[3] R. C. Gonzalez and P. Wintz, *Digital Image Processing*, Reading, MA: Addison-Wesley, 1977, pp. 119-127.
[4] R. M. Haralick, "Statistical and structural approaches to texture," in *Proc. IEEE*, vol. 67, pp. 786-804, 1979.
[5] M. Unser, "Sum and difference histograms for texture classification," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8, pp. 118-125, Jan. 1986.
[6] R. L. Kashyap, "Random field models on torus lattices for finite images," in *Proc. 5th Int. Conf. Pattern Recognition*, Miami Beach, FL, Dec. 1-4, 1980, pp. 1103-1105.
[7] R. L. Kashyap, "Univariate and multivariate random field models for images," *Comput. Graph. Image Processing*, vol. 12, pp. 257-270, 1980.
[8] ____, "Analysis and synthesis of image patterns by spatial interaction models," in *Progress in Pattern Recognition*, L. N. Kanal and A. Rosenfeld, Eds., vol. 1. New York: North-Holland, 1981, pp. 149-186.
[9] R. L. Kashyap and R. Chellappa, "Estimation and choice of neighbors in spatial interaction models of images," *IEEE Trans. Inform. Theory*, vol. IT-29, pp. 60-72, 1983.
[10] R. L. Kashyap, R. Chellappa, and A. Khotanzad, "Texture classification using features derived from random field models," *Pattern Recognition Lett.*, vol. 1, pp. 43-50, 1982.
[11] R. L. Kashyap and A. Khotanzad, "A model-based method for rotation invariant texture classification," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8, pp. 472-481, 1986.
[12] A. Khotanzad, "Stochastic model-based techniques for classification and segmentation of textures," Ph.D. dissertation, School of Electrical Eng., Purdue Univ., W. Lafayette, IN, 1983.

Learning Automata with Changing Number of Actions

M. A. L. THATHACHAR AND BHASKAR R. HARITA

***Abstract*—A new reinforcement scheme based on the linear reward-inaction updating algorithm is presented for a learning automaton whose action set is changing from instant to instant. A learning automaton using the new algorithm is shown to be both absolutely expedient and ϵ-optimal. The simulation results verify the ϵ-optimality of the algorithm. The results can be extended to the design of general nonlinear absolutely expedient learning algorithms.**

I. Introduction

The study of deterministic automata operating in a random environment was initiated by Tsetlin [1] to model the behavior of biological systems. This was extended by Varshavskii and Vorontsova [2] to an investigation of variable structure stochastic automata. These automata are now called learning automata and their theory has been extensively developed over the last few years. Many basic results are documented in a survey [3], a special journal issue [4], and a book on the subject [5].

At every instant, the learning automaton chooses an action on the basis of a probability distribution over a finite and fixed set of actions. The response received from the environment is used to update the action probabilities using a reinforcement scheme or learning algorithm. Through this process of interaction with the environment the automaton learns to choose, asymptotically with a high probability, the optimal action. The learning algorithm has to be suitably designed to get a desired asymptotic behavior from the automaton. Two performance measures of learning automaton that have been extensively studied are ϵ-optimality and absolute expediency [3]-[5].

However, nearly all research so far has dealt with the learning behavior of automata that have action sets that contain a fixed number of actions. The objective of this correspondence is to consider the performance of learning automata with changing number of actions. In such automata, the set of available actions at every instant need only be a subset of the complete set of actions and could change from instant to instant. It is assumed that an external agency selects a subset at each instant on the basis of a probability distribution defined over all the possible subsets. This probability distribution can change with time. The mechanism of selection of an action subset by the external agency need not be known to the automaton.

Manuscript received September 2, 1986; revised August 19, 1987.
The authors are with the Department of Electrical Engineering, Indian Institute of Science, Bangalore 560012, India.
IEEE Log Number 8717376.

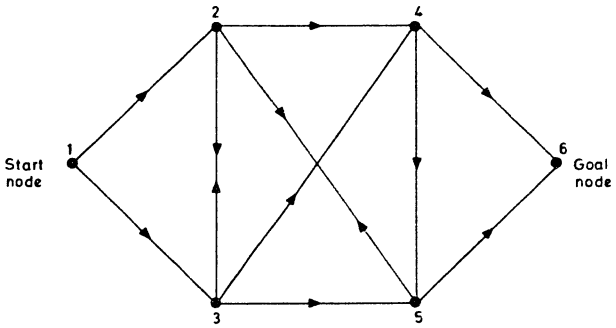

Fig. 1. Directed network with random edge weights.

The need for studying learning automata with changing number of actions arose from the work of Ramakrishnan and Thathachar [6], who proposed an automaton model for the CPU job scheduling problem. In this model the behavior of jobs is characterised by a random CPU burst followed by a random delay to simulate I/O and paging. The N jobs present in the system are identified with the N actions of the automaton scheduler. Initially the action probabilities are equalized, an action (job) is selected, and assigned the CPU. The normalized CPU burst duration is the response of the stationary random environment and is used to update the action probabilities. Once a job relinquishes the CPU, it is no longer available for some time to use the CPU and this is modeled by its re-entry into the CPU queue after a random delay. As the jobs randomly leave and re-enter the queue, the effective number of actions of the automaton available for choice is time varying. When a job has satisfied all its CPU requirements it leaves the system. A new job enters the CPU queue and the action probability vector is suitably adjusted. Using conventional learning algorithms with certain ad hoc changes, Ramakrishnan and Thathachar conducted simulation studies on the automaton scheduler. However, they did not attempt any analysis of the problem.

Learning automata with changing number of actions are also relevant in the modeling of the optimal path problem in stochastic networks [7]. Such networks, whose edge weights are random variables with unknown distribution functions, have one learning automaton at each node whose out-degree is greater than one. The actions of each automaton are the outward directed edges from the node. A path through the network is a sequence of actions from a distinguished start node to a goal node avoiding cycles or closed paths. The optimal path has the minimum expected weight among the set of all paths. The present authors have designed distributed learning algorithms, using which the network of automata converges to the globally optimal sequence of actions.

In the network of Fig. 1, there are automata at the nodes labeled 1, 2, 3, 4 and 5. An edge from node a to node b is represented by the ordered pair (a, b). The edges (2,3) and (2,5) are bidirectional.

The actions of an automaton at an intermediate node in the network depend on how that node was reached from the start node. For example consider the actions of the automaton at node 2. If the partial path to node 2 is constituted by the edge (1,2),

0018-9472/87/1100-1095$01.00

the actions available for choice by the automaton are the edges (2,3), (2,4) and (2,5). When the partial path consists of the sequence of edges (1,3), (3,2) the available actions are the edges (2,4) and (2,5). When the partial path is (1,3), (3,5), (5,2) the only available action at node 2 is (2,4), as cycles are not permitted. This example, although simplistic, demonstrates the need for automata with changing number of actions. One could attempt to avoid the varying action set by having a number of automata at node 2, one corresponding to each partial path. However, each of these automata would be active only when the corresponding partial path is chosen. Thus the automata would be updated less frequently and could take a longer time to converge in comparison with the single-automaton case.

In this correspondence, a learning algorithm suitable for an automaton with a changing number of actions is presented. This algorithm is based on the well-known L_{R-I} reinforcement scheme [3]–[5] and is called the variable action set L_{R-I} algorithm. An automaton with a fixed number of actions using the conventional L_{R-I} scheme is known to be absolutely expedient and ϵ-optimal [5], [8]. It shall be proven that even with a changing number of actions, an automaton using the new algorithm continues to display both properties. From the simulation results it is observed that in all trials where a sufficiently small value of the updating parameter was used, the optimal action is asymptotically chosen verifying ϵ-optimality of the variable action set L_{R-I} algorithm.

After a brief introduction to learning automata in Section II, the new algorithm for a single automaton is presented in Section III. In Section IV we prove the absolute expediency and ϵ-optimality of the algorithm. Simulation results are discussed in Section V, while conclusions are presented in the last section.

II. Learning Automata

A learning automaton is a stochastic automaton that is connected in a feedback loop with a random environment. A stochastic automaton can be denoted by the quadruple $\langle V, P, T, R \rangle$. $V = \{\alpha_1, \alpha_2, \alpha_3, \cdots, \alpha_N\}$ is the set of all actions of the automaton, $2 \leq N < \infty$. $\alpha(k)$ denotes the action chosen by the automaton at the instant k. P denotes a probability distribution over V and is defined at the instant k by the vector $\boldsymbol{p}(k)$ given by

$$\boldsymbol{p}(k) = [p_1(k), p_2(k), p_3(k), \cdots, p_N(k)]$$

where

$$p_i(k) = \text{prob}[\alpha(k) = \alpha_i]$$

and

$$\sum_{i=1}^{N} p_i(k) = 1, \forall k.$$

$\boldsymbol{p}(k)$ is called the action probability vector.

R is the set of reactions of the environment. $\beta(k)$ denotes the reaction at instant k and $\beta(k) \in R$, for all k. A P-model environment shall be considered, with $R = \{0,1\}$. $\beta(k)$ is binary valued with $\beta(k) = 1$ being called a penalty while $\beta(k) = 0$ is a reward.

Associated with each action $\alpha_i \in V$, is a penalty probability c_i, defined as

$$c_i = \text{prob}[\beta(k) = 1 | \alpha(k) = \alpha_i]$$

In this paper only stationary environments in which the c_i are independent of k shall be considered. Let $c_L = \min_i \{c_i\}$. α_L is the optimal action.

T is the updating operator defined by

$$\boldsymbol{p}(k+1) = T(\boldsymbol{p}(k), \alpha(k), \beta(k)).$$

T is usually given as an algorithm and is referred to as a learning algorithm or reinforcement scheme.

In order to judge the effectiveness of a given automaton there are several performance measures and two of them are as follows.

1) ϵ-optimality [3]: A learning automaton is said to be ϵ-optimal if for any given $\epsilon > 0$, we can have

$$\liminf P_L(k) \geq 1 - \epsilon$$

with probability 1 as $k \to \infty$ by suitable choice of the parameters of the learning algorithm.

2) Absolute expediency [3], [8]: A learning automaton is said to be absolutely expedient if

$$E[M(k+1) | \boldsymbol{p}(k)] < M(k)$$

for all k, all $\boldsymbol{p} \in S$ and in all stationary random environments except in the trivial one where $c_i = c_j$, for all i, j, where

$$M(k) = E[\beta(k) | \boldsymbol{p}(k)] = \sum_{i=1}^{N} p_i(k) c_i$$

is the expected penalty at the instant k from the environment and

$$S = \left\{ \boldsymbol{p} : p_i > 0, \sum_{i=1}^{N} p_i = 1 \right\}$$

is the N-dimensional open simplex.

Absolute expediency implies that the average penalty $M(k)$ is a super martingale and the corresponding algorithm has good convergence properties. Necessary and sufficient conditions for the design of absolutely expedient algorithms are available [8]. It has also been shown that absolute expediency results in ϵ-optimality in all stationary random environments [5].

One of the most extensively studied reinforcement schemes is the linear reward-inaction (L_{R-I}) scheme [3] that can be written as

$$\begin{aligned} p_i(k+1) &= p_i(k) + a(1 - p_i(k)); \quad \alpha(k) = \alpha_i, \beta(k) = 0 \\ &= p_i(k); \quad \alpha(k) = \alpha_i, \beta(k) = 1 \\ &= p_i(k) - a p_i(k); \quad j \neq i, \alpha(k) = \alpha_j, \beta(k) = 0 \\ &= p_i(k); \quad j \neq i, \alpha(k) = \alpha_j, \beta(k) = 1, \end{aligned}$$

where $0 < a < 1$ is the updating parameter.

An automaton using the L_{R-I} scheme is ϵ-optimal in all stationary random environments and also absolutely expedient [5], [8].

III. Variable Action Set L_{R-I} Learning Algorithm

In this section a learning automaton with a changing number of actions (variable action set) that uses an L_{R-I} type reinforcement scheme is considered. Such an automaton has a finite set of N actions, $V = \{\alpha_1, \alpha_2, \alpha_3, \cdots, \alpha_N\}$. However, at every instant k, only a subset $V(k)$ consisting of $1, 2, 3, \cdots$ or N actions is available for choice. The selection of the particular action subset $V(k)$ is made by an external agency, at random, according to a probability distribution $\boldsymbol{Q}(k)$ defined over all the possible subsets of actions. The automaton chooses an action $\alpha(k) \in V(k)$, at random according to a probability distribution $\hat{\boldsymbol{p}}(k)$ defined over $V(k)$. The action chosen elicits a binary valued reaction $\beta(k)$ from the P-model environment. The automaton then computes $\hat{\boldsymbol{p}}(k+1)$ using the variable action set L_{R-I} learning algorithm and the cycle repeats.

Before the algorithm is presented, consider the following notations used and state the assumptions made.

1) At the instant k,

$\alpha(k)$ action chosen by the automaton
$\beta(k)$ binary valued reaction of the environment,

$\boldsymbol{p}(k)$ action probability vector defined over the set of all actions V,
$V(k)$ is the subset of actions selected by an external agency.

2) The actions are numbered from 1 to N and the complete set of actions is denoted by $V = \{\alpha_1, \alpha_2, \alpha_3, \cdots, \alpha_N\}$. V_i represents the ith action subset where $1 \le i \le 2^N - 1$ (there are $2^N - 1$ nonempty subsets). Index i is assigned by ordering the subsets in a lexicographic manner beginning with the single-action subsets, then the double action subsets $\cdots$, and ending with the complete set of actions V.

For example, if $V = \{\alpha_1, \alpha_2, \alpha_3\}$ where $N = 3$, the subsets are ordered as

$$V_1 = \{\alpha_1\}, \quad V_2 = \{\alpha_2\}, \quad V_3 = \{\alpha_3\}, \quad V_4 = \{\alpha_1, \alpha_2\},$$
$$V_5 = \{\alpha_1, \alpha_3\}, \quad V_6 = \{\alpha_2, \alpha_3\}, \quad V_7 = V = \{\alpha_1, \alpha_2, \alpha_3\}.$$

3) The equation

$$q_i(k) = \text{prob}[V(k) = V_i]$$

represents the probability of selecting the ith action subset at the kth instant, where $i \le i \le 2^N - 1$, the selection being done by an external agency.

Further,

$$\sum_{i=1}^{2^N - 1} q_i(k) = 1, \qquad \forall k.$$

The probability distribution $\boldsymbol{Q}(k)$, defined over all the possible action subsets is

$$\boldsymbol{Q}(k) = \{q_1(k), q_2(k), q_3(k), \cdots, q_{W-1}(k), q_W(k)\}$$

where $W = 2^N - 1$.

4) The equation

$$K(k) = \sum_{\alpha_i \in V(k)} p_i(k)$$

represents the sum of the probabilities of the actions of the subset $V(k)$ at the instant k.

5) $\hat{p}_i(k) = p_i(k)/K(k)$ is the scaled probability of action α_i in the subset $V(k)$ when $\alpha_i \in V(k)$. We have

$$\hat{p}_i(k) = \text{prob}[\alpha(k) = \alpha_i | V(k), \alpha_i \in V(k)]$$

since it is the probability of choosing the ith action from $V(k)$, conditioned on the event that the subset $V(k)$ has already been selected and α_i belongs to this subset. We define

$$\hat{\boldsymbol{p}}(k) = \{\hat{p}_i(k)\}, \qquad \forall i \text{ such that } \alpha_i \in V(k),$$

which is the scaled probability distribution over $V(k)$.

We now state the following assumptions:

Assumption 1: The selection of the particular action subset at each instant is made randomly by an external agency in accordance with an unknown probability distribution $\boldsymbol{Q}(k)$.

Assumption 2: At least one of the action subsets containing more than one action must have a nonzero probability of being selected.

Comments

a) The first assumption implies that the selection of the action subsets is not necessarily connected in any way with the operation of the automaton. The distribution $\boldsymbol{Q}(k)$ can change from instant to instant in an unknown and arbitrary manner.

b) Assumption 2 is reasonable since its nonsatisfaction would make the problem trivial, as updating is not required for singleton subsets.

Algorithm 1

Consider a time instant k, $k = 0,1,2,\cdots V(k)$ is the action subset selected.

The choice of action $\alpha(k)$ at instant k and the updating of $\boldsymbol{p}(k)$ proceeds through the following stages

1) Form

$$K(k) = \sum_{\alpha_i \in V(k)} p_i(k)$$

2) Scale the probabilities of the actions in the chosen subset. For all i, such that $\alpha_i \in V(k)$, do

$$\hat{p}_i(k) = p_i(k)/K(k).$$

3) Choose an action $\alpha(k)$ from $V(k)$, at random according to the scaled probability vector $\hat{\boldsymbol{p}}(k)$.

4) Update $\hat{\boldsymbol{p}}(k)$ as follows:

$$\hat{p}_q(k+1) = \hat{p}_q(k) + a(1 - \hat{p}_q(k)); \qquad \alpha(k) = \alpha_q, \beta(k) = 0$$
$$= \hat{p}_q(k); \qquad \alpha(k) = \alpha_q, \beta(k) = 1$$
$$= \hat{p}_q(k) - a\hat{p}_q(k); \qquad j \ne q, \alpha(k) = \alpha_j, \beta(k) = 0$$
$$= \hat{p}_q(k); \qquad j \ne q, \alpha(k) = \alpha_j, \beta(k) = 1$$

where

$$\alpha_q, \alpha_j \in V(k)$$

and

$$p_t(k+1) = p_t(k); \qquad \forall t \text{ such that } \alpha_t \notin V(k).$$

The updating parameter a lies in (0,1).

5) Rescale the action probabilities of the actions of the chosen subset. For all i, such that $\alpha_i \in V(k)$, do

$$p_i(k+1) = \hat{p}_i(k+1) \cdot K(k).$$

Comments

a) The second stage of the algorithm involves the scaling of the action probabilities of $V(k)$, a feature drawn from the work of Luce [9]. Since the choice of an action at the instant k is only from among the members of the subset $V(k)$, it is natural to scale the action probabilities in a manner that retains the original ratios the action probabilities bear to one another in the complete set of actions. It simultaneously ensures that the components of $\hat{\boldsymbol{p}}(k)$ sum to unity. This is achieved by dividing each action probability in $V(k)$ by the sum of all the action probabilities in $V(k)$, that is the term $K(k)$.

b) The fourth stage of the algorithm is the L_{R-I} scheme applied to the scaled action probability vector $\hat{\boldsymbol{p}}(k)$.

c) When the initial action probabilities are in the open simplex, the fourth stage of the algorithm ensures that for all i, $1 \le i \le N$, $\hat{p}_i(k)$ would tend to either zero or unity only asymptotically. Hence for all finite k, the divisor term $K(k)$ would be nonzero.

However, during the simulations it was noticed that in some trials the smallness of $K(k)$ resulted in floating point underflow. This problem was avoided by rewriting Algorithm 1 in the following more elegant form:

Alternate Form of the Algorithm

1) Form

$$K(k) = \sum_{\alpha_i \in V(k)} p_i(k).$$

2) Update $\boldsymbol{p}(k)$ as follows. For all q, such that $\alpha_q \in V(k)$, do

$$\begin{aligned} p_q(k+1) &= p_q(k) - ap_q(k) + aK(k); \\ &\qquad \alpha(k) = \alpha_q,\ \beta(k) = 0 \\ &= p_q(k); \qquad \alpha(k) = \alpha_q,\ \beta(k) = 1 \\ &= p_q(k) - ap_q(k); \\ &\quad j \neq q,\ \alpha(k) = \alpha_j,\ \beta(k) = 0,\ \alpha_j \in V(k) \\ &= p_q(k); \\ &\quad j \neq q,\ \alpha(k) = \alpha_j,\ \beta(k) = 1,\ \alpha_j \in V(k). \end{aligned}$$

For all i, such that $\alpha_i \notin V(k)$

$$p_i(k+1) = p_i(k).$$

The updating parameter a lies in (0,1).

This algorithm is obtained by combining the last four stages of Algorithm 1 and does not contain $K(k)$ as a divisor term. It differs from the conventional L_{R-I} scheme only in the first updating step when $\alpha(k) = \alpha_q$ and a reward ($\beta(k) = 0$) is obtained.

IV. Analysis of Algorithm 1

It is known that when the action set is invariant a learning automaton using the L_{R-I} updating scheme is absolutely expedient and ϵ-optimal in all stationary random environments. In this section it will be shown that an automaton using the variable action set L_{R-I} updating scheme continues to be both absolutely expedient and ϵ-optimal.

Absolute Expediency

First to be shown is that

$$\Delta M(k) < 0$$

where

$$\Delta M(k) = E[M(k+1) - M(k) | \boldsymbol{p}(k)]$$

and

$$M(k) = \sum_{i=1}^{N} c_i p_i(k)$$

is the average penalty at instant k.

Proof: Consider a time instant k. Let $V(k) = V_j$ be the action subset selected.

If $\alpha_i \in V(k)$, then from the fourth stage of Algorithm 1 we have

$$\begin{aligned} &E[\hat{p}_i(k+1) - \hat{p}_i(k) | \boldsymbol{p}(k), V(k) = V_j] \\ &\quad = a(1 - \hat{p}_i(k))\hat{p}_i(k)(1 - c_i) \\ &\qquad - a\hat{p}_i(k) \sum_{\substack{g \neq i \\ \alpha_g \in V(k)}} \hat{p}_g(k)(1 - c_g) \\ &\quad = a\hat{p}_i(k)(1 - c_i) \sum_{\substack{g \neq i \\ \alpha_g \in V(k)}} \hat{p}_g(k) \\ &\qquad - a\hat{p}_i(k) \sum_{\substack{g \neq i \\ \alpha_g \in V(k)}} \hat{p}_g(k)(1 - c_g) \\ &\quad = a\hat{p}_i(k) \sum_{\substack{g \neq i \\ \alpha_g \in V(k)}} \hat{p}_g(k)(c_g - c_i). \end{aligned} \tag{1}$$

Now from the second stage of Algorithm 1,

$$\hat{p}_q(k) = p_q(k)/K(k)$$

for all q, such that $\alpha_q \in V(k)$. Since $V(k) = V_j$

$$\hat{p}_q(k) = p_q(k)/K_j(k) \tag{2}$$

for all q, such that $\alpha_q \in V(k)$, where

$$K_j(k) \triangleq \sum_{\substack{q \\ \alpha_q \in V_j}} p_q(k).$$

Then (1) can be rewritten as,

$$\begin{aligned} &E[\hat{p}_i(k+1) - \hat{p}_i(k) | \boldsymbol{p}(k), V(k) = V_j] \\ &\quad = \frac{a}{[K_j(k)]^2} p_i(k) \sum_{\substack{g \neq i \\ \alpha_g \in V(k) = V_j}} p_g(k)(c_g - c_i). \end{aligned} \tag{3}$$

Let

$$W = 2^N - 1.$$

Now

$$\begin{aligned} &E[p_i(k+1) - p_i(k) | \boldsymbol{p}(k)] \\ &\quad = \sum_{j=1}^{W} E[p_i(k+1) - p_i(k) | \boldsymbol{p}(k), V(k) = V_j] \\ &\qquad \cdot \text{prob}[V(k) = V_j] \end{aligned} \tag{4}$$

and

$$\begin{aligned} &E[p_i(k+1) - p_i(k) | \boldsymbol{p}(k), V(k) = V_j] \\ &\quad = K_j(k) \cdot E[\hat{p}_i(k+1) - \hat{p}_i(k) | \boldsymbol{p}(k), V(k) = V_j]. \end{aligned} \tag{5}$$

Using result (5) in (4)

$$\begin{aligned} E[p_i(k+1) - p_i(k) | \boldsymbol{p}(k)] &= \sum_{j=1}^{W} q_j(k) \\ &\cdot K_j(k) \cdot E[\hat{p}_i(k+1) - \hat{p}_i(k) | \boldsymbol{p}(k), V(k) = V_j] \end{aligned} \tag{6}$$

is obtained.

The summation in (6) is taken over all the possible subsets ordered in a lexicographic manner beginning with the single-action subsets and ending with the complete set of actions, $V = V_W$. However, it is to be noted that for all the single-action subsets and others that do not contain the action α_i, the corresponding term,

$$E[\hat{p}_i(k+1) - \hat{p}_i(k) | \boldsymbol{p}(k), V(k) = V_j] = 0.$$

Then, (6) can be rewritten as

$$\begin{aligned} E[p_i(k+1) - p_i(k) | \boldsymbol{p}(k)] &= \sum_{\substack{j = N+1 \\ (\alpha_i \in V_j)}} q_j(k) \\ &\cdot K_j(k) \cdot E[\hat{p}_i(k+1) - \hat{p}_i(k) | \boldsymbol{p}(k), V(k) = V_j] \end{aligned} \tag{7}$$

where the summation excludes the first N single-action subsets and those that do not contain α_i.

Substituting result (3) in (7) we have

$$\begin{aligned} &E[p_i(k+1) - p_i(k) | \boldsymbol{p}(k)] \\ &\quad = a \sum_{\substack{j = N+1 \\ (\alpha_i \in V_j)}}^{W} \frac{q_j(k)}{K_j(k)} p_i(k) \sum_{\substack{g \neq i \\ \alpha_g \in V_j}} p_g(k)(c_g - c_i). \end{aligned} \tag{8}$$

By definition,

$$\Delta M(k) = \sum_{i=1}^{N} c_i E[\, p_i(k+1) - p_i(k) | \boldsymbol{p}(k)] \tag{9}$$

$$= a \sum_{i=1}^{N} \sum_{\substack{j=N+1 \\ (\alpha_i \in V_j)}}^{W} \frac{q_j(k)}{K_j(k)} p_i(k) \sum_{\substack{g \neq i \\ \alpha_g \in V_j}} p_g(k)\left(c_i c_g - c_i^2\right). \tag{10}$$

$\Delta M(k)$ can be equivalently expressed in the quadratic form [8]

$$\boldsymbol{p}(k) D(k) \boldsymbol{p}^T(k)$$

where

$$\boldsymbol{p}(k) = [\, p_1(k), p_2(k), p_3(k), \cdots, P_N(k)]$$

and $D(k)$ is a $N \times N$ symmetric matrix with

$$d_{ii} = 0$$

$$d_{ij} = -a \frac{(c_i - c_j)^2}{2} \sum_{m=N+1}^{W} \frac{q_m(k)}{K_m(k)}$$

where m is such that, $\alpha_i, \alpha_j \in V_m$. By our second assumption $q_m(k) \neq 0$, for at least one subset V_m, where $N+1 \leq m \leq 2^N - 1$. Hence for all nontrivial environments ($c_i \neq c_j$, some $j \neq i$), $D(k)$ has some negative elements. Thus, if the initial action probabilities lie in the open simplex, the quadratic form

$$\Delta M(k) = \boldsymbol{p}(k) D(k) \boldsymbol{p}^T(k) < 0,$$

which completes the proof that an automaton using the variable action set L_{R-I} updating scheme is absolutely expedient.

ϵ-Optimality

It is known that a learning automaton with a fixed number of actions is ϵ-optimal in all stationary random environments when it uses the L_{R-I} reinforcement scheme [5]. In the present context, the selection of an action subset is equivalent to the selection of a learning automaton with a fixed number of actions and updating within this subset takes place according to the L_{R-I} scheme (stage 4) of Algorithm 1. Then ϵ-optimality within the subset would follow provided the subset is chosen an infinite number of times. Thus additional assumptions are needed on the selection probability distribution $\boldsymbol{Q}(k)$. Let us suppose that the following assumptions are satisfied.

Assumption 3: There exists a collection of action subsets, each of which contains the optimal action and whose union is V, the set of all actions.

Assumption 4: Each subset V_r of the collection in Assumption 3 is selected with a positive probability at all times instants (i.e., $q_r(k) \geq \epsilon_r > 0$ for all k for each of these subsets V_r).

We now show that under the above assumptions, ϵ-optimality follows.

Assumption 4 ensures that each subset in the collection is selected an infinite number of times. As argued earlier, in each subset of the collection, the scaled-action probability $\hat{p}_L(k)$ of the optimal action α_L tends to unity with probability $1 - \epsilon$. Since the collection includes all the actions in V by Assumption 3, $\hat{p}_L(k)$ emerges with a unity value when it is updated along with every other action probability of the actions in V. Thus, all the scaled-action probabilities $\hat{p}_i(k)$ $(i \neq L)$ tend to zero and consequently the unscaled action probability $p_L(k) \to 1$ with a probability arbitrarily close to unity. Hence the learning automaton using the variable action set L_{R-I} scheme is ϵ-optimal in all stationary random environments.

It may be of interest to note that Assumptions 3 and 4 are automatically satisfied in the special case when V, the complete set of actions is selected with a positive probability at all instants.

TABLE I
SIMULATION RESULTS FOR THE VARIABLE ACTION SET L_{R-I} ALGORITHM[a]

Updating Parameter a	Average Number of Iterations	Wrong Convergences Out of 50 trials
0.01	1179	0
0.05	226	0
0.1	117	0
0.5	14	21

[a] Total number of actions equals 5. Selection probability of class of subsets (number of actions in each member of the class): 0.03 (1); 0.07 (2); 0.1 (3); 0.1 (4); 0.7 (5).

TABLE II
SIMULATION RESULTS FOR THE VARIABLE ACTION SET L_{R-I} ALGORITHM[b]

Updating Parameter a	Average Number of Iterations	Wrong Convergences Out of 50 trials
0.01	1717	0
0.05	334	0
0.1	172	0
0.5	27	21

[b] Total number of actions equals 5. Selection probability of class of subsets (number of actions in each member of the class): 0.1 (1); 0.15 (2); 0.2 (3); 0.25 (4); 0.3 (5).

TABLE III
SIMULATION RESULTS FOR THE VARIABLE ACTION SET L_{R-I} ALGORITHM[c]

Updating Parameter a	Average Number of Iterations	Wrong Convergences Out of 50 trials
0.01	2655	0
0.05	495	0
0.1	225	0
0.5	71	7

[c] Total number of actions equals 5. Selection probability of class of subsets (number of actions in each member of the class): 0.1 (1); 0.2 (2); 0.4 (3); 0.2 (4); 0.1 (5). The penalty probabilities of the P Model environment are: 0.7; 0.8; 0.1; 0.6; 0.75.

V. SIMULATION RESULTS

In this section results of computer simulations of the variable action set L_{R-I} type algorithm are presented. The objective of the simulations was to study the asymptotic behavior of the algorithm and in particular to find out whether the automaton chooses the optimal action asymptotically.

Tables I, II and III display the simulation results for three different $\boldsymbol{Q}$ distributions. The $2^5 - 1$ possible nonempty subsets of the five-action automaton considered are divided into classes with each class containing subsets with equal number of actions. Subsets belonging to a particular class have the same probability of selection. This is achieved in simulation by choosing the actions of a subset belonging to the particular class, randomly from the set of all actions.

The penalty probabilities of the P-model environment used are given at the end of the tables. Fifty experiments, each consisting of 3000 iterations were performed for each value of the updating parameter a. The average number of iterations taken for p_l (optimal action probability) to be greater than 0.99 is shown in the tables as are the number of wrong convergences.

The last column in the tables demonstrates how the accuracy of the variable action set L_{R-I} algorithm suffers when we increase the value of a to improve the speed of convergence.

Further, for the same value of a, we observe that the average number of iterations is less for the case where the selection probabilities of each class of actions is biased towards the set of all actions (Table I) than when it is not (Tables II and III). This reiterates that global optimality is better achieved when the complete set of actions is selected more often.

The floating point underflow problem was avoided by using the alternate form of Algorithm 1.

It is evident from the simulations that in all trials where the updating parameter a was sufficiently small, the optimal action was asymptotically chosen, verifying the ϵ-optimality of an automaton using the new algorithm.

VI. Conclusion

In this correspondence it has been shown that a learning automaton with a changing number of actions can be made absolutely expedient and ϵ-optimal by using a modified version of the L_{R-I} algorithm. The ϵ-optimality of the algorithm was shown under certain additional assumptions on the selection probabilities of subsets containing the optimal action. The simulation results verify the property of ϵ-optimality. The results of this paper can be extended to the design of general nonlinear absolutely expedient algorithms for learning automata with changing number of actions.

References

[1] M. L. Tsetlin, "On behaviour of finite automata in random media," *Automat. Telemekh.*, vol. 22, no. 10, pp. 1345–1354, Oct. 1961.

[2] V. I. Varshavskii and I. P. Vorontsova, "On the behaviour of stochastic automata with variable structure," *Automaton and Remote Control*, vol. 24, no. 3, pp. 327–333, Oct. 1963.

[3] K. S. Narendra and M. A. L. Thathachar, "Learning automata—a survey," *IEEE Trans. Syst. Man Cybern.*, vol. SMC-4, no. 4, pp. 323–334, July 1974.

[4] K. S. Narendra (Ed.), *J. Cybern. Inform. Sci.* (*Special Issue on Learning Automata*), vol. 1, no. 2–4, 1977.

[5] S. Lakshmivarahan, "*Learning Algorithms—Theory and Applications*," New York: Springer-Verlag, 1981.

[6] K. R. Ramakrishnan and M. A. L. Thathachar, "A learning scheduler for job processing," presented at the Seminar on Large Scale Systems and Signal Processing, School of Automation, Indian Institute of Science, Bangalore, Jan. 1982.

[7] B. R. Harita, "Learning automata in networks," M. E. Thesis, Dept. of Elec. Engg., Indian Institute of Science, Bangalore, Jan. 1986.

[8] S. Lakshmivarahan and M. A. L. Thathachar, "Absolutely expedient learning algorithms for stochastic automata," *IEEE Trans. Syst., Man and Cybern.*, vol. SMC-3, no. 3, pp. 281–286, May 1973.

[9] R. D. Luce, *Individual Choice Behaviour*. New York: John Wiley, 1959.