A New Learning Automaton for Selecting an Arbitrary Subset of Actions

Junqi Zhang, *Senior Member, IEEE*, Peng Zu, PengZhan Qiu, and MengChu Zhou, *Fellow, IEEE*

***Abstract*—As a powerful reinforcement learning method, the learning automaton (LA) has been studied, analyzed, and applied to various engineering systems for decades. However, the state-of-the-art LA-based methods can select only the optimal action or optimal subset and cannot select an arbitrary target subset like selecting the best and worst actions or the ones in a given rank range. In order to solve the problem of selecting a given arbitrary subset of actions, this work proposes a novel pursuit learning scheme, called a discretized equal pursuit reward-inaction algorithm for arbitrary subset selection (DEP$_{RI}$-AS). The proposed scheme pursues the currently estimated arbitrary action subset and makes the probabilities of selecting each action in the subset equal, so as to increase the convergence speed. The proof of its ϵ-optimality property is presented. Simulation results of comparison experiments, parameter analysis, and a real-world application demonstrate its power in selecting a given subset of user-desired actions.**

***Index Terms*—Arbitrary subset, learning automaton (LA), pursuit LA.**

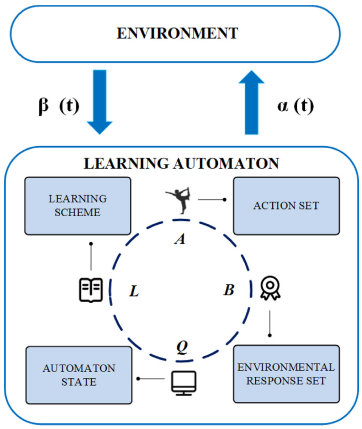


Fig. 1. Interaction between LA and a random environment.

I. Introduction

BY INTERACTING with a random environment, a learning automaton (LA) learns the optimal action with the highest probability of obtaining a reward from a set of allowable actions. Fig. 1 is redrawn and enriched based on the basic logic of the one in [1]. It describes LA's components and environments more explicitly. As shown in Fig. 1, LA is defined as a quadruple consisting of an action set, an environmental response set, an automaton state, and a learning scheme. In each iteration, LA selects an action $\alpha(t)$ and receives a response $\beta(t)$ from an environment. By employing this response and its learning scheme, LA updates its state. LA can be applied in many fields to select the best object, such as selecting the machine with the highest pass rate, the employee with the highest work completion rate, the athlete with the best performance, or the drug with the best therapeutic effect. All of the above scenarios are stochastic environments, in which the response to the same action is stochastic. The learning property of LA makes it have a strong anti-interference ability in real applications and is well suited to such problems. LA can also be applied to the Internet of Things [2], [3], [4], feature selection [5], and data clustering [6].

A. Literature Review

In LA designs, variable structure stochastic automata (VSSA) are proposed in [7], of which the action selecting probabilities are dynamically updated over time. Thathachar et al. [8], [9] introduced discretization and an estimator in VSSA to speed up its convergence. Based on the estimator, Thathachar and Sastry [10] proposed a pursuit scheme. Oommen et al. [1], [11] proposed the discretized pursuit scheme (DP$_{RI}$) to speed up the pursuit scheme. The discretized generalized pursuit algorithm [12] is another well-known scheme. It pursues the actions that have higher reward estimates than the current chosen action to improve the convergence speed. As an efficient pursuit-based scheme, last-position elimination-based LA (LELA) [13] reduces the difficulty of parameter tuning. The computational complexity of fast discretized pursuit LA [14] to select an action and update the action probability does not increase with the

Manuscript received 19 May 2023; accepted 30 August 2023. Date of publication 3 October 2023; date of current version 19 December 2023. This work was supported in part by the Innovation Program of Shanghai Municipal Education Commission under Grant 202101070007E00098; in part by the Fundamental Research Funds for the Central Universities; in part by the National Natural Science Foundation of China under Grant 62073244 and Grant 61876218; in part by the Shanghai Innovation Action Plan under Grant 20511100500; and in part by the Deanship of Scientific Research (DSR) at King Abdulaziz University (KAU), Jeddah, Saudi Arabia, under Grant RG-11-135-43. This article was recommended by Associate Editor L. Chen. *(Corresponding author: MengChu Zhou.)*

Junqi Zhang, Peng Zu, and PengZhan Qiu are with the Department of Computer Science and Technology, Key Laboratory of Embedded System and Service Computing, Ministry of Education, Tongji University, Shanghai 200092, China (e-mail: zhangjunqi@tongji.edu.cn; 1754027@tongji.edu.cn; 1930807@tongji.edu.cn).

MengChu Zhou is with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102 USA (e-mail: zhou@njit.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TSMC.2023.3312282.

Digital Object Identifier 10.1109/TSMC.2023.3312282

TABLE I
COMPARISON OF LEARNING AUTOMATA ALGORITHMS

References	Year	Optimal action	Optimal subset	Best-worst	Arbitrary given rank range
[1]	1990	✓	×	×	×
[12]	2002	✓	×	×	×
[13]	2014	✓	×	×	×
[24]	2014	✓	✓	×	×
[14]	2015	✓	×	×	×
[25]	2015	✓	✓	×	×
[15]	2016	✓	×	×	×
[16]	2016	✓	×	×	×
[17]	2017	✓	×	×	×
[19]	2019	✓	×	×	×
[18]	2020	✓	×	×	×
[20]	2020	✓	×	×	×
[21]	2020	✓	×	×	×
[22]	2021	✓	×	×	×
[23]	2022	✓	×	×	×
Ours	2022	✓	✓	✓	✓

number of actions. Generalized Bayesian stochastic estimator LA presented by Jiang et al. [15] further improves the convergence speed. The optimal computing budget allocation (OCBA)-based LA [16] employs OCBA to accomplish ordinal optimization. Ge et al. [17] proposed a parallel framework for LA that guarantees accuracy while reducing the number of interactions with the environment. Yazidi et al. [18] presented a hierarchical LA for environments with a large number of actions. With the great improvement in speed and accuracy of LA, some researchers have focused on reducing the cost of parameter tuning. Guo and Li [19] proposed a parameter-free LA (PFLA), which can well track sudden environmental changes. Di et al. [20] presented an efficient PFLA that employs a separating function to evaluate actions, thus reducing the computational cost. In addition, some researchers have focused on environments with different types of responses. Yazidi et al. [21] proposed an LA suitable for deterministic optimization environments. LA based on Bayesian inference is proposed by Di et al. [22] to deal with the Q-model environment. Guo et al. [23] proposed LA that can be applied to nonstationary environments, which has better applicability compared with related work.

B. Motivation

In Table I, the functions of popular and latest LA variants are summarized, where "✓" means yes and "×" means no. It can be seen that most state-of-the-art LA variants can learn only the optimal action, which greatly limits the application field of LA. There are problems that demand to select an arbitrary subset of actions, which is more preferred than the optimal one in many scenarios. Fig. 2 shows different examples of selecting an arbitrary subset of actions in an environment with eight allowable actions. The eight actions are ranked in descending order of probability of being rewarded, i.e., action 1 has the highest probability of being rewarded and action 8 has the lowest probability of being rewarded. The blue ones are actions in the subset that meets the required ranking, and the gray ones are other actions. In a discrete-event

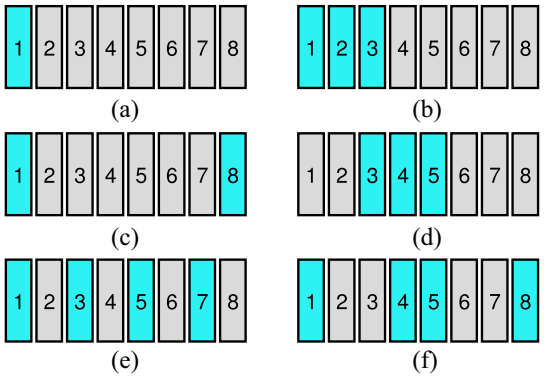


Fig. 2. Examples of an arbitrary action subset. (a) Optimal action. (b) Top-3 actions. (c) Best and worst actions. (d) 3–5th actions. (e) Actions with odd ranking. (f) Actions ranking 1st, 4th, 5th, and 8th.

system, a decision maker might prefer top-k designs, as shown in Fig. 2(b). Zhang et al. [24] introduced a new class of LA algorithms, which enables LA to select an optimal subset of actions. They give four learning schemes that can preliminarily solve this problem but do not specify the best scheme for this problem. It does not prove the ϵ-optimality or give sufficient experiments to verify the effectiveness of the proposed algorithms either. Guo et al. [25] employed LELA to further improve this method and speed up the convergence. Although some LA algorithms can solve the problem of selecting the optimal subset, they cannot work in other tasks of selecting an arbitrary subset. Some practical problems require selecting the best and worst design, as shown in Fig. 2(c). A typical example is that companies usually rank employees according to their performance, and then reward and punish the highest and lowest ranked one, respectively. In addition, some factories prefer the machines whose rank of performance is in a moderate range instead of the highest one due to their constrained budget, which means they require to select the designs within a given range, as shown in Fig. 2(d). In some tasks that require grouping, it is desirable that different groups have equal capabilities. Taking the scenario where eight machines need to be divided into two groups as an example, we can select machines with odd/even performance ranking, as shown in Fig. 2(e). Similarly, we can select two groups of machines with a more irregular distribution of rankings to achieve the same purpose, such as the machines ranking 1st, 4th, 5th, and 8th, and the machines ranking 2nd, 3rd, 6th, and 7th, as shown in Fig. 2(f). Presently, there is no LA algorithm that can solve all of the above problems.

Some existing studies of reinforcement learning focus on the selection of actions under constraints, such as the maximum number of times to select a certain action or safety risks [26], [27]. These constraints occur in the action selection phase of a learning process. Differently, the motivation of this work is to select the set of actions with the desired ranking, which is a learning target rather than a constraint. In addition, it is noteworthy that the above reinforcement learning method works in deterministic environments; while LA works in stochastic environments, in which the response to the same action is stochastic. LA is simple in structure and

easy to be implemented. Based on its advantages, some existing studies introduce LAs into many important fields, such as cooperative tasks [28], [29] and Bayesian networks [30], to improve the performance of existing methods. However, these studies only use the existing LA algorithm but do not improve it. Some existing studies [31], [32] are outstanding works that advance reinforcement learning. However, they fail to select the set of actions with the desired ranking and have a complex structure.

This study proposes a novel LA for selecting an arbitrary subset of actions, called the discretized equal pursuit reward-inaction algorithm for arbitrary subset selection (DEP$_{RI}$-AS). It can address problems, such as selecting the optimal subset, the best and worst actions, actions in a given rank range, actions with odd/even ranking, or those with a more irregular distribution of rankings. Distinctly from traditional discretized learning schemes [1], [11], the proposed scheme pursues the currently estimated arbitrary action subset and makes the probabilities of selecting each action in the subset equal, so as to increase the convergence speed. It possesses the ability to solve all of the above-mentioned selecting problems, thus greatly generalizing LA's application scope.

In Section II, the discretized pursuit scheme DP$_{RI}$ is introduced. In Section III, the proposed DEP$_{RI}$-AS algorithm together with its ϵ-optimality property is presented. Simulation results are given in Section IV to indicate the superiority of DEP$_{RI}$-AS over DUP$_{RI}$ [24]. Finally, Section V concludes this article.

II. Preliminaries

An environment is defined as a triple $\langle A, B, D\rangle$, and LA is defined as a quadruple $\langle A, B, Q, L\rangle$. The explanations of their elements are as follows.

1) $A = \{a_1, \ldots, a_r\}$ is an action set, where r is the number of allowable actions. In each iteration t, LA selects an action $a(t)$ from r allowable actions in the action set A.
2) $B = \{0, 1\}$ is an environmental response set. "1" denotes the reward response and "0" denotes the penalty one. $\beta(t)$ denotes the response received from the environment at time t.
3) $D = (d_1, \ldots, d_r)$ is a reward probability set for the environment to reward actions and is constant at all times.
4) Q is an automaton state consisting of P and $\widetilde{D}$. $Q(t)$ is the automaton state at time t. $P(t) = (p_1(t), \ldots, p_r(t))$ is the action probability vector at time t, where $p_i(t)$ denotes the probability of selecting a_i at time t, satisfying $\sum_{i=1}^{r} p_i(t) = 1$. $\widetilde{D}(t) = (\widetilde{d}_1(t), \ldots, \widetilde{d}_r(t))$ is the estimator vector, where $\widetilde{d}_i(t)$ denotes an estimate of the reward probability of a_i at time t.
5) L is a learning scheme to update Q and consists of the updating rules of $\widetilde{D}$ and P. It is defined by (1)–(4) as follows.

The estimate $\widetilde{d}_i(t)$ is calculated as

$$\widetilde{d}_i(t) = \frac{H_i(t)}{G_i(t)} \tag{1}$$

where $G_i(t)$ is the number of times that a_i has been selected up to time t, and $H_i(t)$ is the number of times that a_i has been rewarded up to time t.

Assume that a_m is the estimated optimal action. The action probability vector of DP$_{RI}$ is updated as follows [1], [11].

If $\beta(t) = 1$ then

$$p_i(t+1) = \max_{i \neq m}(p_i(t) - \Delta, 0) \quad \forall i \in N_r \tag{2}$$

$$p_m(t+1) = 1 - \sum_{i \neq m} p_i(t+1). \tag{3}$$

Else

$$p_i(t+1) = p_i(t) \quad \forall i \in N_r. \tag{4}$$

Endif

where $N_r = \{1, 2, \ldots, r\}$ is an index set. $\Delta = (1/rn)$ is the step size. n is the resolution parameter used to calculate Δ. The larger n is, the smaller Δ is.

III. Proposed Learning Automata

In this section, DEP$_{RI}$-AS is proposed for selecting an arbitrary action subset. It is based on a discretized pursuit learning scheme and reward-inaction learning paradigm. The proof of its ϵ-optimality property is presented.

A. DEP$_{RI}$-AS Learning Scheme

The problems focused in this article can be summarized as selecting an action subset that meets the required ranking. An index set of an arbitrary action subset is defined as $X = \{x_1, x_2, \ldots, x_{\widehat{r}}\}$, where $\widehat{r}$ is the number of actions in the arbitrary action subset and $x_i = \text{rank}_i\{D\}$ represents the index of the ith largest element in D. The largest element in D is defined as the action with the highest reward probability. Thus, the arbitrary action subset is $\widehat{A} = \{a_k | k \in X\}$. For example, selecting the top-k actions means that $X = \{1, \ldots, k\}$, selecting the best and worst actions means that $X = \{1, r\}$, and selecting the u-vth actions means that $X = \{u, u+1, \ldots, v\}$. The notations used are listed in Table II.

The proposed DEP$_{RI}$-AS is based on a discretized pursuit learning scheme and reward-inaction learning paradigm. $\widetilde{A}(t)$ is an estimation of $\widehat{A}$ at time t. In DEP$_{RI}$-AS, probabilities of actions not in $\widetilde{A}(t)$ are decreased by Δ and then the sum of probabilities of actions in $\widetilde{A}(t)$ is increased to keep $\sum_{i=1}^{r} p_i(t) = 1$. In order to better balance the probability of selecting actions to avoid devoting too many resources to actions that have been basically determined in $\widehat{A}$, DEP$_{RI}$-AS equalizes the probability of selecting each action in $\widetilde{A}(t)$. In this way, DEP$_{RI}$-AS can faster distinguish the actions in $\widehat{A}$. Mathematically, the update scheme of DEP$_{RI}$-AS adopts the reward-inaction paradigm to update $P(t)$ and can be formulized as follows.

If $\beta(t) = 1$ then

$$p_i(t+1) = \max(p_i(t) - \Delta, 0) \quad \forall i \notin \widetilde{X}(t) \tag{5}$$

$$p_i(t+1) = \frac{1}{\widehat{r}}\left(1 - \sum_{j \notin \widetilde{X}(t)} p_j(t+1)\right) \quad \forall i \in \widetilde{X}(t). \tag{6}$$

TABLE II
NOTATIONS USED IN DEP$_{RI}$-AS FOR SELECTING AN ARBITRARY SUBSET OF ACTIONS

n	Resolution parameter	r	Number of allowable actions
Δ	Step size	A	Action set
$a(t)$	Action selected at time t	B	Environmental response set
$\beta(t)$	Environment response at time t	D	Reward probability set
d_i	Reward probability of a_i	$Q(t)$	Automaton state at time t
$P(t)$	Action probability vector at time t	$p_i(t)$	Probability of selecting a_i at time t
$\widetilde{D}(t)$	Estimator vector at time t	$\widetilde{d}_i(t)$	Estimated reward probability of a_i at time t
$G_i(t)$	Number of times that a_i has been selected up to time t	$H_i(t)$	Number of times that a_i has been rewarded up to time t
L	Learning scheme	$\widehat{r}$	Number of actions in an arbitrary action subset
X	An index set of an arbitrary action subset	$\widehat{A}$	An arbitrary action subset
$\widetilde{X}(t)$	An index set of estimated arbitrary action subset at time t	$\widetilde{A}(t)$	Estimation of $\widehat{A}$ at time t
N_r	An index set $\{1, 2, ..., r\}$	T	Convergence threshold
I	number of iterations		

Else

$$p_i(t+1) = p_i(t) \quad \forall i \in N_r.$$

Endif.

In (5), LA penalizes the actions that are not in the estimated target action subset $\widetilde{A}(t)$ by reducing their probabilities of being selected. $p_i(t)$ is the probability of selecting action a_i at time t. Δ is the smallest step size, where r is the number of allowable actions and n is the resolution parameter. $\widetilde{X}(t) = \{l | l = \text{rank}_i\{\widetilde{D}(t)\}, i \in X\}$ is an index set of the estimated arbitrary action subset at time t. $\widetilde{A}(t) = \{a_k | k \in \widetilde{X}(t)\}$. The learning algorithm of DEP$_{RI}$-AS is realized by Algorithm 1. In particular, when selecting the optimal action, DEP$_{RI}$-AS reduces to DP$_{RI}$.

Traditional discretized learning schemes pursue only the optimal action, as shown in (2) and (3). This makes LA able to select the optimal action only and fails to solve the problem of selecting the action subset with the desired ranking. In contrast, the learning scheme of our proposed DEP$_{RI}$-AS pursues the currently estimated arbitrary action subset, which enables LA to select an arbitrary subset of actions, thus greatly generalizing LA's application scope. In each iteration, when the selected action is rewarded, traditional discretized learning schemes reward the optimal action by increasing its probability of being selected, as shown in (2) and (3). When pursuing an action subset rather than a single action, a simple way to extend the traditional schemes is to make all actions in the pursued set have an equal probability increasement. However, the probabilities of selecting each action in the pursued set are not equal and may even vary greatly. This results in the LA providing more resources to actions that have already been distinguished and less to those that are still uncertain, such that the LA requires more iterations to accurately distinguish all actions in a given arbitrary subset. To address this problem, in the learning scheme of our proposed DEP$_{RI}$-AS, the probabilities of selecting each action in the pursued set are equal such that they have an equal chance to be selected, thus improving the convergence speed.

Algorithm 1: DEP$_{RI}$-AS

Input: number of allowable actions r, resolution parameter n, action set A, environmental response set B, index set of an arbitrary action subset X, convergence threshold T.

Output: estimation of the arbitrary action subset $\widetilde{A}(t)$, number of iterations I.

BEGIN Initialize $p_i(0) = \frac{1}{r}$, $\forall i \in N_r$.

Initialize $\widetilde{d}_i(t)$ by sampling all actions a certain number of times. $I = 0$

Step 1: $I = I + 1$. At time t, $\alpha(t)$ is selected by $p(t)$. $\alpha(t) = \alpha_i$.

Step 2: Receive a response $\beta_i(t)$. $\widetilde{d}_i$ is updated by:

$$H_i(t+1) = H_i(t) + \beta_i(t)$$

$$G_i(t+1) = G_i(t) + 1$$

$$\widetilde{d}_i(t+1) = H_i(t+1)/G_i(t+1)$$

Step 3: Update $\widetilde{X}(t)$ according to $\widetilde{d}_i(t)$.

Step 4: **If** $\beta(t) = 0$ **Then**

$p_j(t+1) = p_j(t)$, $\forall j \in N_r$.

Goto Step 1.

Endif

Step 5: Update $p(t)$ using (5) and (6).

Step 6: **IF** $\sum_{i \in \widetilde{X}(t)} p_i(t) \geq T$, **Then**

converge to the estimation of the arbitrary action subset $\widetilde{A}(t) = \{a_k | k \in \widetilde{X}(t)\}$

Else *Goto* Step 1.

Endif

End

In each iteration, the operation of DEP$_{RI}$-AS consists of three phases: 1) selecting the next action and updating its estimated reward probability; 2) updating $\widetilde{X}(t)$; and 3) updating the action probability. The computational complexities of phases 1 and 3 are $\mathcal{O}(r)$ and that of phase 2 is $\mathcal{O}(\log r)$, where

r is the number of actions. Therefore, DEP$_{RI}$-AS's computational complexity is $\mathcal{O}(r)$, which is the same as that of other LA variants like DP$_{RI}$.

B. Proof of ϵ-Optimality

In this article, we prove the ϵ-optimality of DEP$_{RI}$-AS using a proof method similar to that proposed in [33], [34], and [35]. More detail can be found in the Supplementary File of this article [36].

IV. Simulation Results

In this section, the performance of DEP$_{RI}$-AS in selecting the arbitrary preferred subset is evaluated experimentally. If $\sum_{i\in\widetilde{X}(t)} p_i(t) \geq T$, LA is considered to have converged. Here, $T(0 < T \leq 1)$ is a threshold. If $\widetilde{A}(t) = \widehat{A}$, i.e., $\widetilde{A}(t)$ obtained from the estimated vector $\widetilde{D}$ is the same as the truly desired subset $\widehat{A}$, LA is considered to have converged correctly. For real applications, an automatic parameter tuning method can be adopted: the resolution parameter n is considered as the best one if it yields the fastest correct convergence in N_E experiments. Generally, when $N_E = 750$, the accuracy of LA is higher than 99.5% by tuning n in this way, where accuracy is the ratio of the number of correctly converged ones to the total number of experiments. Let $T = 0.999$ and $N_E = 750$.

For performance comparison, an unequal scheme called discretized unequal pursuit reward-inaction (DUP$_{RI}$) scheme is extended in a similar way to our proposed DEP$_{RI}$-AS from the one proposed in [24]. The main procedure of DUP$_{RI}$ is the same as Algorithm 1 except that $p_i(t)$ is updated using the following equations in step 5.

If $\beta(t) = 1$ then

$$p_i(t+1) = \max(p_i(t) - \Delta, 0) \quad \forall i \notin \widetilde{X}(t) \tag{7}$$

$$\overline{\Delta} = 1 - \sum_{i\in\widetilde{X}(t)} p_i(t) - \sum_{i\notin\widetilde{X}(t)} p_i(t+1) \tag{8}$$

$$p_i(t+1) = p_i(t) + \frac{\overline{\Delta}}{\widetilde{r}} \quad \forall i \in \widetilde{X}(t). \tag{9}$$

Endif.

Like DEP$_{RI}$-AS, in DUP$_{RI}$, probabilities of actions that are not in $\widetilde{A}(t)$ are decreased by Δ. However, DUP$_{RI}$ only ensures that the probability increasements of actions in $\widetilde{A}(t)$ are equal, but fails to ensure that the probabilities of actions in $\widetilde{A}(t)$ are equal after the update.

Our experiments mainly involve the following five typical types of selection problems: 1) selecting top-k actions; 2) the best and worst actions; 3) actions in a given rank range; 4) actions with odd ranking; and 5) those with a more irregular distribution of rankings. Each algorithm samples all actions ten times to initialize $\widetilde{d}_i(t)$ and these additional iterations are included in the experimental results [37]. Each algorithm runs 100 000 times in each environment. We evaluate the performance of algorithms by presenting their convergence speed, which is measured by the average number of iterations required for convergence.

TABLE III
Reward Probability of Environments for Selecting Top-3 Actions

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}
E_1	**0.55**	**0.50**	**0.45**	0.40	0.35	0.30	0.25	0.20	0.15	0.10
E_2	**0.60**	**0.55**	**0.50**	0.40	0.35	0.30	0.25	0.20	0.15	0.10
E_3	**0.65**	**0.60**	**0.55**	0.40	0.35	0.30	0.25	0.20	0.15	0.10
E_4	**0.71**	**0.68**	**0.65**	0.40	0.35	0.30	0.25	0.20	0.15	0.10
E_5	**0.75**	**0.70**	**0.65**	0.40	0.35	0.30	0.25	0.20	0.15	0.10
E_6	**0.85**	**0.75**	**0.65**	0.40	0.35	0.30	0.25	0.20	0.15	0.10

TABLE IV
Convergence Speed of Contenders for Selecting Top-3 Actions

	DEP$_{RI}$-AS		DUP$_{RI}$	
	n	$\widetilde{I}$	n	$\widetilde{I}$
E_1	4257	**13481**	5158	14976
E_2	1174	**3539**	1400	3888
E_3	544	**1579**	675	1793
E_4	206	**584**	245	643
E_5	197	**547**	239	610
E_6	210	**537**	247	588

A. Top-k Actions

The first kind of experiment focuses on the problem that learns the action subset of top-k actions. Let $k = 3$, i.e., $X = \{1, 2, 3\}$. Six benchmark environments E_1–E_6 with ten actions (d_1–d_{10}) are used to evaluate the performance of DEP$_{RI}$-AS and DUP$_{RI}$. The actions' reward probabilities of these environments are presented in Table III. When LA works in E_1 and selects action d_1, its reward probability is 0.55, which means that it has a 55% probability of being rewarded and a 45% probability of being penalized. Top-3 actions are in boldface.

Table IV presents the convergence speed for selecting top-3 actions. "$\widetilde{I}$" denotes the convergence speed and is measured by the average number of iterations required for convergence in 100 000 experiments. The best result between DEP$_{RI}$-AS and DUP$_{RI}$ is in boldface. Fig. 3 shows the mean of the action probability $E_p = E\{\sum_{i\in\widetilde{X}(t)} p_i(t)\}$ versus iteration count, where t is the number of iterations. LA is considered to have converged if $\sum_{i\in\widetilde{X}(t)} p_i(t) \geq T$. Therefore, the larger E_p, the faster the convergence. It can be seen that DEP$_{RI}$-AS requires fewer iterations for convergence than DUP$_{RI}$ in all benchmark environments, and its convergence speed is faster than DUP$_{RI}$ by 8.7%–11.1%. DEP$_{RI}$-AS has a stronger ability to distinguish the actions in the target subset and can select top-3 actions faster than DUP$_{RI}$ under the same accuracy. By comparing the experimental results of E_1–E_3, it can be found that as the minimum difference between the top-3 action and other actions decreases, the selection difficulty increases and the convergence speed decreases. In E_4–E_6, the minimum difference between top-3 actions and other actions is the same, while the difference among top-3 actions is different. The experimental results show that the larger the difference among top-3 actions, the faster the convergence speed of DEP$_{RI}$-AS.

B. Best and Worst Actions

The second kind of experiment focuses on the problem of selecting the best and worst actions, i.e., $X = \{1, 10\}$.

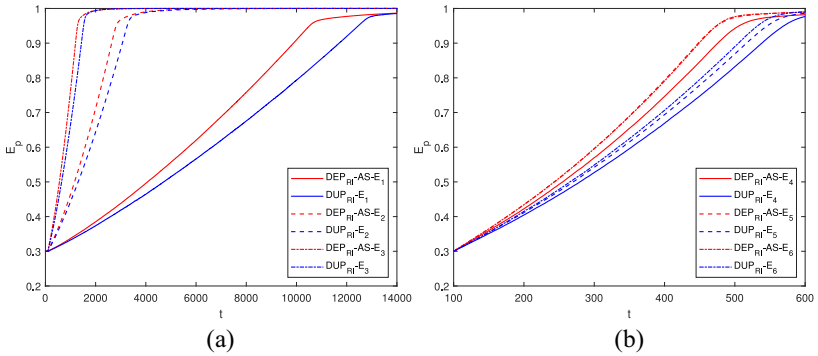


Fig. 3. Mean of the action probability $E_p = E\{\sum_{i\in\widetilde{X}(t)} p_i(t)\}$ versus t for selecting top-k actions. (a) E_1–E_3. (b) E_4–E_6.

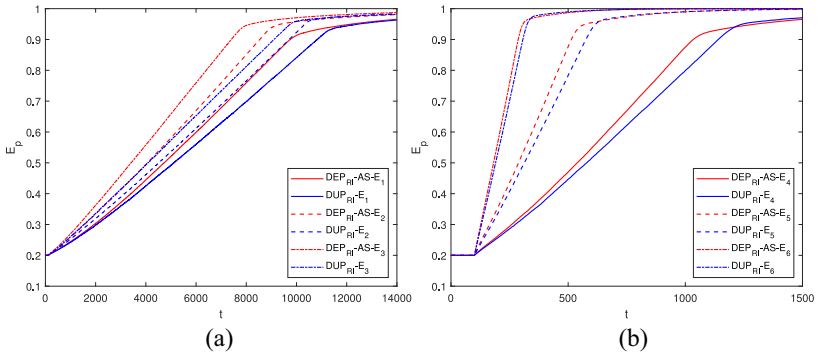


Fig. 4. Mean of the action probability $E_p = E\{\sum_{i\in\widetilde{X}(t)} p_i(t)\}$ versus t for selecting the best and worst actions. (a) E_1–E_3. (b) E_4–E_6.

TABLE V
REWARD PROBABILITY OF ENVIRONMENTS FOR SELECTING THE BEST AND WORST ACTIONS

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}
E_1	**0.75**	0.70	0.65	0.60	0.55	0.50	0.45	0.40	0.35	**0.30**
E_2	**0.85**	0.70	0.65	0.60	0.55	0.50	0.45	0.40	0.35	**0.30**
E_3	**0.75**	0.70	0.65	0.60	0.55	0.50	0.45	0.40	0.35	**0.20**
E_4	**0.85**	0.70	0.65	0.60	0.55	0.50	0.45	0.40	0.35	**0.20**
E_5	**0.90**	0.70	0.65	0.60	0.55	0.50	0.45	0.40	0.35	**0.15**
E_6	**0.95**	0.70	0.65	0.60	0.55	0.50	0.45	0.40	0.35	**0.10**

TABLE VI
CONVERGENCE SPEED OF CONTENDERS FOR SELECTING THE BEST AND WORST ACTIONS

	DEP$_{RI}$-AS		DUP$_{RI}$	
	n	$\widetilde{I}$	n	$\widetilde{I}$
E_1	5113	**16070**	5859	16887
E_2	4872	**13029**	5764	13707
E_3	3874	**11806**	4797	13432
E_4	497	**1587**	581	1674
E_5	229	**746**	270	791
E_6	106	**368**	120	381

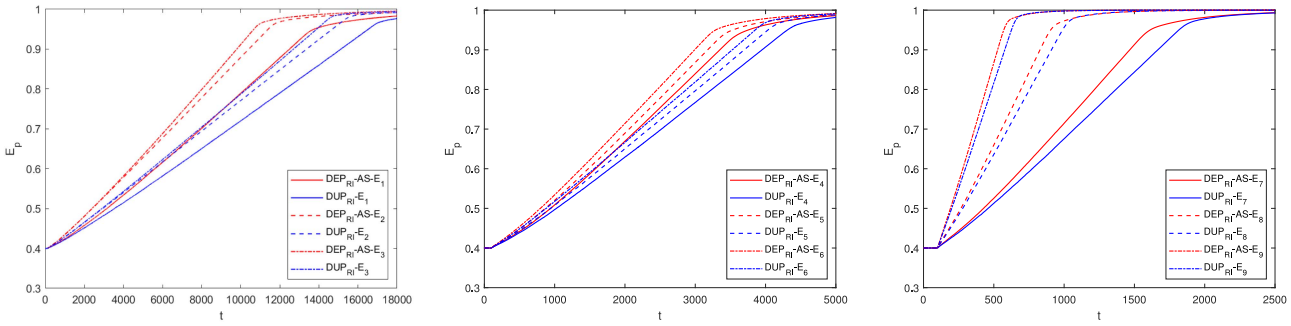
The actions' probabilities of six benchmark test environments are presented in Table V. The best and worst actions are in boldface.

Table VI presents the convergence speed for selecting the best and worst actions. Fig. 4 shows the convergence curves. It can be seen that DEP$_{RI}$-AS requires fewer iterations for convergence than DUP$_{RI}$ in all environments, and its convergence speed is faster than DUP$_{RI}$ by 3.4%–12.1%. We denote $l_1 = d_1 - d_2$ and $l_2 = d_9 - d_{10}$. In E_1–E_3, $\min\{l_1, l_2\} = 0.05$ is the same, while l_1 is larger in E_2 and l_2 is larger in E_3. Even for two required actions with the same reward probability difference from their adjacent remaining actions, DEP$_{RI}$-AS has a stronger ability to select the ones with higher reward probability since the actions with low reward probability are more affected by the stochastic environment and more difficult to be distinguished, as shown in Fig. 4(a). In E_4–E_6, l_1 and l_2 gradually increase. For example, in E_4, $l_1 = 0.15$ and $l_2 = 0.15$, while in E_6, $l_1 = 0.25$ and $l_2 = 0.25$. The increase of l_1 and l_2 decreases the difficulty of distinguishing the best and worst actions. Therefore, compared with E_4, LA can use a larger step size Δ in E_6 to achieve the same accuracy. The sum of probabilities of actions in the estimated arbitrary action subset $\widetilde{A}(t)$ is $\sum_{i\in\widetilde{X}(t)} p_i(t)$. According to (5) and (6), using a larger Δ speeds up the increase of $\sum_{i\in\widetilde{X}(t)} p_i(t)$. Thus, DEP$_{RI}$-AS converges faster in E_6, and the average number of iterations required for its convergence decreases, as shown in Fig. 4(b).

C. Actions in Given Range

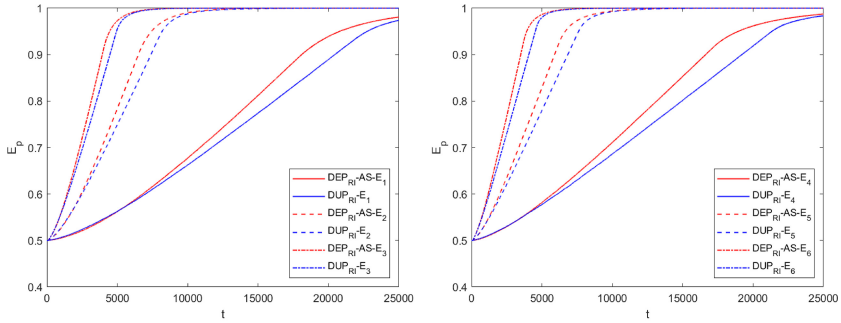
The third kind of experiment focuses on the problem of selecting actions in a given range, i.e., $X = \{4, 5, 6, 7\}$. The actions' probabilities of nine benchmark test environments are presented in Table VII. Actions that rank between the 4th and 7th actions are in boldface.

Table VIII presents the convergence speed for selecting actions that rank between the 4th and 7th actions. Fig. 5 shows the convergence curves. It can be seen that DEP$_{RI}$-AS requires



(a) (b) (c)

Fig. 5. Mean of the action probability $E_p = E\{\sum_{i\in\widetilde{X}(t)} p_i(t)\}$ versus t for selecting actions in a given range. (a) E_1–E_3. (b) E_4–E_6. (c) E_7–E_9.



(a) (b)

Fig. 6. Mean of the action probability $E_p = E\{\sum_{i\in\widetilde{X}(t)} p_i(t)\}$ versus t for selecting actions with odd ranking and actions with an irregular distribution of rankings. (a) E_1–E_3. (b) E_4–E_6.

TABLE VII
REWARD PROBABILITY OF ENVIRONMENTS FOR SELECTING ACTIONS THAT RANK BETWEEN THE 4TH AND 7TH ACTIONS

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}
E_1	0.75	0.70	0.65	**0.60**	**0.55**	**0.50**	**0.45**	0.40	0.35	0.30
E_2	0.85	0.80	0.75	**0.60**	**0.55**	**0.50**	**0.45**	0.40	0.35	0.30
E_3	0.75	0.70	0.65	**0.60**	**0.55**	**0.50**	**0.45**	0.30	0.25	0.20
E_4	0.76	0.71	0.66	**0.56**	**0.53**	**0.50**	**0.47**	0.37	0.32	0.27
E_5	0.80	0.75	0.70	**0.60**	**0.55**	**0.50**	**0.45**	0.35	0.30	0.25
E_6	0.86	0.81	0.76	**0.66**	**0.58**	**0.50**	**0.42**	0.32	0.27	0.22
E_7	0.85	0.80	0.75	**0.60**	**0.55**	**0.50**	**0.45**	0.30	0.25	0.20
E_8	0.90	0.85	0.80	**0.60**	**0.55**	**0.50**	**0.45**	0.25	0.20	0.15
E_9	0.95	0.90	0.85	**0.60**	**0.55**	**0.50**	**0.45**	0.20	0.15	0.10

TABLE VIII
CONVERGENCE SPEED OF CONTENDERS FOR SELECTING ACTIONS THAT RANK BETWEEN THE 4TH AND 7TH ACTIONS

	DEP$_{RI}$-AS		DUP$_{RI}$	
	n	$\widetilde{I}$	n	$\widetilde{I}$
E_1	7022	**18598**	8854	20882
E_2	6207	**15202**	8297	18078
E_3	5526	**13936**	7415	16892
E_4	1779	**4993**	2201	5571
E_5	1733	**4684**	2114	5146
E_6	1697	**4378**	2091	4842
E_7	782	**2122**	926	2289
E_8	425	**1157**	486	1220
E_9	255	**696**	292	741

fewer iterations for convergence than DUP$_{RI}$ in all environments, and its convergence speed is faster than DUP$_{RI}$ by 5.2%–17.5%. We denote $l_3 = d_3 - d_4$ and $l_4 = d_7 - d_8$. In E_1–E_3, $\min\{l_3, l_4\}$ is the same, while l_3 is larger in E_2 and l_4 is larger in E_3. The experimental results show that DEP$_{RI}$-AS has a stronger ability to select actions with higher probability. In E_4–E_6, l_3 and l_4 are the same, while the difference among actions in $\widehat{A}$ is different. The experimental results show that the larger the difference among actions in $\widehat{A}$, the faster the convergence speed of DEP$_{RI}$-AS. Comparing the experimental results of E_7–E_9, it can be found that as l_3 and l_4 increase, the selection difficulty decreases and the convergence speed of DEP$_{RI}$-AS increases.

D. Actions With Odd Ranking and Actions With Irregular Distribution of Rankings

Next, we focus on the problem of selecting actions with odd ranking and actions with an irregular distribution of rankings, i.e., $X = \{1, 3, 5, 7, 9\}$ and $X = \{1, 4, 5, 8, 10\}$. The actions' probabilities of six benchmark test environments are presented in Table IX, where E_1–E_3 are environments for selecting actions with odd ranking and E_4–E_6 are environments for selecting actions with an irregular distribution of rankings. The desired actions are in boldface.

Table X presents the convergence speed of DEP$_{RI}$-AS and DUP$_{RI}$. "$\widetilde{I}$" denotes the convergence speed and is measured by the average number of iterations required for convergence in 100 000 experiments. The best result between DEP$_{RI}$-AS and DUP$_{RI}$ is in boldface. Fig. 6 shows the convergence curves. It can be seen that DEP$_{RI}$-AS requires fewer iterations for convergence than DUP$_{RI}$ in all environments, and its convergence

TABLE IX
REWARD PROBABILITY OF ENVIRONMENTS FOR SELECTING ACTIONS WITH ODD RANKING OR ACTIONS WITH AN IRREGULAR DISTRIBUTION OF RANKINGS

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}
E_1	**0.75**	0.70	**0.65**	0.60	**0.55**	0.50	**0.45**	0.40	**0.35**	0.30
E_2	**0.87**	0.79	**0.71**	0.63	**0.55**	0.47	**0.39**	0.31	**0.23**	0.15
E_3	**0.95**	0.85	**0.75**	0.65	**0.55**	0.45	**0.35**	0.25	**0.15**	0.05
E_4	**0.75**	0.70	0.65	**0.60**	**0.55**	0.50	0.45	**0.40**	0.35	**0.30**
E_5	**0.87**	0.79	0.71	**0.63**	**0.55**	0.47	0.39	**0.31**	0.23	**0.15**
E_6	**0.95**	0.85	0.75	**0.65**	**0.55**	0.45	0.35	**0.25**	0.15	**0.05**

TABLE X
CONVERGENCE SPEED OF CONTENDERS FOR SELECTING ACTIONS WITH ODD RANKING OR ACTIONS WITH AN IRREGULAR DISTRIBUTION OF RANKINGS

	DEP$_{RI}$-AS		DUP$_{RI}$	
	n	$\widetilde{I}$	n	$\widetilde{I}$
E_1	9484	**26702**	11706	29080
E_2	3410	**9603**	4215	10459
E_3	2085	**5815**	2547	6286
E_4	8874	**24826**	11059	27309
E_5	3096	**8794**	3830	9657
E_6	1815	**5169**	2256	5707

speed is faster than DUP$_{RI}$ by 7.5%–9.4%. In each environment, the difference between the reward probability of each action and its adjacent action, i.e., d_i-d_{i+1} $\forall i \in \{1, 2, \ldots, 9\}$, is the same and is defined as l_d. By comparing the experimental results of selecting actions with odd ranking in E_1–E_3, it can be found that as l_d increases, the selection difficulty decreases and the convergence speed increases. The reward probabilities of actions in E_4–E_6 are the same as those in E_1–E_3. In E_4–E_6, as l_d increases, the difficulty of selecting actions with an irregular distribution of rankings decreases, and the convergence speed increases.

E. Parameter Analysis

In this section, the influence of the resolution parameter n on DEP$_{RI}$-AS's performance is analyzed. Table XI presents the environments of three typical types of selection problems. E_1 is the environment of selecting top-3 actions, E_2 is the environment of selecting the best and worst actions, and E_3 is the environment of selecting actions that rank between the 4th and 7th actions. The boldface indicates the required actions.

Table XII presents the convergence speed and accuracy using different resolution parameters n, where $\widetilde{I}$ is the average number of iterations required for convergence and $\widetilde{A}$ is the accuracy. A larger n leads to a smaller step size Δ. In the environments of all three tasks, with the increase of n, the step size Δ decreases, which improves the accuracy and reduces the convergence speed. Therefore, DEP$_{RI}$-AS can converge with an arbitrarily high probability as n increases but requires more iterations to converge.

F. Scalability Analysis

In this section, the scalability of the proposed DEP$_{RI}$-AS is evaluated. In the following experiments, an environment containing 20 actions is considered. The actions' reward probabilities are presented in Table XIII. The experiments involve the following five typical types of selection problems: 1) selecting top-3 actions $X = \{1, 2, 3\}$; 2) the best and worst actions $X = \{1, 20\}$; 3) the 4–7th actions $X = \{4, 5, 6, 7\}$; 4) actions with odd ranking ($X = \{1, 3, 5, 7, 9, 11, 13, 15, 17, 19\}$); and 5) actions with an irregular distribution of rankings ($X = \{1, 4, 5, 8, 10, 11, 13, 16, 17, 20\}$).

TABLE XI
REWARD PROBABILITY OF ENVIRONMENTS FOR PARAMETER ANALYSIS

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}
E_1	**0.60**	**0.55**	**0.50**	0.40	0.35	0.30	0.25	0.20	0.15	0.10
E_2	**0.85**	0.70	0.65	0.60	0.55	0.50	0.45	0.40	0.35	**0.20**
E_3	0.80	0.75	0.70	**0.60**	**0.55**	**0.50**	**0.45**	0.35	0.30	0.25

TABLE XII
CONVERGENCE SPEED AND ACCURACY FOR PARAMETER ANALYSIS

E_1	n	974	1074	1174	1274	1374
	$\widetilde{I}$	3071	3314	3539	3777	4007
	$\widetilde{A}$	99.742%	99.801%	99.849%	99.894%	99.912%
E_2	n	297	397	497	597	697
	$\widetilde{I}$	1152	1376	1587	1793	1994
	$\widetilde{A}$	99.264%	99.668%	99.844%	99.930%	99.955%
E_3	n	1533	1633	1733	1833	1933
	$\widetilde{I}$	4293	4479	4684	4878	5077
	$\widetilde{A}$	99.737%	99.826%	99.852%	99.880%	99.900%

TABLE XIII
REWARD PROBABILITY OF A LARGE ACTION SPACE

d_1	d_2	d_3	d_4	d_5
0.95	0.90	0.85	0.80	0.75
d_6	d_7	d_8	d_9	d_{10}
0.70	0.65	0.60	0.55	0.50
d_{11}	d_{12}	d_{13}	d_{14}	d_{15}
0.45	0.40	0.35	0.30	0.25
d_{16}	d_{17}	d_{18}	d_{19}	d_{20}
0.20	0.15	0.10	0.05	0.00

TABLE XIV
CONVERGENCE SPEED AND ACCURACY OF CONTENDERS IN A LARGE ACTION SPACE

	DEP$_{RI}$-AS			DUP$_{RI}$		
	n	$\widetilde{I}$	$\widetilde{A}$	n	$\widetilde{I}$	$\widetilde{A}$
Top-3	7142	**15570**	99.885%	8366	16308	99.862%
B-W	2333	**8789**	99.896%	2479	8871	99.884%
4-7th	11784	**30029**	99.883%	13886	31389	99.872%
Odd	17687	**57119**	99.885%	22367	61816	99.873%
Irr	16327	**53248**	99.864%	20435	57102	99.851%

Table XIV shows the convergence speed and accuracy of DEP$_{RI}$-AS and DUP$_{RI}$. "Top-3," "B-W," "4-7th," "Odd," and "Irr" denote the five problems mentioned above, respectively. In all cases, the accuracy of both algorithms is higher than 99.8%. DEP$_{RI}$-AS requires fewer iterations for convergence than DUP$_{RI}$, and its convergence speed is faster than DUP$_{RI}$ by 0.9%–7.6%. Therefore, the proposed DEP$_{RI}$-AS can effectively deal with the large action space, and its performance is better than DUP$_{RI}$ in all experiments.

G. Real-World Environment

With the rapid development of information technology, cloud manufacturing becomes a new networked manufacturing model [38]. It collects manufacturing resources in a virtual

TABLE XV
PASS RATES OF MACHINING EQUIPMENTS

d_1	d_2	d_3	d_4	d_5	d_6
0.97	0.81	0.90	0.86	0.94	0.78
d_7	d_8	d_9	d_{10}	d_{11}	d_{12}
0.99	0.87	0.78	0.79	0.87	0.98

TABLE XVI
CONVERGENCE SPEED AND ACCURACY OF CONTENDERS FOR SELECTING MACHINING EQUIPMENTS

	DEP$_{RI}$-AS			DUP$_{RI}$		
	n	$\widetilde{I}$	$\widetilde{A}$	n	$\widetilde{I}$	$\widetilde{A}$
1-4th	6005	**8818**	99.886%	7143	9456	99.876%
5-8th	7641	**13551**	99.866%	8997	14454	99.844%
9-12th	7052	**13011**	99.874%	8487	13936	99.853%

resource pool to provide manufacturing services for different types of users. As one of the most important manufacturing resources, machining equipment is widely used in automobile, aerospace, and other fields. In order to meet the needs of users in different fields, machining equipment needs to be classified and selected according to its performance, such as pass rate.

In this section, DEP$_{RI}$-AS is applied to the selection of machining equipment in cloud manufacturing to evaluate the proposed method. In the following experiments, a resource pool containing 12 machines is considered. Their pass rates (i.e., reward probabilities) are presented in Table XV. The purpose of the task is to classify the 12 machines into three groups according to their performances, i.e., to select the best 4 ($X = \{1, 2, 3, 4\}$), the worst 4 ($X = \{9, 10, 11, 12\}$), and the remaining ones ($X = \{5, 6, 7, 8\}$).

Table XVI presents the convergence speed and accuracy for selecting various subsets of machines. It can be seen that the accuracy of both algorithms is higher than 99.8% in all tasks of selecting machines with different performance rankings. In addition, DEP$_{RI}$-AS requires fewer iterations for convergence than DUP$_{RI}$ in all tasks, and its convergence speed is faster than DUP$_{RI}$ by 6.2%–6.7%. Therefore, the proposed DEP$_{RI}$-AS can effectively deal with the application of selecting machines in cloud manufacturing, and its performance is better than DUP$_{RI}$ in all experiments.

V. CONCLUSION

Based on the discretized pursuit learning scheme, this article presents a discretized equal pursuit reward-inaction algorithm. The proposed algorithm tackles problems of selecting an arbitrary subset of actions that have never been considered before in the LA field, e.g., selecting an optimal subset, the best and worst actions, or those in a given range. It pursues the currently estimated arbitrary action subset and makes the probabilities of selecting each action in the currently estimated subset equal to increase the convergence speed. This article also presents the proof of DEP$_{RI}$-AS's ϵ-optimality property. Simulation results demonstrate its power for selecting an arbitrary action subset. The proposed algorithm is suitable for the *P*-model environment, but it cannot be applied to environments with more complex responses, such as *Q*-model and *S*-model environments. Our future work plans to expand the application environment of the proposed algorithm, improve the accuracy and convergence speed of LA in selecting an arbitrary action subset, and intend to apply it to complex optimization problems and real applications [39], [40], [41], [42], [43], [44], [45], [46].

REFERENCES

[1] B. J. Oommen and J. K. Lanctot, "Discretized pursuit learning automata," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, no. 4, pp. 931–938, Jul./Aug. 1990.

[2] A. Wheeldon, R. Shafik, T. Rahman, J. Lei, A. Yakovlev, and O.-C. Granmo, "Learning automata based energy-efficient AI hardware design for IoT applications," *Philos. Trans. Royal Soc. A*, vol. 378, no. 2182, 2020, Art. no. 20190593.

[3] X. Deng et al. "Learning-automata-based confident information coverage barriers for smart ocean Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9919–9929, Oct. 2020.

[4] A. Saleem, M. K. Afzal, M. Ateeq, S. W. Kim, and Y. B. Zikria, "Intelligent learning automata-based objective function in RPL for IoT," *Sustain. Cities Soc.*, vol. 59, Aug. 2020, Art. no. 102234.

[5] C. Dey, R. Bose, K. K. Ghosh, S. Malakar, and R. Sarkar, "LAGOA: Learning automata based grasshopper optimization algorithm for feature selection in disease datasets," *J. Ambient Intell. Humanized Comput.*, vol. 13, no. 6, pp. 3175–3194, 2022.

[6] S. Barshandeh, R. Dana, and P. Eskandarian, "A learning automata-based hybrid MPA and JS algorithm for numerical optimization problems and its application on data clustering," *Knowl.-Based Syst.*, vol. 236, Jan. 2022, Art. no. 107682.

[7] V. Varshavskii and I. Vorontsova, "On the behavior of stochastic automata with a variable structure," *Avtomatika Telemekhanika*, vol. 24, no. 3, pp. 353–360, 1963.

[8] M. Thathachar and O. John, "Discretized reward-inaction learning automata," *J. Cybern. Inf. Sci.*, vol. 2, pp. 24–29, 1979.

[9] M. Thathachar and P. Sastry, "Estimator algorithms for learning automata," presented at Platinum Jubilee Conf. Syst. Signal Process., 1986.

[10] M. A. L. Thathachar and P. S. Sastry, "A class of rapidly converging algorithms for learning automata," in *Proc. IEEE Int. Conf. Cybern. Soc.*, 1984, pp. 602–606.

[11] B. J. Oommen and M. Agache, "Continuous and discretized pursuit learning schemes: Various algorithms and their comparison," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 31, no. 3, pp. 277–287, Jun. 2001.

[12] M. Agache and B. J. Oommen, "Generalized pursuit learning schemes: New families of continuous and discretized learning automata," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 32, no. 6, pp. 738–749, Dec. 2002.

[13] J. Zhang, C. Wang, and M. Zhou, "Last-position elimination-based learning automata," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2484–2492, Dec. 2014.

[14] J. Zhang, C. Wang, and M. Zhou, "Fast and epsilon-optimal discretized pursuit learning automata," *IEEE Trans. Cybern.*, vol. 45, no. 10, pp. 2089–2099, Oct. 2015.

[15] W. Jiang, B. Li, S. Li, Y. Tang, and C. L. P. Chen, "A new prospective for learning automata: A machine learning approach," *Neurocomputing*, vol. 188, pp. 319–325, May 2016.

[16] J. Zhang, C. Wang, D. Zang, and M. Zhou, "Incorporation of optimal computing budget allocation for ordinal optimization into learning automata," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 1008–1017, Apr. 2016.

[17] H. Ge, J. Li, S. Li, W. Jiang, and Y. Wang, "A novel parallel framework for pursuit learning schemes," *Neurocomputing*, vol. 228, pp. 198–204, Mar. 2017.

[18] A. Yazidi, X. Zhang, L. Jiao, and B. J. Oommen, "The hierarchical continuous pursuit learning automation: A novel scheme for environments with large numbers of actions," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 2, pp. 512–526, Feb. 2020.

[19] Y. Guo and S. Li, "A non-Monte-Carlo parameter-free learning automata scheme based on two categories of statistics," *IEEE Trans. Cybern.*, vol. 49, no. 12, pp. 4153–4166, Dec. 2019.

[20] C. Di, Q. Liang, F. Li, S. Li, and F. Luo, "An efficient parameter-free learning automaton scheme," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 11, pp. 4849–4863, Nov. 2021.

[21] A. Yazidi, N. Bouhmala, and M. Goodwin, "A team of pursuit learning automata for solving deterministic optimization problems," *Appl. Intell.*, vol. 50, no. 9, pp. 2916–2931, 2020.
[22] C. Di, F. Li, S. Li, and J. Tian, "Bayesian inference based learning automaton scheme in Q-model environments," *Appl. Intell.*, vol. 51, no. 10, pp. 7453–7468, 2021.
[23] Y. Guo, C. Di, and S. Li, "A novel reduced parameter S-model of estimator learning automata in the switching non-stationary environment," *Neural Comput. Appl.*, vol. 34, pp. 6811–6824, Mar. 2022.
[24] J. Zhang, Z. Li, Q. Kang, and M. Zhou, "A new class of learning automata for selecting an optimal subset," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, 2014, pp. 3429–3434.
[25] X. Guo, W. Jiang, H. Ge, and S. Li, "A new learning automata algorithm for selection of optimal subset," in *Proc. Int. Conf. Intell. Comput.*, 2015, pp. 693–702.
[26] S. Shen, M. S. Ausin, B. Mostafavi, and M. Chi, "Improving learning & reducing time: A constrained action-based reinforcement learning approach," in *Proc. 26th Conf. User Model. Adapt. Personalization*, 2018, pp. 43–51.
[27] J. Garcıa and F. Fernández, "A comprehensive survey on safe reinforcement learning," *J. Mach. Learn. Res.*, vol. 16, no. 1, pp. 1437–1480, 2015.
[28] Z. Zhang, D. Wang, and J. Gao, "Learning automata-based multiagent reinforcement learning for optimization of cooperative tasks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 10, pp. 4639–4652, Oct. 2021.
[29] Z. Yang, Y. Liu, Y. Chen, and L. Jiao, "Learning automata based Q-learning for content placement in cooperative caching," *IEEE Trans. Commun.*, vol. 68, no. 6, pp. 3667–3680, Jun. 2020.
[30] K. Asghari, M. Masdari, F. S. Gharehchopogh, and R. Saneifard, "A fixed structure learning automata-based optimization algorithm for structure learning of Bayesian networks," *Expert Syst.*, vol. 38, no. 7, 2021, Art. no. e12734.
[31] B. H. Abed-Alguni and M. A. Ottom, "Double delayed Q-learning," *Int. J. Artif. Intell.*, vol. 16, no. 2, pp. 41–59, 2018.
[32] B. H. Abed-Alguni, "Action-selection method for reinforcement learning based on cuckoo search algorithm," *Arab. J. Sci. Eng.*, vol. 43, no. 12, pp. 6771–6785, 2018.
[33] X. Zhang, O.-C. Granmo, B. J. Oommen, and L. Jiao, "On using the theory of regular functions to prove the ε-optimality of the continuous pursuit learning automaton," in *Proc. Int. Conf. Ind., Eng. Appl. Appl. Intell. Syst.*, 2013, pp. 262–271.
[34] X. Zhang, B. J. Oommen, O.-C. Granmo, and L. Jiao, "Using the theory of regular functions to formally prove the ε-optimality of discretized pursuit learning algorithms," in *Proc. Int. Conf. Ind., Eng. Appl. Appl. Intell. Syst.*, 2014, pp. 379–388.
[35] X. Zhang, O.-C. Granmo, B. J. Oommen, and L. Jiao, "A formal proof of the ε-optimality of absorbing continuous pursuit algorithms using the theory of regular functions," *Appl. Intell.*, vol. 41, no. 3, pp. 974–985, 2014.
[36] [Online]. Available: https://github.com/windycoffee/SF-DEPRI
[37] Y. Tang, X. Li, Y. Zhang, X. Xia, and L. Gui, "Dynamic multi-swarm global particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2019, pp. 1030–1037.
[38] S.-L. Wang, L. Guo, L. Kang, C.-S. Li, X.-Y. Li, and Y. M. Stephane, "Research on selection strategy of machining equipment in cloud manufacturing," *Int. J. Adv. Manuf. Technol.*, vol. 71, no. 9, pp. 1549–1563, 2014.
[39] Y. Xia, M. Zhou, X. Luo, Q. Zhu, J. Li, and Y. Huang, "Stochastic modeling and quality evaluation of infrastructure-as-a-service clouds," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 1, pp. 162–170, Jan. 2015.
[40] G. Tian, H. Zhang, M. Zhou, and Z. Li, "AHP, gray correlation, and TOPSIS combined approach to green performance evaluation of design alternatives," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 7, pp. 1093–1105, Jul. 2018.
[41] W.-N. Chen, D.-Z. Tan, Q. Yang, T. Gu, and J. Zhang, "Ant colony optimization for the control of pollutant spreading on social networks," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 4053–4065, Sep. 2020.
[42] M. Ghahramani, Y. Qiao, M. C. Zhou, A. O'Hagan, and J. Sweeney, "AI-based modeling and data-driven evaluation for smart manufacturing processes," *IEEE/CAA J. Automatica Sinica*, vol. 7, no. 4, pp. 1026–1037, Jul. 2020.
[43] X. Guo, M. Zhou, A. Abusorrah, F. Alsokhiry, and K. Sedraoui, "Disassembly sequence planning: A survey," *IEEE/CAA J. Automatica Sinica*, vol. 8, no. 7, pp. 1308–1324, Jul. 2021.
[44] Z. Liu, N. Wu, Y. Qiao, and Z. Li, "Performance evaluation of public bus transportation by using DEA models and Shannon's entropy: An example from a company in a large city of China," *IEEE/CAA J. Automatica Sinica*, vol. 8, no. 4, pp. 779–795, Apr. 2021.
[45] X. Luo, W. Qin, A. Dong, K. Sedraoui, and M. Zhou, "Efficient and high-quality recommendations via momentum-incorporated parallel stochastic gradient descent-based learning," *IEEE/CAA J. Automatica Sinica*, vol. 8, no. 2, pp. 402–411, Feb. 2021.
[46] Y. Tang, et al., "A personalized learning system for parallel intelligent education," *IEEE Trans. Comput. Soc. Syst.*, vol. 7, no. 2, pp. 352–361, 2020.

Junqi Zhang (Senior Member, IEEE) received the Ph.D. degree in computing science from Fudan University, Shanghai, China, in 2007.

He was a Postdoctoral Research Fellow with Peking University, Beijing, China, in 2007. From 2014 to 2015, he was a Visiting Scholar with the New Jersey Institute of Technology, Newark, NJ, USA. He is currently a Full Professor with Tongji University, Shanghai. He has published 10+ papers in IEEE TRANSACTIONS and 30+ papers in conferences. His current research interests include swarm intelligence, swarm robots, multiagent systems, learning automata, reinforcement learning, and big data.

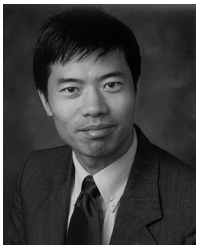
Prof. Zhang was a recipient of the Outstanding Postdoctoral Award with Peking University in 2007.

Peng Zu received the B.S. degree in computer science and technology from Tongji University, Shanghai, China, in 2021, where he is currently pursuing the M.S. degree in computer science and technology with the Department of Computer Science and Technology.

His current research interests include swarm robots, multiagent systems, and learning automaton and its applications.

PengZhan Qiu received the B.S. degree in computer science and technology from the China University of Mining and Technology, Xuzhou, China, in 2019. He is currently pursuing the M.S. degree in computer science and technology with Tongji University, Shanghai, China.

His current research interests include evolutionary computation, learning automaton and its applications, and optimal computing budget allocation.



MengChu Zhou (Fellow, IEEE) received the B.S. degree in control engineering from the Nanjing University of Science and Technology, Nanjing, China, in 1983, the M.S. degree in automatic control from the Beijing Institute of Technology, Beijing, China, in 1986, and the Ph.D. degree in computer and systems engineering from Rensselaer Polytechnic Institute, Troy, NY, USA, in 1990.

In 1990, he joined the New Jersey Institute of Technology, Newark, NJ, USA, where he is currently a Distinguished Professor. He has over 1100 publications, including 12 books, 750+ journal papers (600+ in IEEE TRANSACTIONS), 31 patents, and 32 book chapters. His interests are in Petri nets, automation, Internet of Things, and big data.

Dr. Zhou is a Fellow of IFAC, AAAS, CAA, and NAI.