Counter Example for Q-bucket-brigade under Prediction Problem [*]

Atsushi Wada
ATR NIS
2-2-2 Hikaridai, Seika-cho
Soraku-gun, Kyoto, Japan
wada@atr.jp

Keiki Takadama
Tokyo Institute of Technology
4259 Nagatsuta-cho
Midori-ku, Kanagawa, Japan
keiki@dis.titech.ac.jp

Katsunori Shimohara
ATR NIS
2-2-2 Hikaridai, Seika-cho
Soraku-gun, Kyoto, Japan
katsu@atr.jp

ABSTRACT

Aiming to clarify the convergence or divergence conditions for Learning Classifier System (LCS), this paper explores: (1) an extreme condition where the reinforcement process of LCS diverges; and (2) methods to avoid such divergence. Based on our previous work that showed equivalence between LCS's reinforcement process and Reinforcement Learning (RL) with Function approximation (FA) method, we present a counter-example for LCS with Q-bucket-brigade based on the 11-state star problem, a counter-example originally proposed to show the divergence of Q-learning with linear FA. Furthermore, the empirical results applying the counter-example to LCS verified the results predicted from the theory: (1) LCS with Q-bucket-brigade diverged under the prediction problem, where the action selection policy was fixed; and (2) such divergence was avoided by using implicit-bucket-brigade or applying residual gradient algorithm to Q-bucket-brigade.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning–Parameter learning.

General Terms

Algorithms, Design, Theory.

Keywords

Learning classifier systems, genetic-based machine learning, convergence, reinforcement learning, function approximation.

[*]This research was conducted as part of 'Research on Human Communication' with funding from the National Institute of Information and Communications Technology, and supported in part by a Grant-in-Aid for Scientific Research (B) (No. 17360424) of Ministry of Education, Culture, Sports, Science and Technology (MEXT).

1. INTRODUCTION

Learning Classifier Systems (LCSs) are rule-based adaptive systems intended for a general framework to realize intelligent behavior by combining two biologically inspired adaptive mechanisms – *learning* and *evolution* – with each essentially connecting to the fields of Reinforcement Learning (RL) and Evolutionary Computation (EC).

Since LCSs were mainly developed in the field of EC, most of the theoretical works focused on the analysis of Genetic Algorithms (GAs) for LCSs' rule discovery process[3]. However, when focusing on the RL side, few works[6] have contributed to connect LCSs' reinforcement process to the firm mathematical basis of RL, which is necessary for delivering the findings of recent development in the RL fields to the LCS field, especially the convergence proofs of learning.

Toward our goal of building the foundations of LCS seamlessly connected to the basis of RL, this paper addresses the issue of the convergence proof for LCS's reinforcement process. In our previous work[12, 13], we revealed that the reinforcement process of ZCS[16] with Q-bucket-brigade is equivalent to Q-learning with the Function Approximation (FA) method[9] within the class of linear approximation. Also, a disappointing issue was referred that currently there exists no convergence proof for Q-learning with linear FA but a counter example exists for such category. This problem motivated us to propose ZCS with the *residual gradient algorithm*[1], an RL technique that can introduce convergence to Q-learning with linear FA[14].

In this paper, we proceed to further steps exploring: (1) an extreme condition where the reinforcement process of LCS diverges; and (2) methods to avoid such divergence. For this objective, we present a counter-example for LCS with Q-bucket-brigade based on the 11-state star problem, which was originally proposed as the counter-example for Q-learning with linear FA, and we present empirical results by applying it to Reinforcement learning-based XCS (RXCS), an LCS based on XCS[17] but modified to be consistent with Q-learning with FA[13].

The rest of this paper is organized as follows. Section 2 introduces the current state of the convergence proofs for RL methods, then Section 3 introduces RXCS. In Section 4, we proposes the 11-state star problem, the counter-example for LCS, and Section 5 gives the experimental results of applying the 11-state star problem to RXCS. Finally, Section 6 includes our discussions and conclusion.

2. STATE OF CONVERGENCE PROOFS FOR REINFORCEMENT LEARNING

In this section, we first explain the properties of RL methods identifying the types of RL methods, which affects the availability of the convergence proofs. Then, based on the difference between these properties, the current state of the convergence proofs for RL methods is introduced.

2.1 Properties of reinforcement learning

Prediction and control problems. RL has two aspects regarding its learning: (1) the *policy evaluation*, which estimates the action values, which are often *Q-values*, for an arbitrary policy π^1; and (2) the *policy improvement*, which improves the current policy π to a better policy π' referring to the current action values. In the prediction problem, the policy is fixed through the learning and the action values for that policy is estimated. In the control problem, the policy evaluation and the policy improvement is performed at the same time.

On-policy and off-policy methods. An *on-policy* method, such as Sarsa, estimates the action values of the policy that controls the actual action selection. On the other hand, an *off-policy* method, such as Q-learning, estimates the policy that is different to the policy controlling the actual action selection.

Classes of the approximated action value function. Several representations for designing the generalized action value function have been proposed for RL methods, such as state-aggregation, tile-coding, and highly sophisticated representations such as RBF networks and Neural Networks are also applicable. These representations can be categorized into the following classes by their mathematical properties: (I) tabular; (II) state-aggregation; (III) linear approximation; and, (IV) non-linear approximation.

2.2 Convergence proofs

Referring to literature on RL, the current state of the convergence proofs for RL methods under the prediction and control problems can be described as follows. In the cases of Classes (I) and (II), convergence proofs are available under the prediction and control problems regardless of on-policy or off-policy methods[5]. In class (III), an on-policy methods are proved to converge under the prediction problem and osciallates near optimal under control problem [11]. On the other hand, an off-policy methods are shown to diverege under the prediction problem[1].

2.3 Relation between LCSs' reinforcement process

We presented in our previous work that the reinforcement process of ZCS with Q-bucket-brigade is equivalent to Q-learning with the Function Approximation (FA) method within the class of linear approximation. Thus, using categories for off-policy methods with class (III), the linear approximation will tell us the availability of the convergence proofs for ZCS's reinforcement process. From Section 3.3, we can see that no convergence proof is available for that under both the prediction and the control problems. However, we can say that if we modify the off-policy method to an on-policy method within class (III), convergence proof is available under the prediction problem, and the proof for oscillating near the optimal is available under the control problem.

Another approach to avoid the risk of divergence for the categories of the off-policy method within FA class (III) is to apply the *residual gradient algorithm*[1] to the update equation of the off-policy methods. In [14], we applied the residual gradient algorithm to the Q-bucket-brigade of ZCS, which resulted in an LCS with the off-policy update but avoided the risk of divergence.

Note that these discussions are not specific to ZCS but are also applicable to any LCSs having RL with FA-equivalent reinforcement processes, including Reinforcement learning based XCS (RXCS), which is presented in the next section.

^{1}In RL literature, the *policy* π_{xa} is defined as a set of probability for all the possible combinations of taking a possible action a at a possible state x.

3. REINFORCEMENT LEARNING BASED XCS (RXCS)

Reinforcement learning-based XCS (RXCS) is an LCS originally proposed in [13] comparing XCS's reinforcement process to Q-learning with linear FA. In this work, RXCS is designed by modifying XCS to become equivalent with RL with linear FA with respect to its reinforcement process. Here we only describe the modifications from the original XCS. See [4] for the entire XCS algorithm.

3.1 Payoff definition

The payoff definition of RXCS is defined as:

$$P(a_i) \quad = \quad \frac{\sum_{cl_k \in [\mathrm{M}]|_{a_i}} p_k \times num_k}{\sum_{cl_k \in [\mathrm{M}]|_{a_i}} num_k}, \tag{1}$$

where the fitness-weighted average of the classifier predictions in the XCS's original payoff definition is modified to the numerosity-weighted average of the classifier predictions.

3.2 Update process

The update equation of the classifier prediction is defined as:

$$p_j \quad \leftarrow \quad p_j + \beta(P - P_{-1}) \frac{num_j}{\sum_{cl_k \in [\mathrm{A}_{-1}]} num_k}, \tag{2}$$

where P_{-1} is a numerosity-weighted prediction for the classifiers included in the previous action set $[\mathrm{A}_{-1}]$ defined as:

$$P_{-1} \quad = \quad \frac{\sum_{cl_k \in [\mathrm{A}_{-1}]} p_k \times num_k}{\sum_{cl_k \in [\mathrm{A}_{-1}]} num_k}. \tag{3}$$

3.2.1 Q-bucket-brigade and implicit-bucket-brigade

In the original XCS, the target value P for the update in Equation 2 is defined:

$$P \quad \leftarrow \quad r + \gamma \max_a P(a), \tag{4}$$

which is defined as the sum of the current reward and the discounted payoff value for the current greedy action $a^* = \arg\max_a P(a)$. This type of update, namely, *Q-bucket-brigade*, was originally introduced for ZCS in [16] with another alternative update named *implicit-bucket-brigade*, which can be described by modifying Equation 4 as:

$$P \quad \leftarrow \quad r + \gamma P(a), \tag{5}$$

where the max operator is removed from the update equation for the Q-bucket-brigade.

3.2.2 Residual-bucket-brigade

Following the same process where we applied the residual gradient algorithm to ZCS in [14], RXCS with a residual gradient algorithm can be obtained. This is simply realized by adding the update process for the classifiers in the greedy action set $[\mathrm{M}]|_{a^*}$ defined as:

$$p_j \quad \leftarrow \quad p_j - \gamma\beta(P - P_{-1})\frac{num_j}{\sum_{cl_k \in [\mathrm{A}_{-1}]} num_k}, \tag{6}$$

which works complementarily with the ordinary update process for the classifiers in the previous action set $[\mathrm{A}_{-1}]$ defined in Equation 2. Here, we name this update process *residual-bucket-brigade* for convenience.

3.3 Convergence proofs

As RXCS is designed to be equivalent with RL with linear FA, the discussion in Section 2.3 is also applicable to RXCS, which means that: (i) RXCS with the Q-bucket-brigade lies within the the category of the off-policy method, inside the FA class of (III) the linear approximation, which might carry the risk of divergence; and (ii) such risk can be avoided in RXCS with the implicit-bucket-brigade or the residual-bucket-brigade.

4. COUNTER-EXAMPLE FOR OFF-POLICY UPDATE

In this section, a counter-example for RXCS with Q-bucket-brigade is proposed, which is based on the 11-state star problem originally presented in [2].

4.1 The 11-state star problem

The 11-state star problem was originally proposed by Baird as a counter-example for Q-learning with linear FA, whose state transitions are described as Fig. 1. The circles in the figure denote the states $\{\texttt{X1}, \ldots, \texttt{X11}\}$. Every transition receives zero reward, and each state has two actions: action `A1` represented by a solid line, and action `A2` represented by a dotted line. In all states, action `A1` transitions to state `X11`. Action `A2` transitions to one of the randomly chosen states within `X1` through `X10` with probability 9/10, or otherwise transitions to the terminal state with probability 1/10. The discount factor γ is set to 0.9.

The designed approximated action value function is described in Table 1, where the value of each state is given by the single approximation parameter or the linear combination of two approximation parameters. The function-approximation system is simply a lookup table, except for one additional approximation parameter $\theta(0)$ that provides generalization. Note that the coefficient of the parameter $\theta(0)$ in the action value for $(\texttt{X11}, \texttt{A1})$ is twice as large as that of other action values for `A1`. This difference in the coefficient regarding the center state `X11` is known to cause a monotonic increase in the value of the approximation parameter, and thus derives the divergence of the learning.

Table 1: Action value function for the 11-state star problem.

State	Action values for `A1`	Action values for `A2`
`X1`	$\theta(0) + 2\theta(1)$	$\theta(12)$
`X2`	$\theta(0) + 2\theta(2)$	$\theta(13)$
`X3`	$\theta(0) + 2\theta(3)$	$\theta(14)$
`X4`	$\theta(0) + 2\theta(4)$	$\theta(15)$
`X5`	$\theta(0) + 2\theta(5)$	$\theta(16)$
`X6`	$\theta(0) + 2\theta(6)$	$\theta(17)$
`X7`	$\theta(0) + 2\theta(7)$	$\theta(18)$
`X8`	$\theta(0) + 2\theta(8)$	$\theta(19)$
`X9`	$\theta(0) + 2\theta(9)$	$\theta(20)$
`X10`	$\theta(0) + 2\theta(10)$	$\theta(21)$
`X11`	$2\theta(0) + \theta(11)$	$\theta(22)$

Table 3: RXCS's classifier population designed for the 11-state star problem.

State	Action values for `A1`	Action values for `A2`
`X1`	$(2p_0 + 4p_1)/6$	p_{12}
`X2`	$(2p_0 + 4p_2)/6$	p_{13}
`X3`	$(2p_0 + 4p_3)/6$	p_{14}
`X4`	$(2p_0 + 4p_4)/6$	p_{15}
`X5`	$(2p_0 + 4p_5)/6$	p_{16}
`X6`	$(2p_0 + 4p_6)/6$	p_{17}
`X7`	$(2p_0 + 4p_7)/6$	p_{18}
`X8`	$(2p_0 + 4p_8)/6$	p_{19}
`X9`	$(2p_0 + 4p_9)/6$	p_{20}
`X10`	$(2p_0 + 4p_{10})/6$	p_{21}
`X11`	$(2p_0 + p_{11})/3$	p_{22}

4.2 Representation for LCS

To apply the 11-star problem to LCS, the following steps are required: (1) represent the states and the actions using LCS's representation; and (2) represent the approximated action value function as a classifier population. Here, we adopt the ternary representation, the most common representation for LCS.

For the former step, a simple conversion rules are designed that converts: (a) the states $\{\texttt{X1}, \ldots, \texttt{X11}\}$ into corresponding 4-bit binary strings $\{\texttt{0000}, \ldots, \texttt{1011}\}$; and (b) the actions `A1` and `A2` into single bits `0` and `1`, respectively.

For the latter step, we propose a classifier population design that represents the approximated action value function for the 11-state star problem. Table 2 shows how the population is composed. For each state-action pair, there exists a corresponding classifier that identically matches that state-action pair. The classifier 0 is the only exception that matches with any states and has an action part of `0`, that is, the action `A1`. In the case of RXCS, this composition of the classifier population represents the approximated action value function described as Table 3. From the table, we can see that the essential property of the 11-state problem is successfully expressed, where the coefficient of p_0, the prediction of the classifier 0 in the action value for $(\texttt{X11}, \texttt{A1})$ is twice as large as that of other action values for `A1`. This is due to the payoff definition of RXCS, where the predictions of classifiers in the action set is weighted by the value of each classifier's numerosity and averaged.

5. SIMULATION EXPERIMENTS

In this section, we apply the 11-state star problem to RXCS with Q-bucket-brigade, implicit-bucket-brigade and residual-bucket-brigade. These three cases are tested for the

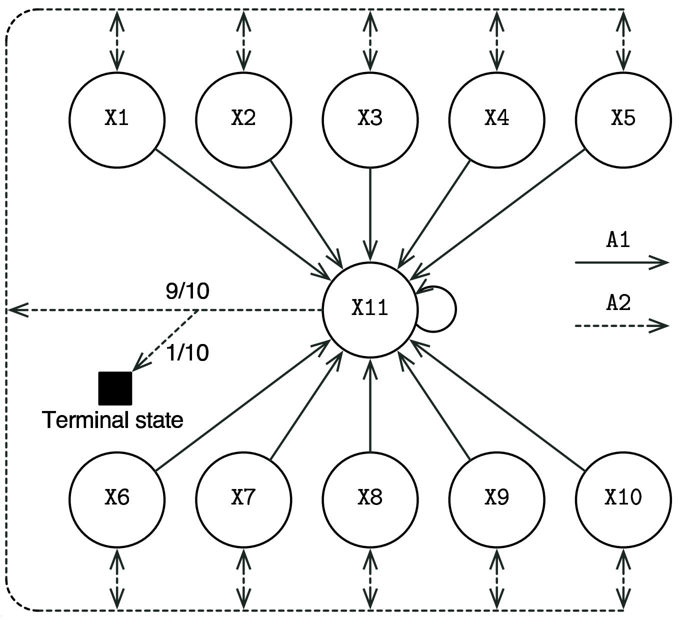


Figure 1: The state transition diagram for the 11-state star problem, which was originally proposed by Baird as a counter-example for Q-learning with linear FA.

Table 2: The design of classifier population for representing the 11-state star problem.

ID	Matching states	Action	Numerosity	$condition_j$: $action_j$
0	{X1, X2, ..., X11}	A1	2	#### : 0
1	{X1}	A1	4	0001 : 0
2	{X2}	A1	4	0010 : 0
3	{X3}	A1	4	0011 : 0
4	{X4}	A1	4	0100 : 0
5	{X5}	A1	4	0101 : 0
6	{X6}	A1	4	0110 : 0
7	{X7}	A1	4	0111 : 0
8	{X8}	A1	4	1000 : 0
9	{X9}	A1	4	1001 : 0
10	{X10}	A1	4	1010 : 0
11	{X11}	A1	1	1011 : 0
12	{X1}	A2	1	0001 : 1
13	{X2}	A2	1	0010 : 1
14	{X3}	A2	1	0011 : 1
15	{X4}	A2	1	0100 : 1
16	{X5}	A2	1	0101 : 1
17	{X6}	A2	1	0110 : 1
18	{X7}	A2	1	0111 : 1
19	{X8}	A2	1	1000 : 1
20	{X9}	A2	1	1001 : 1
21	{X10}	A2	1	1010 : 1
22	{X11}	A2	1	1011 : 1

prediction problem with the fixed and the decaying learning rate.

In both the cases of adopting the fixed and the decaying learning rate, the learning rate β is initially set to 0.01. In the case of adopting the decaying learning rate, β is decreased by the decaying coefficient of $1/n$, where n is initially set to 1 but incremented by 1 every 1,000 steps[2]

The common conditions used for the experiments are as follows. The action selection policy is fixed, where the probabilities of taking the actions `A1` and `A2` are fixed to 1/10 and 9/10, respectively. The discount factor γ is set to 0.9. RXCS is suppressed with its rule-discovery process of GA invocation, covering, and deletion. In all cases, ten simulations are performed with each, including a total of 100,000 episodes.

Here, the empirical results regarding the prediction problems are presented. Figures 2, 3 and 4 present the results for RXCS with the Q-bucket-brigade, the implicit-bucket-brigade and the residual-bucket-brigade, respectively. In all the figures, the graphs on the left-hand side, labeled (a), represent the cases for the fixed learning rate while the graphs on the right-hand side, labeled (b), represent the cases for the decaying learning rate. In all the graphs, the x-axis, y-axis and z-axis measure the number of episodes, the identification number of the classifier, and the value of the corresponding classifier prediction, respectively.

The Q-bucket-brigade with both the fixed and the decaying learning rate displayed a monotonic increase of the classifier predictions, as Figures 2 (a) and (b) indicate, while the implicit-bucket-brigade and the residual-bucket-brigade converged to the correct value of 0 in both the cases of the fixed and the decaying learning rate.

6. DISCUSSION AND CONCLUSION

So far, the convergence regarding the reinforcement process of LCS has been discussed from both the aspects of theory and practice. From the theoretical aspect, we referred to the convergence proofs of RL and clarified that: (1) RXCS with the Q-bucket-brigade is inside the the category of the off-policy method within the FA class of (III) linear approximation, which might carry the risk of divergence of the learning; and (2) such risk can be avoided in RXCS with the implicit-bucket-brigade or the residual-bucket-brigade. From the practical aspect, we presented the LCS version of the 11-state star problem, the counter-example for the off-policy RL methods with linear FA. Furthermore, the empirical results applying the counter-example to RXCS verified the results predicted from the theory: (1) RXCS with Q-bucket-brigade diverged under the prediction problem, where the action selection policy was fixed; and (2) such divergence was avoided by using implicit-bucket-brigade or applying residual gradient algorithm to Q-bucket-brigade.

Presenting such extreme conditions enabled us to verify the results from the theoretical aspect, and we believe that such a rigorous approach is effective and essential for understanding the nature of LCS.

For future research, the influence of rule discovery should be taken into the analysis, aiming at contributing to the formal understanding of Genetic Algorithms and Reinforcement Learning interacting within Learning Classifier Systems.

[2]This decaying condition satisfies the general convergence condition required for β_t, the learning rate at the total steps t within an episode as follows: $\sum_t \beta_t \to \infty$, $\sum_t \beta_t^2 \to 0$.

7. REFERENCES

[1] L. C. Baird. Residual algorithms: Reinforcement learning with function approximation. In *International Conference on Machine Learning*, pages 30–37, 1995.

[2] L. C. Baird. *Reinforcement Learning Through Gradient Descent*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA 15213, 1999.

[3] Butz, M., Kovacs, T., Lanzi, P.L., Wilson, S.W.: Toward a theory of generalization and learning in xcs. IEEE Transactions on Evolutionary Computation **8** (2004) 28–46

[4] M. V. Butz and S. W. Wilson. *Advances in Learning Classifier Systems*, volume LNAI 1996, chapter An Algorithmic Description of XCS, pages 253–272. Berlin: Springer-Verlag, 2001.

[5] G. J. Gordon. Stable function approximation in dynamic programming. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 261–268, 1995. Morgan Kaufmann.

[6] P. L. Lanzi. Learning classifier systems from a reinforcement learning perspective. *Soft Computing*, 6:162–170, 2002.

[7] S. P. Singh, T. Jaakkola, and M. I. Jordan. Reinforcement learning with soft state aggregation. In *Advances in Neural Information Processing Systems*, volume 7, pages 361–368. The MIT Press, 1995.

[8] R. Sutton and A. Barto. *An introduction to reinforcement learning*. MIT Press, 1998.

[9] R. S. Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In *Advances in Neural Information Processing Systems*, volume 8, pages 1038–1044. The MIT Press, 1996.

[10] J. N. Tsitsiklis and B. V. Roy. Feature-based methods for large scale dynamic programming. *Machine Learning*, 22(1-3):59–94, 1996.

[11] J. N. Tsitsiklis and B. V. Roy. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5):674–690, 1997.

[12] A. Wada, K. Takadama, K. Shimohara, and O. Katai. Comparison between Q-learning and ZCS Learning Classifier System: From aspect of function approximation. In *The 8th Conference on Intelligent Autonomous Systems,*, 422-429, 2004.

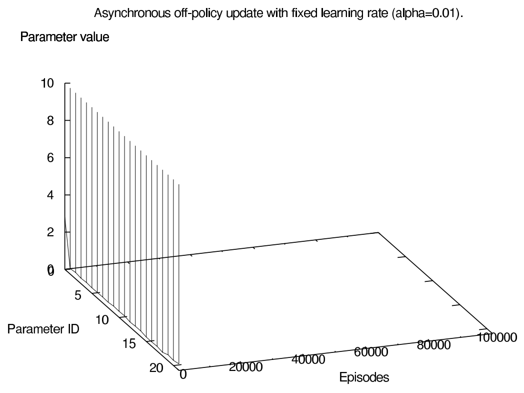
[13] A. Wada, K. Takadama, K. Shimohara, and O. Katai. Learning classifier system equivalent with reinforcement learning with function approximation. In *The Eighth International Workshop on Learning Classifier Systems,*, 2005. (accepted).

[14] A. Wada, K. Takadama, K. Shimohara, and O. Katai. *Foundations on Learning Classifier Systems*, chapter Learning Classifier Systems with Convergence and Generalization. Springer, in press.

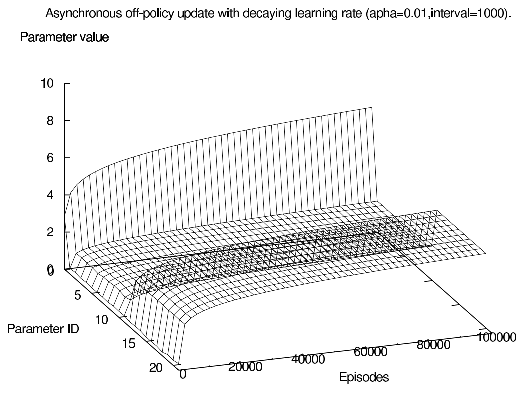
[15] J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, Cambridge University, 1989.

[16] S. W. Wilson. ZCS: A zeroth level classifier system. *Evolutionary Computation*, 2(1):1–18, 1994.

[17] S. W. Wilson. Classifier fitness based on accuracy. *Evolutionary Computation*, 3(2):149–175, 1995.

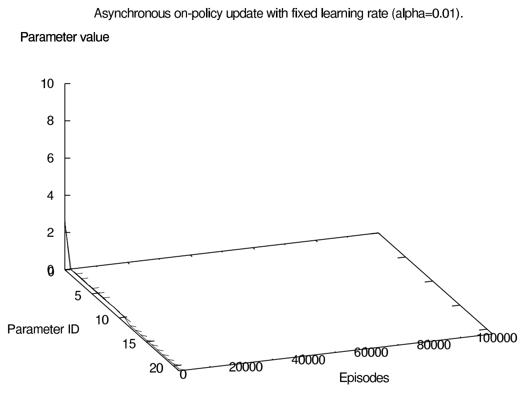


(a) Fixed learning rate.

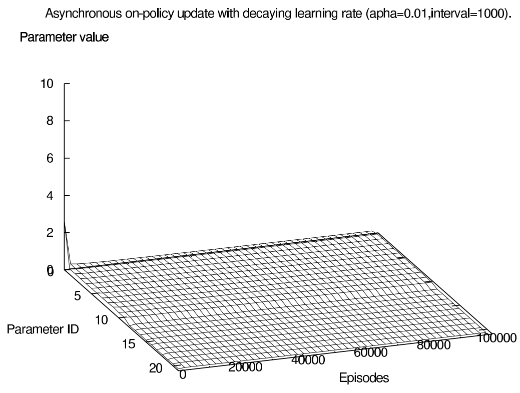


(b) Decaying learning rate.

Figure 2: The dynamics of the prediction value for each classifier in the RXCS classifier population, with the Q-bucket-brigade.

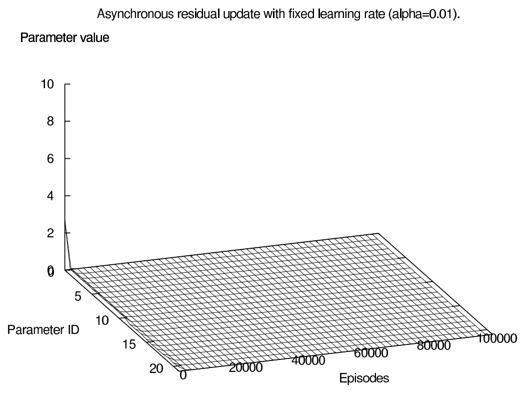


(a) Fixed learning rate.

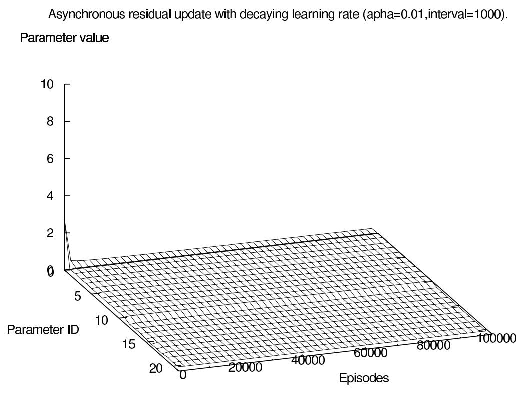


(b) Decaying learning rate.

Figure 3: The dynamics of the prediction value for each classifier in the RXCS classifier population, with the implicit-bucket-brigade.



(a) Fixed learning rate.



(b) Decaying learning rate.

Figure 4: The dynamics of the prediction value for each classifier in the RXCS classifier population, with the residual-bucket-brigade.