A Generalized Learning Algorithm for an Automaton Operating in a Multiteacher Environment

Arif Ansari, *Member, IEEE*, and George P. Papavassilopoulos, *Senior Member, IEEE*

***Abstract*— Learning algorithms for an automaton operating in a multiteacher environment are considered. These algorithms are classified based on the number of actions given as inputs to the environments and the number of responses (outputs) obtained from the environments. In this paper, we present a general class of learning algorithm for multi-input multi-output (MIMO) models. We show that the proposed learning algorithm is absolutely expedient and ϵ-optimal in the sense of average penalty. The proposed learning algorithm is a generalization of Baba's GAE algorithm [16] and has applications in solving, in a parallel manner, multi-objective optimization problems in which each objective function is disturbed by noise [20].**

***Index Terms*— Learning algorithms, MIMO systems, multiteacher environment, stochastic automata.**

I. Introduction

The study of deterministic automata operating in a random environment was initiated by Tsetlin [1] to model the behavior of biological systems. Varshavskii and Vorontsova [2] extended the concept to variable structure stochastic automata. Norman [3]–[5] studied a stochastic automaton with two states and showed that ϵ-optimality can be ensured for the L_{R-I} scheme, and later, this scheme was proved to be ϵ-optimal in the general n-state case [6], [7]. Lakshmivarahan and Thathachar [8] introduced the concept of absolute expediency and proved this class of algorithms to be ϵ-optimal under additional constraints. Many of these results are documented in a survey by Narendra and Thathachar [11]. The book by Lakshmivarahan [10] provides a introduction to learning algorithm theory. Recent research results and current applications of learning algorithms are also available in books by Narendra and Thathachar [12] and Najim and Poznyak [25]. Most of these results apply to a single automaton operating in a single-teacher environment. Some authors studied the learning behavior of a stochastic automaton operating in a multiteacher environment (Fig. 1). Kodischek and Narendra [13] considered the learning behavior of a fixed-structure automaton acting in a multiteacher environment. Thathachar and Bhakthavathsalam [14] studied learning behavior of a variable-structure stochastic automaton in a two-distinct-teacher environment. Baba extensively studied the learning behavior of a stochastic automaton operating in a multiteacher environment [16]–[20]. Baba considered the case in which a single action selected by the automaton at stage k is given as input to all the environments. All of the aforementioned learning algorithms update the action probability vector based on the multiresponses (or single response) obtained from the multiteacher (or single teacher) for the same action as input. These types of algorithm exclude the cases in which there are m teachers and they are provided with different actions as inputs. In this paper, we consider the case in which different actions selected by the automaton are given as inputs to multiteacher environment.

Learning algorithms can be classified as linear or nonlinear, stationary or nonstationary, etc. Let us classify the learning algorithms based on the number of actions given as inputs to the environments and number of responses (outputs) obtained from the environments. A single automaton operating in a single-teacher environment can be considered as a single-input, single-output model (SISO). Baba's automaton operating in a multiteacher environment is a single-input multiple-output model (SIMO), since all the teachers (environments) are provided with the same action as input and the environments provide multiple probabilistic responses based on multicriteria. The case in which different actions are applied as inputs to the multiteacher environment at the same time k and according to the same action probability and each environment providing a response according to its own criteria has not been studied. This case corresponds to multi-input multi-output (MIMO) model. MIMO model studies the learning behavior of a student who poses different questions to different teachers at the same time and obtains multiple answer to his questions. In the real world, we are often confronted with different alternatives and different teachers providing answers to different questions at the same time. MIMO model addresses these types of learning situations. Also, MIMO algorithm is a generalization of Baba's GAE algorithm. The learning automaton considered is a P-type[1] automaton [12].

The MIMO learning algorithm can be applied to both single criterion and multicriterion models. In the single-criterion model, all the different teachers (environments) use the same criterion, i.e., the reward–penalty probabilities are the same for all teachers. In the multicriterion model, each environment has a different set of reward–penalty probabilities for different actions. Kodischek and Narendra [13] imposed a condition on the multicriterion structure, and they assumed that there exists

Manuscript received April 15, 1996; revised March 28, 1997 and April 17, 1999. This work was supported in part under National Science Foundation Grant CCR-9222734. This paper was recommended by Associate Editor K. Pattipati.

The authors are with the Department of Electrical and Systems Engineering, University of Southern California, Los Angeles, CA 90089 USA (e-mail: aansari@scf.usc.edu).

Publisher Item Identifier S 1083-4419(99)07938-8.

[1] A P-type environment provides a 0 or 1 response to a given action, 0 indicates success and 1 indicates failure.

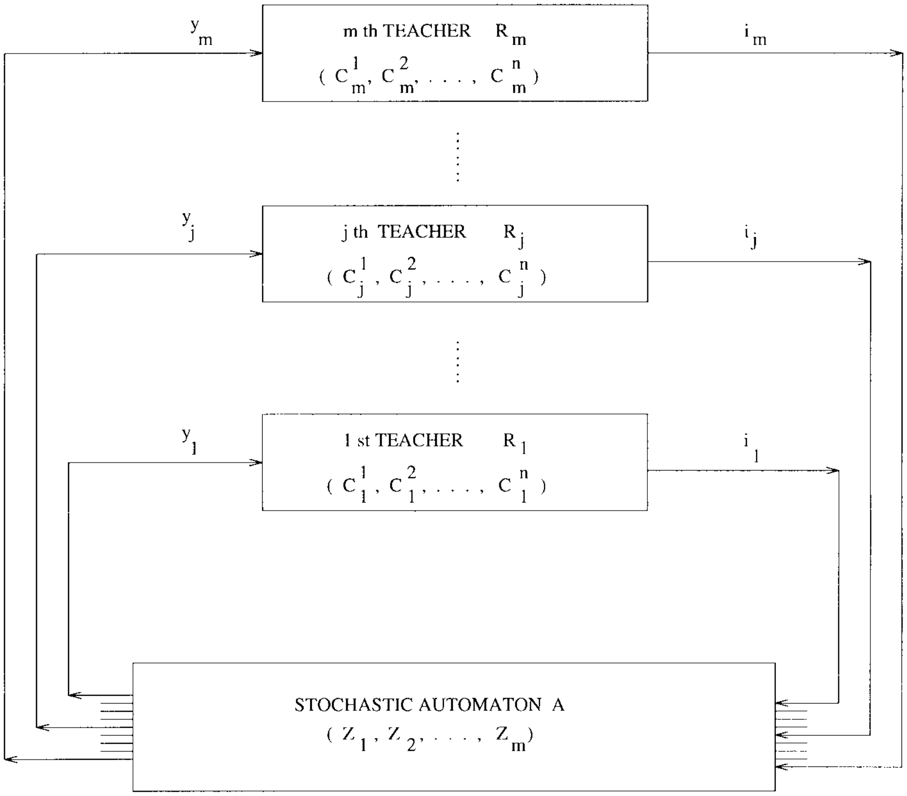

Fig. 1. Stochastic automaton A operating in a MIMO environment.

an action l (α_l) which all teachers agree is the best action, i.e.,

$$c_j^l < c_j^i \text{ for all } j \ (1 \le j \le m) \\ \text{and all } i \ (1 \le i \le n, i \ne l) \qquad (1)$$

where c_j^i denotes the probability of failure (penalty probability) for action i according to environment j. This condition corresponds to the statement that the teachers "agree" that lth action α_l is the best one. Baba [16]–[18] used a more general condition than (1) in which there exists an action α_l such that its total penalty probability is smaller than the total penalty probability of other actions, i.e.,

$$c_1^l + \cdots + c_m^l < c_1^i + \cdots + c_m^i, \text{ for all } i, \\ 1 \le i \le n, (i \ne l). \qquad (2)$$

In this paper, we make the same assumption as given by (2) to find the best action with the smallest total penalty probability.

The proposed learning algorithm is better than Baba's GAE algorithm, in the sense it has a parallel structure and more flexibility. In the GAE algorithm, all the m teacher-environments have to be provided with the same action as input. In the proposed algorithm, actions given to the environments need not be the same. Also, instead of selecting different actions as inputs, we can send the action probability vector as input to all the teacher-environments and let them select their own actions according to $p(k)$. This flexible nature of the algorithm decentralizes the processing, reduces the computational overhead for the automaton and can make efficient use of parallel processors, if they play the role of environments.

The proposed learning algorithm can be used to solve multi-objective optimization problems [20] in which each objective function is disturbed by noise. Since the proposed algorithm is a parallel version of SISO for the single-criterion case, it can be used for a wide range of application given by Najim and Poznyak [25].

This paper is organized as follows. In the Section I, a brief introduction to the learning algorithm for a stochastic automaton operating in a multiteacher environment is given and the learning algorithms are classified based on the number of actions given as inputs to the environments and number of responses (outputs) obtained from the environments. In Section II, a mathematical description of the problem is given. Section III describes the learning automaton and various definitions needed to study the problem are given. In Section IV, a learning algorithm for the MIMO case is given, and we show that the proposed learning algorithm is absolutely expedient and under additional constraints is ϵ-optimal in the sense of average penalty. In the Section V, the proposed algorithm is simulated for various cases and the results of the simulations are presented.

II. Statement of the Problem

Let us consider that we have m stationary P-type teacher-environments. Each of these P-type environments evaluates a finite number of actions (n) probabilistically. If the outcome of an evaluation is 0, it is a success, and if the outcome is 1, it is a failure. The evaluation of an action by an environment is done according to its own reward–penalty criterion. Let d_j^i

be the probability of success (reward probability) for action i according to environment j. Let c_j^i be the probability of failure (penalty probability) for action i according to environment j and

$$c_j^i = 1 - d_j^i, \qquad 1 \le j \le m \text{ and } 1 \le i \le n. \tag{3}$$

Each environment has a different penalty probability criterion and hence the above problem is called a multicriterion problem. We can use these m environments to probabilistically evaluate m actions selected independently from the finite group of n actions. Each action can be selected more than once, and the actions are selected according to the action probability vector $p(k)$ at time k. The penalty probabilities of these actions are not known to us.

Koditschek and Narendra [13] introduced the concept of learning automaton under a multiteacher environment. They studied the learning behavior of a fixed-structure automaton acting in a P-type multiteacher environment. Their objective was to find the best action with the smallest penalty probability. They made an assumption that the teachers "agree" that lth action y_l is the best one, the assumption is given by (1). The condition given by (1) is very restrictive. Baba [16]–[18] used a more general condition than (1) in which there exists an action y_l such that its total penalty probability is smaller than the total penalty probability of other actions. The condition is given by (2). Dividing both sides of (2) by m, we get average penalty

$$\frac{c_1^l + \cdots + c_m^l}{m} < \frac{c_1^i + \cdots + c_m^i}{m}, \quad \text{for all } i, 1 \le i \le n, (i \ne l). \tag{4}$$

Let us denote the above inequality as follows:

$$ac_l < ac_i, \quad \text{for all } i, 1 \le i \le n, (i \ne l). \tag{5}$$

where

$ac_i = (c_1^i + \cdots + c_m^i/m)$ denotes the average penalty for action i;

ac_l $= \min\{ac_1, \cdots, ac_n\}$.

The goal is to find the best action y_l with the smallest average penalty.

III. Learning Automaton and its Performance Measures

Let us briefly describe the learning mechanism of a stochastic automaton A operating in a multiteacher P-type stationary environment. The stochastic automaton is defined by the set {S, Y, α, Z, G, p(k), T}where S denotes the set of inputs $(i_1, i_2, \cdots, i_m)$ to the automaton, where i_j $(j = 1, 2, \cdots, m)$ is the response from teacher R_j $(c_j^1, c_j^2, \cdots, c_j^n)$ and has binary values of 0 and 1. The 0 indicates reward response from R_j and 1 indicates penalty response from R_j. Y denotes the set of outputs $(y_1, y_2, \cdots, y_m)$ applied to each of the m environments, where $y_j \in \alpha$ $(j = 1, 2, \cdots, m)$. The finite action set is represented by $\alpha = \{\alpha_1, \alpha_2, \cdots, \alpha_n\}$, the set from which the best action is chosen. Z $= \{z_1, z_2, \cdots, z_m\}$ represents the m internal states of the automaton, $z_j \in \alpha, (j = 1, 2, \cdots, m)$ and each of the z_j at time k is selected according to the action probability vector $p(k)$. The action probability vector $p(k) = [p_1(k), p_2(k), \cdots, p_n(k)]^T$ represents the probability distribution with which action α_j is selected for internal state z_j at time k. $G = \{g_1, g_2, \cdots, g_m\}$ denotes the m deterministic one to one output mapping of the internal states $z_j(k)$ to the output $y_j(k)$ of the automaton, $y_j(k) = g_j(z_j(k))$. In this problem, g_j is an identity mapping and $y_j(k) = z_j(k)$. T represents the reinforcement scheme (learning algorithm) which generates $p(k+1)$ from $p(k)$. The initial condition $p(0)$ is given by

$$p_1(0) = \cdots = p_n(0) = \frac{1}{n}. \tag{6}$$

Also the $p(k)$ at every time k should satisfy the following requirements:

$$\sum_{i=1}^{n} p_i(k) = 1, \qquad 0 \le p_i(k) \le 1. \tag{7}$$

To evaluate the effectiveness of the proposed learning algorithm for a stochastic automaton operating in a multiteacher environment, we need various performance measures. The different performance measures such as average penalty, optimality, etc., have been set up by various authors for the single teacher environment and multiteacher environments [12], [16]. These measures are modified for the multicriterion model and are given below.

Definition 1: The average penalty $M_j(k)$ for a given action vector $p(k)$ based on teacher j reward–penalty criterion at stage k for a learning automaton operating under a multiteacher environment is given by

$$M_j(k) = \sum_{i=1}^{n} c_j^i p_i(k) \tag{8}$$

where c_j^i denotes the penalty probability for action i according to environment j.

Definition 2: The average weighted penalty is defined as follows:

$$AM(k) = \frac{\sum_{j=1}^{m} M_j(k)}{m} = \sum_{i=1}^{n} p_i(k) ac_i. \tag{9}$$

Definition 3: A learning algorithm for a stochastic automaton is said to be *absolutely expedient* in the general multiteacher environment if

$$E[AM(k+1)|p(k)] < AM(k) \tag{10}$$

for all k, all $p_i(k) \in (0, 1)$, $(i = 1, 2, \cdots, n)$ and for all possible sets $\{c_j^i\}$ $(j = 1, 2, \cdots, m)$, $(i = 1, 2, \cdots, n)$.[2]

[2] It is assumed that cases with all actions having equal average penalty probabilities are excluded.

Definition 4: A learning algorithm for a stochastic automaton is said to be ϵ-optimal in the general multiteacher environment if

$$\lim_{k\to\infty} E[AM(k)] < ac_l + \epsilon, \epsilon > 0. \tag{11}$$

To generalize the learning algorithm for MIMO case, we need to define the sample space of the outputs of the teachers, and they are stated as follows.

Definition 5: Let the sample space of the outputs of teacher j operating in a multiteacher environment be

$$\Omega_j = \{\alpha_j^1 S, \alpha_j^2 S, \cdots, \alpha_j^n S, \alpha_j^1 F, \alpha_j^2 F, \cdots, \alpha_j^n F\} \tag{12}$$

where $\alpha_j^i S$ denotes a success response from teacher j, when the input given to it by the automaton is α_i and $\alpha_j^i F$ denotes a failure response from teacher j, when the input given to it by the automaton is α_i.

Definition 6: Let Ω denote the sample space of the outputs of all the teachers operating in a multiteacher environment, and it is given by

$$\Omega = \Omega_1 \times \Omega_2 \times \cdots \times \Omega_m \tag{13}$$

and $|\Omega| = (2n)^m$.

IV. Proposed Learning Algorithm

Let $p(k)$ be the action probability vector at time k and $p_i(k)$ denote the probability of selecting action α_i at time k as input for environment $j, j = 1, 2, \cdots, m$. Let the stochastic automaton select m actions as inputs for m teacher environments independently and with replacement from a set of n actions according to $p(k)$. Let $r_i(k)$ be the number of environment receiving action α_i as the input and $\Sigma_i^n r_i(k) = m$. Let $s_i(k)$ denote the number of favorable responses (rewards) obtained from the environments for action α_i. Let $f_i(k)$ denote the number of unfavorable responses (penalties) obtained from the environments for action α_i and $s_i(k) + f_i(k) = r_i(k)$. We propose the following learning algorithm for a stochastic automaton operating in a general multiteacher environment as follows. For any i $(i = 1, 2, \cdots, n)$

$$\begin{aligned} p_i(k+1) = {} & p_i(k) + \frac{s_i(k)}{m}\left(\sum_{j\neq i}^{n} g_j(p(k))\right) \\ & - \frac{(r_i(k) - s_i(k))}{m}\left(\sum_{j\neq i}^{n} h_j(p(k))\right) \\ & - \frac{\left(\sum_{j\neq i}^{n} s_j(k)\right)}{m} g_i(p(k)) \\ & + \frac{\left(\sum_{j\neq i}^{n} (r_j(k) - s_j(k))\right)}{m} h_i(p(k)). \end{aligned} \tag{14}$$

The following assumptions are made regarding all the functions g_j and $h_j, j = 1, 2, \cdots, n$.

Assumption 1: g_j and h_j are continuous functions.

Assumption 2: g_j and h_j are nonnegative functions.

Assumption 3:

$$\begin{aligned} & 0 < g_j(p) < p_j \\ & 0 < \sum_{j\neq i}^{n} (p_j + h_j(p)) < 1 \end{aligned} \tag{15}$$

for all $p_j \in (0, 1)$ and all $j = 1, 2, \cdots, n$.

Remark 1: The Assumption 3 of (15) ensures that all the components of $p(k+1) \in (0, 1)$ when all the components of $p(k) \in (0, 1)$.

Remark 2: From the proposed general learning scheme, general learning algorithm for SISO model given by [12, Eq. (4.61)] and Baba's GAE algorithm given by [18, Eqs. (14) and (15)] can be obtained as special cases.

Note 1: Let us denote $p_i(k+1)$ of (14) for MIMO model as $p_i^{\text{MIMO}}(k+1)$. The proposed general learning scheme can be written as an average of the general learning algorithms for SISO models and is stated as a lemma as follows.

Lemma 1: The proposed general learning scheme (14) for MIMO model $p_i^{\text{MIMO}}(k+1)$ can be expressed as an average of the general learning algorithms for SISO models $p_i^{\text{SISO}}(k+1)$ given by (4.61) in [12] and it is given as follows. For all i $(i = 1, 2, \cdots, n)$

$$p_i(k+1) = \frac{1}{m}\sum_{l=1}^{m} p_l^i(k+1) \tag{16}$$

where $p_l^i(k+1)$ denotes the general learning algorithm $p_i^{\text{SISO}}(k)$ for SISO model for a single automaton operating in a single teacher-environment (Teacher l).[3]

Proof: First, let us show that the general learning scheme $p_i^{\text{SISO}}(k+1)$ for a single automaton operating in a single teacher-environment (Teacher l) given by (4.61) in [12], can be written as follows.

For all i $(i = 1, 2, \cdots, n)$

$$\begin{aligned} & p_l^i(k+1) \\ & \quad = p_l^i(k) + I_{\alpha_l^i S}\left(\sum_{j\neq i}^{n} g_j(p(k))\right) - I_{\alpha_l^i F}\left(\sum_{j\neq i}^{n} h_j\ (p(k))\right) \\ & \qquad - \sum_{j\neq i}^{n} (I_{\alpha_l^j S}) g_i(p(k)) + \sum_{j\neq i}^{n} (I_{\alpha_l^j F})\ h_i(p(k)) \end{aligned} \tag{17}$$

where $I_{\alpha_l^j S}$ is an indicator random variable denoting the occurrence of event $\alpha_l^j S$ and $I_{\alpha_l^j F}$ is an indicator random variable denoting the occurrence of event $\alpha_l^j F$. Since $I_{\alpha_l^i S}, I_{\alpha_l^i F}, I_{\alpha_l^j S}$, and $I_{\alpha_l^j F}$ are indicator random variables, we know $\Sigma_{i=1}^n (I_{\alpha_l^i S} + I_{\alpha_l^i F}) = 1$.

Let $\alpha(k) = \alpha_i$ be the action selected by the automaton and represented to the environment as input at time k. Let the response of the environment for the given input be a success signal, then $I_{\alpha_l^i S} = 1$ and $I_{\alpha_l^i F} = 0$. Also $I_{\alpha_l^j S} = I_{\alpha_l^j F} = 0$,

[3] We represent the teacher by label l for generalization purposes.

for $j \neq i$. Substituting the above in (17), we get,

$$p_l^i(k+1) = p_l^i(k) + \sum_{j\neq i}^{n} g_j(p(k)). \tag{18}$$

Similarly, we can show when $\alpha(k) = \alpha_i$ and the response of the environment for the given input is a failure signal

$$p_l^i(k+1) = p_l^i(k) - \sum_{j\neq i}^{n} h_j(p(k)). \tag{19}$$

When $\alpha(k) = \alpha_q$ and the response of the environment for the given input is a success signal

$$p_l^i(k+1) = p_l^i(k) - g_i(p(k)). \tag{20}$$

When $\alpha(k) = \alpha_q$ and the response of the environment for the given input is a failure signal

$$p_l^i(k+1) = p_l^i(k) + h_i(p(k)). \tag{21}$$

Hence, we obtain the SISO algorithm given by (4.61) in [12]. Now using (17) we can write the (16) as

$$\begin{aligned} p_i(k+1) \\ = \frac{1}{m}\Bigg\{ &\left(\sum_{l=1}^{m} p_l^i(k)\right) + \sum_{l=1}^{m} I_{\alpha_l^i S}\left(\sum_{j\neq i}^{n} g_j(p(k))\right) \\ &- \sum_{l=1}^{m} I_{\alpha_l^i F}\left(\sum_{j\neq i}^{n} h_j(p(k))\right) \\ &- \sum_{l=1}^{m}\sum_{j\neq i}^{n} (I_{\alpha_l^j S}) g_i(p(k)) \\ &+ \sum_{l=1}^{m}\sum_{j\neq i}^{n} (I_{\alpha_l^j F}) h_i(p(k)) \Bigg\}. \end{aligned} \tag{22}$$

Since the action probability is the same for all teachers, we have

$$p_1^i(k) = p_2^i(k) = \cdots = p_n^i(k) = p_i(k). \tag{23}$$

Substituting the above (23) in (22) and simplifying, we get

$$\begin{aligned} p_i(k+1) \\ = p_i(k) + \frac{1}{m}\Bigg\{ &+\left(\sum_{l=1}^{m} I_{\alpha_l^i S}\right)\left(\sum_{j\neq i}^{n} g_j(p(k))\right) \\ &- \left(\sum_{l=1}^{m} I_{\alpha_l^i F}\right)\left(\sum_{j\neq i}^{n} h_j(p(k))\right) \\ &- \left(\sum_{l=1}^{m}\sum_{j\neq i}^{n} (I_{\alpha_l^j S})\right) g_i(p(k)) \\ &+ \left(\sum_{l=1}^{m}\sum_{j\neq i}^{n} (I_{\alpha_l^j F})\right) h_i(p(k)) \Bigg\}. \end{aligned} \tag{24}$$

Also, $\Sigma_{l=1}^{m} I_{\alpha_l^i S}$ denotes the total number of successes due to action i and is given by

$$\sum_{l=1}^{m} I_{\alpha_l^i S} = s_i(k). \tag{25}$$

$\Sigma_{i=1}^{m} I_{\alpha_l^i F}$ denotes the total number of failures due to action i and is given by

$$\sum_{l=1}^{m} I_{\alpha_l^i F} = f_i(k) = r_i(k) - s_i(k). \tag{26}$$

$\Sigma_{l=1}^{m} \Sigma_{j\neq i}^{n} (I_{\alpha_l^j S})$ denotes the total number of successes due to other actions j and is given by

$$\sum_{l=1}^{m}\sum_{j\neq i}^{n} (I_{\alpha_l^j S}) = \sum_{j\neq i}^{n} s_j(k). \tag{27}$$

$\Sigma_{l=1}^{m} \Sigma_{j\neq i}^{n} (I_{\alpha_l^j F})$ denotes the total number of failures due to other actions j and is given by

$$\sum_{l=1}^{m}\sum_{j\neq i}^{n} (I_{\alpha_l^j F}) = \sum_{j\neq i}^{n} f_j(k) = \sum_{j\neq i}^{n} (r_j(k) - s_j(k)). \tag{28}$$

Substituting (25)–(28) in (24) we obtain (14). □

The following theorem can be stated for the proposed learning algorithm to show it is absolutely expedient.

Theorem 1: The general learning scheme given by (14) and used by a single automaton operating in a multiteacher environment is absolutely expedient, if and only if, the $g_i(\cdot)$ and $h_i(\cdot)$ satisfy the following symmetry conditions:[4]

$$\frac{g_1(p)}{p_1} = \frac{g_2(p)}{p_2} = \cdots = \frac{g_n(p)}{p_n} = \lambda(p) \tag{29}$$

and

$$\frac{h_1(p)}{p_1} = \frac{h_2(p)}{p_2} = \cdots = \frac{h_n(p)}{p_n} = \mu(p). \tag{30}$$

Proof: Using (9), we can write the average weighted penalty $AM(k+1)$ at stage k for a given action vector $p(k+1)$ in the multiteacher environment as

$$AM(k+1) = \frac{1}{m}\left(\sum_{i=1}^{n} p_i(k+1)(c_1^i + \cdots + c_m^i)\right). \tag{31}$$

Using Lemma 1, we can write the above (31) as

$$AM(k+1) = \frac{1}{m}\sum_{i=1}^{n}\left(\frac{1}{m}\sum_{l=1}^{m} p_l^i(k+1)(c_1^i + \cdots + c_m^i)\right). \tag{32}$$

Rearranging the order of summation and taking conditional expectation with respect to $p(k)$, we get

$$\begin{aligned} E(AM(k+1)|p(k)) \\ = E\Bigg(\frac{1}{m}\sum_{l=1}^{m}\frac{1}{m} \\ \cdot\left(\sum_{i=1}^{n} p_l^i(k+1)(c_1^i + \cdots + c_m^i)\right)\Bigg| p(k)\Bigg). \end{aligned} \tag{33}$$

[4] p denotes $p(k)$.

Since expectation is a linear operator, we get

$$\begin{aligned} &E(AM(k+1)|p(k)) \\ &\quad = \frac{1}{m}\sum_{l=1}^{m}\frac{1}{m} \\ &\qquad \cdot E\left(\left(\sum_{i=1}^{n} p_l^i(k+1)(c_1^i+\cdots+c_m^i)\right)\middle|p(k)\right). \quad (34)\end{aligned}$$

We know from [12] that the given learning algorithm $p_i^{\mathrm{SISO}}(k)$ is absolutely expedient for all possible sets[5] of $\{c_j^i i\}$ $(j = 1,2,\cdots,m),(i=1,2,\cdots,n)$. Absolutely expedient for SISO implies

$$E(M_{SISO}(k+1)|p(k)) < E(M_{SISO}(k)|p(k)) \quad (35)$$

iff (29) and (30) are satisfied.

The above (35) can be written as

$$\begin{aligned} &E\left(\sum_{i=1}^{n} c_i p_i^{\mathrm{SISO}}(k+1)|p_i(k)\right) \\ &\qquad < E\left(\sum_{i=1}^{n} c_i p_i^{\mathrm{SISO}}(k)|p_i(k)\right) \quad (36)\end{aligned}$$

for all k, all $p_i(k) \in (0,1)$, $(i = 1,2,\cdots,n)$ and for all possible sets $\{c_i\}$ $(i=1,2,\cdots,n)$. Hence using (35) in (34), we get

$$\begin{aligned} E(AM(k+1)|p(k)) &< \frac{1}{m}\sum_{l=1}^{m}\frac{1}{m} \\ E&\left(\left(\sum_{i=1}^{n} p_l^i(k)(c_1^i+\cdots+c_m^i)\right)|p(k)\right). \quad (37)\end{aligned}$$

Using (8), we can simplify the (37) and get

$$E(AM(k+1)|p(k)) < \frac{1}{m}\sum_{l=1}^{m}\frac{1}{m}E(M_l(k)|p(k)) \quad (38)$$

simplifying further, we get

$$E(AM(k+1)|p(k)) < E(AM(k)|p(k)). \quad (39)$$

We can now rewrite the learning algorithm given by (14) in terms of $\lambda(p)$ and $\mu(p)$ as given by Lemma 2.

Lemma 2: Let $\delta p(k) = p_i(k+1) - p_i(k)$ denote the change in action probability at stage k, then using Theorem 1, we can write the proposed learning algorithm equation (14) in a simpler vector form as follows:

$$\begin{aligned}\delta p(k) = \frac{\lambda(p(k))}{m}&\begin{bmatrix} s_1(k) - p_i(k)s_{\mathrm{total}}(k) \\ \vdots \\ s_n(k) - p_n(k)s_{\mathrm{total}}(k)\end{bmatrix} \\ +\frac{\mu(p(k))}{m}&\begin{bmatrix} -f_1(k) + p_1(k)f_{\mathrm{total}}(k) \\ \vdots \\ -f_n(k) + p_n(k)f_{\mathrm{total}}(k)\end{bmatrix} \quad (40)\end{aligned}$$

where $s_{\mathrm{total}}(k) = \Sigma_{i=1}^{n} s_i(k)$ and $f_{\mathrm{total}}(k) = \Sigma_{i=1}^{n} f_i(k), \lambda(p)$[6] and $\mu(p)$ are continuous functions satisfying the following conditions given by (41) to make $0 \leq p(k) \leq 1$, for all values of $k, k = 1,2,\cdots$

$$\begin{aligned} &0 < \lambda(p) < 1 \\ &0 \leq \mu(p) < \min_i\left\{\frac{p_i}{1-p_i}\right\}. \quad (41)\end{aligned}$$

Proof: Substituting (29) and (30) into (14) and arranging in vector form, we get (40). The Assumptions 1 to 3 given by (15) made on functions g_j and h_j translates into conditions on $\lambda(p)$ and $\mu(p)$.

Remark 3: If we consider $p(k+1)$ as an action vector generated from $p(k)$, then the probability with which $p(k+1)$ occurs is given by

$$\begin{aligned}&\mathrm{Prob}(p(k+1)|p(k)) \\ &\quad = \frac{m!\prod_{l=1}^{n}\prod_{q=1}^{m}\left(p_l(k)d_q^l(k)\right)^{I_{\alpha_q^l S}}\left(p_l(k)c_q^l(k)\right)^{I_{\alpha_q^l F}}}{\prod_{i=1}^{n} s_l(k)!\prod_{l=1}^{n} f_l(k)!}. \quad (42)\end{aligned}$$

To show that the proposed learning algorithm is ϵ-optimal we need additional assumptions regarding the function $\lambda(p)$ and the penalty probabilities. Let us make the following assumptions regarding the $\lambda(p)$ and the penalty probabilities.

Assumption A:

$$\lambda(p) = 0, \text{only if } p \text{ is a unit vector.} \quad (43)$$

Assumption B: The average penalty probabilities $ac_1, ac_2, \cdots, ac_n$ are distinct.

Also, to prove ϵ-optimality, we need to show

1) the vertices of s_n are the only absorbing states;
2) the process $\{p(k)\}$ converges *w.p. 1*.

Let us now show that the proposed algorithm satisfies the above two properties.

Lemma 3: Under the assumptions A and B, The set of unit n-vectors of s_n form the set of all absorbing states of the Markov process $\{p(k), k \geq 0.\}$ generated by the absolutely expedient learning algorithm (40).

Proof: Let $s(k) = [s_1(k)\cdots s_n(k)]^T$ and $f(k) = [f_1(k)\cdots f_n(k)]^T$ and then the absolutely expedient scheme of (40) in vector form can be written as

$$\begin{aligned}\delta p(k) = &\frac{\lambda(p(k))}{m}(s(k) - p(k)s_{\mathrm{total}}(k)) \\ &- \frac{\mu(p(k))}{m}(f(k) - p(k)f_{\mathrm{total}}(k)). \quad (44)\end{aligned}$$

This occurs with probability

$$\begin{aligned}&\mathrm{Prob}(p(k+1)|p(k)) \\ &\quad = \frac{m!\prod_{l=1}^{n}\prod_{q=1}^{m}\left(p_l(k)d_q^l(k)\right)^{I_{\alpha_q^l S}}\left(p_l(k)c_q^l(k)\right)^{I_{\alpha_q^l F}}}{\prod_{i=1}^{n} s_l(k)!\prod_{l=1}^{n} f_l(k)!}. \quad (45)\end{aligned}$$

[5] We omit cases for which the average penalties are equal for all actions.

[6] $\lambda(p)$ denotes $\lambda(p(k))$ for all $k, k = 1,2,\cdots$ and $\mu(p)$ denotes $\mu(p(k))$ for all $k, k = 1,2,\cdots$

Suppose $p(k) = e_i, e_i$ is unit vector i of simplex s_n. Then only action α_i will be selected and therefore

$$\left.\begin{aligned} s_l(k) &= 0 \\ f_l(k) &= 0 \end{aligned}\right\} \quad \text{for } l \neq i \tag{46}$$

and

$$\begin{aligned} s_{\text{total}}(k) &= s_i(k) \\ f_{\text{total}}(k) &= f_i(k) \end{aligned} \tag{47}$$

also

$$\begin{aligned} p_i(k) &= 1 \\ p_l(k) &= 0. \end{aligned} \tag{48}$$

Substituting (46)–(48) in (44) we get $\delta p(k) = 0$, for all possible values of $s_i(k)$ and $f_i(k)$, thus $\delta p(k) = 0$, *w.p. 1.* Hence all unit vectors of simplex s_n are absorbing states.

To show that there are no other absorbing states of $\{p(k)\}$, observe that, $\delta p_i(k) = 0$ can be written as

$$p_i(k) = \frac{-\lambda(p(k))s_i(k) + \mu(p(k))f_i(k)}{-\lambda(p(k))s_{\text{total}}(k) + \mu(p(k))f_{\text{total}}(k)} \tag{49}$$

and this occurs with probability

$$\begin{aligned} &\text{Prob}(p(k+1)|p(k)) \\ &= \frac{m! \prod_{l=1}^{n} \prod_{q=1}^{m} \left(p_l(k)d_q^l(k)\right)^{I_{\alpha_q^l S}} \left(p_l(k)c_q^l(k)\right)^{I_{\alpha_q^l F}}}{\prod_{i=1}^{n} s_l(k)! \prod_{l=1}^{n} f_l(k)!}. \end{aligned} \tag{50}$$

An absorbing state corresponds to the value of $p(k)$ for which

$$p(k+1) = p(k) w.p.1 \tag{51}$$

For (51) to be satisfied, at least one of the following two conditions should be satisfied:

- (49) should hold for all values $s_i, f_i, s_{\text{total}}$ and f_{total};
- the probability given by (50) should be zero.

If $p(k)$ is not equal to unit vector, then there exists at least one $p_i(k)$ such that $0 < p_i(k) < 1$. Since $p_i(k) \neq 1$, $s_{\text{total}}(k) \neq s_i(k)$ and $f_{\text{total}}(k) \neq f_i(k)$ *w.p. 1.* Hence different values for $s_i(k), f_i(k), s_{\text{total}}(k)$, and $f_{\text{total}}(k)$ are possible. Now the (49) should satisfy for all possible values of $s_i(k), f_i(k), s_{\text{total}}(k)$ and $f_{\text{total}}(k)$ which is not possible. Also the probability given by (50) is not zero, hence there can be no other absorbing states. □

Note 2: The Assumption B is not needed for the above proof.

Lemma 4: Under Assumptions A and B, the Markov process $\{p(k)\}$ converges *w.p. 1* to the set of unit n-vectors.

Proof: Using Lemma 1, we can write

$$p_i(k) = \frac{1}{m} \sum_{l=1}^{m} p_l^i(k). \tag{52}$$

From [12] we know each $p_l^i(k)$ converges *w.p. 1*, this implies $\Sigma_{l=1}^m p_l^i(k)$ also converges *w.p. 1.* □

The following theorem can now be stated for the proposed learning algorithm to show it is ϵ-optimal.

TABLE I
(CA3): Three-Teacher Environment with Teacher 1–3.
(CA1): Single-Teacher Environment with Teacher 1

	Penalty Probabilities				
	c_j^1	c_j^2	c_j^3	c_j^4	c_j^5
Teacher 1	0.27	0.65	0.88	0.57	0.46
Teacher 2	0.35	0.58	0.29	0.87	0.77
Teacher 3	0.12	0.75	0.69	0.38	0.65
Teacher 4	0.52	0.46	0.75	0.78	0.39
Teacher 5	0.18	0.32	0.83	0.65	0.78

TABLE II
(CB3): Three-Teacher Environment with Teacher 1–3.
(CB1): Single-Teacher Environment with Teacher 1

	Penalty Probabilities				
	c_j^1	c_j^2	c_j^3	c_j^4	c_j^5
Teacher 1	0.55	0.29	0.38	0.76	0.91
Teacher 2	0.26	0.15	0.63	0.59	0.86
Teacher 3	0.94	0.21	0.66	0.48	0.38
Teacher 4	0.77	0.26	0.48	0.30	0.61
Teacher 5	0.39	0.12	0.87	0.56	0.47

Theorem 2: Suppose that $\lambda(p(k)) = \theta\lambda_1(p(k))$ and $\mu(p(k)) = \theta\mu_1(p(k))$, $0 \leq \theta < 1$. $\lambda_1(p(k))$ and $\mu_1(p(k))$ are bounded functions which satisfy the conditions given by (41) and if $\lambda_1(p(k)) + \mu_1(p(k)) > 0 \; \forall \, p(k) \in [s_n - \{e_i\}_{i=1}^n]$. Then the stochastic automaton with the given learning algorithm is ϵ-optimal in the general multiteacher environment, as $\theta \to 0$.[7]

Proof: The proof is similar to proof of ϵ-optimality in [12] with the role of c_i replaced by ac_i and we omit the proof. □

V. Simulation

In this section, we present the simulation studies of the learning behaviors of the proposed scheme under multiteacher environment. We used the same test cases as [17]. Two examples are presented, In the first example (CA5), there are five teacher-environments and five actions satisfying (2). In the other example (CB5) there are five teacher environments and five actions satisfying the more restrictive (1). In both examples the learning behaviors of single-teacher and three-teacher environment are simulated and compared with the five teacher environment. We simulated the proposed learning algorithm for the reward-inaction case to compare with the GL_{R-I} scheme, under the conditions $\lambda = 0.04$ and $\mu = 0$. The penalty probabilities are given in Tables I and II. We averaged the action probabilities and total weighted rewards every five steps to smooth the plots. The total number of steps used in simulation is 500 and the number of runs is 100.

Example 1: The proposed learning algorithm is simulated for the five teacher-environment model (CA5). The learning behavior of the above simulation is compared with the simulation results of three-teacher environment model (CA3) and the single-teacher environment model (CA1). The penalty probabilities for CA1, CA3, and CA5 cases are given in

[7] It is shown in [9] that absolutely expedient algorithms are ϵ-optimal in all stationary random environments. If the above argument is used then there is no need to prove Theorem 2.

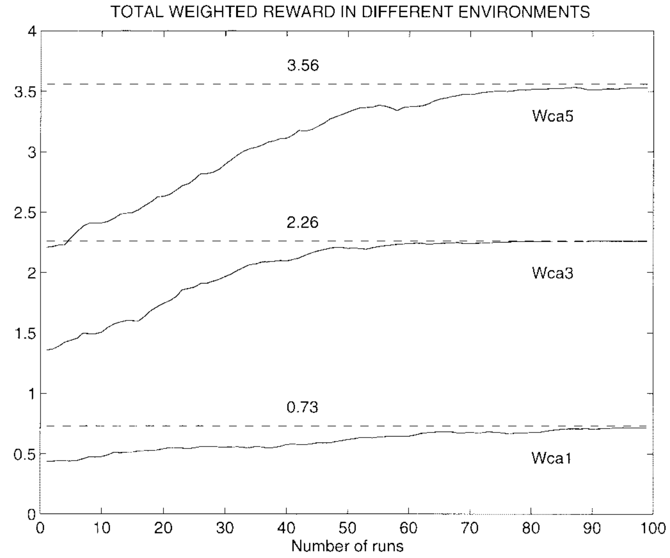

Fig. 2. Total weighted reward for cases CA1, CA3, and CA5.

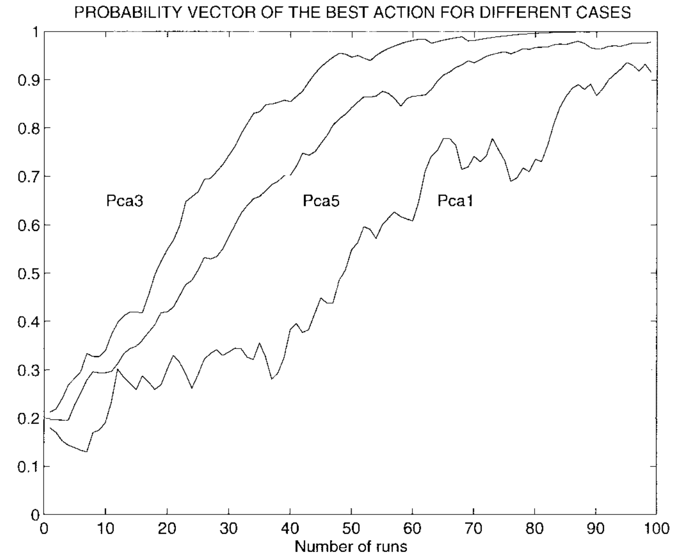

Fig. 3. Penalty probabilities of best action for cases CA1, CA3, and CA5.

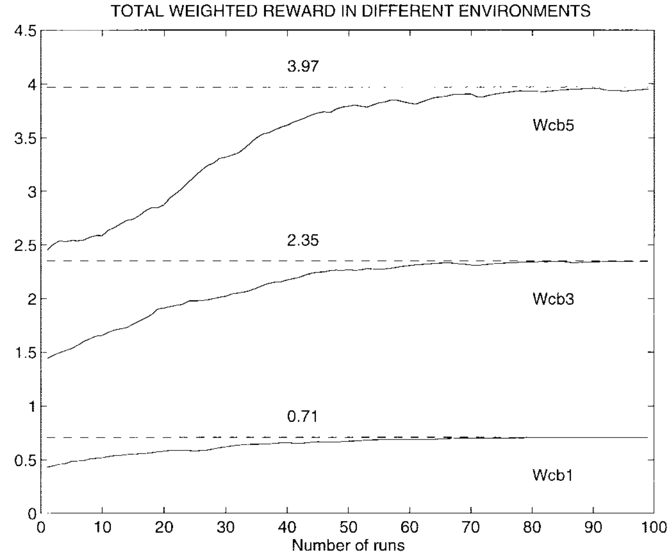

Fig. 4. Total weighted reward for cases CB1, CB3, and CB5.

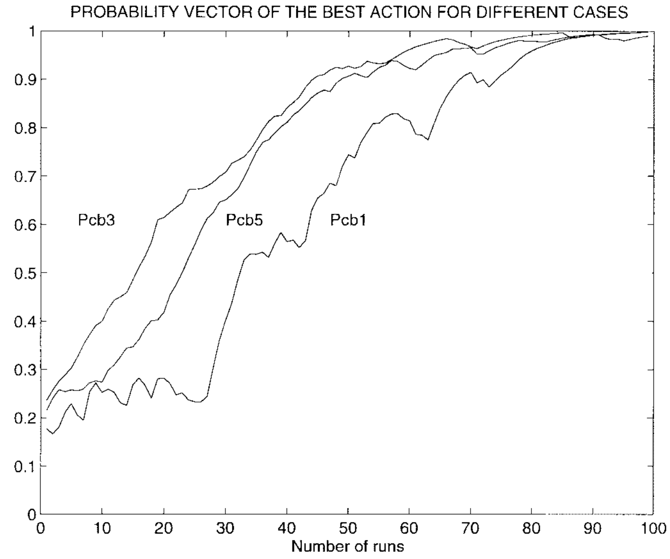

Fig. 5. Penalty probabilities of best action for cases CB1, CB3, and CB5.

Table I. The changes in the total weighted reward $W(k)$[8] for cases CA1, CA3, and CA5 are given by $\mathrm{W_{ca1}}$, $\mathrm{W_{ca3}}$, and $\mathrm{W_{ca5}}$, and these are shown in Fig. 2. The probability vector of the best action (in this case the best action is action 1) for cases CA1, CA3, and CA5 are given by $\mathrm{P_{ca1}}$, $\mathrm{P_{ca3}}$, and $\mathrm{P_{ca5}}$, and these shown in Fig. 3.

Note 3: The speed of convergence[9] depends principally on the difference between the minimum average penalty probability ac_l and the next higher average penalty probability. In this example, CA3 has the highest convergence rate followed by CA5 and CA1. This fact is reflected in Fig. 3.

Example 2: The proposed learning algorithm is simulated for the five-teacher environment model (CB5). The learning behavior of the above simulation is compared with the simulation results of three-teacher environment model (CB3) and the single-teacher environment model (CB1). The penalty probabilities for CB1, CB3, and CB5 cases are given in Table II. The changes in the total weighted reward $W(k)$ for cases CB1, CB3, and CB5 are given by $\mathrm{W_{cb1}}$, $\mathrm{W_{cb3}}$, and $\mathrm{W_{cb5}}$, and these are shown in Fig. 4. The probability vector of the best action (in this case, the best action is action 2) for cases CB1, CB3, and CB5 are given by $\mathrm{P_{cb1}}$, $\mathrm{P_{cb3}}$, and $\mathrm{P_{cb5}}$, and these are shown in Fig. 5.

Note 4: In this example CB3 has the highest convergence rate followed by CB5 and CB1.

Remark 4: The above two experimental results show that the proposed MIMO algorithm has a nice convergence property, and converges very close to optimality.

Remark 5: The above two experimental results show that the proposed MIMO algorithm has comparable convergence properties as the Baba GL_{R-I} algorithm.

[8] Total weighted reward is given by $W(k) = \Sigma_{i=1}^{n} p_i(k)(c_1^i + \cdots + c_m^i)$.

[9] The authors would like to thank the anonymous reviewer for pointing out this fact.

VI. Conclusion

The learning behaviors of variable structure stochastic automaton operating in a P-type multiteacher environment have been considered. The learning algorithms were classified based on the number of actions given as inputs to the multiteacher environment and the number of outputs obtained from the environments. Based on this type of classification the learning algorithms in the literatures can be classified as SISO and SIMO algorithms. A general class of nonlinear algorithms for MIMO model was proposed. It has been shown that the algorithm for MIMO model is absolutely expedient and ϵ-optimal in the general multiteacher environment. The proposed algorithm is a generalized version of nonlinear reward-penalty learning algorithm. From this algorithm, Baba's GAE [16] scheme and SISO scheme can be obtained as special cases. We have also shown that the proposed algorithm is a average of the sum of SISO algorithms and the algorithm converges to the action with the least average penalty in the ϵ-optimal sense. Simulation study was done for the test case given by [17] and the simulation indicates that the proposed learning algorithm has nice convergence properties. It appears that this work can be further extended by assigning weights to different teachers. In many real-world problems not all the teacher environments are equally reliable. Hence by assigning weights we can find the best action with the least weighted average penalty in the ϵ-optimal sense.

References

[1] M. L. Tsetlin, "On the behavior of finite automata in random media," *Automat. Remote Contr.*, vol. 22, pp. 1210–1219, 1962.

[2] V. I. Varshavskii and I. P. Vorontsova, "On the behavior of stochastic automata with a variable structure," *Automat. Remote Contr.*, vol. 24, pp. 327–333, 1963.

[3] M. F. Norman, "On linear models with two absorbing barriers," *J. Math. Psychol.*, vol. 5, pp. 225–241, 1968.

[4] M. F. Norman, *Markov Processes and Learning Models.* New York: Academic, 1972.

[5] ——, "Markov learning process," *SIAM Rev.*, vol. 16, pp. 143–162, 1974.

[6] R. Viswanathan and K. S. Narendra, "A note on linear reinforcement scheme for variable structure stochastic automata," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-2, pp. 292–294, 1972.

[7] Y. Sawaragi and N. Baba, "A note on the learning behavior of variable structure stochastic automata," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-3, pp. 644–647, 1973.

[8] S. Lakshmivarahan and M. A. L. Thathachar, "Absolutely expedient learning algorithms for stochastic automata," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-3, pp. 281–286, 1973.

[9] ——, "Bounds on the convergence probabilities of learning automata," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-6, pp. 756–763, 1976.

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

[11] K. S. Narendra and M. A. L. Thathachar, "Learning automata—A survey," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-14, pp. 323–334, 1974.

[12] K. S. Narendra and M. A. L. Thathachar, *Learning Automata—An Introduction.* Englewood Cliffs, NJ: Prentice-Hall, 1989.

[13] D. E. Koditschek and K. S. Narendra, "Fixed structure automata in a multiteacher environment," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-7, pp. 616–624, 1977.

[14] M. A. L. Thathachar and R. Bhakthavathsalam, "Learning automaton operating in parallel environments," *J. Cybern. Inf. Sci.*, vol. 1, pp. 121–127, 1978.

[15] M. A. L. Thathachar and K. R. Ramakrishnan, "A hierarchical system of learning automata," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-11, pp. 236–241, 1981.

[16] N. Baba, "The absolutely expedient nonlinear reinforcement schemes under the unknown multiteacher environment," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-13, pp. 100–108, 1983.

[17] ——, "On the learning behaviors of variable-structure stochastic automaton in the general N-teacher environment," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-13, pp. 224–231, 1983.

[18] ——, "New topics in learning automata theory and applications," *Lecture Notes in Control and Information Sciences.* Berlin, Germany: Springer-Verlag, 1984, vol. 71.

[19] ——, "Learning behaviors of hierarchical structure stochastic automata operating in a general multiteacher environment," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, pp. 585–587, 1985.

[20] ——, "Three approaches for solving the stochastic multiobjective programming problem," *Stochastic Optimization, Numerical Methods and Technical Applications, Proceedings, Neubiberg, FRG, May 29–31, 1990, Lecture Notes in Economics and Mathematical Systems.* Berlin, Germany: Springer-Verlag, 1990, vol. 379, pp. 93–109.

[21] Y. M. El-Fattah, "Stochastic automata models of certain problems of collective behavior," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-10, pp. 304–314, 1980.

[22] O. V. Nedzelnitsky, Jr., and K. S. Narendra, "Nonstationary models of learning automata routing in data communication networks," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-17, pp. 1004–1015, 1987.

[23] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis.* New York: Wiley, 1973

[24] A. G. Barto and P. Anandan, "Pattern-recognizing stochastic learning automata," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, May/June 1985.

[25] K. Najim and A. S. Poznyak, *Learning Automata: Theory and Applications.* New York: Pergamon, 1994.

Arif Ansari (S'91–M'92) was born in Madras, India. He received the B.S. degree in electrical engineering from Anna University, Madras, in 1986, the M.Tech degree in controls systems from the Indian Institute of Technology, Madras, in 1987, and the M.S. degree in applied mathematics from the University of Southern California (USC), Los Angeles, in 1992. He is currently pursuing the Ph.D. degree in stochastic learning algorithms at USC

From 1988 to 1989, he was a Research Assistant, Rensselaer Polytechnic Institute, Troy, NY, and worked on the NASA SCOLE project. Currently, he is teaching statistics in the Marshall Business School, USC. His research interests include model reference adaptive control, neural networks, fuzzy logic, stochastic learning algorithms, and financial engineering.

George P. Papavassilopoulos (SM'96) received the Diploma degree in mechanical and electrical engineering from NTVA, Greece, in 1975 and the Ph.D. degree in 1979 from the University of Illinois, Urbana-Champaign.

He is a Professor of electrical engineering systems at the University of Southern California (USC), Los Angeles. His research interests are in the areas of game theory, optimization, learning algorithms, BMI's, and parallel algorithms for nonconvex problems with applications in engineering and economics.