frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 23 September 2022
DOI 10.3389/fnbot.2022.999658

OPEN ACCESS

EDITED BY
Yusen He,
Grinnell College, United States

REVIEWED BY
Yang Wang,
East China Normal University, China
Chong Di,
Qilu University of Technology, China
Yujin Zhang,
Shanghai University of Engineering Sciences, China

*CORRESPONDENCE
Xudie Ren
renxudie@sjtu.edu.cn

RECEIVED 21 July 2022
ACCEPTED 22 August 2022
PUBLISHED 23 September 2022

CITATION
Ren X, Li S and Ge H (2022) A parameter-free learning automaton scheme.
Front. Neurorobot. 16:999658.
doi: 10.3389/fnbot.2022.999658

A parameter-free learning automaton scheme

Xudie Ren[1]*, Shenghong Li[1] and Hao Ge[2]

[1]School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China, [2]Shanghai Data Miracle Intelligent Technology Co., Ltd., Shanghai, China

For a learning automaton, a proper configuration of the learning parameters is crucial. To ensure stable and reliable performance in stochastic environments, manual parameter tuning is necessary for existing LA schemes, but the tuning procedure is time-consuming and interaction-costing. It is a fatal limitation for LA-based applications, especially for those environments where the interactions are expensive. In this paper, we propose a parameter-free learning automaton (PFLA) scheme to avoid parameter tuning by a Bayesian inference method. In contrast to existing schemes where the parameters must be carefully tuned according to the environment, PFLA works well with a set of consistent parameters in various environments. This intriguing property dramatically reduces the difficulty of applying a learning automaton to an unknown stochastic environment. A rigorous proof of ϵ-optimality for the proposed scheme is provided and numeric experiments are carried out on benchmark environments to verify its effectiveness. The results show that, without any parameter tuning cost, the proposed PFLA can achieve a competitive performance compared with other well-tuned schemes and outperform untuned schemes on the consistency of performance.

KEYWORDS

parameter-free, Monte-Carlo simulation, Bayesian inference, learning automaton, parameter tuning

1. Introduction

Learning Automata (LA) are simple self-adaptive decision units that were firstly investigated to mimic the learning behavior of natural organisms (Narendra and Thathachar, 1974). The pioneering work can be traced back to the 1960s by the Soviet scholar (Tsetlin, 1961, 1973). Since then, LA has been extensively explored and it is still under investigation as well in methodological aspects (Agache and Oommen, 2002; Papadimitriou et al., 2004; Zhang et al., 2013, 2014; Ge et al., 2015a; Jiang et al., 2015) as in concrete applications (Song et al., 2007; Horn and Oommen, 2010; Oommen and Hashem, 2010; Cuevas et al., 2013; Yazidi et al., 2013; Misra et al., 2014; Kumar et al., 2015; Vahidipour et al., 2015). One intriguing property that popularizes the learning automata-based approaches in engineering is that LA can learn the stochastic characteristics of the external environment it interacts with, and maximize the long-term reward it obtains through interacting with the environment. For a detailed overview of LA, one may refer to a new comprehensive survey (Oommen and Misra, 2009) and a classic book (Narendra and Thathachar, 2012).

In the case of LA, *accuracy* and *convergence rate* become two major measurements to evaluate the effectiveness of a LA scheme. The former is defined as the probability of a correct convergence and the latter as the average iterations for a LA to get converged[1]. Most of the reported schemes in the field of LA have two or more tunable parameters, making themselves capable of adapting to a particular environment. An automaton's accuracy and convergence rate highly depend on the selection of those parameters. Generally, ensuring a high accuracy is of uppermost priority. According to the ϵ-*optimality* property of LA, the probability of converging to the optimal action can be arbitrarily close to one, as long as the learning resolution is large enough. However, it will raise another problem. Taking the classic Pursuit scheme for example, as Figure 1 illustrates, the number of iterations required for convergence grows nearly linearly with the resolution parameter, while the accuracy grows logarithmically. This implies a larger learning resolution can lead to higher accuracy, but at the cost of much more interactions with the environment. This dilemma necessitates parameter tuning to find a balance between convergence rate and accuracy.

In literature, the performance of various LA schemes is evaluated by comparing their convergence rates on the premise of a certain accuracy. The learning parameters of various schemes are tuned through a standard procedure to ensure the accuracies are kept at the same level, so that the convergence rates can be fairly compared. For deterministic estimator-based learning automata, the smallest value of the resolution parameter that yielded a hundred percent accuracy in a certain number of experiments is selected. The situation is more sophisticated when concerning the stochastic estimator-based schemes (Papadimitriou et al., 2004; Ge et al., 2015a; Jiang et al., 2015), because extra configurable parameters should be set to control the perturbation added. Parameter tuning is intended to balance the trade-off between speed and accuracy. However, the interaction cost can be tremendous itself[3], due to its trial and error nature. In practical applications, especially where interacting with environments could be expensive, e.g., drug trials, destructive tests, and financial investments, the enormous cost for parameter tuning is undesired. Therefore, we believe, the issue of learning parameter configurations deserves more attention in the community, which gives impetus to our work.

The scope of this research is confined to designing a learning scheme for LA in which the parameter tuning can be omitted,

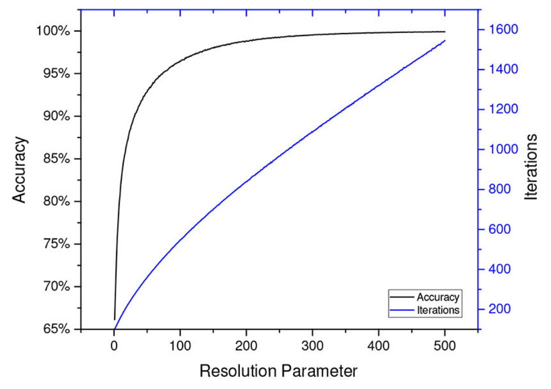

FIGURE 1
The accuracy and iterations with different resolution parameters for DP$_{ri}$ (Oommen and Lanctôt, 1990) in benchmark environment E_1, which is defined in Papadimitriou et al. (2004). The results are averaged over 250,000 replications.[2]

and that's why it is called *parameter-free* in the title. It is noted that the term *parameter-free* does not imply that no configurable parameters are involved in the proposed model, but indicates a set of parameters for the scheme that can be universally applicable to all environments. This paper is an extension of our preliminary work (Ge et al., 2015b). The proposed scheme in Ge et al. (2015b) can only operate in two-action environments, whereas in this paper, our proposed scheme can operate in both two-action environments as well as multi-action environments. In addition, in this paper, optimistic initial values are utilized to improve the performance further. Moreover, a rigorous theoretical analysis of the proposed scheme and a comprehensive comparison among recently proposed LA schemes are provided in this paper which was not included in Ge et al. (2015b).

The contribution of this paper can be summarized as follows:

1. To the best of our knowledge, we present the first *parameter-free* scheme in the field of LA, for learning in any stationary P-model stochastic environment. The meaning of the terminology *parameter-free* is two-fold: (1) The learning parameters do not need to be manually configured. (2) Unlike other estimator-based schemes, initializations of estimators are also unnecessary in our scheme.
2. Most conventional LA schemes in literature employ a stochastic exploration strategy, on the contrary, we design a deterministic gradient descent-like method instead of probability matching as the exploration strategy to further accelerate the convergence rate of the automaton.
3. The statistics behavior of the proposed parameter-free learning automata (PFLA) is analyzed and rigorous proof of the ϵ-optimality property is provided as well.

1 For this reason, the terms *convergence rate* and *iteration* are used interchangeably.

2 E_1 defined in Papadimitriou et al. (2004) corresponds to E_5 defined in Section 5 of this paper.

3 The details will be elaborated in Section 5.

4. Comprehensive comparison among recently proposed LA schemes is given to validate the theoretical analyses and demonstrate that PFLA is superior to other methods concerning tuning cost.

This paper proceeds as follows. Section 2 describes our philosophy and some related works. Section 3 presents the primary results of the paper: a parameter-free learning automaton scheme. Section 4 discusses the theoretical performance of the proposed scheme. Section 5 provides a numerical simulation for verifying the proposed scheme. Finally, Section 6 concludes this paper.

2. Related works

Consider a P-model environment which could be mathematically defined by a triple $< \mathbb{A}, \mathbb{B}, \mathbb{C} >$, where

- $\mathbb{A} = \{a_1, a_2, \ldots, a_r\}$ represents a finite action set
- $\mathbb{B} = \{0, 1\}$ denotes a binary response set
- $\mathbb{C} = \{c_1, c_2, \ldots, c_r\}$ is a set of reward probabilities corresponding to $\mathbb{A}$, which means $\Pr\{a_i \text{ gets rewarded}\}=c_i$. Each c_i is assumed to lie in the open interval $(0, 1)$.

Some other major notations that are used throughout this paper are defined in Table 1.

The aim of LA is to identify the optimal action a_m, which has the maximum reward probability, from $\mathbb{A}$ through interacting with the environment. The general philosophy is to collect feedback from the environment and use this information to extract evidence that supports an optimal assertion.

Then we are faced with two challenges:

TABLE 1 Notations used in this paper.

Symbol	Explanation
r	The cardinality of the action set $\mathbb{A}$
E	A vector of estimates
N	The number of repetitions in the Monte Carlo simulation
η	The threshold to terminate the iteration
a_i	The ith action in $\mathbb{A}$
α_i	A parameter of a_i's beta distribution
β_i	A parameter of a_i's beta distribution
S_i	The number of times that a_i has been selected
$\mathcal{H}_i$	The hypothesis that a_i is the optimal action
$Beta(\alpha, \beta)$	A beta distribution with parameter α and β
$Norm(\mu, \sigma)$	A normal distribution with mean μ and variance σ^2
$B(\alpha, \beta)$	The beta function
$B(x; \alpha, \beta)$	The incomplete beta function

1. How to organize the information gathered and make full use of them?
2. When is the time to make an assertion that claims one of the actions is optimal?

2.1. Information utilization

Lots of work have been done for the first challenge. Although the reward probabilities $\mathbb{C}$ are unknown to us, we can construct consistent estimators to guarantee that the estimates of the reward probabilities can converge to their true values as the quantity of samples increases.

As the feedback for one action can be modeled as a Bernoulli distributed random variable in P-model environments, there are two ways to construct such estimators currently.

1. One is from the frequentist's perspective. The most intuitive approach is to utilize the likelihood function, which is a basic quantitative measure over a set of predictions with respect to observed data. In the context of parameter estimation, the likelihood function is naturally viewed as a function of the parameters c_i to be estimated. The parameter that maximizes the likelihood of the observed data is referred to as the *maximum likelihood estimate* (MLE). MLE-based LA (Oommen and Lanctôt, 1990; Agache and Oommen, 2002) are proved to be a great success, achieving a tremendous improvement in the rate of convergence compared with traditional variable structure stochastic automata. However, as we revealed in Ge et al. (2015a), MLE suffers from one principle weakness, i.e., MLE is unreliable when the quantity of samples is small.

   Several efforts have been devoted to improving MLE. The concept of *stochastic estimator* was employed in Papadimitriou et al. (2004) so that the influence of lacking samples can be reduced by introducing controlled randomnesses to MLE. In Ge et al. (2015a), we proposed an interval estimator-based learning automata DGCPA, in which the upper bound of a 99% confidence interval of c_i is used as estimates of reward probabilities. Both of these two LA schemes broke the records of convergence rate when proposed, which confirmed the defect of traditional MLE.
2. On the other hand, there are attempts from the Bayesian perspective. Historically, one of the major reasons for avoiding Bayesian inference is that it can be computationally intensive under many circumstances. The rapid improvements in available computing power over the past few decades can, however, help overcome this obstacle, and Bayesian techniques are becoming more widespread not only in practical statistical applications but also in theoretical approaches to modeling human cognition. In Bayesian statistics, parameter estimation involves placing a probability distribution over model parameters. Concerning LA, the

posterior distribution of c_i with respect to observed data is a beta distribution.

In Zhang et al. (2013), DBPA was proposed where the posterior distribution of estimated $\hat{c}_i$ is represented by a beta distribution $Beta(\alpha, \beta)$, the parameter α and β record the number of times that a specific action has been rewarded and penalized, respectively. Then the 95th percentile of the cumulative posterior distribution is utilized as an estimation of c_i.

One of the main drawbacks of the way that information is being used by existing LA schemes is that they summarize beliefs about c_i, such as the likelihood function or the posterior distribution, into a point estimate, which obviously may lead to information loss. In the proposed PFLA, we insist on taking advantage of the entire Bayesian posterior distribution of c_i for further statistical inference.

2.2. Optimal assertion

For the second challenge, as the collected information accumulates, we become more and more confident to make an assertion. But when is the exact timing?

The quantity of samples before the convergence of existing strategies is indirectly controlled by its learning parameters. Actually, the LA is not aware of whether it has collected enough information or not, as a consequence, its performance completely relies on the manual configuration of learning parameters inevitably. As far as we're concerned, there is no report describing a parameter-free scheme for learning in multi-action environments, and this research area remains quite open.

However, there are efforts from other research areas that shed some light on this target. In Granmo (2010), a Bayesian learning automaton (BLA) was proposed for solving the two-armed Bernoulli bandit (TABB) problem. The TABB problem is a classic optimization problem that explores the trade-off between exploitation and exploration in reinforcement learning. One distinct difference between learning automata and bandit-playing algorithms is the metrics used for performance evaluation. Typically, *accuracy* is used for evaluating LA algorithms while *regret* is usually used in bandit playing algorithms. Despite being presented with different objectives, BLA is somewhat related to our study and inspired our work. Therefore, the philosophy of BLA is briefly summarized as follows: The BLA maintains two beta distributions as estimates of the reward probabilities for the two arms (corresponding to actions in the LA field). At each time instance, two values are randomly drawn from the two beta distributions, respectively. The arm with the higher random value is selected, and the feedback is utilized to update the parameter of the beta distribution associated with the selected arm. One advantage of BLA is that it doesn't involve any explicit computation of Bayesian expression. In Granmo (2010), it has been claimed that BLA performs better than UCB-tuned, the best performing algorithm reported in Auer et al. (2002).

Inspired by Granmo (2010), we constructed the PFLA by using Bayesian inference to enable convergence self-judgment in this paper. In contrast to Granmo (2010), however, the probability of each arm being selected must be explicitly computed to judge the convergence of the algorithm. In addition, due to the poor performance of probability matching, we developed a deterministic exploration strategy. The technical details are provided in the next section.

3. A parameter-free learning automaton

In this section, we introduce each essential mechanism of our scheme in detail.

3.1. Self-judgment

Consider a P-model environment with r available actions, as we have no prior knowledge about these actions, each of them is possible to be the optimal one. We refer to these r possibilities as r hypotheses $\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_r$ so that each hypothesis $\mathcal{H}_i$ represents the event that action a_i is the optimal action.

As we discussed in Section 2, the Bayesian estimates of each action's reward probability just intuitively are beta distributed random variables, denoted as $E = \{e_1, e_2, \ldots, e_r\}$, where $e_i \sim Beta(\alpha_i, \beta_i)$.

Because the propositions $\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_r$ are mutually exclusive and collectively exhaustive, apparently we have $\sum_i Pr(\mathcal{H}_i) = 1$. Therefore, we can simply assert that α_i is the optimal action once $Pr(\mathcal{H}_i)$ is greater than some predefined threshold η. For this reason, the explicit computation of $Pr(\mathcal{H}_i)$ is necessary here to make that assertion.

3.1.1. Two-action environments

In the two-action case, $Pr(\mathcal{H}_1)$ can be formulated in the following equivalent forms:

$$\begin{aligned} Pr(\mathcal{H}_1) &= Pr(e_1 > e_2) && (1) \\ &= \sum_{i=0}^{\alpha_1 - 1} \frac{B(\alpha_2 + i, \beta_1 + \beta_2)}{(\beta_1 + i) B(1 + i, \beta_1) B(\alpha_2, \beta_2)} && (2) \\ &= \sum_{i=0}^{\beta_2 - 1} \frac{B(\beta_1 + i, \alpha_1 + \alpha_2)}{(\alpha_2 + i) B(1 + i, \alpha_2) B(\alpha_1, \beta_1)} && (3) \\ &= 1 - Pr(\mathcal{H}_2) && (4) \end{aligned}$$

$$= 1 - \sum_{i=0}^{\alpha_2-1} \frac{B(\alpha_1 + i, \beta_1 + \beta_2)}{(\beta_2 + i)B(1 + i, \beta_2)B(\alpha_1, \beta_1)} \tag{5}$$

$$= 1 - \sum_{i=0}^{\beta_1-1} \frac{B(\beta_2 + i, \alpha_1 + \alpha_2)}{(\alpha_1 + i)B(1 + i, \alpha_1)B(\alpha_2, \beta_2)} \tag{6}$$

The above formulas can be easily implemented by a programming language with a well-defined log-beta function, thus the exact calculation of $Pr(\mathcal{H}_1)$ can be completed within $\mathcal{O}(\min(\alpha_1, \alpha_2, \beta_1, \beta_2))$. However, in multi-action cases, the closed-form of $Pr(\mathcal{H}_i)$ is too complex and it's somewhat computationally intensive to calculate it directly. So in our scheme, a Monte Carlo simulation is adopted for evaluating $Pr(\mathcal{H}_i)$ in a multi-action environment.

3.1.2. Multi-action environments

The closed-form calculation of $Pr(\mathcal{H}_i)$ is feasible for a small action set, but it becomes much more difficult as the number of actions increases.

Monte Carlo methods are a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results.

In multi-action environments, in order to evaluate $Pr(\mathcal{H}_i)$, an intuitive approach is to generate random samples from the r beta distributions and count how often the sample from $Beta(\alpha_i, \beta_i)$ is bigger than any other samples. In that way, the following Monte-Carlo simulation procedure is proposed.

Suppose the number of simulation replications is N. Since e_i follows $Beta(\alpha_i, \beta_i)$, let x_i^n be one of the r random samples at the nth replication.

Then, $Pr(\mathcal{H}_i)$ can be simulated as

$$\widehat{Pr}(\mathcal{H}_i) = \frac{1}{N}\sum_{n=1}^{N} I(x_i^n) \tag{7}$$

where $I(x_i^n)$ is an indicator function such that

$$I(x_i^n) = \begin{cases} 1 & \text{if } x_i^n > x_j^n, \forall j \neq i \quad \text{(8a)} \\ 0 & \text{otherwise} \quad \text{(8b)} \end{cases}$$

It is simple to verify that $\sum_i Pr(\mathcal{H}_i) = 1$.

3.2. Exploration strategy

In conventional estimator-based learning schemes, which are the majority family of LA, a stochastic exploration strategy is employed. A probability vector for choosing each action is maintained in the automaton and is properly updated under the guidance of the estimator and environment feedback after every interaction. However, such a probability vector does not exist in our scheme. Instead, a vector of probabilities indicating the chance of each action being the best one is maintained in our scheme. The exploration strategy in Granmo (2010) is the so-called *probability matching*, which occurs when an action is chosen with a frequency equivalent to the probability of that action being the best choice. In Ge et al. (2015b), we constructed a learning automata by adding an absorbing barrier to BLA and applying it as a baseline for comparison. The numerical simulation shows the low performance of the probability matching strategy in designing parameter-free LA. Therefore, a novel deterministic exploration strategy is proposed accordingly to overcome this pitfall.

Because $\max\{Pr(\mathcal{H}_i)\} > \eta$ is the stop criterion of our scheme, in order to pursue a rapid convergence, one straightforward and obvious approach is maximizing the expected increment of $\max\{Pr(\mathcal{H}_i)\}$ over the action set.

3.2.1. Two-action environments

In two-action environments, if $Pr(\mathcal{H}_1)$ is greater than $Pr(\mathcal{H}_2)$, then we suppose action a_1 is more likely to be the optimal one, and thus attempt to find out the action that will lead to the maximal expected increment of $Pr(\mathcal{H}_1)$, or vice versa.

We denote $Pr(\mathcal{H}_1)$ as $g(\alpha_1, \beta_1, \alpha_2, \beta_2)$, and the following recurrence relations are derived (Cook, 2005):

$$g(\alpha_1 + 1, \beta_1, \alpha_2, \beta_2) = g(\alpha_1, \beta_1, \alpha_2, \beta_2) + h(\alpha_1, \beta_1, \alpha_2, \beta_2)/\alpha_1 \tag{9}$$

$$g(\alpha_1, \beta_1 + 1, \alpha_2, \beta_2) = g(\alpha_1, \beta_1, \alpha_2, \beta_2) - h(\alpha_1, \beta_1, \alpha_2, \beta_2)/\beta_1 \tag{10}$$

$$g(\alpha_1, \beta_1, \alpha_2 + 1, \beta_2) = g(\alpha_1, \beta_1, \alpha_2, \beta_2) - h(\alpha_1, \beta_1, \alpha_2, \beta_2)/\alpha_2 \tag{11}$$

$$g(\alpha_1, \beta_1, \alpha_2, \beta_2 + 1) = g(\alpha_1, \beta_1, \alpha_2, \beta_2) + h(\alpha_1, \beta_1, \alpha_2, \beta_2)/\beta_2 \tag{12}$$

where $h(\alpha_1, \beta_1, \alpha_2, \beta_2) = \frac{B(\alpha_1+\alpha_2, \beta_1+\beta_2)}{B(\alpha_1,\beta_1)B(\alpha_2,\beta_2)}$.

Hence, given that action a_1 is chosen, the conditional expected increment of $Pr(\mathcal{H}_1)$ is:

$$\mathbb{E}[\Delta Pr(\mathcal{H}_1) \mid a_1 \text{ is chosen}] \tag{13}$$

$$= c_1 \times h(\alpha_1, \beta_1, \alpha_2, \beta_2)/\alpha_1 - (1 - c_1) \times h(\alpha_1, \beta_1, \alpha_2, \beta_2)/\beta_1 \tag{14}$$

$$= h(\alpha_1, \beta_1, \alpha_2, \beta_2)(c_1/\alpha_1 - (1 - c_1)/\beta_1) \tag{15}$$

because c_1 is unknown to us, we can approximate the above equation as

$$\mathbb{E}[\Delta Pr(\mathcal{H}_1) \mid a_1 \text{ is chosen}] \tag{16}$$

$$\approx h(\alpha_1, \beta_1, \alpha_2, \beta_2)(\frac{\alpha_1}{\alpha_1 + \beta_1}/\alpha_1 - \frac{\beta_1}{\alpha_1 + \beta_1}/\beta_1) \tag{17}$$

$$= 0 \tag{18}$$

In the same way, we have

$$\mathbb{E}[\Delta Pr(\mathcal{H}_1) \mid a_2 \text{ is chosen}] \approx 0 \tag{19}$$

(18) and (19) indicate that no matter which action is picked, the expected difference of $\max\{Pr(\mathcal{H}_i)\}$ will approximately be zero, which makes it difficult for us to make decisions.

Our solution is to select the action that gives the expected maximum possible increment to $\max\{Pr(\mathcal{H}_i)\}$, as we did in Ge et al. (2015b). More specifically, if $Pr(\mathcal{H}_1)$ is greater than $Pr(\mathcal{H}_2)$, then we try to find out the action that could probably lead to the expected maximal increment of $Pr(\mathcal{H}_1)$, that is

$$\underset{i}{\operatorname{argmax}}\, \mathbb{E}[\max\{\Delta Pr(\mathcal{H}_1)\} \mid a_i \text{ is chosen}] \tag{20}$$

Otherwise, we try to maximize

$$\underset{i}{\operatorname{argmax}}\, \mathbb{E}[\max\{\Delta Pr(\mathcal{H}_2)\} \mid a_i \text{ is chosen}] \tag{21}$$

The events that can lead to increments of $Pr(\mathcal{H}_1)$ are "*action a_1 is selected and rewarded*" and "*action a_2 is selected and punished*." Hence the optimization objective of (20) can be simplified as:

$$\begin{cases} \frac{c_1}{\alpha_1} h(\alpha_1, \beta_1, \alpha_2, \beta_2) & a_1 \text{ is chosen} \quad (22a) \\ \frac{1-c_2}{\beta_2} h(\alpha_1, \beta_1, \alpha_2, \beta_2) & a_2 \text{ is chosen} \quad (22b) \end{cases}$$

By employing the Maximum Likelihood Estimate of c_1 and c_2, (22) can be written as

$$\begin{cases} \frac{h(\alpha_1, \beta_1, \alpha_2, \beta_2)}{(\alpha_1 + \beta_1)} & a_1 \text{ is chosen} \quad (23a) \\ \frac{h(\alpha_1, \beta_1, \alpha_2, \beta_2)}{(\alpha_2 + \beta_2)} & a_2 \text{ is chosen} \quad (23b) \end{cases}$$

The same conclusion holds also for situation $Pr(\mathcal{H}_1) < Pr(\mathcal{H}_2)$.

As a result, the strategy adopted in two-action environments is selecting the action which has been observed less between the two candidate actions at every time instance, as (24) reveals.

$$\begin{cases} \underset{i}{\operatorname{argmin}}(\alpha_i + \beta_i) & \text{when } S_1 \neq S_2 \quad (24a) \\ \text{randomly chosen} & \text{when } S_1 = S_2 \quad (24b) \end{cases}$$

3.2.2. Multi-action environments

In multi-action environments, the automaton has to distinguish the best action from the action set. Intuitively, we can maximize the expected increment of $Pr(\mathcal{H}_i)$ over the selection of actions, however, the closed form of $Pr(\mathcal{H}_i)$ is complicated, making the exact solution computationally intractable.

However, from an alternative perspective, the automaton only needs to determine which is the best of the top two possibly optimal actions. That is, for the two actions which are most possible to be the optimal action, denoted as action a_{i1} and action a_{i2}, we only have to maximize the probability $Pr(e_{i1} > e_{i2})$ or $Pr(e_{i2} > e_{i1})$, exactly the same as it in two-action environments. So we come to the conclusion that, in the proposed scheme, our exploration strategy is similar to (24).

3.3. Initialization of beta distributions

In our scheme, each estimation e_i is represented by a beta distribution $e_i \sim Beta(\alpha_i, \beta_i)$. The parameters α_i and β_i record the number of times that action a_i has been rewarded and punished, respectively.

In the beginning, as we know nothing about the actions, a non-informative (uniform) prior distribution is advised to infer the posterior distribution. So α_i and β_i should be set identically to 1, exactly the same as in Granmo (2010) and Zhang et al. (2013).

However, as clarified in Sutton and Barto (2018), initial action values can be used as a simple way of encouraging exploration. The technique of *optimistic initial values* is applied, which has been reported as a quite effective simple trick on stationary problems.

Therefore, in our scheme, the prior distribution is $Beta(2, 1)$ for inferring the posterior distribution, i.e., all beta random variables are initialized as $\alpha_i = 2, \beta_i = 1$.

The estimates of all actions' reward probability are intentionally biased toward 1. The impact of the bias is permanent, though decreasing over iterations. When an action has been sampled just a few times, the bias contributes a large proportion to the estimate, thus further exploration is encouraged. By the time an action has been observed many times, the impact of the biased initial value is negligible.

Finally, the overall process of PFLA is summarized in Algorithm 1.

4. Performance analysis

In this section, the statistical performance of the proposed scheme is analyzed, an approximate lower bound of the accuracy is derived and the ϵ-optimality of the proposed scheme is further proved.

4.1. An approximate lower bound of the accuracy

As declared in Owen (2013), from the central limit theorem (CLT), we know that the error of Monte Carlo simulation has approximately a normal distribution with zero mean and variance σ^2/N. Hence, if we denote the error between $Pr(\mathcal{H}_i)$ and its Monte-Carlo estimate as ϵ_i, then we get

$$\begin{aligned} Pr(\mathcal{H}_i) &= \widehat{Pr}(\mathcal{H}_i) + \epsilon_i && (27) \\ &\geq \eta + \epsilon_i && (28) \\ &\sim \eta + Norm(0, \frac{\sigma_i^2}{N}) && (29) \\ &\geq \eta - \mid \epsilon_i \mid && (30) \end{aligned}$$

```
Require: $\eta$: a convergence threshold; $N$: the number
    of replications of Monte Carlo simulation
1: Initial $\alpha_i = 2, \beta_i = 1$ for $i = 1, 2, 3, \ldots, r$;
2: repeat
3:     Evaluate the probability $\widehat{Pr}(\mathcal{H}_i)$ according to
    (7) for each action $i = 1, 2, 3, \ldots, r$;
4:     Choose the two actions with top two $\widehat{Pr}(\mathcal{H}_i)$,
    denoted as $a_{i1}$ and $a_{i2}$. If there are two or more
    actions with identical maximum $\widehat{Pr}(\mathcal{H}_i)$, then
    choose two from them randomly.
5:     Select one from $a_{i1}$ and $a_{i2}$ according to
       $a_i = \begin{cases} a_{i1} & \text{if } S_{i1} < S_{i2} \\ a_{i2} & \text{if } S_{i1} > S_{i2} \\ \text{randomly chosen} & \text{if } S_{i1} = S_{i2} \end{cases}$
    and interacts with the environment.
6:     Receive feedback from the environment and
    update the parameters of beta distributions for
    action $a_i$:
       $\begin{cases} \alpha_i = \alpha_i + 1 & \text{if a reward is received} \\ \beta_i = \beta_i + 1 & \text{if a penalty is received} \end{cases}$
7: until $\max\{\widehat{Pr}(\mathcal{H}_i)\} > \eta$
```

Algorithm 1. Parameter-free learning automaton.

where $\widehat{Pr}(\mathcal{H}_i)$ is the Monte-Carlo estimate of $Pr(\mathcal{H}_i)$ and σ_i^2 is the variance of $I(x_i)$.

We may note that the right-hand side of (29) is irrelevant to the characteristics of the environment. In other words, the performance of the proposed scheme only depends on the selection of η and N. That is the theoretical foundation of the parameter-free property.

As the outcome of $I(x_i)$ is binary, in the worst case, the maximum of σ_i^2 is 0.25. When N equals 1,000, the probability density function of ϵ_i is shown in Figure 2, which quantitatively depicts the error. Obviously, the error is so small that could be ignored.

Therefore, the approximate lower bound of $Pr(\mathcal{H}_i)$ is η. According to the Bayesian theory, the accuracy of our scheme is approximately larger than η.

Next, we shall describe the behavior of the proposed scheme more precisely. Like the pioneers have done in previous literature, the ϵ-optimality of the proposed scheme will be derived.

4.2. Proof of ε-optimality

Recall that e_i is defined as the estimated reward probability of action a_i and follows $Beta(\alpha_i, \beta_i)$, which is the posterior

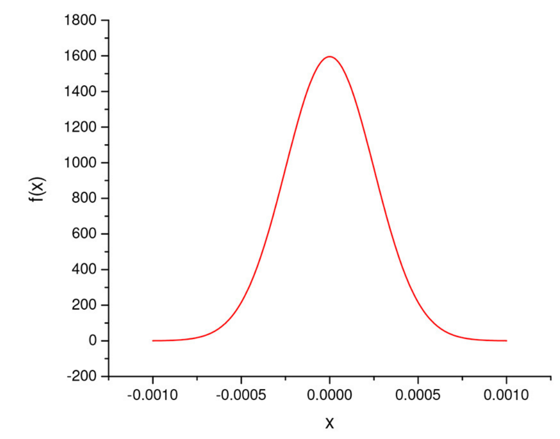

FIGURE 2
The probability density function of $Norm(0, \frac{1}{4{,}000})$.

distribution of the estimated reward probability. The probability density function of $Beta(\alpha_i, \beta_i)$ is $f(x_i; \alpha_i, \beta_i) = C_i x_i^{\alpha_i - 1}(1 - x_i)^{\beta_i - 1}$, where $C_i = \frac{1}{B(\alpha_i, \beta_i)}$ serves as a normalizing factor such that $\int_0^1 f(x_i; \alpha_i, \beta_i) = 1$. Let $Z_i = \alpha_i - 2$ and $W_i = \beta_i - 1$ denote the numbers of times that action a_i has been rewarded and penalized, respectively, and $S_i = Z_i + W_i = \alpha_i + \beta_i - 3$ be the number of times that action a_i has been selected.

Based on these preliminaries, the following Lemmas and Theorems are proposed:

Lemma 1. *The beta distribution $Beta(\alpha_i, \beta_i)$ becomes 1-point Degenerate distribution with a Dirac delta function spike at c_i, provided that the number of selecting action a_i approaches infinity, i.e., $\forall \varepsilon > 0$,*

$$\lim_{S_i \to \infty} \int_{|x_i - c_i| \le \varepsilon \bigcap [0,1]} f(x_i; \alpha_i, \beta_i) dx_i = 1 \tag{31}$$

$$\lim_{S_i \to \infty} \int_{|x_i - c_i| > \varepsilon \bigcap [0,1]} f(x_i; \alpha_i, \beta_i) dx_i = 0 \tag{32}$$

Proof 1. *According to the law of large numbers, we have $\frac{Z_i}{S_i} \to c_i$, as $S_i \to \infty$.*

Hence

$$\left.\begin{aligned} \lim_{S_i \to \infty} \frac{\alpha_i - 1}{S_i} &= \frac{Z_i + 1}{S_i} = c_i \\ \lim_{S_i \to \infty} \frac{\beta_i - 1}{S_i} &= \frac{S_i - Z_i}{S_i} = (1 - c_i) \end{aligned}\right\} \Rightarrow \begin{cases} \alpha_i - 1 = c_i S_i \\ \beta_i - 1 = (1 - c_i) S_i \end{cases} \tag{33}$$

The probability density function takes the form:

$$\lim_{S_i \to \infty} f(x_i; \alpha_i, \beta_i) = C_i x_i^{\alpha_i - 1}(1 - x_i)^{\beta_i - 1} \tag{34}$$

$$= C_i [x_i^{c_i}(1 - x_i)^{1 - c_i}]^{S_i} \tag{35}$$

$$= C_i g^{S_i}(x_i) \tag{36}$$

where $g(x_i) = x_i^{c_i}(1-x_i)^{1-c_i}$.

Note that $g(x_i)$ *is a non-negative integrable function, we have*

$$\lim_{S_i\to\infty}\left(\int_0^1 g^{S_i}(x_i)\right)^{\frac{1}{S_i}} dx_i = ||g||_\infty. \tag{37}$$

Therefore,

$$\lim_{S_i\to\infty} C_i^{\frac{1}{S_i}} = \frac{1}{\left(\int_0^1 g^{S_i}(x_i)dx_i\right)^{\frac{1}{S_i}}} = \frac{1}{||g||_\infty} \tag{38}$$

This reveals, as $S_i \to \infty$

$$\begin{aligned}&\left(\int_{|x_i-c_i|>\varepsilon \bigcap [0,1]} f(x_i;\alpha_i,\beta_i)dx_i\right)^{\frac{1}{S_i}}\\ =&C_i^{\frac{1}{S_i}}\left(\int_{|x_i-c_i|>\varepsilon \bigcap [0,1]} g^{S_i}(x_i)dx_i\right)^{\frac{1}{S_i}} \to \frac{\|g\|_{\infty,\varepsilon}}{\|g\|_\infty}\end{aligned} \tag{39}$$

where $\|g\|_{\infty,\varepsilon}$ *is the* L^∞ *norm of* g *when restricted to* $|x_i - c_i| > \varepsilon$.

By taking both sides of (39) to the S_i *power, we obtain*

$$\int_{|x_i-c_i|>\varepsilon \bigcap [0,1]} f(x_i;\alpha_i,\beta_i)dx_i \to \left(\frac{\|g\|_{\infty,\varepsilon}}{\|g\|_\infty}\right)^{S_i} \tag{40}$$

Obviously $\frac{\|g\|_{\infty,\varepsilon}}{\|g\|_\infty} < 1$, *for the fact that* g *is continuous and has a unique maximum at* c_i, *thus*

$$\int_{|x_i-c_i|>\varepsilon \bigcap [0,1]} f(x_i;\alpha_i,\beta_i)dx_i \to 0 \tag{41}$$

as $S_i \to \infty$.

Note that $\int_0^1 f(x_i;\alpha_i,\beta_i)dx_i = 1$ *and the proof is finished.*

Lemma 2. *For two or more random variables* $e_i \sim Beta(\alpha_i, \beta_i)$, *assume* m *is the index of action that has the maximum reward probability such that* $c_m = \max(c_i)$, *then*

$$\lim_{S_i\to\infty} Pr\{e_m > \max_{i\neq m}(e_i)\} = 1 \tag{42}$$

Proof 2.

$$\begin{aligned}&Pr\{e_m > \max_{i\neq m}(e_i)\}\\ =&\int_0^1 f(x_m;\alpha_m,\beta_m)\prod_{i\neq m}[\int_0^{x_m} f(x_i;\alpha_i,\beta_i)dx_i]dx_m\end{aligned} \tag{43}$$

From Lemma 1, we know that $f(x_i;\alpha_i,\beta_i) \to \delta(x_i - c_i)$ *as* $S_i \to \infty$.

By using the sampling property of the Dirac delta function, (43) can be simplified as

$$\lim_{S_i\to\infty} \Pr\{e_m > \max_{i\neq m}(e_i)\} = \lim_{S_i\to\infty}\prod_{i\neq m}\int_0^{c_m} f(x_i;\alpha_i,\beta_i)dx_i \tag{44}$$

$$= \lim_{S_i\to\infty}\prod_{i\neq m}\int_0^{c_m}\delta(x_i - c_i)dx_i \tag{45}$$

Note that $\forall i \neq m$, *as* $c_i \in [0, c_m]$, $\int_0^{c_m}\delta(x_i - c_i)dx_i = 1$. *And finally*

$$\lim_{S_i\to\infty} Pr\{e_m > \max_{i\neq m}(e_i)\} = 1 \tag{46}$$

This completes the proof.

Remark 1. *It is noted that, Lemma 2 implies* $\lim_{S_i\to\infty} Pr\{\mathcal{H}_m\} = 1$

Lemma 3. *Suppose one component of the vector* $\{Pr(\mathcal{H}_1), Pr(\mathcal{H}_2), \ldots, Pr(\mathcal{H}_r)\}$, *say* $Pr(\mathcal{H}_i)$ *approaches 1 only if the number of each action been selected* $S_i \to \infty$, *for all* $i \in \{1, 2, \ldots, r\}$.

Proof 3. *As* $Pr(\mathcal{H}_i) \to 1$, *for any* $\delta > 0$, *we have* $Pr(\mathcal{H}_i) \geq 1 - \delta$, *hence*

$$Pr(\mathcal{H}_i) = \int_0^1 f(x_i;\alpha_i,\beta_i)\prod_{j\neq i}[\int_0^{x_i} f(x_j;\alpha_j,\beta_j)dx_j]dx_i \tag{47}$$

$$\begin{aligned}=&\int_0^1 f(x_i;\alpha_i,\beta_i)\int_0^{x_i} f(x_j;\alpha_j,\beta_j)dx_j\\ &\prod_{k\neq i,k\neq j}[\int_0^{x_i} f(x_k;\alpha_k,\beta_k)dx_k]dx_i\end{aligned} \tag{48}$$

$$\begin{aligned}\leq&\int_0^1 f(x_i;\alpha_i,\beta_i)\int_0^{x_i} f(x_j;\alpha_j,\beta_j)dx_j\\ &\prod_{k\neq i,k\neq j}[\int_0^{1} f(x_k;\alpha_k,\beta_k)dx_k]dx_i\end{aligned} \tag{49}$$

$$=\int_0^1 f(x_i;\alpha_i,\beta_i)\int_0^{x_i} f(x_j;\alpha_j,\beta_j)dx_jdx_i \tag{50}$$

$$=Pr\{e_i > e_j\} \tag{51}$$

As a result, for all $j \neq i$,

$$Pr\{e_i > e_j\} \geq Pr(\mathcal{H}_i) \to 1 \tag{52}$$

$$\Rightarrow Pr\{e_j > e_i\} \to 0 \tag{53}$$

$$\Rightarrow \int_0^1 f(x_j;\alpha_j,\beta_j)\int_0^{x_j} f(x_i;\alpha_i,\beta_i)dx_idx_j \to 0 \tag{54}$$

By denoting $F(x) = f(x;\alpha_j,\beta_j)B(x;\alpha_i,\beta_i) = f(x;\alpha_j,\beta_j)\int_0^x f(x_i;\alpha_i,\beta_i)dx_i$, *we have*

$$\int_0^1 F(x)dx \to 0 \tag{55}$$

Suppose at least one of S_i *and* S_j *is not infinity, thus three possible cases should be discussed.*

1. Case $S_i < \infty$ and $S_j < \infty$.

   In this case, $f(x_j;\alpha_j,\beta_j)$ *is a continuous function and strictly positive on* $(0, 1)$. *As* $\frac{dB(x;\alpha_i,\beta_i)}{dx} = f(x;\alpha_i,\beta_i)$ *is continuous,* $B(x;\alpha_i,\beta_i)$ *is continuously differentiable which implies it is a continuous function. In addition,* $B(x;\alpha_i,\beta_i)$ *is* *strictly positive on* $(0, 1)$*. Clearly, the product of two strictly positive continuous functions* $F(x)$ *is continuous and* $F(x) > 0$ *on the interval* $(0, 1)$*, hence*

$$\int_0^1 F(x)dx > 0 \tag{56}$$

which contradicts (55).

2. *Case* $S_i < \infty$ *and* $S_j = \infty$*.*

   Similarly, we can prove that $B(x; \alpha_i, \beta_i)$ *is strictly positive and continuous on* $(0, 1)$*, and* $f(x; \alpha_j, \beta_j) \to \delta(x - c_j)$*.*

   Hence, (54) can be written as:

$$B(c_j; \alpha_i, \beta_i) \to 0 \tag{57}$$

   that contradicts the fact that $B(x; \alpha_i, \beta_i)$ *is strictly positive on* $(0, 1)$*.*

3. *Case* $S_i = \infty$ *and* $S_j < \infty$*.*

   Similarly, we can prove that $f(x_j; \alpha_j, \beta_j)$ *is strictly positive and continuous on* $(0, 1)$*, and* $f(x_i; \alpha_i, \beta_i) \to \delta(x - c_i)$*.*

   Hence, (54) can be written as:

$$\int_{c_i}^1 f(x; \alpha_j, \beta_j)dx \to 0 \tag{58}$$

   which implies $f(x_j; \alpha_j, \beta_j) = 0$ *on* $(c_i, 1)$*, that contradicts the fact that* $f(x_j; \alpha_j, \beta_j)$ *is strictly positive on* $(0, 1)$*.*

By summarizing the above three cases, we conclude that the supposition is false and both S_i *and* S_j *must be infinity.*

As i, j *enumerate all the action indexes, the proof is completed.*

Remark 2. *From Lemma 3 and Remark 1, one can immediately see that given a threshold* $\eta \to 1$*, PFLA will converge to the optimal action w.p.1 whenever it gets converged.*

Lemma 4. *The Monte Carlo estimation of* $Pr(\mathcal{H}_i)$ *will converge almost surely as the number of Monte Carlo replications N tends to infinity, i.e.,*

$$Pr\{\lim_{N\to\infty} \widehat{Pr}(\mathcal{H}_i) = Pr(\mathcal{H}_i)\} = 1 \tag{59}$$

Proof 4. *This lemma can be easily derived according to the strong law of large numbers.*

Let us now state and prove the main result for algorithm PFLA.

Theorem 1. *PFLA is* ϵ*-optimal in every stationary random environment. That is, given any* $\varepsilon > 0$*, there exists a* $N_0 < \infty$*, a* $t_0 < \infty$*, and a* $\eta_0 < 1$ *such that for all* $t \geq t_0$*,* $N \geq N_0$ *and* $\eta > \eta_0$*:*

$$\widehat{Pr}(\mathcal{H}_m) > 1 - \varepsilon \tag{60}$$

Proof 5. *The theorem is equivalent to showing that,*

$$Pr\{\lim_{\substack{N\to\infty \\ t\to\infty \\ \eta\to 1}} \widehat{Pr}(\mathcal{H}_m) = 1\} = 1 \tag{61}$$

From Lemma 4, we know that (61) is equivalent to

$$Pr\{\lim_{\substack{t\to\infty \\ \eta\to 1}} Pr(\mathcal{H}_m) = 1\} = 1 \tag{62}$$

And according to Remark 2, we only need to prove that the scheme can definitely get converged, i.e., at least one of the components $\{Pr(\mathcal{H}_1), Pr(\mathcal{H}_2), \ldots, Pr(\mathcal{H}_r)\}$ *approaches 1, as* $t \to \infty$ *and* $\eta \to 1$*.*

Suppose the scheme has not converged yet at time t_1*, because exactly one action will be explored at each time instant, we have* $\sum_i S_i = t_1$*.*

As $t_1 \to \infty$*, a finite series has an infinite sum, which indicates that at least one of the terms* S_i *has an infinite value.*

Then denote the set of actions, whose corresponding observation times $S_i(t_1) \to \infty$*, as* $\mathbb{A}_1$*, and denote the absolute complement set of* $\mathbb{A}_1$ *as* $\mathbb{A}_2$*.*

1. *If* $\mathbb{A}_2 = \emptyset$*, then for any action* a_i*, we have* $S_i \to \infty$*.*

   By considering Remark 1, we have

$$Pr(\mathcal{H}_m) \to 1 \tag{63}$$

2. *We will show that if* $\mathbb{A}_2 \neq \emptyset$*, then it is impossible that both the top two possibly optimal actions belong to set* $\mathbb{A}_1$*.*

   Denote the action in $\mathbb{A}_1$ *with the highest reward probability as* a_{m1}*, then according to Lemma 2,* $\forall a_i \in \mathbb{A}_1$ *and* $i \neq m1$*,*

$$Pr(\mathcal{H}_i) \to 0. \tag{64}$$

   While for actions $a_j \in \mathbb{A}_2$*,*

$$Pr(\mathcal{H}_j) = \int_0^1 f(x_j; \alpha_j, \beta_j) \prod_{k\neq j} [\int_0^{x_j} f(x_k; \alpha_k, \beta_k)dx_k]dx_j \tag{65}$$

$$= \int_0^1 f(x_j; \alpha_j, \beta_j) \prod_{a_{k1}\in\mathbb{A}_1} [\int_0^{x_j} f(x_{k1}; \alpha_{k1}, \beta_{k1})dx_{k1}] \prod_{k2\neq i, a_{k2}\in\mathbb{A}_2} [\int_0^{x_j} f(x_{k2}; \alpha_{k2}, \beta_{k2})dx_{k2}]dx_j \tag{66}$$

$$= \int_0^1 f(x_j; \alpha_j, \beta_j) \prod_{a_{k1}\in\mathbb{A}_1} I(x_j \geq c_{k1}) \prod_{k2\neq i, a_{k2}\in\mathbb{A}_2} [\int_0^{x_j} f(x_{k2}; \alpha_{k2}, \beta_{k2})dx_{k2}]dx_j \tag{67}$$

$$= \int_{c_{m1}}^1 f(x_j; \alpha_j, \beta_j) \prod_{k2\neq i, a_{k2}\in\mathbb{A}_2} [\int_0^{x_j} f(x_{k2}; \alpha_{k2}, \beta_{k2})dx_{k2}]dx_j \tag{68}$$

TABLE 2 Accuracy (*number of correct convergences/number of experiments*) of the compared algorithms in environments E_1 to E_9, when using the "best" learning parameters (250,000 experiments were performed for each scheme in each environment).

Env.	DP$_{ri}$	DGPA	DBPA	DGCPA	SE$_{ri}$	GBSE	LELA$_R$	PFLA
E_1	0.997	0.998	0.997	0.997	0.998	0.998	0.998	0.999
E_2	0.997	0.997	0.998	0.998	0.997	0.998	0.998	0.999
E_3	0.996	0.996	0.996	0.997	0.997	0.997	0.997	0.998
E_4	0.998	0.997	0.998	0.997	0.998	0.998	0.998	0.999
E_5	0.995	0.997	0.996	0.997	0.997	0.997	0.997	0.997
E_6	0.994	0.996	0.994	0.996	0.996	0.996	0.996	0.999
E_7	0.993	0.995	0.993	0.995	0.995	0.995	0.995	0.996
E_8	0.996	0.997	0.996	0.998	0.998	0.998	0.997	0.999
E_9	0.994	0.997	0.994	0.997	0.997	0.997	0.997	0.997

TABLE 3 Comparison of the average number of iterations required for convergence of the compared algorithms in environments E_1 to E_9.

Env.		E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9
DP$_{ri}$	Para.[a]	$n = 22$	$n = 29$	$n = 74$	$n = 18$	$n = 298$	$n = 653$	$n = 2,356$	$n = 216$	$n = 881$
	Iter.[b]	46	61	127	64	1,086	2,500	9,613	783	2,363
DGPA	Para.	$n = 20$	$n = 28$	$n = 70$	$n = 32$	$n = 33$	$n = 65$	$n = 204$	$n = 28$	$n = 55$
	Iter.	52	66	141	72	880	1,677	5,191	754	1,445
DBPA	Para.	$n = 20$	$n = 24$	$n = 57$	$n = 13$	$n = 102$	$n = 216$	$n = 820$	$n = 57$	$n = 326$
	Iter.	44	54	105	52	646	1,419	5,423	432	1,384
DGCPA	Para.	(12, 1)	(18, 2)	(38, 3)	(18, 3)	(3, 5)	(6, 9)	(17, 20)	(2, 4)	(5, 7)
	Iter.	41	52	99	50	351	678	2032	298	598
SE$_{ri}$	Para.	(15, 3)	(18, 3)	(38, 5)	(12, 3)	(16, 8)	(32, 12)	(105, 25)	(13, 6)	(33, 12)
	Iter.	43	51	99	54	426	834	2,540	325	729
GBSE	Para.	(12, 3)	(14, 3)	(22, 5)	(8, 3)	(1, 7)	(3, 9)	(6, 17)	(1, 5)	(3, 8)
	Iter.	43	53	97	55	401	772	2,262	306	612
LELA$_R$	Para.	$n = 13$	$n = 16$	$n = 41$	$n = 9$	$n = 9$	$n = 17$	$n = 59$	$n = 9$	$n = 24$
	Iter.	51	65	137	63	629	1,129	3,733	586	1,072
PFLA	Iter.	44	51	102	54	510	934	2,737	538	735

For all schemes, the "best" learning parameters for each environment are used (250,000 experiments were performed for each scheme in each environment).
[a]Para., Parameter. And for methods that have more than one tunable parameter, a tuple is used to represent the parameters. For example, $n = 38$, $\gamma = 5$ is represented as (38, 5) in the table.
[b]Iter., Iteration.

As $c_{m1} < 1$, and the integrand is strictly positive and continuous. Obviously, (68) is larger than zero trivially.

For actions in $\mathbb{A}_1$ other than a_{m1}, $Pr(\mathcal{H}_i) \to 0$, while for actions in $\mathbb{A}_2$, all $Pr(\mathcal{H}_i)$ equal some constants that are larger than zero. Hence, at least one action of the top two most probably optimal actions is from $\mathbb{A}_2$ and this action will be chosen to draw feedback.

As time $t \to \infty$, once $\mathbb{A}_2 \neq \emptyset$, one action in $\mathbb{A}_2$ will be explored. As a consequence, we can always find a $t_0 > t_1$ such that all actions in $\mathbb{A}_2$ will be explored infinite times and yield an empty $\mathbb{A}_2$.

Combining the above two cases, we may infer that all actions will be explored an infinite number of times and $Pr(\mathcal{H}_m) \to 1$.

This completes the proof.

5. Simulation results

During the last decade, SE$_{ri}$ has been considered as the state-of-the-art algorithm for a long time, however, some recently proposed algorithms (Ge et al., 2015a; Jiang et al., 2015) claim a faster convergence than SE$_{ri}$. To make a comprehensive comparison among currently available techniques, as well as to verify the effectiveness of the proposed parameter-free scheme, in this section, PFLA is compared with several classic parameter-based learning automata schemes, including DP$_{ri}$ (Oommen and Lanctôt, 1990), DGPA (Agache and Oommen, 2002), DBPA (Zhang et al., 2013), DGCPA* (Ge et al., 2015a), SE$_{ri}$ (Papadimitriou et al., 2004), GBSE (Jiang et al., 2015), and LELA$_R$ (Zhang et al., 2014).

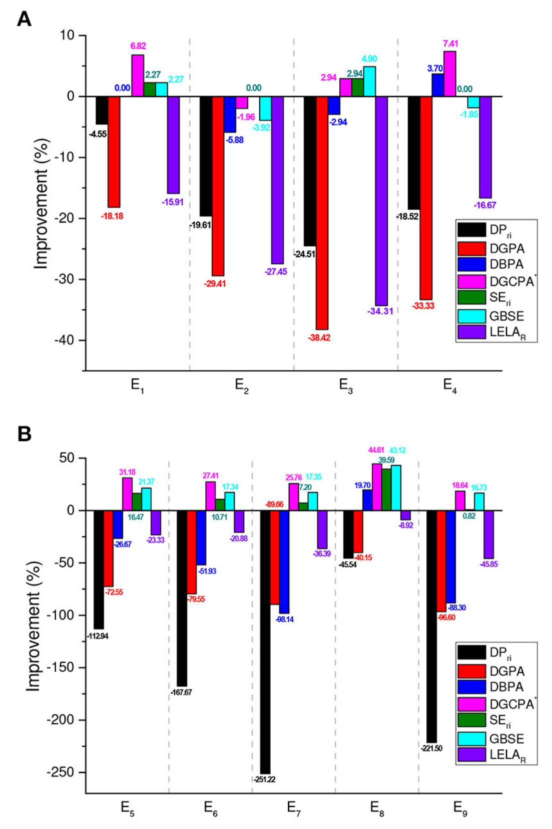

FIGURE 3
Convergence rate improvements of the compared algorithms relative to PFLA in benchmark environments, calculated by using $\frac{\text{Iterations}_{\{PFLA\}} - \text{Iterations}_{\{ComparedAlgorithm\}}}{\text{Iterations}_{\{PFLA\}}}$. **(A)** Two-action environments. **(B)** Ten-action environments.

All the schemes are evaluated in four two-action benchmark environments (Ge et al., 2015b) and five 10-action benchmark environments (Papadimitriou et al., 2004). The actions' reward probabilities for each environment are as follows:

E_1 :{0.90, 0.60}
E_2 :{0.80, 0.50}
E_3 :{0.80, 0.60}
E_4 :{0.20, 0.50}
E_5 :{0.65, 0.50, 0.45, 0.40, 0.35, 0.30, 0.25, 0.20, 0.15, 0.10}
E_6 :{0.60, 0.50, 0.45, 0.40, 0.35, 0.30, 0.25, 0.20, 0.15, 0.10}
E_7 :{0.55, 0.50, 0.45, 0.40, 0.35, 0.30, 0.25, 0.20, 0.15, 0.10}
E_8 :{0.70, 0.50, 0.30, 0.20, 0.40, 0.50, 0.40, 0.30, 0.50, 0.20}
E_9 :{0.10, 0.45, 0.84, 0.76, 0.20, 0.40, 0.60, 0.70, 0.50, 0.30}

The comparison is organized in two ways:

1. Comparison between PFLA and parameter-based schemes with their learning parameters being carefully tuned.
2. Comparison between PFLA and parameter-based schemes without parameter tuning, using either pre-defined or randomly selected learning parameters.

5.1. Comparison with well-tuned schemes

Firstly, the parameter-based schemes are simulated with carefully tuned best parameters. The procedure for obtaining the best parameters is elaborated in the Appendix. The proposed PFLA, by contrast, takes identical parameter values of $\eta = 0.99$ and $N = 1,000$ in all nine environments.

The results of the simulations are summarized in Tables 2, 3. The *accuracy* is defined as *the ratio between the number of correct convergence and the number of experiments*, whilst the *iteration* as *the averaged number of required interactions between automaton and environment for a correct convergence*. It is noted that the initialization cost of estimators is also included. The number of initializations for each action is 10.

In Table 2, PFLA converges with relatively high accuracy consistently, coinciding with our analytical results in Section 4, and verifying the effectiveness of our proposed parameter-free scheme. And since the accuracies of all schemes are close, their convergence rates can be "fairly" compared[4].

In the aspect of convergence rate, obviously, in Table 3, PFLA is outperformed by the top performers, namely SE$_{ri}$, GBSE, and DGCPA*. Figure 3 depicts the improvements of the competitors over PFLA. Take E_7 as an example, the convergence rate of PFLA is improved by DGCPA*, SE$_{ri}$, and GBSE with 25.76, 7.20, and 17.35%, respectively. While other schemes, DP$_{ri}$, DGPA, and LELA$_R$ are outperformed by PFLA significantly. Generally speaking, FPLA is faster than deterministic estimator-based schemes and slower than stochastic estimator-based algorithms.

However, taking the parameter tuning cost of the competitors into consideration, the parameter-free property begins to show its superiority. In order to clarify that point, we record the number of interactions between automaton and environment during the process of parameter tuning for each parameter-based scheme. The results are summarized in Table 4[5]. It can be seen that the extra interactions required for parameter tuning by deterministic estimator-based schemes

4 Technically speaking, the comparison is not completely fair, that's the reason the word "fairly" are quoted. An explanation will be given in later subsections.

5 It is noted that the numerical value shown in Table 4 may differ according to the way parameter tuning being implemented, still it gives qualitatively evidence to the heavy parameter tuning cost of the parameter-based schemes. The technical details of the parameter tuning procedure used here are provided in the Appendix.

TABLE 4 The parameter tuning cost (number of extra interactions) of the compared algorithms in environments $E_1 - E_9$.

Env.	DP_{ri}	DGPA	DBPA	DGCPA	SE_{ri}	GBSE	$LELA_R$
E_1	3.075×10^6	3.023×10^6	2.523×10^6	3.046×10^7	2.062×10^7	1.881×10^7	2.471×10^6
E_2	3.866×10^6	4.552×10^6	3.373×10^6	3.633×10^7	2.669×10^7	2.241×10^7	3.346×10^6
E_3	1.521×10^7	1.554×10^7	1.045×10^7	9.192×10^7	8.704×10^7	6.180×10^7	1.042×10^7
E_4	2.616×10^6	5.331×10^6	2.147×10^6	3.445×10^7	2.215×10^7	2.025×10^7	2.362×10^6
E_5	3.947×10^8	6.248×10^7	1.033×10^8	2.421×10^8	1.268×10^9	3.443×10^8	2.437×10^7
E_6	1.813×10^9	1.708×10^8	4.117×10^8	7.442×10^8	6.905×10^9	9.331×10^8	6.262×10^7
E_7	1.503×10^{10}	1.369×10^9	4.931×10^9	7.618×10^9	1.207×10^{11}	9.158×10^9	3.910×10^8
E_8	2.008×10^8	5.264×10^7	4.146×10^7	1.808×10^8	7.079×10^8	2.714×10^8	2.209×10^7
E_9	1.802×10^9	1.208×10^8	5.933×10^8	5.495×10^8	6.266×10^9	8.092×10^8	7.029×10^7

(DGPA, DBPA, and $LELA_R$, except DP_{ri}) are slightly less than stochastic estimator-based schemes (DGCPA*, SE_{ri}, and GBSE). Both families of schemes cost millions of extra interactions for seeking the *best parameter*. The proposed scheme can achieve a comparative performance without relying on any extra interactions/information.

For better visualization, a scatter map is used to illustrate the performance of different methods. In the scatter map, each dot represents a specific method discussed in this section. The x-axis indicates the averaged accuracy achieved by each method in the benchmark environments, and the y-axis indicates the averaged iterations need for each method to get converged in the benchmark environments, as shown in Figure 4. It is noted that the iterations are normalized with respect to each environment before being averaged over different environments for each method. As we are always pursuing a method with higher accuracy and convergence rate, the method approaching the right bottom corner of the figure is better than the others. From Figure 4B, we can draw the conclusion that taking the parameter tuning cost of the competitors into consideration, the proposed PFLA is the best of all competitors.

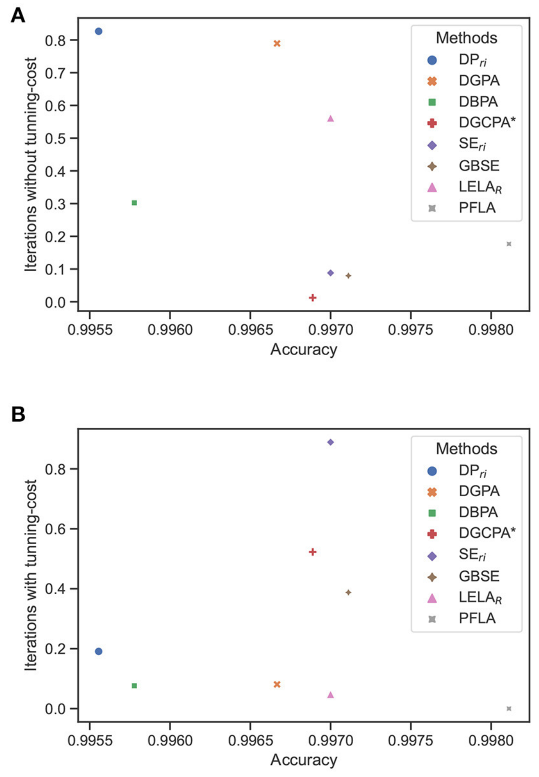

FIGURE 4
Averaged iterations vs. averaged accuracy in the nine benchmark environments of compared methods. **(A)** Without considering the parameter tunning cost. **(B)** With consideration of the parameter tunning cost.

5.2. Comparison with untuned schemes

In this part, the parameter-based algorithms are simulated in benchmark environments without their learning parameter specifically tuned. Their performance will be compared with PFLA under the same condition—no extra information about the environment is available.

5.2.1. Using generalized learning parameter

Firstly, the best parameter in E_2 and E_6 are applied for learning in other environments respectively to evaluate how well they can "generalize" in other environments. The results are shown in Tables 5, 6, respectively.

5.2.2. Using random learning parameter

Secondly, randomly selected learning parameters are adopted to evaluate the expected performance of each algorithm in fully unknown environments. The random resolution parameter takes value in the range from 90% of the minimal value to 110% of the maximal value of the best resolution

TABLE 5 Comparison of convergence rate and accuracy of the parameter-based algorithms in all environments other than E_2, when using the "best" learning parameters in E_2.

Env.	DP_{ri}		DGPA		DBPA		DGCPA		SE_{ri}		GBSE		$LELA_R$	
	Iter.	Acc.	Iter.	Acc.	Iter.	Acc.	Iter.	Acc.	Iter.	Acc.	Iter.	Acc.	Iter.	Acc.
E_1	55	0.998	65	0.999	49	0.998	53	0.995	47	0.999	48	0.999	59	0.998
E_3	63	0.975	70	0.976	57	0.976	61	0.972	57	0.979	62	0.983	67	0.976
E_4	90	0.999	66	0.996	79	0.999	45	0.994	69	0.999	80	0.999	96	0.999
E_5	264	0.895	767	0.995	301	0.967	640	0.999	286	0.962	701	0.998	1,026	0.999
E_6	319	0.804	835	0.971	408	0.927	821	0.996	351	0.897	836	0.987	1,068	0.995
E_7	393	0.658	976	0.858	577	0.806	1,249	0.957	443	0.748	1,093	0.905	1,155	0.909
E_8	253	0.918	752	0.997	280	0.979	616	0.999	273	0.981	648	0.999	954	0.999
E_9	246	0.801	827	0.981	275	0.869	751	0.997	299	0.903	636	0.984	758	0.986

TABLE 6 Comparison of convergence rate and accuracy of the parameter-based algorithms in all environments other than E_6, when using the "best" learning parameters in E_6.

Env.	DP_{ri}		DGPA		DBPA		DGCPA		SE_{ri}		GBSE		$LELA_R$	
	Iter.	Acc.[a]	Iter.	Acc.	Iter.	Acc.	Iter.	Acc.	Iter.	Acc.	Iter.	Acc.	Iter.	Acc.
E_1	812	1	125	0.999	282	1	29	0.783	95	1	26	0.769	61	0.999
E_2	923	1	126	0.999	320	1	28	0.777	103	0.999	29	0.781	68	0.998
E_3	899	1	133	0.996	317	1	31	0.725	124	0.998	29	0.716	70	0.978
E_4	1572	1	126	0.999	535	1	28	0.791	137	1	46	0.846	101	0.999
E_5	1879	0.999	1,582	0.999	969	0.999	501	0.972	641	0.999	599	0.999	1,085	0.999
E_7	3961	0.942	1,939	0.945	2,358	0.965	1,055	0.929	1,203	0.942	1,110	0.948	1,219	0.917
E_8	1641	0.999	1,555	0.999	845	0.999	495	0.974	629	0.999	592	0.999	1008	0.999
E_9	1923	0.987	1,667	0.998	1,078	0.988	673	0.973	725	0.996	675	0.997	799	0.989

[a] Acc., Accuracy.

parameter in the nine benchmark environments[6], and a range from 1 to 20 for the perturbation parameter if needed. The simulation results are demonstrated in Table 7.

From the three tables, there is a significant decline in accuracy in some environments. As the accuracies differ greatly in those cases, the convergence rates cannot be compared directly. Still, several conclusions can be drawn. One is that the performance of untuned parameter-based algorithms is unstable when learning in an unknown environment, and thus cannot be used in practical applications without parameter tuning. Another conclusion is that those algorithms, that use generalized learning parameters or random learning parameters, are either have a lower accuracy or a slower convergence rate than PFLA in the benchmark environment. In other words, none of them can outperform PFLA in both accuracy and convergence rate without the help of prior information.

6 For example, the resolution parameter of DPri is range from $\lfloor 90\% * 18 \rfloor$ to $\lceil 110\% * 2356 \rceil$, i.e., from 16 to 2,592.

5.3. Discussion of the fairness of the comparison

Technically speaking, the comparison between PFLA and well-tuned schemes is not fair. This is because the interactions can be perceived as information exchanges between automaton and the environment. So if the number of interactions is unlimited, the algorithm can simply use the empirical distributions. The outperforming of the well-tuned schemes owes to their richer knowledge about the environment acquired during the parameter tuning process. And for this reason, a fair comparison between PFLA and untuned schemes is carried out. Despite the unfairness of the first comparison, the significance lies in providing baselines for evaluating the convergence rate of PFLA qualitatively.

By the way, the comparison within parameter-based algorithms is not fair either, because the amount of prior information acquired is different. This method is widely

TABLE 7 Comparison of the average number of iterations required for convergence of the parameter-based algorithms in environments E_1 to E_9.

Env.	DP_{ri}		DGPA		DBPA		DGCPA		SE_{ri}		GBSE		LELA_R	
	Iter.	Acc.	Iter.	Acc.	Iter.	Acc.	Iter.	Acc.	Iter.	Acc.	Iter.	Acc.	Iter.	Acc.
E_1	1,606	0.999	216	0.999	574	0.999	76	0.924	121	0.999	71	0.943	108	0.999
E_2	1,824	0.999	217	0.999	652	0.999	73	0.922	132	0.999	78	0.943	121	0.999
E_3	1,767	0.999	224	0.996	638	0.998	87	0.896	152	0.996	95	0.927	124	0.990
E_4	3,121	0.999	217	0.999	1,105	0.999	66	0.928	189	0.999	109	0.953	192	0.999
E_5	3,253	0.996	2,821	0.999	1,476	0.997	836	0.977	687	0.993	925	0.997	2,153	0.999
E_6	3,995	0.988	2,922	0.995	2,072	0.993	1,042	0.975	886	0.979	1,158	0.991	2,229	0.996
E_7	6,260	0.951	3,309	0.963	3,626	0.970	1,647	0.952	1,302	0.911	1,699	0.952	2,395	0.958
E_8	2,859	0.997	2,774	0.999	1,288	0.999	810	0.978	647	0.996	878	0.998	2003	0.999
E_9	3,031	0.985	2,919	0.997	1,615	0.987	1,041	0.976	768	0.979	988	0.988	1547	0.993

The randomly selected learning parameters are used and 250,000 experiments were performed for each scheme in each environment.

used by the research community to compare the theoretically best performance of their proposed algorithms, however, the hardness of the algorithm can achieve theoretically best is usually ignored.

6. Conclusion

In this paper, we propose a parameter-free learning automaton scheme for learning in stationary stochastic environments. The proof of the ε-optimality of the proposed scheme in every stationary random environment is presented. Compared with existing schemes, the proposed PFLA possesses a parameter-free property, i.e., a set of parameters that can be universally applicable to all environments. Furthermore, our scheme is evaluated in four two-action and five 10-action benchmark environments and compared with several classic and state-of-the-art schemes in the field of LA. Simulations confirm that our scheme can converge to the optimal action with high accuracy. Although the rate of convergence is outperformed by some schemes that are well-tuned for specific environments, the proposed scheme still shows its intriguing property of not relying on the parameter-tuning process. Our future work includes optimizing the exploration strategy further.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

All authors listed have made a substantial, direct, and intellectual contribution to the work and approved it for publication.

Funding

This research work was funded by the National Nature Science Foundation of China under Grant 61971283, Shanghai Municipal Science and Technology Major Project under Grant 2021SHZDZX0102, and Shanghai AI Innovation and Development Project under Grant 2020-RGZN-02026.

Conflict of interest

Author HG was employed by company Shanghai Data Miracle Intelligent Technology Co., Ltd.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Agache, M., and Oommen, B. J. (2002). Generalized pursuit learning schemes: new families of continuous and discretized learning automata. *IEEE Trans. Syst. Man Cybern. Part B* 32, 738–749. doi: 10.1109/TSMCB.2002.1049608

Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002). Finite-time analysis of the multi-armed bandit problem. *Mach. Learn.* 47, 235–256. doi: 10.1023/A:1013689704352

Cook, J. D. (2005). *Exact Calculation of Beta Inequalities*. Technical report, Technical report, UT MD Anderson Cancer Center Department of Biostatistics.

Cuevas, E., Wario, F., Zaldivar, D., and Pérez-Cisneros, M. (2013). *Circle Detection on Images Using Learning Automata*. Berlin; Heidelberg: Springer Berlin Heidelberg. 545–570. doi: 10.1007/978-3-642-29694-9_21

Ge, H., Jiang, W., Li, S., Li, J., Wang, Y., and Jing, Y. (2015a). A novel estimator based learning automata algorithm. *Appl. Intell.* 42, 262–275. doi: 10.1007/s10489-014-0594-1

Ge, H., Yan, Y., Li, J., Ying, G., and Li, S. (2015b). "A parameter-free gradient bayesian two-action learning automaton scheme," in *Proceedings of the International Conference on Communications, Signal Processing, and Systems* (Berlin; Heidelberg).

Granmo, O.-C. (2010). Solving two-armed Bernoulli bandit problems using a Bayesian learning automaton. *Int. J. Intell. Comput. Cybern.* 3, 207–234. doi: 10.1108/17563781011049179

Horn, G., and Oommen, B. (2010). Solving multiconstraint assignment problems using learning automata. *IEEE Trans. Syst. Man Cybern. Part B* 40, 6–18. doi: 10.1109/TSMCB.2009.2032528

Jiang, W., Li, B., Tang, Y., and Philip Chen, C. L. (2015). A new prospective for learning automata: a machine learning approach. *Neurocomputing* 188, 319–325. doi: 10.1016/j.neucom.2015.04.125

Kumar, N., Misra, S., and Obaidat, M. (2015). Collaborative learning automata-based routing for rescue operations in dense urban regions using vehicular sensor networks. *IEEE Syst. J.* 9, 1081–1090. doi: 10.1109/JSYST.2014.2335451

Misra, S., Krishna, P., Kalaiselvan, K., Saritha, V., and Obaidat, M. (2014). Learning automata based QoS framework for cloud IAAS. *IEEE Trans. Netw. Service Manage.* 11, 15–24. doi: 10.1109/TNSM.2014.011614.130429

Narendra, K. S., and Thathachar, M. (1974). Learning automata-a survey. *IEEE Trans. Syst. Man Cybern.* 4, 323–334. doi: 10.1109/TSMC.1974.5408453

Narendra, K. S., and Thathachar, M. A. (2012). *Learning Automata: An Introduction*. New York, NY: Courier Dover Publications.

Oommen, B., and Hashem, M. (2010). Modeling a student-classroom interaction in a tutorial-like system using learning automata. *IEEE Trans. Syst. Man Cybern. Part B* 40, 29–42. doi: 10.1109/TSMCB.2009.2032414

Oommen, B. J., and Agache, M. (2001). Continuous and discretized pursuit learning schemes: various algorithms and their comparison. *IEEE Trans. Syst. Man Cybern. Part B* 31, 277–287. doi: 10.1109/3477.931507

Oommen, B. J., and Lanctôt, J. K. (1990). Discretized pursuit learning automata. *IEEE Trans. Syst. Man Cybern.* 20, 931–938. doi: 10.1109/21.105092

Oommen, J., and Misra, S. (2009). "Cybernetics and learning automata," in *Springer Handbook of Automation*, ed S. Y. Nof (Berlin; Heidelberg: Springer), 221–235. doi: 10.1007/978-3-540-78831-7_12

Owen, A. B. (2013). Monte carlo theory, methods and examples.

Papadimitriou, G. I., Sklira, M., and Pomportsis, A. S. (2004). A new class of ε-optimal learning automata. *IEEE Trans. Syst. Man Cybern. Part B* 34, 246–254. doi: 10.1109/TSMCB.2003.811117

Song, Y., Fang, Y., and Zhang, Y. (2007). "Stochastic channel selection in cognitive radio networks," in *Global Telecommunications Conference, 2007, GLOBECOM '07* (Washington, DC), 4878–4882. doi: 10.1109/GLOCOM.2007.925

Sutton, R. S., and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. Cambridge, MA; London: The MIT Press.

Tsetlin, M. (1973). *Automaton Theory and Modeling of Biological Systems*. New York, NY: Academic Press.

Tsetlin, M. L. (1961). On the behavior of finite automata in random media. *Avtom. Telemekh.* 22, 1345–1354.

Vahidipour, S., Meybodi, M., and Esnaashari, M. (2015). Learning automata-based adaptive petri net and its application to priority assignment in queuing systems with unknown parameters. *IEEE Trans. Syst. Man Cybern. Syst.* 45, 1373–1384. doi: 10.1109/TSMC.2015.2406764

Yazidi, A., Granmo, O.-C., and Oommen, B. (2013). Learning automaton based online discovery and tracking of spatiotemporal event patterns. *IEEE Trans. Cybern.* 43, 1118–1130. doi: 10.1109/TSMCB.2012.2224339

Zhang, J., Wang, C., and Zhou, M. (2014). Last-position elimination-based learning automata. *IEEE Trans. Cybern.* 44, 2484–2492. doi: 10.1109/TCYB.2014.2309478

Zhang, X., Granmo, O.-C., and Oommen, B. J. (2013). On incorporating the paradigms of discretization and Bayesian estimation to create a new family of pursuit learning automata. *Applied Intell.* 39, 782–792. doi: 10.1007/s10489-013-0424-x

Appendix

The standard parameter tuning procedure of learning automata

As emphasized in Section 1, parameter tuning is intended to balance the trade-off between speed and accuracy. And the standard procedure of parameter tuning is pioneered in Oommen and Lanctôt (1990) and become a common practice in follow-up researches (Oommen and Agache, 2001; Agache and Oommen, 2002; Papadimitriou et al., 2004; Zhang et al., 2014; Ge et al., 2015a; Jiang et al., 2015).

The basic idea is, that the smallest value of resolution parameter n that yielded the fastest convergence and that simultaneously resulted in zeros errors in a sequence of NE experiments are defined as "best" parameters. Besides, to reduce the variance coefficient of the "best" values of n, Papadimitriou et al. (2004) advocate performing the same procedure 20 times and computing the average "best" value of n in these experiments. For tuning stochastic estimator-based learning automata, which have a perturbation parameter γ in addition to the well-known resolution parameter n. A two-dimensional grid search should be performed to seek the best parameter pair (n, γ). The method used in Papadimitriou et al. (2004) is to obtain the "best" resolution parameter n for each value of γ, and then evaluate the speed of convergence for each of the (n, γ) pairs and choose the best pair.

Based on these instructions, we use the following procedure for parameter tuning in our experiment:

The resolution parameter is initialized to 1 and increased by 1 each time a wrong convergence emerges until a certain number of successive *No Error* experiments is carried out. Repeat this process 20 times, averaging over these 20 resulting values, and denote it as the best resolution parameter. The value of number of successive No Error experiments is set as $NE = 750$, as the same value in Papadimitriou et al. (2004), Zhang et al. (2014), Jiang et al. (2015), and Ge et al. (2015a). For tuning the "best" γ, in our simulation settings, for the four two-action environments, the search range of γ is from 1 to 10; For the five ten-action environments except E_7, the search range of γ is from 1 to 20, while for E_7, the most difficult one, the range is a little wider, from 1 to 30.

It is noted that the above standard procedure has been widely adopted by the research community, but it does not mean this is the most efficient way. Apparently, it can be improved by several methods, such as random search or two-stage coarse-to-fine search. This issue is worth further investigation and is beyond the scope of this paper.