Evolution, Development and Learning in the Prisoner's Dilemma Game

Yukimasa OGAWA Takaya ARITA
Graduate School of Information Science, Nagoya University
Furo-cho, Chikusa-ku, Nagoya 464-8601, Japan
E-mail: ogawa@create.human.nagoya-u.ac.jp, arita@nagoya-u.jp

Abstract

Evolution, learning and development are the three main adaptive processes that enable living systems to adapt to environments on different time scales. The purpose of our study is to investigate the relationships among evolution, learning and development, especially in dynamic environments where there is no explicit optimal solution through generations, and the fitness of an individual depends on the interactions in a population. To do this, we construct a computational model using the iterated Prisoner's Dilemma game as dynamic environments. In the model, evolution and learning is achieved by a genetic algorithm and a Meta-Pavlov learning, respectively. Development is handled by two alternative computation-universal mechanisms: a tag system and a Turing machine. The results showed that almost all experiments we conducted finally established cooperation through evolution, learning and development, while there were various scenarios in which cooperative relationships were established, corresponding to the flexibility in the respective roles of them.

Keywords: development, genetic algorithm, trilateral adptation, the iterated prisoner's dilemma, artificial life.

1 Introduction

Evolution, learning and development are the three main adaptive processes that enable living systems to adapt to environments on different time scales. These processes do not occur in isolation, and interactions among them are very complex and not clearly understood in both biology and engineering. Combinations of them have a long history in the fields of artificial life, evolutionary computation, and engineering. Also, recently, interactions between evolution and development, so-called "evo-devo" have attracted researchers in both biology and artificial life. However, very few models combine all these three adaptive processes, among which Downing introduced a developmental process into Hinton and Nowlan's very simple model in which environment was static and the optimal solution was fixed, and focused on the evolution of developmental process by analyzing the feature of the genomes arising in the process of the Baldwin effect [1].

The purpose of our study is to investigate the relationships among evolution, learning and development, especially in dynamic environments where there is no explicit optimal solution through generations, and the fitness of an individual depends on the interactions in a population. To do this, we have adopt a synthetic approach and construct a computational model using the Iterated Prisoner's Dilemma (IPD) game as dynamic environments. In the model, evolution is achieved by a genetic algorithm and learning is achieved by a simple improvement algorithm termed Meta-Pavlov. Development is a kind of mapping process from genotype to phenotype, and is the least understood one of the three. We focus on two computation-universal mechanisms for development: a tag system and a Turing machine, and compare them with the results of the experiments.

2 Model

We construct a computational model in which three processes (evolution, learning and development) create the strategies for the Iterated Prisoner's Dilemma (IPD) game (Figure 1). The IPD game is a simple 2-player non-zero-sum game, in each round of which each player independently chooses an action from cooperate (C) or defect (D) without knowing the other's choice, and obtains the score according to the payoff matrix (Table 1). We adopt the framework proposed by Downing [1], while we incorporate a tag system [4] as an alternative mechanism for development, and interpret the developed phenotypes as strategies for the IPD game. The genotype encodes both a mechanism performing a developmental process, which is either a tag system or a Turing machine, and an initial tape on which the mechanism runs. The final developed tape as a phenotype is interpreted as a deterministic strat-

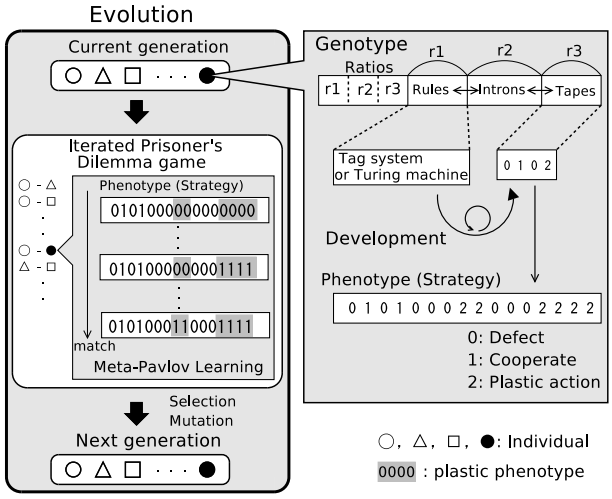

Figure 1: Evolution, development and learning in the model.

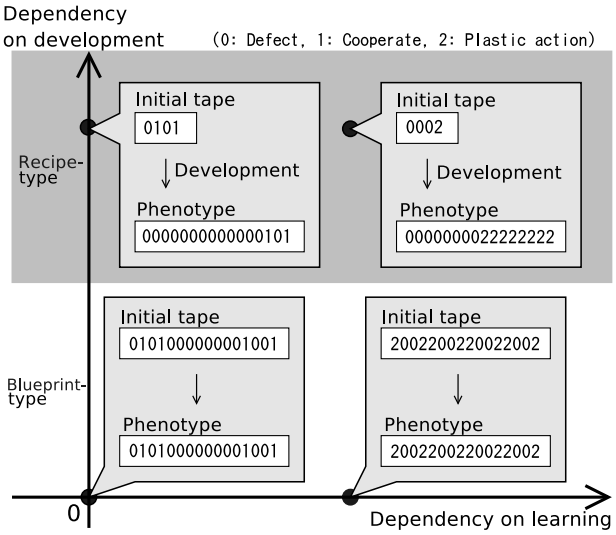

Figure 2: Individuals in the space of learnability and developability.

Table 1: Payoff matrix for the Prisoner's Dilemma game.

player \ opponent	cooperate	defect
cooperate	(3, 3)	(0, 5)
defect	(5, 0)	(1, 1)

(player's score, opponent's score)

egy of the IPD game. The action of the plastic phenotype is changed via a simple improvement algorithm termed "Meta-Pavlov" [3] during a game. The average score of each individual is regarded as a fitness value, new population is generated by the roulette wheel selection according to the scores, and then mutation is performed on a bit-by-bit basis.

The essential point of the model is that the developmental process can determine not only the strategies but also the possible amount of improvement by learning during game play, and at the same time the evolutionary process can determine this developmental process (Figure 2). Individuals evolved in the model are characterized in the space shown in this figure, of which the horizontal axis represents the possible amount of improvement by learning and the vertical axis represents the level of the degree of dependence on development. For example, evolution can generate the individual (at the coordinate origin) that never undergoes a developmental process, which means the genotype-phenotype mapping is 1 to 1, and has no learning ability, which means the phenotype has no plasticity.

2.1 Genotype

Each genotype has *genotype-length* bits, and is composed of four fields: ratio, rule, intron, and tape. The ratio field have $3t$ bits, which encodes three integers: r_1, r_2 and r_3, each representing the ratio in length of the rules, the introns and the tapes, respectively. The intron field just separates the rule and the tape field, and is not used at all in subsequent development and learning. The tape field is used as an initial tape on which the developmental process runs.

In case of a tag system, the rule field encodes the transition rules, each of which specifies the elements to be removed from the beginning of a tape, and the elements to be appended onto the end of the tape. Each transition rule requires $(P_{ts}+Q_{ts})\times n_{ts}$ bits, where P_{ts} and Q_{ts} are the numbers of removed and appended elements, respectively, and n_{ts} is the number of bits used to encode these elements. The tape symbol encodes an integer between 0 and $2^{n_{ts}}-1$. These integers are converted to 0, 1 or 2 with nearly equal probability.

In case of a Turing machine, the rule field encodes transition rules as 5-tuples of the form (s, x, s^*, x^*, a), in which s is the current state, x is the tape symbol being read, s^* is the next state, x^* is the next symbol, and a is the action, that is either overwriting x with x^*, or inserting x^* to the immediate right of x on the tape. Each transition rule requires $2m_{tm}+2n_{tm}+1$ bits, where n_{tm} and m_{tm} are the numbers of bits used to

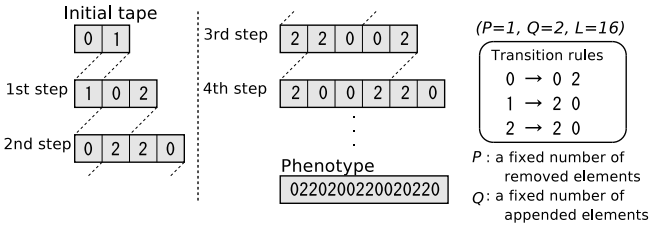

Figure 3: An example of development in case of a tag system.

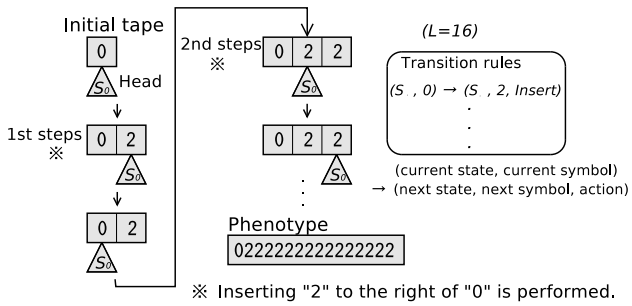

Figure 4: An example of development in case of a Turing machine.

encode a tape symbol and a state, respectively. Each element is expressed with 0, 1 or 2 as in the case of a tag system.

2.2 Development

The phenotype is generated from both the transition rules for a tag system or a Turing system and an initial tape into which developmental process decodes the genotype.

Figure 3 shows an example of development in case of a tag system. In general, the transition rules for a tag system specify that a fixed number of elements should be removed from the beginning of the sequence, and depending on these elements, several number of elements should be appended onto the end of the sequence. In the current model, both the number of the elements removed and the number of the elements appended are fixed, and are 1 and 2 respectively. In the example shown in Figure 3, first, 0 is removed from the beginning of the tape, and then the rule corresponding to 0 is retrieved from the transition rules. Since $0 \rightarrow 02$ is included in the transition rules, 02 is appended onto the end of the tape in this case. This type of replacement process continues until the tape size reaches L.

Figure 4 shows an example of development in case of a Turing machine, which is the same as the one adopted in [1]. The head always begins the developmental process in state 0 at the left edge of the cell on the tape. In this figure, the current state and the tape symbol are s_0 and 0, respectively, and then the rule corresponding to $(s_0, 0)$ is retrieved from the transition rules. As $(s_0, 0) \rightarrow (s_0, 2, \mathit{Insert})$ is included in the transition rules, 2 is inserted to the immediate right of 0 on the tape, and the head moves right one cell. If the head reaches the right edge of the cell on the tape, it is again set to the left edge of the cell on the tape. This type of movement and replacement process continues until either the tape size reaches L or the number of the steps reaches $max\text{-}devp\text{-}steps_{tm}$.

2.3 Phenotype

The final developed tape as a phenotype is interpreted as a deterministic strategy for the IPD game, and it defines the next action according to the history of actions of both players, which is the same as in Lindgren's model [2] but is introduced plasticity as in Suzuki 's model [3]. Each strategy is represented as a string of 0's (defect), 1's (cooperate) and 2's (plastic action). "x" is used to express this plastic phenotype in this paper.

A strategy S of memory m can be expressed by associating an action A_k (0, 1 or 2) with each history k as follows:

$$h_m = (a_{m-1}, \cdots, a_1, a_0)_2, \tag{1}$$

where a_0 is the opponent's previous action (0 and 1), a_1 is the previous player's action. a_2 is the opponent's next to previous action, and so on. S for a strategy of memory m can be expressed by associating an action A_k (0, 1 or 2) with each history k as follows:

$$S = [A_0 A_1 \cdots A_{n-1}] \ \ (n = 2^m). \tag{2}$$

2.4 Learning and Evolution

A plastic phenotype can be changed by learning based on Meta-Pavlov during a game. Each agent changes its phenotypes according to the result of each round by referring to the Meta-Pavlov learning matrix (Table 2). It does not express a strategy but the way to change its own strategy (phenotype) according to the result of the current round, though this matrix is the same as that of the Pavlov strategy. The learning process is described as follows: At the beginning of the game, the plastic phenotype is set randomly to either 0 or 1. If the phenotype used in the round is plastic, the phenotype is changed to the corresponding value in this matrix based on the result of the round. The

Table 2: Strategy matrix for Meta-Pavlov learning.

player \ opponent	cooperate	defect
cooperate	C	D
defect	D	C

agent uses the new strategy specified by the changed phenotype from the next round on.

We shall consider a population of N individuals interacting according to the IPD. Each bit of gene is set randomly in the initial population. The round robin tournament is conducted among strategies under the condition in which the performed actions could be changed by noise with probability p_n. Each plastic phenotype is set randomly at the beginning of games. Rounds are repeated with the probability p_f, which is decided at the end of each round. The tournament is "ecological": The average score of each individual is regarded as a fitness value, a new population of individuals is generated by the roulette wheel selection according to the scores, and mutation is performed on a bit-by-bit basis with the probability p_m.

3 Evolutionary Experiments

We conducted 20 trials for 10000 generations in each of the experiment with the tag system and that with the Turing machine, focusing on the strategies of memory 4 ($m = 4$). The other parameters were as follows: *genotype-length* $= 300$, $N = 1000$, $p_m = 1/5000$, $p_n = 1/25$, $p_f = 0.995$, $t = 5$, $P_{ts} = 1$, $Q_{ts} = 2$, $n_{ts} = 5$, $L = 16$, $m_{tm} = 3$, $n_{tm} = 5$, $max\text{-}devp\text{-}step_{tm} = 100$.

First of all, we have found that cooperation was finally established through evolution, learning and development in most of the trials we conducted, while there were various scenarios in which cooperative relationships were established, corresponding to the flexibility in the respective roles of them. The last generation in each trial was classified into the following five types: XX, XL, DX, DL and US as shown in Table 3, in which D represents strong dependency on development, L represents strong dependency on learning, X represents weak dependency and US represents the gray type owing to instability in the boundary areas. For example, XX represents that the average individual depended neither on development nor on learning, and DL represents that the average individual depended both on development and on learning. Dependency on development (D-dependent) was decided in case that the average times of rule application in development was more than 5, and dependency on learning (L-dependent) was decided in case that the average ratio of plastic phenotypes chosen during game play was more than 0.5, in the last generation. The average score was kept around 2.4 in the last generation in almost all trials in L-demendent case (XL and DL), while DX trials varied in the score from trial to trial, and achieved the highest score, 2.6.

Table 3: Classification of the results by dependence on development or learning.

Classification				Tag system	Turing machine
stable	D-independent	L-independent	XX	0	3
		L-dependent	XL	1	7
	D-dependent	L-independent	DX	2	9
		L-dependent	DL	15	1
unstable			US	2	0

From this table, we see that 15 trials of 20 trials were DL in case of the tag system. This means that the tag system tended to work efficiently to help evolution explore the cooperative roles of learning and development in dynamic environments, in other words, three adaptive processes evolved the individuals depending on both learning and development, which could result in the smooth establishment of cooperative relationships.

Figure 5 shows an evolutionary transition of the average score, dependency on learning and dependency on development in a typical trial of DL. Dependency on learning and dependency on development were calculated as the average ratio of "2"s in all of the developed expressions (which is different from the definition of L-dependent) and the number of times of applying rules (in development) divided by 50, respectively. The dashed line shows the threshold that separates D-dependent and D-independent. The evolutionary scenario in this figure is summarized as follows: Defect strategy spread until around the 50th generation, which decreased the score (which is difficult to see from this figure). Simultaneously, partially plastic strategies (e.g. [0xxx0x0x0x0x010x]) spread in the population. Then, fully plastic strategies (e.g. [xxxxxx0xxxxxxxxx]) occupied the population, which increased the plasticity and the average score. Around the 1300th generation, some strategies (e.g. [xx0x000x0x0000xx]) spread, which decreased the plasticity, but around the 1600th generation, the plasticity increased again. Finally, some robust strategies (e.g. [x00x000x000x000x]) occupied the population. The number of times of applying developmental

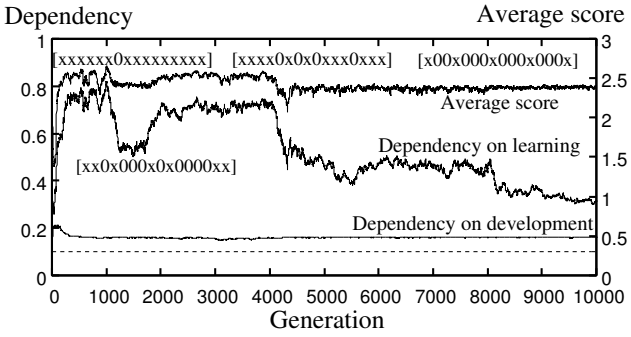

Figure 5: An experimental result (tag system).

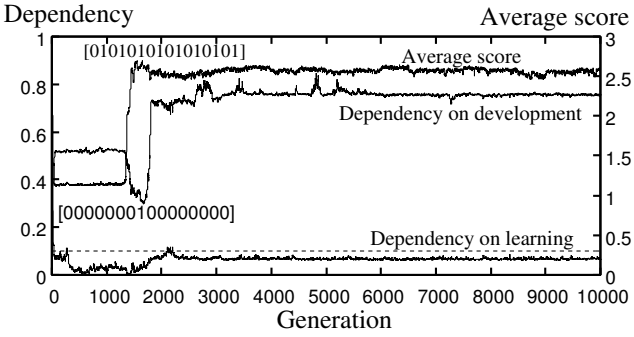

Figure 6: An experimental result (Turing machine).

rules was around 8 throughout the generations.

11 trials of 15 DL trials showed the similar evolutionary scenario to the above described one, while 4 trials had other scenarios like the one, for example, in which dependency on learning did not reach such a high value and some robust strategies like [x00x0xx00xx0x00x] evolved to occupy the population while keeping the average score high.

As for the cases of the Turing machine, we see from this table that 16 trials of 20 trials were classified into XL or DX, which means that either learning or development tended to work exclusively to build cooperative relationships. The possible cause of this is the flexibility in the complementary roles of learning and development in evolution of cooperation. Just 2 trials of the 9 DX were the special case in which cooperative relationships could not be established since defect-oriented strategies continued to occupy the population.

A typical case of XL is summarized as follows: D-dependent defect strategies spread in the population in initial phase. However, dependency on learning did not increase and the strategies like [x00x00x00x00001x] finally evolved with the average score converging to about 2.4 and with dependency on learning hovering around 0.4. Figure 6 shows a transition in a typical trial of DX. Firstly, D-dependent defect strategies (e.g. [0000000100000000]) grew in the population, which decreased the average score until around the 1300th generation. But then, "Tit-for-Tat"-like non-plastic strategies (e.g. [0101010101010101]) spread in the population, which increased the average score sharply. Finally, some cooperative strategies without plasticity (e.g. [110x000000000001]) occupied the population, so that average score converged to about 2.6.

4 Conclusion

We have constructed a computational model using the iterated Prisoner's Dilemma game as dynamic environments in order to investigate the relationships among evolution, learning and development. Development is handled by two alternative computation-universal mechanisms: a tag system and a Turing machine in the model. The evolutionary experiments have shown that when adopting a tag system, each individual tended to depend on both learning and development, which often resulted in the smooth establishment of cooperative relationships. While, when adopting a Turing machine, either development or learning worked exclusively and built cooperative relationships. These results show the flexibility in the roles of learning and development in establishment of cooperation. Differences between the results of the two developmental mechanisms could possibly be due to the fact that the tag system could work more efficiently to help evolution explore cooperative roles of learning and development in dynamic environments.

References

[1] Downing, K. L.: Development and the Baldwin Effect, *Artificial Life*, Vol. 10, No. 1, pp. 39–63, 2004.

[2] Lindgren, K.: Evolutionary Phenomena in Simple Dynamics, *Artificial Life II*, pp. 295–311, 1991.

[3] Suzuki, R. and Arita, T.: Interactions between Learning and Evolution: The Outstanding Strategy Generated by the Baldwin Effect, *Biosystems*, Vol. 77, Issues 1–3, pp. 57-71, 2004.

[4] Wolfram, S.: *A New Kind of Science*, Wolfram Media, pp. 93–96, 894–895, 2002.