F** players" to other strategies; they intend to give only zero or one point to other players (for instance, (1) and (2) in Table 3). Strategies that construct $\lambda x.x\lceil 3 \rceil\lceil 5 \rceil$ behave as "**T** players" to other strategies in the similar sense (for instance, (3) and (4) in Table 3). Examples of $\mathbf{G}'$ in this type are $\lambda x.x\lceil 0 \rceil\lceil 1 \rceil$, $\lambda x.x\lceil 3 \rceil\lceil 5 \rceil$, $\lambda x.x\lceil 5 \rceil\lceil 1 \rceil$, $\lambda x.x\lceil 1 \rceil\lceil 1 \rceil$, $\lambda x.x\lceil 0 \rceil\lceil 0 \rceil$, $\lambda x.x\lceil 5 \rceil\lceil 0 \rceil$, and $\lambda x.x\lceil 1 \rceil\lceil 5 \rceil$. In Table 3, strategies (1)–(5) are of this type.

(2) Type II (constant function)

$\mathbf{G}'$ is computed as

$$\mathbf{G}' = \lambda x.\,\lceil n \rceil\,, \tag{14}$$

where $\lceil n \rceil$ is a Barendregt numeral. These $\mathbf{G}'$s are evidently not reactive to other strategies, and do not have a conditional form. Therefore, we label these $\mathbf{G}'$s as constant functions. Entries (6)–(10) in Table 3 are strategies of type II. A $\mathbf{G}'$ of type II always outputs $\lceil n \rceil$ for any other strategies. Players who construct a $\mathbf{G}'$ of this type give the opponent a constant $\lceil n \rceil$. Therefore, these strategies are analogous to AllD and AllC behavior. Practical examples of $\mathbf{G}'$ in type II are: $\lambda x.\lceil 0 \rceil$, $\lambda x.\lceil 1 \rceil$, $\lambda x.\lceil 5 \rceil$, $\lambda x.\lceil 3 \rceil$, $\lambda x.\lceil 4 \rceil$, $\lambda x.\lceil 2 \rceil$.

(3) Type III (others)

The rest of the population is labeled as type III. $\mathbf{G}'$s of this type, like type II strategies, do not possess a conditional form, but differ from a type II strategy, because a type III strategy is reactive to opponents. It returns numeral rewards of a Barendregt type against many strategies through interference reduction processes with the game master. However, only a few strategies can return numerals. Entries (11)–(15) in Table 3 are examples of type III strategies. Examples are $\lambda x.x\mathbf{F}\ulcorner 2\urcorner(x\mathbf{F}\ulcorner 4\urcorner)$, $\lambda x.x\mathbf{F}\ulcorner 0\urcorner\ulcorner 3\urcorner\ulcorner 5\urcorner$, $\lambda x.x\ulcorner 3\urcorner\ulcorner 5\urcorner(x\ulcorner 0\urcorner\ulcorner 1\urcorner)$, $\lambda x_1x_2.x_2\mathbf{F}(x_1\ulcorner 3\urcorner\ulcorner 5\urcorner)$, $\lambda x.x\mathbf{F}\ulcorner 0\urcorner\ulcorner 0\urcorner$, $\lambda x_1.x_1\ulcorner 3\urcorner\ulcorner 5\urcorner(\lambda x_2.x_2\ulcorner 3\urcorner\ulcorner 5\urcorner(x_1x_2))$.

The above 4407 distinct λ-terms can generate 57 $\mathbf{G}'$s. The major $\mathbf{G}'$ form is type I, where 62% of λ-terms have $\lambda x.x\ulcorner 0\urcorner\ulcorner 1\urcorner$ as their $\mathbf{G}'$, and 22% of λ-terms have $\lambda x.x\ulcorner 3\urcorner\ulcorner 5\urcorner$. Approximately 13% of λ-terms generate $\mathbf{G}'$s of type II. The constant function $\lambda x.\ulcorner 0\urcorner$ is the major practical form of type II. $\mathbf{G}'$s of type III comprise only 3% of the entire population.

Strategies with the same $\mathbf{G}'$ give the same effect, i.e., reward to the opponent, irrespective of the original strategy form. We have seen in Fig. 1 that each strategy takes the same reward repeatedly in the round-robin tournament. The reason for this is that the entire population only used a few forms of $\mathbf{G}'$.

5.2. Interference with a conditional

Now, we discuss behavior against $\mathbf{G}'$. The second step of Eq. (11) is as follows:

$$\mathbf{G}'_{\mathrm{opp}}S_{\mathrm{self}} \rightarrow \mathrm{Reward}_{\mathrm{self}}, \qquad (15)$$

where $\mathbf{G}'_{\mathrm{opp}}$, is the form of $\mathbf{G}'$ given by the opposing player. It is clear that the score of each player strongly depends on the $\mathbf{G}'$ constructed by the opponent. We will examine how players obtain their rewards from each type of $\mathbf{G}'$.

A $\mathbf{G}'$ of type II (constant function) is not reactive and actually functions as: $(\lambda x.\ulcorner n\urcorner)S_{\mathrm{self}} \rightarrow \ulcorner n\urcorner$, for any strategy S_{self}. Strategies playing against $\mathbf{G}'$ of type II, therefore, can merely acquire a constant reward, which is prepared in $\mathbf{G}'_{\mathrm{opp}}$. A strategy that uses a $\mathbf{G}'$ of this type can control the reward of the opponent completely. Those strategies with $\lambda x.\ulcorner 0\urcorner$ can avoid being exploited, but they never cooperate with themselves. Similarly, strategies with $\lambda x.\ulcorner 5\urcorner$ always cooperate but are never able to punish the opponent.

On the other hand, a $\mathbf{G}'$ of type I (conditional) is a reactive strategy and it functions against $\mathbf{T}$ and $\mathbf{F}$ as follows: $(\lambda x.x\ulcorner m\urcorner\ulcorner n\urcorner)\mathbf{T} \rightarrow \ulcorner m\urcorner$, $(\lambda x.x\ulcorner m\urcorner\ulcorner n\urcorner)\mathbf{F} \rightarrow \ulcorner n\urcorner$. Some good strategies (such as (1) and (2) in Table 3) take not zero or one point but two points from $\lambda x.x\ulcorner 0\urcorner\ulcorner 1\urcorner$. Moreover, the best-rewarded strategy ((1) in Table 3) takes six points from $\lambda x.x\ulcorner 3\urcorner\ulcorner 5\urcorner$. These numerals do not appear in the original $\mathbf{G}'$s of $\lambda x.x\ulcorner 0\urcorner\ulcorner 1\urcorner$ and $\lambda x.x\ulcorner 3\urcorner\ulcorner 5\urcorner$. Those scores are the outcome of the interference with the $\mathbf{G}'$ of the opponent.

Examining the actual reduction process shows that the interference comes from the numeric function *successor*. In the reduction process, the *successor* is applied to the numeral in $\mathbf{G}'$ to give other numerals. The *successor* for Barendregt numerals is defined as:

$$\mathbf{succ} \equiv \lambda x_1x_2.x_2(\lambda x_3x_4.x_4)x_1$$
$$(= \lambda x_1x_2.x_2\mathbf{F}x_1). \qquad (16)$$

This combinator actually functions as $\mathbf{succ}\ulcorner n\urcorner \rightarrow \ulcorner n+1\urcorner$ for the Barendregt numeral $\ulcorner n\urcorner$. The reduction process of the strategies that take two points against $\mathbf{F}$ is as follows: $(\lambda x.x\ulcorner 0\urcorner\ulcorner 1\urcorner)S_{\mathrm{self}} \rightarrow \mathbf{succ}\ulcorner 1\urcorner \rightarrow \ulcorner 2\urcorner$.

Moreover the best strategy (1) in Table 3 is able to construct **succ** when playing against the $\mathbf{G}'$ of $\lambda x.x\ulcorner 3\urcorner\ulcorner 5\urcorner$ as: $(\lambda x.x\ulcorner 3\urcorner\ulcorner 5\urcorner)S_{\mathrm{self}} \rightarrow \mathbf{succ}\ulcorner 5\urcorner \rightarrow \ulcorner 6\urcorner$. This strategy can acquire six points from $\mathbf{T}$ and those strategies that have $\lambda x.x\ulcorner 3\urcorner\ulcorner 5\urcorner$ as their $\mathbf{G}'$. The majority of $\mathbf{G}'$s in the population are $\lambda x.x\ulcorner 0\urcorner\ulcorner 1\urcorner\ulcorner 1\urcorner$ and $\lambda x.x\ulcorner 3\urcorner\ulcorner 5\urcorner$, as they receive the best reward in the round-robin tournament.

What must be emphasized is that they use the combinator $\ulcorner 0\urcorner$ in $\mathbf{G}'$ not as a numeral but as a function. This interference allows those strategies to compose the **succ**. Then they apply the **succ** to the latter numeral $\ulcorner 1\urcorner$ and obtain two points. We interpret this as deceiving the game master in the sense that the players use the original λ-term with a different meaning.

5.3. Discrimination ability

However, constructing **succ** against any conditional is not advantageous. Because the game master $\mathbf{G}$ is also a conditional, such a strategy also constructs **succ** when applied to the game master. In this case, the reduction process of making its own $\mathbf{G}'$ is as follows: $\mathbf{G}S_{\mathrm{self}} \rightarrow \mathbf{succ}(\lambda x{\cdot}x\ulcorner 0\urcorner\ulcorner 1\urcorner) \rightarrow \lambda y.y\mathbf{F}(\lambda x.x\ulcorner 0\urcorner\ulcorner 1\urcorner)$. Obviously this $\mathbf{G}'$ cannot give Barendregt numerals for **T** and **F**, and those λ-terms are distinguished as syntactically incorrect strategies.

The successful strategies can discriminate between **G** and **G**′ when they are input to the conditionals. They produce the syntactically correct **G**′ when applied to the game master **G**, while they construct **succ** and apply it to a numeral when applied to $\lambda x.x\lceil 0\rceil\lceil 1\rceil$ of **G**′. In addition, most such strategies construct $\lambda x.x\lceil 0\rceil\lceil 1\rceil$ when applied to the game master **G**. Therefore, they can obtain two points against themselves.

Specifically, these strategies can discriminate between two types of conditionals by the former part of the conditional. Simple strategies such as **T** and **F** cannot distinguish these two conditionals; they simply return the former part or the latter part of a given conditional. They merely behave against the conditional $\lambda x{\cdot}xPQ$ as:

$$(\lambda x.xP\#)\mathbf{T} \to P, \qquad (\lambda x.x\#Q)\mathbf{F} \to Q \tag{17}$$

where # represents an arbitrary combinator. On the other hand, the successful strategies can change their behavior according to the components of conditionals. Strategy (2) of Table 3 can distinguish between two types of conditionals as follows.

$$(\lambda x.x(\lambda x.\underline{x}\#\#)Q)\mathrm{Str} \to Q, \tag{18}$$

$$(\lambda x.x(\lambda \underline{x}.x)Q)\mathrm{Str} \to \mathbf{succ}\, Q. \tag{19}$$

In other words, they deceive the given conditional by "dissecting" the components of the function.

6. Evolutionary simulation of the λ-game

6.1. Model

In the previous section, we classified and analyzed the types of strategies in the λ-game system. Here, we study the evolution of strategies of λ-terms.

Our evolutionary dynamics consists of the population of λ-terms. We employ genetic programming (GP) as an evolutionary operator, because a λ-term is expressed by a parse tree. GP starts with an initial population of n players, consisting of elementary strategies, namely **T** and **F**. At each generation, each player (λ-term) plays the λ-game against all other players. The fitness of the λ-term is the average score of all matches against the other players. At each generation, the k players with the lowest value of fitness are replaced with the k players having the highest fitness. Then the reproduced strategies are mutated by genetic operations. In this evolutionary simulation, we employ the same game master **G**, defined in Eq. (8).

6.2. Genetic programming in the λ-calculus

We take the following process for the genetic operations. Our mutation and crossover operators were inspired by the work of Heiss-Gzedik and Fontana (unpublished).

(1) Mutation

According to the definition of the λ-term, mutation is naturally introduced as adding or removing abstractions and applications.

Abstraction of new variables can be introduced by inserting variables before any sub-λ-term. Similarly, any abstraction that does not bind a variable can be deleted.

The procedure for insertion of applications is as follows. A sub-λ-term is chosen randomly and is made either an operator or operand. Then, the missing operand or operator term is chosen randomly from variables and the set of combinators (**T**(=λ*xy*.*x*), **F**(=λ*xy*.*y*), **I**(=λ*x*.*x*), **S**(=λ*xyz*.*xz*(*yz*)), **V**(=λ*xyz*.*zxy*)). To insert a variable, it should be bound where the missing λ-term is inserted. Similarly, an application can be removed by erasing either the operator or the operand λ-term. The mutation occurs with probability R_{mutation} for each player at each generation. The type of mutation is randomly selected from the above four mutations (inserting/removing abstraction/application) and occurs on one randomly selected subterm.

(2) Crossover

Crossover in λ-terms would correspond to the exchange of subterms. However, the exchange of arbitrary subterms of combinators may cause free variables in the offspring λ-terms. To ensure conservation of the closure, we allow only combinators to be exchanged. The crossover occurs with probability $R_{\mathrm{crossover}}$ for each player, and the crossover partner is randomly chosen.

6.3. Simulation results

The population consists of 100 individuals ($n = 100$) and the initial population consists of 50 **T**s and 50 **F**s. Other parameters were chosen as $R_{\mathrm{mutation}} = 0.1$, $R_{\mathrm{crossover}} = 0.05$, and the number of replaced players as $k = 10$ throughout the experiments.

Fig. 2 shows an example of evolution dynamics. The horizontal axis corresponds to the generation, while the vertical axis corresponds to the average scores of the players by the number of games played and by the size of population (upper figure) and the average of the maximal depth of each λ-term (lower figure).

6.3.1. Time evolution of population

We see that the average score increases in a stepwise way with generations. In early generations, the strategy **F** spreads in the population rapidly, and the population is dominated by **F**. In these generations, therefore, the average score is distributed around one. Then, new strategies that are similarly effective but different shapes of λ-term invade and spread neutrally in the population. In other words, these strategies give the same $\mathbf{G}'$ as **F** (i.e., $\lambda x.x\lceil 0\rceil\lceil 1\rceil$) as their own $\mathbf{G}'$, and they acquire one point from this $\lambda x.x\lceil 0\rceil\lceil 1\rceil$.

As the generations continue, larger λ-terms and then a new strategy that acquires two points on average will appear (see steps 6000–9000 in the lower part of Fig. 2). The population is dominated by the strategies that have $\lambda x.x\lceil 0\rceil\lceil 1\rceil$ as their $\mathbf{G}'$. The new strategy, having the same $\mathbf{G}'$, can obtain two points from the dominant strategies, while the dominant strategies obtain one point. Therefore, the new strategy spreads through the population.

After the new strategy dominates the population, similarly effective strategies invade in the same way as **F** was invaded in the earlier generations. In other words, strategies that have $\lambda x.x\lceil 0\rceil\lceil 1\rceil$ as their intermediate $\mathbf{G}'$ forms and acquire two points from this $\mathbf{G}'$ spread neutrally in the population. These strategies obtain two points from the conditional $\mathbf{G}'$ in the same way as the strategy described in Section 5.2. They deceive the conditional $\mathbf{G}'$ and compose **succ**$(=\lambda xy.y(\lambda xy.y)x)$ through the interference process with the conditional $\mathbf{G}'$.

As evolution proceeds, new strategies that obtain higher scores appear. They obtain more than two points by making not only one **succ** but several **succ**s. Hence, the average score increases step by step with the generations. We show here an example of a strategy that obtains four points on average at the 15,000th generation:

$$\begin{aligned}\mathrm{Str} = {} & \lambda x_1.x_1(\lambda x_2x_3x_4x_5.x_5(\lambda x_6.x_4\mathbf{F})\\ & (\lambda x_9.x_9(x_3(\lambda x_{10}x_{11}x_{12}.\mathbf{F}))\\ & (\lambda x_{15}.x_{15}\mathbf{F}(\lambda x_{18}.x_{18}\mathbf{F}x_4))))\\ & (\lambda x_{21}x_{22}x_{23}x_{24}.x_{24}x_1(\lambda x_{25}.\mathbf{F})\\ & (\lambda x_{28}x_{29}.\mathbf{F}))(\lambda x_{32}.\mathbf{F})\end{aligned}$$

It should be noted that almost all these strategies generate $\lambda x.x\lceil 0\rceil\lceil 1\rceil$ as their $\mathbf{G}'$, and they acquire the higher points from this $\mathbf{G}'$. This suggests that the strategies with conditional $\mathbf{G}'$ have the advantage in explor-

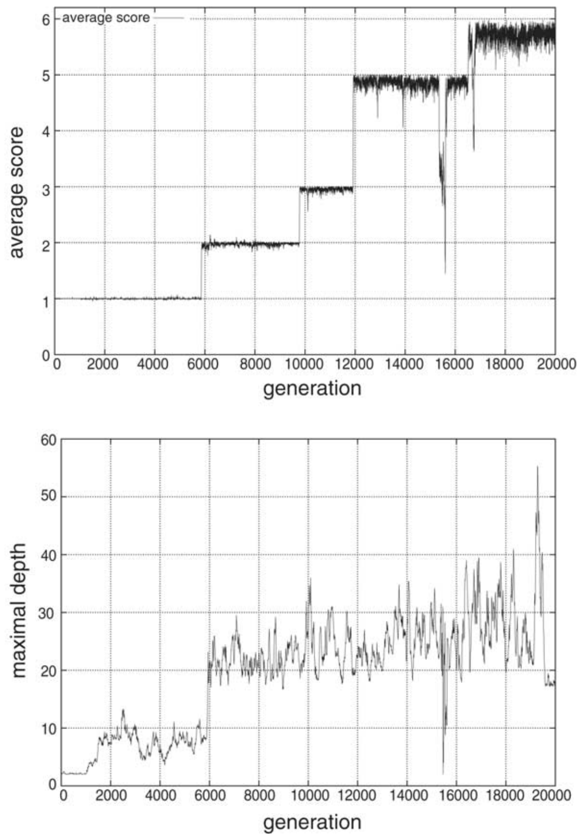

Fig. 2. Time evolution of the average score in the upper figure and the average of the maximal depth of λ-terms in the lower figure.

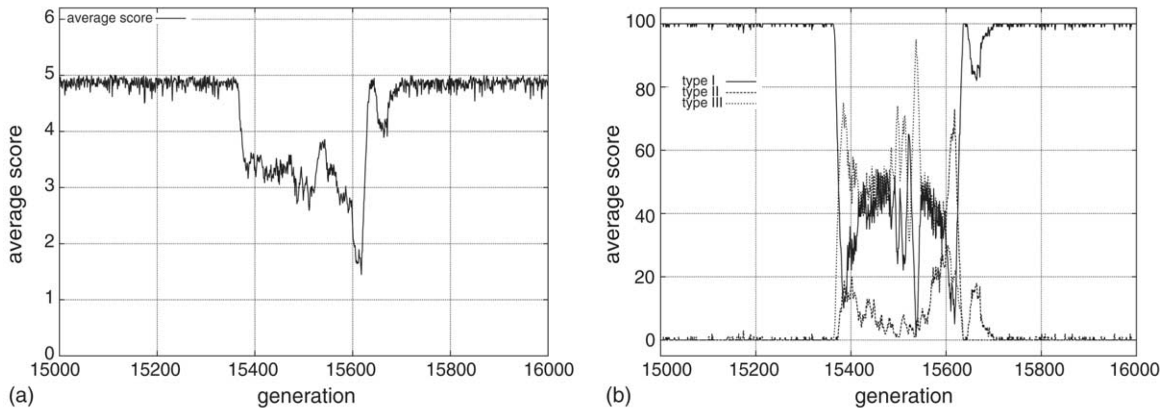

Fig. 3. The invasion of a strategy of other type of **G**′. The horizontal axis corresponds to the generation. The vertical axis corresponds to the average score in (a) and to the number of population of each type of **G**′ in (b).

ing suitable tree structures through the evolutionary processes.

6.3.2. Robustness of players with the conditional ***G***′

Populations of type I players (i.e., players with conditional **G**′) are sometimes invaded by players of other types. This behavior is observed in generations 15,300–15,700, and Fig. 3 is a scale-up of these generations. These new strategies produce their **G**′ of type III: a **G**′ that is reactive but not conditional. However, after several hundred generations, a new strategy with **G**′ of type I appears and takes over the strategy, and the population is again dominated by strategies of type I. A similar evolutionary process is observed in different evolutionary paths. This result suggests that the strategy with **G**′ of $\lambda x.x\lceil 0\rceil\lceil 1\rceil$ is robust against the invasion of strategies with different **G**′ types.

Strategies of type I can generate their **G**′ by relatively simple operations, because $\lambda x.x\lceil 0\rceil\lceil 1\rceil$ is prepared as a component of the game master **G**, i.e., it is a subtree of **G**. Therefore, players can make their own subtrees vary to obtain higher rewards from the new **G**′ of the invader, while they continue to generate the same **G**′ of $\lambda x.x\lceil 0\rceil\lceil 1\rceil$. On the other hand, players with strategies of other types find it difficult to adjust to the two processes of producing their own **G**′ and gaining from the **G**′ of other players simultaneously. We say that these strategies of type I have greater adaptability to new circumstances. Players of type I have a rather simple process to generate their own **G**′, whereas they have a rather complicated process to obtain higher rewards from the **G**′ of other players.

7. Discussion

An abstract game system has been proposed in terms of a λ-calculus formalism. A great advantage of λ-calculus formalism compared with dynamical systems formalism is that it can take both variables and functions as its input values. This advantage has allowed us to study a meta-game structure.

Howard tried to analyze a meta-game system, but while his analysis is interesting, we require a tool to systematize the analysis. A λ-game system provides one possible realization.

For a λ-game version of the PD game, we found that there were many meta-strategies that were syntactically correct but interfered with the rules of the game. This observation is based on the following analysis.

First, we focused on the form of **G**′ that was constructed by each strategy as an intermediate form. These **G**′s represent the kind of “attitude” players take towards other players and therefore the game situation strongly depends on the characteristics of the intermediate **G**′. Some strategies have a non-reactive **G**′ to return the same output irrespective of the opponent’s strategy. These strategies correspond to AllD and AllC in the iterated PD game. Other strategies have reactive **G**′s. Some of the **G**′s are interpreted as conditionals of the λ-calculus. They realize Howard's meta-level strategies in the sense that a game is one-shot but players can interact reactively against each other. That is to say, they have a *function* as their strategy, so that they can change their attitude depending on the opponent's strategy. This is similar to what Howard calls meta-players. They reinterpret the rules of the game by inferring the opponent's behavior. Because the game score can be rewritten by such strategies, we say that the matrix of the meta-game is the outcome of interpretation by inference of the opponent's behavior.

Second, we examined how the meta-strategies obtain their rewards from **G**′. Some strategies that obtained the higher rewards can interfere with **G**′s of conditional form. They interfere with the **G**′ by using a *constant numeral* in **G**′ as a *function*. They construct the combinators **succ** and apply them to numerals. Their behavior is different from Howard's meta-strategies, because while Howard's meta-strategies cannot play with the same-level player, meta-strategies in the λ-game can play against the same-level player. This becomes possible by describing variables (constant numerals) and functions (*successors*) at the same level.

We have also developed a model of evolution of the meta-strategies through genetic programming. In the evolutionary simulations, we found that they improve the composition of **succ** to make not only one **succ** but several, and therefore the strategies evolved into obtaining higher scores. We also observed the advantage of strategies of type I (conditional) from the viewpoint of evolutionary stability. This is explained by the simplicity of the reduction process of **G**′.

In this study, dynamics was introduced to the strategies as evolutionary dynamics. However, this is insufficient to investigate the endogenous dynamics of rules, which are the significant features of open-ended game systems, i.e., "play" behavior. It is interesting to introduce dynamics into rules i.e., the game master. Hashimoto and Kumagai (2003) proposed a meta-evolutionary game dynamics for studying the dynamics of rules and individual behaviors. The coevolution of strategies and the game master as in Hills' experiment (1990) is also an interesting future problem. A flexible λ-game formalism would help to ground these social systems studies.

Acknowledgment

This work is partially supported by Grant-in-aid (No. 09640454) from the Ministry of Education, Science, Sports and Culture. This work was also partially supported by the research project "Experimental Economics: A New Method of Teaching Economics and the Research on Its Impact on Society," which is supported and funded by the Ministry of Education, Culture, Sport, Science and Technology of Japan and Kyoto Sangyo University.

References

Anderlini, L., 1990. Some notes on Church's Thesis and the theory of games. Theory Decision 29, 19–52.

Axelrod, R., 1984. The Evolution of Cooperation. Basic Books, New York.

Bacharach, M.O.L., 1997. The epistemic structure of a theory of a game. In: Bacharach, M.O.L., et al. (Eds.), Epistemic Logic and the Theory of Games and Decisions. Kluwer Academic Publishers, Dordrecht, pp. 303–344.

Barendregt, H.G., 1981. The Lambda Calculus: Its Syntax and Semantics. Elsevier, Amsterdam.

Binmore, K., 1987. Modeling rational players I and II. Econ. Philos. 3, 179–214; 4, 9–55.

Fontana, W., Buss, L.W., 1994a. The arrival of the fittest: toward a theory of biological organization. Bull. Math. Biol. 56, 1–64.

Fontana, W., Buss, L.W., 1994b. What would be conserved if 'the tape were played twice'? Proc. Natl. Acad. Sci. U.S.A. 91, 757–761.

Hashimoto, T., Kumagai, Y., 2003. Meta-evolutionary game dynamics for mathematical modelling of rules dynamics. In: Banzhaf, W., et al. (Eds.), Advances in Artificial Life. Springer, pp. 107–117.

Heiss-Gzedik, D., Fontana, W. Evolution of λ-expressions through genetic programming, unpublished. http://www.santafe.edu/∼walter/Pages/publications.html.

Hills, D., 1990. Co-evolving parasites improve simulated evolution as an optimization procedure. Physica D 42, 228–234.

Hofstadter, D.R., 1985. Metamagical Themas. Basic Books, New York.

Howard, N., 1966a. The theory of meta-games. Gen. Syst. 11, 167–186.

Howard, N., 1966b. The mathematics of meta-games. Gen. Syst. 11, 187–200.

Howard, N., 1971. Paradoxes of Rationality: Theory of Metagames and Political Behavior. MIT Press, Cambridge, MA.

Ikegami, T., 1994. From genetic evolution to emergence of game strategies. Physica D 75, 310–327.

Ikegami, T., Taiji, M., 1998. Structures of possible worlds in a game of players with internal models. Acta Polytech. Scand. Ma 91, 283–292.

Ikegami, T., Taiji, M., 1999. Imitation and cooperation in coupled dynamical recognizers. In: Floreano, D., et al. (Eds.), Advances in Artificial Life. Springer-Verlag, pp. 545–554.

Kaneko, M., Nagashima, T., 1996. Game logic and its applications I. Studia Logica 57, 325–354.

Langton, C.G. (Ed.), 1992. Artificial Life II. Addison-Wesley, CA.

Lindgren, K., 1992. Evolutionary phenomena in simple dynamics. In: Langton, C.G. (Ed.), Artificial Life II. Addison-Wesley, CA, pp. 295–311.

Masumoto, G., Ikegami, T., 2001. The lambda-game system: an approach to a meta-game. In: Kelemen, J., Sosik, P. (Eds.), Advances in Artificial Life, LINAI 2159. Springer, pp. 695–699.

Suber, P., 1990. Nomic, A Game of Self-Amendment. The Paradox of Self-Amendment: A Study of Logic, Law, Omnipotence and Change. Peter Lang Publishing, New York.

Taiji, M., Ikegami, T., 1999. Dynamics of internal models in game players. Physica D 134, 253–266.

Vreeswijk, G.A.W., 1995a. Formalizing Nomic: working on a theory of communication with modifiable rules of procedure. Technical Report CS 95-02, Department of Computer Science, FdAW, University of Limburg, The Netherlands.

Vreeswijk, G.A.W., 1995b. Several experiments in elementary self-modifying protocol games, such as Nomic. Technical Report CS 95-06, Department of Computer Science, FdAW, University of Limburg, The Netherlands.