Constructing an Autonomous Agent with an Interdependent Heuristics

Koichi Moriyama and Masayuki Numao

Department of Computer Science,
Graduate School of Information Science and Engineering,
Tokyo Institute of Technology.
2–12–1, Ookayama, Meguro, Tokyo, 152–8552, Japan.
koichi@nm.cs.titech.ac.jp, numao@cs.titech.ac.jp

Abstract. When we construct an agent by integrating modules, there appear troubles concerning the autonomy of the agent if we introduce a heuristics that dominates the whole agent. Thus, we design an agent that has an interdependent heuristics influenced by a module controlled by the heuristics, and we apply these agents into a problem of obtaining cooperation of Multi-Agents. We enable a present method that can solve the problem in a reinforcement learning context to be applied into a dynamic environment, and the improved method is embodied into the agent as the interdependent heuristics. We conduct experiments comparing the proposed agents with agents such as those ones each of which has a heuristics controlled by a supervisor, then we empirically confirm that the proposed agent having the interdependent heuristics is the most flexible of all the tested agent.

1 Introduction

Many researchers want to construct autonomous agents that do not need to be controlled directly by a human during execution time. It might seem easy to construct the autonomous agent by introducing in advance some heuristics into the agent. If these heuristics are *absolute* for the whole agent, however, there appear troubles concerning the autonomy of the agent. Therefore, in this paper, we design an agent that does *not* have any absolute heuristics, but has an *interdependent* heuristics that is influenced by a module controlled by the heuristics.

We apply the agents to a problem of obtaining cooperation of Multi-Agents (MA) as an example of auto-adaptation of the agent to a changing environment. To solve this problem in a reinforcement learning context, Mikami et al. [4, 5] proposed the reward filtering method that uses a *filtered* reward instead of a reward itself in a reinforcement learning process. This method can be regarded as a kind of heuristics of the problem. However, since their method uses several fixed filtering functions only one of which is selected in advance by a designer, it is useless in a problem whose class we do not know in advance or whose conditions are variable. Therefore, we propose a parameterized filtering function

called *General Filter*. Since General Filter can work as one of all the original filtering functions by changing its parameters, it is important for the agent to get a way of controlling the parameters. Thus, in this paper, we embody General Filter into the agent with providing a learning mechanism for the parameters. To learn the parameters, however, the evaluation of the parameters is needed. So we introduce two evaluation methods: an *interdependent* one described above and a *subordinate* one that is subject to an absolute factor introduced by a designer. We also introduce another type of General Filter whose parameters are *controlled by a supervisor* that can check up on all the agents in the problem. We conduct experiments comparing several fixed filters and these three General Filters in a problem of obtaining cooperation of MA in order to determine the comparative performance of our proposed General Filter *with the interdependent evaluation*.

This paper is organized as follows. In Section 2, we describe troubles caused by an absolute heuristics and give an outline of an agent having an interdependent heuristics. In Section 3, we introduce the reward filtering method and propose General Filter. Then we embody General Filter into the agent outlined in Section 2 with the interdependent evaluation. In Section 4, we conduct experiments in a problem of obtaining cooperation of MA in order to compare the proposed agents embodied in Section 3 with agents such as those ones each of which has General Filter controlled by a supervisor. In Section 5, we discuss the result of the experiments and the social role of filters. In Section 6, we conclude this paper and mention several future works.

2 Outline of an Agent

We suppose that an agent has a learning module and a heuristic module for the learning module, perceives the outside environment through a sensor, acts to it through an actuator, and receives rewards from it. When we introduce a heuristics into the agent in advance, we usually tend to give *absolute power* for it. The structure of an agent having an absolute heuristics is shown in Fig. 1.

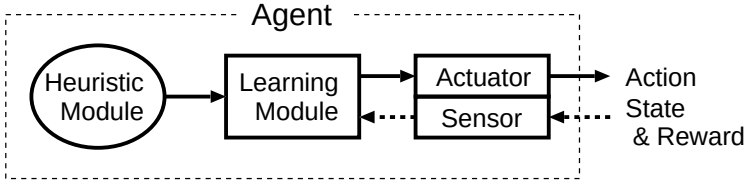

Fig. 1. Outline of an agent having an absolute heuristic module: solid arrows mean controls and dotted arrows mean data flows.

When an agent has a heuristic module that has absolute power, there appear troubles concerning the autonomy of the agent. If the absolute heuristics is fixed,

it should be effective for all inputs of the agent. This might be feasible in a very simple environment, but, in a complex environment, a fixed heuristics causes troubles like the frame problem [6]. On the other hand, if the absolute heuristics is variable, the problem switches to determining who changes the heuristics. If the heuristics directly controls actions of the agent, we are able to avoid these troubles because we can give the agent a learning mechanism that learns the actions from the *outside information*. However, in this paper, since the heuristic module controls the learning module in the agent, the heuristics can only control actions of the agent indirectly. Then, if a human is needed to change the heuristics, the agent is not autonomous anymore, and if it is supposed that a meta heuristics changes the heuristics, on the other hand, similar troubles described in this paragraph occur to the meta heuristics again.

In order to avoid the troubles, by extending the *interdependency* between the agent and the environment described above, we design an agent that does *not* have any absolute heuristics, but has an *interdependent* heuristics that is influenced by the learning module controlled by the heuristics. The structure of the agent having an interdependent heuristics is shown in Fig. 2. The arrow labeled by an asterisk ($*$) in Fig. 2 represents the interdependency between the learning module and the heuristic module.

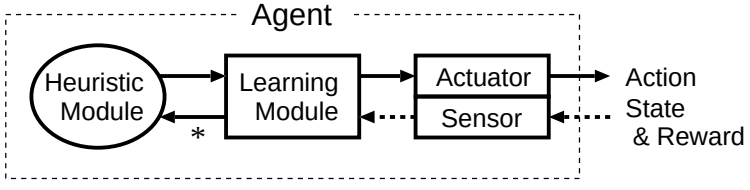

Fig. 2. Outline of an agent having an interdependent heuristic module: the arrow labeled by an asterisk ($*$) represents the interdependency between the learning module and the heuristic module.

3 Obtaining Cooperation of Multi-Agents

3.1 Classification of Multi-Agent Problems

In [2], problems for obtaining cooperation of Multi-Agents (MA) are classified into two classes[1].

- Cooperative Problem Solving
  Achieving common purposes among agents.

[1] In fact, in [2], this classification is given for Distributed Artificial Intelligence (DAI). At first, the studies of MA were formulated as a part of the studies of DAI, but the difference between MA and the rest of DAI is fading away now [2].

– Negotiation and Balancing
  Dealing with competition among agents.

Agents engaged in a problem belonging to the former class have common purposes, so their motives for cooperation, if they have, are to solve the problem faster, more effectively, and so on. Agents in a problem in the latter class, on the other hand, have independent purposes, so their motives for cooperation are to avoid deadlocks caused by shared resources.

3.2 Reward Filtering Method

To solve problems in the two classes described above in a reinforcement learning context, Mikami et al. [4, 5] proposed a method using a *filtered* reward instead of a reward itself in a reinforcement learning process. In their proposal, in order to filter its own reward, an agent uses a fixed filter constructed as a function of the mean reward of agents in the neighborhood of the agent, including the agent itself. These agents are called *neighbors* of the agent. Mikami et al. defined two types of filtering functions: Average Filter and Enhancement Filter. Average Filter flattens a reward that is over/under the mean reward of neighbors. Enhancement Filter, on the other hand, exaggerates a reward that is under the mean reward of neighbors. It has been shown in [4] that Average Filter and Enhancement Filter are effective for a problem in Cooperative Problem Solving and for a problem in the class called Deadlock Avoidance Problem similar to Negotiation and Balancing, respectively. Therefore, these filters can be regarded as a kind of heuristics for learning.

3.3 General Filter

Since the filters defined by Mikami et al. are fixed, they are useless in a problem whose class we do not know in advance and/or whose conditions are variable. So we propose a parameterized filter defined as follows:

$$r' = \begin{cases} M + \alpha(r - M) & \text{if } r < M \\ M + \beta(r - M) & \text{otherwise.} \end{cases}$$

In this definition, for a particular agent, r means a reward of the agent, r' means a filtered reward of the agent, M means the mean reward of neighbors of the agent, and α and β are parameters. This filter can represent all the original filters defined in [4] and other filters through a proper setting of the parameters. So we call this parameterized filter *General Filter*.

In the following, we use the term "the parameters" to refer to the parameters α and β of General Filter defined above. If an agent can adjust the parameters appropriately according to the changes in the environment, then the agent is able to make a heuristics that is effective for the problem whose class we do not know in advance and/or whose conditions are variable. Therefore, it is important for the agent to get a way of controlling the parameters. In this paper, we consider that the agent *learns* proper modifications of the parameters.

3.4 Embodying General Filter into an Agent

In this subsection, we discuss how to embody General Filter into an agent having an interdependent heuristics. On the agent structure shown in Fig. 2, the following modifications should be done:

- Divide the learning module into two components, one with Reinforcement Learning Module (RLM), and the other one with General Filter.
- Change the heuristic module to Parameters Learning Module (PLM), which is in charge of controlling General Filter.

By these modifications alone, however, there are two issues: the agent has no arrow labeled by an asterisk in Fig. 2, and, in order to learn the parameters in PLM, PLM needs the evaluation of the parameters. To solve these issues, we propose that the parameters are evaluated by RLM.

As a result of the modifications, the structure of the agent having General Filter with the interdependent evaluation takes the form shown in Fig. 3.

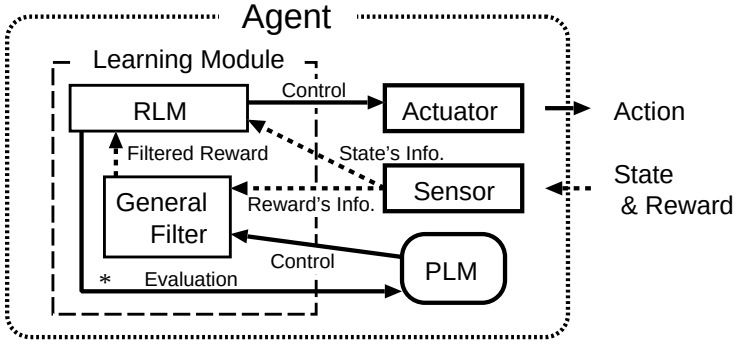

Fig. 3. The agent having General Filter with the interdependent evaluation

4 Experiments

In order to determine the comparative performance of our proposal, we conduct two experiments comparing the agent shown in Fig. 3 with several variants of the agent in a problem of obtaining cooperation of Multi-Agents (MA). Due to space limitations, we only show here the result applying the agents to a problem in the class Cooperative Problem Solving.

4.1 Preparation

As the experimental problem, owing to [4], we use a game similar to the *tragedy of the commons* [1] for the experiment. This problem is famous in the domain of game theory [7]. The tragedy of the commons means that if each player acts according to his/her *individual* rationality, it is happened that certain *common* property among the players is decreased and, in consequence, his/her own payoff is also decreased. Since the players must cooperate with one another to get more payoffs, this problem is an example in the class Cooperative Problem Solving.

An agent in our game has three actions: selfish, cooperative, and altruistic. In every cycle, each agent in the game shows an action simultaneously and gets a reward calculated by the actions of *all* the agents in the cycle. It is defined that the reward given for a particular agent in a particular situation is maximum when the agent acts selfishly, and minimum when the agent acts altruistically. However, it is also defined that negative common property (common cost) among the agents is increased when an agent acts selfishly and decreased when an agent acts altruistically, so the reward of each agent is small when all the agents act selfishly. If the variation of the common cost r_c is represented to Δr_c, the common cost r_c, initialized to zero at every cycle beginning, is added by Δr_c if an agent acts selfishly, is not changed if an agent acts cooperatively, and is subtracted by Δr_c if an agent acts altruistically. Once the common cost r_c has been calculated by the actions of all the agents, each agent gets a reward of $3 - r_c$ if the agent acts selfishly, $1 - r_c$ if the agent acts cooperatively, and $-3 - r_c$ if the agent acts altruistically.

We conduct two experiments: one is in a *static* environment, the other is in a *dynamic* environment. The static environment is the one where all factors except for agents are static. The dynamic environment is, on the other hand, the one where some factors besides agents are also variable. In this paper, in order to realize these environments, we make the parameter Δr_c fixed in the static environment and make it variable according to the number of cycles in the dynamic environment.

We use ten agents each of which has an unique ID in the experiments. Neighbors N_i of an agent a_i $(0 \leq i \leq 9)$ are defined as follows:

$$N_i \triangleq \{a_k \mid k = (i + j) \bmod 10,\ j = 0, 1, 2, 3\}.$$

Note that the neighbors include the agent itself.

In order to determine the comparative performance of our proposal, we have conducted the experiments comparing the following six types of agent. These types are divided into two groups: having a fixed filter or having General Filter. We describe three fixed filters first, then describe three General Filters. The three fixed filters are No Filter (NF), Full Average Filter (FAF), and Full Enhancement Filter (FEF). The agent having NF uses rewards themselves in learning. FAF is a filter into which the original two Average Filters defined in [4] are merged. FEF is a filter into which the original Enhancement Filter defined in [4] and an

opposite Enhancement Filter are merged[2]. NF, FAF, and FEF are represented as $r' = r$, $r' = M$, and $r' = 2r - M$, respectively. Symbols in the equations are identical with the ones described in Section 3.3.

The three General Filters are distinguished according to the way of evaluation of the parameters. The first type is that the parameters in an agent are evaluated by the *interdependent* method described so far (GF-Int). As the interdependent evaluation, the filtered reward, which is used in Reinforcement Learning Module (RLM), is *sign-inversed* in RLM and this sign-inversed filtered reward is sent to Parameters Learning Module (PLM). The second type is that the parameters in an agent are evaluated by a *subordinate* method using the sum of rewards the agent gets in a certain time interval fixed in advance (GF-Sub). As the fixed interval, we use ten cycles. The third type is that we introduce a *supervisor* that can check up on rewards of all the agents, and the parameters in the agents are controlled by the supervisor directly (GF-Sv).

In RLM, for an agent, states used for learning are defined as the combination of the actions of neighbors of the agent. This means that the agent can get only the local information about the environment. The mean reward of the neighbors, necessary for filtering a reward of the agent, is calculated by filtered rewards of them. The reinforcement signal used for learning suitable actions is a reward filtered by the filter the agent has inside.

For General Filter, initial values of the parameters are both set to 1, then the filter is identical with No Filter. We set upper and lower bounds of the parameters to 0 and 2, respectively. The parameters are modified, in a particular cycle, *before* a reward got by the agent is filtered.

When the parameters are changed in an agent itself, PLM learns three modifications, namely increasing, maintaining, and decreasing *each* parameter, through a reinforcement learning method. States used for learning the parameters are the modification itself of each parameter, and the range of this modification is determined randomly after considering the frequency of learning and modifying the parameters in PLM. This frequency is identical with the frequency of evaluating the parameters. Therefore, in this paper, when PLM is evaluated by the interdependent method, PLM modifies the parameters at every cycle, or when PLM is evaluated by the subordinate method, on the other hand, PLM modifies the parameters at every ten cycles.

When the parameters are changed by a supervisor, on the other hand, the supervisor learns the same three modifications of each parameter described above through reinforcement signals each of which is the sum of rewards got by all agents in a cycle. The supervisor directly modifies the parameters in all the agents at every cycle.

As the reinforcement learning method, we use Q-learning [8] with the learning rate and the discount factor both set to 0.5. Action selections using Q-values fol-

[2] Since the original Enhancement Filter exaggerates a reward *under* the mean reward of neighbors, we introduce an opposite Enhancement Filter that exaggerates a reward *over* the mean reward of neighbors. The original Enhancement Filter has a parameter determining the degree of exaggeration of rewards, and we set this parameter to 1.

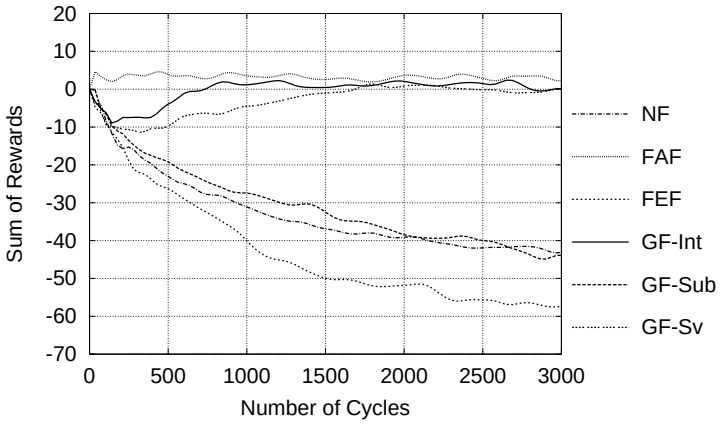

Fig. 4. Result in a static environment: each line in the plot shows the sum of rewards of the agents when applying the filter whose label is printed in the right part of the figure. NF stands for No Filter. FAF stands for Full Average Filter. FEF stands for Full Enhancement Filter. GF-Int stands for General Filter evaluated interdependently. GF-Sub stands for General Filter evaluated subordinately. GF-Sv stands for General Filter controlled by a supervisor.

low the probability calculated by Boltzman distribution [3] with the temperature set to 1.

4.2 Result in a Static Environment

Here we show the result of a experiment in a static environment. In this paper, to make the environment static, we set the parameter Δr_c to the constant value 1.

Figure 4 shows the result of this experiment. The horizontal axis designates the cycles of the game, and the vertical axis represents the sum of rewards got by all the agents. Each line is plotted every five cycles and approximated with a Bézier curve.

This figure shows that

- When using whether NF, FEF, or GF-Sub, the sum of rewards decreases as the number of cycles increases. This means that the tragedy of the commons is occurring.
- When using FAF, the line in the plot is almost horizontal. This indicates that FAF manages to keep a high value of the sum of rewards, thereby avoiding the occurrence of the tragedy of the commons.
- When using whether GF-Int or GF-Sv, although the sum of rewards decreases at the beginning, it stops decreasing halfway and starts increasing until it sets close to the sum of rewards of FAF. This means that both evaluations of General Filter are able to learn the proper parameters that can avoid the tragedy of the commons.

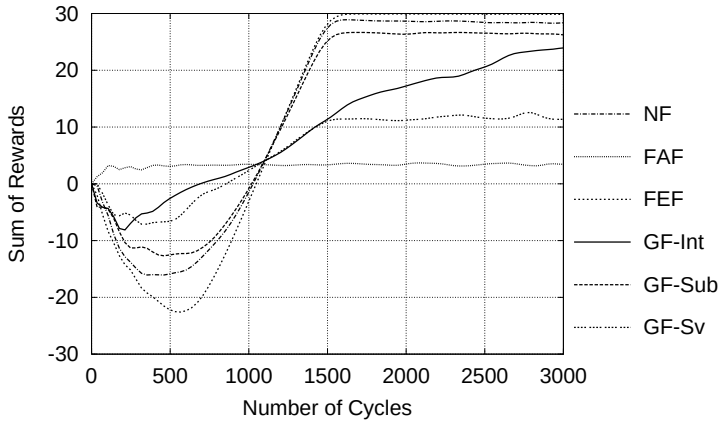

Fig. 5. Result in a dynamic environment: the label of each line is identical with the one in Fig. 4.

4.3 Result in a Dynamic Environment

On the other hand, we show here a result of a experiment in a dynamic environment. To make the environment dynamic, we make the parameter Δr_c variable according to the number of cycles as follows:

$$\Delta r_c = \begin{cases} 1 - \dfrac{cycles}{1500} & \text{if } cycles < 1500 \\ 0 & \text{otherwise.} \end{cases}$$

This definition of the parameter Δr_c implies that the agents should be cooperative at the beginning and do not have to be at the end.

Figure 5 shows the result of this experiment. The plot in the figure was constructed in a similar way to the one shown in Fig. 4.

This figure shows that

- When using whether NF, FEF, or GF-Sub, the sum of rewards decreases at the beginning, and it increases at the end. This means that the performance of these filters is simply influenced by the change of Δr_c.
- When using FAF, the line in the plot is almost horizontal. This means that FAF can not adapt to the changes in the environment.
- When using whether GF-Int or GF-Sv, at the beginning, the filter tries to be like FAF, but it tries to be like NF and others at the end. This means that GF-Int and GF-Sv can learn the proper parameters to be the best filter for the conditions of the environment. Note that although the sum of rewards of GF-Sv stopped increasing halfway, that of GF-Int continues to increase to that of NF and others. This means that the adaptability of GF-Int is superior to that of GF-Sv.

5 Discussion

In the experiment, although it is natural that General Filters were more adaptable than the fixed filters, General Filter with the interdependent evaluation, in particular, was the most flexible of all the tested filters. We used the sign-inversed filtered reward got in a cycle as the interdependent evaluation of General Filter. Here we consider the reason why the operation of sign-inversion is good for learning the proper modifications of the parameters in a changing environment first.

It is true that, in a cycle, the parameters of General Filter are changed *before* the filter controls a reward, but we must consider that there is a time gap between the cycle in which the parameters are modified by Parameters Learning Module (PLM) and the cycle in which the agent performs an action influenced by these modified parameters. Since the parameters have an immediate effect on the learning module and not on the actuator of the agent, the influence of modification of the parameters is lately appeared in the action of the agent. Thus, for a particular cycle, an action of the agent producing a reward is the result of learning from filtered rewards that were calculated using the parameters *before modified*. Therefore, although the parameters have been renewed by PLM before the agent learns its behavior through Reinforcement Learning Module, the filtered reward is mainly influenced by the *former* values of the parameters. If the filtered reward is small, that is, the reward of the agent is also small, the former values of the parameters should be modified because they are not suitable for the environment. This means that the modification of the parameters is laudable, so the modification should receive a big reward. Similarly, if the filtered reward is big, the modification of the parameters should receive a small reward. These relations between the filtered reward and the evaluation of the parameters may be represented in the operation of sign-inversion. Thus, General Filter with the interdependent evaluation was the most flexible of all.

Next, we argue about the social role of filters. In the experiment, Full Average Filter (FAF) produced good results when cooperation among agents was needed. FAF cut a reward of an agent over and under the mean reward of neighbors of the agent, and the agent learns its behavior using this cut reward. Note that this cutting is executed in the agent itself, namely, is executed *not* out of the agent, even if the agent has General Filter controlled by a supervisor. Thus, since the filter can change the *internal* evaluation, which the agent has inside, of the environment, the filter might be regarded as a kind of characteristic or emotional parameter of the agent. Now we can consider that the character of an agent having FAF is reformed not to pursue rewards by the filter. This agent was not rational anymore in the sense used in game theory, so Multi-Agents in which each agent has FAF were able to avoid the tragedy of the commons.

6 Conclusion

In this paper, we have designed an agent that has two interdependent modules, the learning module and the heuristic module, which both influence each other.

Then we applied the agent to a problem of obtaining cooperation of Multi-Agents. We introduced the reward filtering method to solve this problem, and constructed General Filter that could adapt to the changes in the environment. In the proposed agent having General Filter with the interdependent evaluation, the parameters of General Filter were evaluated by Reinforcement Learning Module in order to realize the interdependency between the modules. From the result of the experiment with several alternatives for the structure of the agent, it was confirmed that the agent whose modules were interdependent, proposed in this paper, was the most flexible of all the tested agent for the changes in the environment. In the discussion, we considered that the filter was like a kind of characteristic or emotional parameter of the agent.

We point out the future works as follows. First, we must formulate the mathematical properties of the interdependent evaluation used in this paper. We might be able to use knowledge from the mathematical analyses done in system theory and game theory. Second, we must realize the interdependency without sign-inversion that is a kind of heuristics. In other words, the learning module in the agent must automatically obtain how to evaluate the heuristic module.

References

1. Garrett Hardin. The Tragedy of the Commons. *Science*, 162:1243–1248, 1968.
2. Toru Ishida and Kazuhiro Kuwabara. Distributed Artificial Intelligence (1): Cooperative Problem Solving. *Journal of Japanese Society for Artificial Intelligence*, 7(6):945–954, 1992. (in Japanese).
3. Leslie P. Kaelbling, Michael L. Littman, and Andrew W. Moore. Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.
4. Sadayoshi Mikami and Yukinori Kakazu. Co-operation of Multiple Agents Through Filtering Payoff. In *1st European Workshop for Reinforcement Learning*, 1994.
5. Sadayoshi Mikami, Yukinori Kakazu, and Terence C. Fogarty. Co-operative Reinforcement Learning By Payoff Filters. In *Proc. 8th European Conference on Machine Learning, ECML-95*, (Lecture Notes in Artificial Intelligence 912), pages 319–322, Heraclion, Crete, Greece, 1995.
6. Stuart J. Russell and Peter Norwig. *Artificial Intelligence: A Modern Approach*. Englewood Cliffs, NJ: Prentice-Hall, 1995.
7. Mitsuo Suzuki. *Shin Gehmu Riron* (New Game Theory). Tokyo: Keiso Shobo, 1994. (in Japanese).
8. Christopher J. C. H. Watkins and Peter Dayan. Technical Note: Q-learning. *Machine Learning*, 8:279–292, 1992.