Decentralized Learning of Nash Equilibria in Multi-Person Stochastic Games With Incomplete Information

P. S. Sastry, *Member, IEEE*, V. V. Phansalkar, and M. A. L. Thathachar, *Fellow, IEEE*

Abstract—**A multi-person discrete game where the payoff after each play is stochastic is considered. The distribution of the random payoff is unknown to the players and further none of the players know the strategies or the actual moves of other players. A learning algorithm for the game based on a decentralized team of Learning Automata is presented. It is proved that all stable stationary points of the algorithm are Nash equilibria for the game. Two special cases of the game are also discussed, namely, game with common payoff and the relaxation labelling problem. The former has applications such as pattern recognition and the latter is a problem widely studied in computer vision. For the two special cases it is shown that the algorithm always converges to a desirable solution.**

I. Introduction

In this paper we consider decentralized algorithms for learning Nash equilibria in general multiperson stochastic games with incomplete information. We will be concerned with Learning Automata models [1] for solving this problem.

The game we consider is a discrete stochastic game played by N players. Each of the players has finitely many actions one of which he plays at each instant. After each play, the payoffs to individual players are random variables. Further, the game is one of incomplete information [2]. Thus, nothing is known regarding the distributions of the random payoffs. For learning optimal strategies, the game is played repeatedly. We are interested in (asymptotically) learning equilibrium strategies, in the sense of Nash, with respect to the expected value of the payoff. Our interest will be in decentralized learning algorithms. Hence, after each play, each of the players updates his strategy based solely on his current action or move and his payoff. None of the players has any information regarding the other players. As a matter of fact, none of the players need to even know the existence of other players. Thus the game we tackle is also of imperfect information [2].

Such games are useful in tackling problems in many areas such as decentralized control, optimization, pattern recognition and computer vision. In many such problems Nash equilibria, in fact, represent the desired solutions. (For a good discussion on the rationality of Nash equilibria see [3, Ch. 2]).

Manuscript received July 18, 1992; revised June 11, 1993. This work was supported by the Office of Naval Research Grant No. N 00014-92-J-1324 under an Indo-U.S. project.

The authors are with the Department of Electrical Engineering, Indian Institute of Science, Bangalore-560 012.

IEEE Log Number 9400634.

We use a team of learning automata [1] for evolving to the optimal strategies. Games of learning automata have been used as models for adaptive decision making in many applications [4]–[6]. In Learning Automata theory, algorithms for learning optimal strategies have been developed for many special types of finite stochastic games. Some of the models considered are: Two person zero-sum games [7], [8], N-person games with common payoff [4], [9, [10], [11] and non-cooperative games such as Prisoner's Dilemma and Stackelberg games [12], [13]. In this paper, we consider a team of learning automata involved in an N-person game with different payoffs to different players, in general. We will investigate the asymptotic behaviour of a class of learning algorithms which are implementable in a decentralized fashion. We prove, using weak convergence techniques [14], [15], that all the stable stationary points of the algorithm are Nash equilibria. (See Remark 3.1 where the notion of this convergence is explained). We show that most of the known results for automata games can be derived as special cases of our general result.

We begin by formulating the learning problem in Section II. Section III presents the learning algorithm and its analysis. Section IV discusses a special case of the game where all players get identical payoff. This model has been used in many pattern recognition problems [4], [11]. We derive the currently known results for this game with common payoff as special cases of our result and further show that we can get stronger results based on our analysis. Another interesting problem that can be considered as an instance of the N-person game, is the consistent labelling problem. A team of automata algorithm for relaxation labelling was presented in [6] which was successfully employed for many computer vision problems [16], [17]. The analysis presented in [6] leaves some questions regarding the long term behaviour of the algorithm unanswered. Here, in Section V, we show that our result gives a complete characterization of the relaxation labelling algorithm. Section 6 presents a discussion of the results of the paper and Section 7 concludes the paper.

II. Problem Formulation

In this section we introduce our notation and derive a few results regarding Nash equilibria which will be used later on in the analysis of our algorithm. Most of the formulation in this section can be found in any standard book on Game Theory (e.g., [18]–[20]).

Consider a N-person stochastic game. Each player i has a finite set of actions or pure strategies, S_i, $1 \leq i \leq N$. Let cardinality of S_i be m_i, $1 \leq i \leq N$. (It may be noted that the sets S_i, $1 \leq i \leq N$, need not have any common elements and we assume no structure on these sets). Each play of the game consists of each of the players choosing an action. The result of each play is a random payoff to each player. Let r_i denote the random payoff to player i, $1 \leq i \leq N$. It is assumed that $r_i \in [0,1]$. Define functions $d^i : \Pi_{j=1}^N S_j \rightarrow [0,1]$, $1 \leq i \leq N$, by

$$d^i(a_1, \ldots, a_N) = E[r_i | \text{player } j \text{ chose action } a_j, \quad a_j \in S_j, 1 \leq j \leq N] \tag{1}$$

The function d^i is called the payoff function or utility function of player i, $1 \leq i \leq N$. The objective of each player is to maximize his payoff. A strategy for player i is defined to be a probability vector $\mathbf{q}_i = [q_{i1}, \ldots, q_{im}]^t$, where player i chooses action j with probability q_{ij}. The strategy of a player can be time varying, as it would be, for example, during learning. Each of the pure strategies or actions of the i^{th} player can be considered as a strategy. Let $\mathbf{e}_i$ be a unit probability vector (of appropriate dimension) with i^{th} component unity and all others zero. Then $\mathbf{e}_i$ is the strategy corresponding to the action i. (It may be noted that any unit probability vector represents a pure strategy). A strategy that does not correspond to a pure strategy is called a mixed strategy.

Given the actions chosen by the players, (1) specifies the expected payoff. We can easily extend the functions d^i, to the set of all strategies. If g^i is this extension, then it is defined as

$$\begin{aligned} g^i(\mathbf{q}_1, \ldots, \mathbf{q}_N) &= E[r_i \mid j^{\text{th}} \text{ player employs strategy } \mathbf{q}_j, 1 \leq j \leq N] \\ &= \sum_{j_1, \ldots, j_N} d^i(j_1, \ldots, j_N) \prod_{s=1}^{N} q_{sj_s} \end{aligned} \tag{2}$$

Definition 2.1 The N-tuple of strategies $(\mathbf{q}_1^0, \ldots, \mathbf{q}_N^0)$ is said to be a **Nash equilibrium**, *if for each i, $1 \leq i \leq N$, we have*

$$\begin{aligned} &g^i(\mathbf{q}_1^0, \ldots, \mathbf{q}_{i-1}^0, \mathbf{q}_i^0, \mathbf{q}_{i+1}^0, \ldots, \mathbf{q}_N^0) \\ &\quad \geq g^i(\mathbf{q}_1^0, \ldots, \mathbf{q}_{i-1}^0, \mathbf{q}, \mathbf{q}_{i+1}^0, \ldots, \mathbf{q}_N^0) \\ &\quad \forall \text{ probability vectors } \mathbf{q} \in [0,1]^{m_i} \end{aligned} \tag{3}$$

In general, each $\mathbf{q}_i^0$ above will be a mixed strategy and then we refer to $(\mathbf{q}_1^0, \ldots, \mathbf{q}_N^0)$, satisfying (3), as a Nash equilibrium in mixed strategies. Every N-person game will have at least one Nash equilibrium in mixed strategies [3, 18].

We say we have a Nash equilibrium in pure strategies if $(\mathbf{q}_1^0, \ldots, \mathbf{q}_N^0)$ is a Nash equilibrium with each $\mathbf{q}_i^0$ a unit probability vector. In view of (2), for verifying a Nash equilibrium in pure strategies, we can simplify the condition (3) as given in the definition below.

Definition 2.2 The $N-$tuple of actions $(a_1^0, \ldots, a_N^0)$ (or equivalently the set of strategies $(\mathbf{e}_{a_1^0}, \ldots, \mathbf{e}_{a_N^0})$) is called a **Nash equilibrium in pure strategies** *if for each i, $1 \leq i \leq N$,*

$$\begin{aligned} &d^i(a_1^0, \ldots, a_{i-1}^0, a_i^0, a_{i+1}^0, \ldots, a_N^0) \\ &\quad \geq d^i(a_1^0, \ldots, a_{i-1}^0, a_i, a_{i+1}^0, \ldots, a_N^0), \forall a_i \in S_i. \end{aligned} \tag{4}$$

Here, for all j, $a_j \in S_j$, the set of pure strategies of player j.

Definition 2.3 A Nash equilibrium is said to be **strict** *if the inequalities in (3)(equivalently, (4), for pure strategies) are strict.*

Since each of the sets S_i is finite, each of the functions, $d^i : \Pi_{j=1}^N S_j \rightarrow [0,1]$, can be represented by a hyper matrix of dimension $m_1 \times \cdots \times m_N$. These N hyper matrices together constitute what can be called the reward structure of the game. Since the game is one of incomplete information these payoff matrices are unknown. Now the learning problem for the game can be stated as follows.

Let G be a N-person stochastic game with incomplete information. At any instant k, let the strategy employed by the i^{th} player be $\mathbf{q}_i(k)$. Let $a_i(k)$ and $r_i(k)$ be the actual actions selected by i and the pay off received by i respectively at k, $k = 0, 1, 2, \ldots$. Find a decentralized learning algorithm for the players (that is, design functions T_i, where $\mathbf{q}_i(k+1) = T_i(\mathbf{q}_i(k), a_i(k), r_i(k)))$ such that $\mathbf{q}_i(k) \rightarrow \mathbf{q}_i^0$ as $k \rightarrow \infty$ where $(\mathbf{q}_1^0, \ldots, \mathbf{q}_N^0)$ is a Nash equilibrium of the game.

In the next section we present the team of automata model for solving this problem. In the remaining part of this section, we state a simple result regarding Nash equilibria, which is needed for the analysis later on. Define $\mathbf{K} \subset [0,1]^{m_1 + \cdots + m_N}$ by

$$\begin{aligned} \mathbf{K} = \{ &Q \in [0,1]^{m_1 + \cdots + m_N} : \\ &Q = (\mathbf{q}_1, \ldots, \mathbf{q}_N), \text{ and } \forall i, 1 \leq i \leq N, \\ &\mathbf{q}_i \text{ is a } m_i\text{-dimensional probability vector} \} \end{aligned} \tag{5}$$

It is easy to see that $\mathbf{K}$ is the set of all $N-$tuples of mixed strategies or the set of possible strategies for the game. Let $\mathbf{K}^* \subset \mathbf{K}$ denote the set of possible pure strategies for the game. $\mathbf{K}^*$ is defined by

$$\begin{aligned} \mathbf{K}^* = \{ &Q \in [0,1]^{m_1 + \cdots + m_N} : Q = (\mathbf{q}_1, \ldots, \mathbf{q}_N), \\ &\text{and } \forall i, 1 \leq i \leq N, \mathbf{q}_i \text{ is a} \\ &m_i\text{-dimensional probability vector} \\ &\text{with one component unity} \} \end{aligned} \tag{6}$$

It is easy to see that $\mathbf{K}^*$ can be put in one to one correspondence with the set $\prod_{j=1}^N S_j$. Hence we can think of the function d^i, given by (1) as defined on $\mathbf{K}^*$. Similarly, functions g^i, given by (2), are defined over $\mathbf{K}$. Define functions h_{is}, $1 \leq s \leq m_i$, $1 \leq i \leq N$, on $\mathbf{K}$ by

$$\begin{aligned} h_{is}(Q) &= E[r_i | \text{ player } j \text{ employed strategy } \mathbf{q}_j, \\ &\quad 1 \leq j \leq N, j \neq i, \\ &\quad \text{and player } i \text{ chose action } s] \\ &= \sum_{j_1, \ldots, j_{i-1}, j_{i+1}, \ldots, j_N} d(j_1, \ldots, j_{i-1}, s, j_{i+1}, \ldots, j_N) \prod_{s \neq i} p_{sj_s} \end{aligned} \tag{7}$$

where $Q = (\mathbf{q}_1, \ldots, \mathbf{q}_N)$. From (2) and (7), we have

$$\sum_{s=1}^{m_i} h_{is}(Q) q_{is} = g^i(Q) \tag{8}$$

Lemma 2.1 *Any* $Q^0 = (\mathbf{q}_1^0, \ldots, \mathbf{q}_N^0) \in \mathbf{K}$ *is a Nash equilibrium if and only if*

$$h_{is}(Q^0) \leq g^i(Q^0), \quad \forall s, i.$$

Corollary 2.1 *Let* $Q^0 = (\mathbf{q}_1^0, \ldots, \mathbf{q}_N^0)$ *be a Nash equilibrium. Then for each* i,

$$h_{is}(Q^0) = g^i(Q^0) \ \forall\, s \text{ such that } q_{is}^0 > 0.$$

Both the above results follow directly from the definition of Nash equilibria and are standard results in Game theory (see, for example, [20, Thm. 3.1], and [19, Ch. 3]).

III. Algorithm for Learning Nash Equilibria

We will employ a team of learning automata to evolve to Nash equilibria in the game. A learning automaton is a simple adaptive decision making device that is capable of learning the optimal action (from a finite set of actions) through interactions with a random environment. The automaton keeps a probability distribution over the set of actions and at each instant it selects one action at random based on this distribution. The environment responds with a random reaction for this choice of action. The automaton uses this reaction to update its action probability distribution through a learning algorithm and the cycle repeats. Define the optimal action to be the one with highest expected value of environmental reaction. The problem of interest in learning automata theory is the design of learning algorithms so that asymptotically the automaton chooses only the optimal action. For the purposes of this paper our interest is in the collective behaviour of a number of such automata. Here the expected value of reaction to an automaton depends on the actions of other automata also. In what follows we discuss such a model for the multiperson stochastic game. For a general introduction to learning automaton theory, the reader is referred to [1].

Recall that we consider an N-person game where the i^{th} player has m_i pure strategies. We will represent each player by a learning automaton and the actions of the automaton are the pure strategies of the player. Let $\mathbf{p}_i(k) = [p_{i1}(k) \cdots p_{im_i}(k)]^t$ denote the action probability distribution of the i^{th} automaton player. $p_{ij}(k)$ denotes the probability with which i^{th} automaton player chooses the j^{th} pure strategy at instant k. Thus $\mathbf{p}_i(k)$ is the strategy employed by the i^{th} player at instant k (same as what we called $\mathbf{q}_i$ in section II). $\mathbf{p}_i(0)$ is the initial mixed strategy of player i. In the absence of extra information, the player can assign equal probabilities to all actions. Each play of the game consists of each of the automata choosing an action independently and at random according to their current action probabilities. The payoff to the i^{th} player will be the reaction to the i^{th} automaton which will be denoted by $r_i(k)$. The learning algorithm used by each of the player is as given below.

Learning Algorithm

1) At every time step, each player (automaton) chooses an action according to his action probability vector. Thus, the i^{th} player chooses action a_i at instant k, based on the probability distribution $\mathbf{p}_i(k)$.
2) Each player obtains a payoff based on the set of all actions. The reward to player i is $r_i(k)$, given by (1).
3) Each player updates his action probability according to the rule

$$\mathbf{p}_i(k+1) = \mathbf{p}_i(k) + br_i(k)(\mathbf{e}_{a_i} - \mathbf{p}_i(k)), \quad i = 1, \ldots, N. \tag{9}$$

where $0 < b < 1$ is a parameter and $\mathbf{e}_{a_i}$ is a unit vector of appropriate dimension with a_ith component unity. (It may be recalled that we assumed $0 \leq r_i \leq 1$, $\forall i$). This algorithm is known as Linear Reward-Inaction ($L_{R\text{-}I}$) algorithm [1]. It is easy to see from (9) that this algorithm is completely decentralized.

Let $P(k) = (\mathbf{p}_1(k), \ldots, \mathbf{p}_N(k))$ denote the state of the team at instant k. As discussed in Section II, our interest is in the asymptotic behaviour of $P(k)$. The analysis presented in the next subsection addresses this question.

A. Analysis of the Learning Algorithm

Consider $P(k)$, the state of the team. Since each $\mathbf{p}_i$ is a probability vector, we have $P(k) \in \mathbf{K}$ where $\mathbf{K}$ is as defined by (5).

Under the learning algorithm specified by (9), $\{P(k), k \geq 0\}$ is a Markov process. The analysis of this process is done in two stages. In the first part we derive an ordinary differential equation (ODE) whose solution approximates the asymptotic behaviour of $P(k)$ if the parameter b used in (9) is sufficiently small. In the second part we characterize the solutions of the ODE and thus obtain the long term behaviour of $P(k)$.

Obtaining the Equivalent ODE The learning algorithm given by (9) can be represented as

$$P(k+1) = P(k) + bG(P(k), a(k), r(k)) \tag{10}$$

where $a(k) = (a_1(k) \cdots a_N(k))$ denotes the actions selected by the automata team at k and $r(k) = (r_1(k) \cdots r_N(k))$ are the resulting payoffs. $G(.,.,.)$ represents the updating specified by equation (9).

Consider a piecewise-constant interpolation of $P(k), P^b(.)$, defined by

$$P^b(t) = P(k), t \in [kb, (k+1)b) \tag{11}$$

where b is the parameter used in (9).

$P^b(.) \in D^{m_1+\cdots+m_N}$, the space of all functions from $\mathbf{R}$ into $[0,1]^{m_1+\cdots+m_N}$, which are right continuous and have left hand limits. (It may be noted that $P^b(t) \in \mathbf{K}, \forall t$). Now consider the sequence $\{P^b(.) : b > 0\}$. We are interested in the limit of this sequence as $b \to 0$.

Define the function $f : \mathbf{K} \to [0,1]^{m_1+\cdots+m_N}$ by

$$f(P) = E[G(P(k), a(k), r(k)) \mid P(k) = P] \tag{12}$$

The following theorem gives the limiting behaviour of P^b as $b \to 0$.

Theorem 3.1 *Consider the sequence of interpolated processes* $\{P^b(.)\}$. *Let* $X_0 = P^b(0) = P(0)$. *Then the sequence converges weakly, as* $b \to 0$, *to* $X(.)$ *which is the solution of the ODE,*

$$\frac{dX}{dt} = f(X), X(0) = X_0 \tag{13}$$

Proof: The theorem is a particular case of a general result due to Kushner ([15], Theorem 3.2). We note the following about the learning algorithm given by (10)

1) $\{P(k), (a(k-1), r(k-1)), k \geq 0\}$ is a Markov process. $(a(k), r(k))$ take values in a compact metric space.
2) The function $G(.,.,.)$ is bounded and continuous and it does not depend on b.
3) If $P(k) = P$, a constant, then $\{(a(k), r(k)), k \geq 0\}$ is an i.i.d. sequence. Let M^P denote the distribution of this process.
4) The ODE (13) has a unique solution for each initial condition.

Hence by [15, Thm. 3.2], the sequence $\{P^b(.)\}$ converges weakly as $b \to 0$ to the solution of the ODE,

$$\frac{dX}{dt} = \overline{G}(X), X(0) = X_0$$

where $\overline{G}(P) = E^P G(P(k), a(k), r(k))$ and E^P denotes the expectation with respect to the invariant measure M^P.

Since for $P(k) = P, (a(k), r(k))$ is i.i.d. whose distribution depends only on P and the payoff matrices, we have

$$\overline{G}(P) = E[G(P(k), a(k), r(k)) \mid P(k) = P] = f(P), \text{ by (12)}$$

Hence the theorem. □

Remark 3.1 The convergence of functionals implied by weak convergence ensured by the above Theorem, along with the knowledge of the nature of the solutions of the ODE (13), enables one to understand the long term behaviour of $P(k)$. One can show that with probability increasingly close to 1 as b decreases, $P(k)$ follows the trajectory $X(t)$ of the ODE (13), with an error bounded above by some arbitrary, apriori fixed, $\epsilon > 0$. (See [21, Chapter 2, Theorem 1] and the discussion therein). The result, for example, can be specialised to derive long term behaviour of the algorithm when started in the neighbourhood of an asymptotically stable equilibrium point.

Specifically, let $X(.)$ be the solution to the ODE (13) and suppose that the initial condition $X(0) = X_0$ is sufficiently close to an asymptotically stable stationary point of the ODE, say, $P^0 \in \mathbf{K}$.

The sequence $\{P^b\}$ is a sequence of random variables taking values in $D^{m_1+\cdots+m_N}$, the space of all right continuous functions with left hand limits defined over $[0, \infty)$ and taking values in a bounded subset of $\mathbf{R}^{m_1+\cdots+m_N}$, namely $\mathbf{K}$.

Consider a bounded continuous function $h_T(.)$ defined over $D^{m_1+\cdots+m_N}$ for every $T < \infty$ given by

$$h_T(Y) = \sup_{0 \leq t \leq T} ||Y(t) - X(t)||$$

($h_T(.)$ is bounded because in our case all functions in $D^{m_1+\cdots+m_N}$ take values in $\mathbf{K}$, a bounded set).

Because of the weak convergence result of Theorem 3.1, $E[h_T(P^b)] \to E[h_T(X)]$ as $b \to 0$, where X is the solution to the ODE. With the particular initial condition used, let P^0 be the equilibrium point to which the solution of the ODE converges. This, coupled with the nature of the interpolation, implies that for the given initial condition and any $\epsilon > 0$ and integers K_1, K_2, $0 < K_1 < K_2 < \infty$, there exists a b_0 such that

$$E[\sup_{K_1 \leq k \leq K_2} ||P(k) - P^0||] < \epsilon, \forall b < b_0 \qquad (14)$$

Thus if the ODE has an asymptotically stable (in the small) stationary point P^0, then for all initial conditions sufficiently close to it, the algorithm essentially converges to P^0. In view of this, for the rest of the analysis we analyse stability properties of the ODE and talk in terms of some points in $\mathbf{K}$ being 'stable', 'unstable' etc. for the algorithm. While this sort of convergence over compact sets is admittedly a very weak type of convergence, for stochastic algorithms of this kind this is the best result that can be hoped for [21].

Analysis of the ODE By Theorem 3.1, the limit P of the interpolated process P^b, given by (11) satisfies the ODE (13). Recall that P consists of N probability vectors. The phase space of the ODE is the set $\mathbf{K}$ defined by (5). As discussed in Section II, the points in $\mathbf{K}^*$ (where $\mathbf{K}^*$ is as defined by (6)), represent pure strategies. We refer to all points in $\mathbf{K}^*$ as corners of $\mathbf{K}$. P contains $m_1 + \cdots + m_N$ components which are denoted by p_{iq}, $1 \leq q \leq m_i$, $1 \leq i \leq N$. f also has the same number of components which will be denoted as f_{iq}. The component equations of (13) are

$$\frac{dp_{iq}}{dt} = f_{iq}(P), 1 \leq q \leq m_i, 1 \leq i \leq N. \qquad (15)$$

Using (9) and (12) we get f_{iq} as

$$\begin{aligned} f_{iq}(P) &= p_{iq}(1 - p_{iq})E[r_i \mid P, a_i = q] \\ &\quad + \sum_{s \neq q} p_{is}(-p_{iq})E[r_i | P, a_i = s] \\ &= p_{iq} \sum_s p_{is}[h_{iq}(P) - h_{is}(P)] \end{aligned} \qquad (16)$$

where the functions h_{is} are as defined by (7) and we have used the fact that $\sum_{s \neq q} p_{is} = (1 - p_{iq})$. (Recall that a_i is the action selected by the i^{th} player and r_i is his payoff).

Using (16), the ODE (13) can be written as

$$\frac{dp_{iq}}{dt} = p_{iq} \sum_s p_{is}[h_{iq}(P) - h_{is}(P)], 1 \leq q \leq m_i, 1 \leq i \leq N. \qquad (17)$$

The following theorem characterizes the solutions of the ODE and hence characterises the long term behaviour of our learning algorithm.

Theorem 3.2 The following are true of the learning algorithm (if the parameter b in (9) is sufficiently small).

1) *All corners of $\mathbf{K}$ are stationary points.*
2) *All Nash equilibria are stationary points.*
3) *All stationary points that are not Nash equilibria are unstable.*
4) *All corners of $\mathbf{K}$ that are strict Nash equilibria (in pure strategies) are asymptotically stable.*

Proof:

1) By inspection from (16), if P is a corner then $f_{iq}(P) = 0$ because at a corner, for each i, either $p_{iq} = 0$ or $p_{is} = 0, \forall s \neq q$.

2) Using (8) we can rewrite (17) as

$$\frac{dp_{iq}}{dt} = f_{iq}(P) = p_{iq}[h_{iq}(P) - g^i(P)] \tag{18}$$

Let P^0 be a Nash equilibrium. Then by Corollary 2.1, for each i, either $p^0_{iq} = 0$ or $h_{iq}(P^0) = g^i(P^0)$. Hence $f_{iq}(P^0) = 0, \forall i, q$.

3) Let P^0 be a zero of $f(P)$ which is not a Nash equilibrium. Then by Lemma 2.1, there is an i and an s such that

$$h_{is}(P^0) > g^i(P^0) \tag{19}$$

Due to the continuity of the functions involved, the inequality (19) will hold in a small open neighbourhood around P^0. This implies, by (18), that for all points in this neighbourhood $\frac{dp_{is}}{dt} > 0$ if $p_{is} \neq 0$. Hence in all sufficiently small neighbourhoods of P^0, there will be infinitely many points starting from which $P(k)$ will eventually leave the neighbourhood. This implies P^0 is unstable.

4) Let $P^0 = (\mathbf{e}_{a_1}, \ldots, \mathbf{e}_{a_N})$ be a corner of $\mathbf{K}$ that is a Nash equilibrium. Use the transformation $P \rightarrow \overline{P}$, defined by

$$\begin{aligned} \overline{p}_{iq} &= p_{iq} \text{ if } q \neq a_i \\ &= 1 - p_{ia_i} \text{ otherwise} \end{aligned} \tag{20}$$

In $\overline{P}$, only $(m_1 - 1) + \cdots + (m_N - 1)$ components are independent. Choose $\overline{p}_{iq}$, $q \neq a_i$, as the independent components. For these, we get

$$\frac{d\overline{p}_{iq}}{dt} = H_{iq}(\overline{P}) + \text{ second and higher order terms in components of } \overline{P}.$$

where, by Taylor expansion from (8), (18)

$$H_{iq}(\overline{P}) = \overline{p}_{iq}[d^i(a_1, \ldots, a_{i-1}, q, a_{i+1}, \ldots, a_N) - d^i(a_1, \ldots, a_N)]$$

Consider the Lyapunov function

$$V(\overline{P}) = \sum_{q, i: q \neq a_i} \overline{p}_{iq}.$$

We have $V(\overline{P}) \geq 0$ and is zero when $\overline{p}_{iq} = 0$ for all i, q.

$$\begin{aligned} \frac{dV(\overline{P})}{dt} &= \sum_{i,q:q\neq a_i} \frac{d\overline{p}_{iq}}{dt} \\ &= \sum_{i,q:q\neq a_i} \overline{p}_{iq}[d^i(a_1, \ldots, a_{i-1}, q, a_{i+1}, \ldots, a_N) \\ &\quad - d^i(a_1, \ldots, a_N)] + \text{ higher order terms} \\ &< 0, \end{aligned} \tag{21}$$

for all $\overline{P} \neq 0$ in a sufficiently small neighbourhood around the origin because P^0 is a strict Nash equilibrium. This proves P^0 is asymptotically stable. □

Remark 3.2 Because of the above theorem, we can conclude that our learning algorithm will never converge to a point in $\mathbf{K}$ which is not a Nash equilibrium and strict Nash equilibria in pure strategies are locally asymptotically stable. This still leaves two questions unanswered. (i) Do Nash equilibria in mixed strategies form stable attractors for the algorithm, and (ii) is it possible that $P(k)$ does not converge to a point in $\mathbf{K}$ which would be the case, for example, if the algorithm exhibits limit cycle behaviour. At present we have no results concerning the first question. Regarding the second question we provide a sufficient condition for $P(k)$ to converge to some point in $\mathbf{K}$. This is proved in Theorem 3.3 below.

Theorem 3.3 Suppose there is a bounded differentiable function

$$F : \mathbf{R}^{m_1 + \cdots + m_N} \rightarrow \mathbf{R}$$

such that for some constant $c > 0$,

$$\frac{\partial F}{\partial p_{iq}}(P) = c\, h_{iq}(P), \forall i, q \text{ and all } P \in \mathbf{K}. \tag{22}$$

Then the learning algorithm, for any initial condition in $\mathbf{K} - \mathbf{K}^*$, *always converges to a Nash equilibrium.*

Proof: Consider the variation of F along the trajectories of the ODE. We have

$$\begin{aligned} \frac{dF}{dt} &= \sum_{i,q} \frac{\partial F}{\partial p_{iq}} \frac{dp_{iq}}{dt} \\ &= \sum_{i,q} \frac{\partial F}{\partial p_{iq}}(P) p_{iq} \sum_s p_{is}[h_{iq}(P) - h_{is}(P)], \text{ by (17)} \\ &= c \sum_i \sum_q \sum_s p_{iq} p_{is}[(h_{iq}(P))^2 - h_{iq}(P) h_{is}(P)], \text{ by (22)} \\ &= c \sum_i \sum_q \sum_{s>q} p_{iq} p_{is}[h_{iq}(P) - h_{is}(P)]^2 \\ &\geq 0 \end{aligned} \tag{23}$$

Thus F is nondecreasing along the trajectories of the ODE. Also, due to the nature of the learning algorithm given by (9), all solutions of the ODE (17), for initial conditions in $\mathbf{K}$, will be confined to $\mathbf{K}$ which is a compact subset of $\mathbf{R}^{m_1 + \cdots + m_N}$. Hence by [[22], Theorem 2.7], asymptotically all the trajectories will be in the set $\mathbf{K}_1 = \{P \in [0,1]^{m_1 + \cdots + m_n} : \frac{dF}{dt}(P) = 0\}$.

From (23) and (17) it is easy to see that

$$\begin{aligned} \frac{dF}{dt}(P) = 0 &\Rightarrow p_{iq} p_{is}[h_{iq}(P) - h_{is}(P)] = 0 \ \forall q, s, i \\ &\Rightarrow f_{iq}(P) = 0 \ \forall i, q \\ &\Rightarrow P \text{ is a stationary point of the ODE.} \end{aligned} \tag{24}$$

Thus all solutions have to converge to some stationary point. Since by Theorem 3.2 all stationary points that are not Nash equilibria are unstable, the theorem follows. □

Theorem 3.2, and Theorem 3.3 together characterise the long term behaviour of the learning algorithm. For any general N-person game, all strict Nash equilibria in pure strategies are asymptotically stable in the small. Further the algorithm can not converge to any point in $\mathbf{K}$ which is not a Nash equilibrium. If the game satisfies the sufficiency condition needed for Theorem 3.3 then the algorithm will converge to a Nash equilibrium. (If the game does not satisfy this condition we cannot be sure that the algorithm will converge rather than, e.g., exhibit a limit cycle behaviour). In general, we cannot establish that, in a general game, all mixed strategy equilbria are stable attractors.

In the next two sections we discuss these results in two special cases of N-person games where we illustrate existence of the function F needed in Theorem 3.3.

IV. Games With Common Payoff

In a game with common payoff all the players receive the same payoff after each play of the game. Thus we have $r_i = r$, for all i and hence $d^i(a_1, \ldots, a_N) = d^j(a_1, \ldots, a_N)$, $\forall i, j$ where the d^i are as defined by (1). Equivalently we have

$$g^i(P) = g^j(P), \forall i, j$$

where g^i are as defined in (2).

Hence the reward structure of the game can be represented by a single hyper matrix D of dimension $m_1 \times \cdots \times m_N$, whose elements are

$$d_{a_1 \ldots a_N} = E[r \mid \text{Player } j \text{ played action } a_j], 1 \leq j \leq N \tag{25}$$

Definition 4.1 The tuple of actions $(a_1, \ldots, a_N)$ is called a **mode** *of the matrix D if for all permissible values of the index s we have*

$$\begin{aligned} d_{a_1 \ldots a_N} &\geq d_{s a_2 \ldots a_N} \\ d_{a_1 \ldots a_N} &\geq d_{a_1 s \ldots a_N} \\ &\vdots \\ d_{a_1 \ldots a_N} &\geq d_{a_1 a_2 \ldots s} \end{aligned} \tag{26}$$

It is easy to see that if $(a_1, \ldots, a_N)$ is a Nash equilibrium in pure strategies then $(a_1, \ldots, a_N)$ is a mode of the matrix D and vice versa. There always will be at least one mode, namely the highest element in the matrix D. Thus a game with common payoff will have at least one Nash equilibrium in pure strategies. Further the Nash equilibrium in pure strategies is strict if the corresponding mode is strict (i.e., the inequalities in (26) are strict).

Games with common payoff have been used as parallel distributed algorithms in applications such as Pattern Recognition and Optimization [1], [4], [11] and for concept learning [23]. It is known that if all the automata use the $L_{R\text{-}I}$ algorithm as given in Section III, then the expected payoff, conditioned on the current action probabilities of all automata, increases monotonically [1], [9]. Further if the D matrix has a single mode then the learning algorithm converges to it [1], [10].

We will now present a convergence result for the game based on our analysis which ensures convergence to one of the Nash equilibria (modes) even if the game matrix is multimodal.

Theorem 4.1 Consider a game with common payoff. Then, under the learning algorithm given by (9)*, the automata team converges to one of the Nash equilibria.*

Proof: In view of Theorem 3.2, all that needs to be shown is that the ODE has only point attractors. That is, for example, it will not have limit cycles.

Define

$$F(P) = \sum_{j_1 \ldots j_N} d(j_1, \ldots, j_N) \prod_{s=1}^{N} p_{sj_s} \tag{27}$$

Hence

$$\begin{aligned} \frac{\partial F}{\partial p_{iq}} &= \sum_{j_1, \ldots, j_{i-1}, j_{i+1}, \ldots, j_N} d(j_1, \ldots, j_{i-1}, q, j_{i+1}, \ldots, j_N) \prod_{s \neq i} p_{sj_s} \\ &= h_{iq}(P) \end{aligned} \tag{28}$$

where h_{iq} is as defined by (7). Hence this function F satisfies the condition needed for Theorem 3.3. The proof is completed by applying that theorem. □

Remark 4.1 By this theorem we can conclude that all modes in the game matrix are stable and strict modes are asymptotically stable. Further, the algorithm always converges to a point in $\mathbf{K}$. In practice it is found that the algorithm always converges to a mode of D rather than to a mixed strategy.

Remark 4.2 In some applications such as optimization, it is desirable to converge to the highest valued mode (which would correspond to the global maximum) rather than any mode (local maximum). With the type of decentralized algorithm that we have, it is not possible to achieve convergence to global optimum. Such a convergence, however, is possible if we allow some information exchange among the players. The only known automata algorithms at present that can guarantee convergence to the global optima are the estimator algorithms [4, 11, 13] and algorithms with a random walk term added to escape local optima [24, 25].

V. The Consistent Labelling Problem

Another special case of the general game that we consider is the so called consistent labelling problem [26]. A team of automata model for solving the labelling problem was presented in [6]. This algorithm was found useful in many computer vision problems [16], [17]. The consistent labelling problem is a generalization of the idea of constraint satisfaction to include cases where the constraints may be 'soft' [26]. Recently it was pointed out by Ricci [27] that constraint satisfaction can be viewed as finding Nash equilibria in N-person games. In this section we show that finding consistent labellings can be viewed as a special case of our game and then illustrate the analysis of Section III.A for this case. For defining a labelling problem we follow the formalism of Hummel and Zucker [26].

In a labelling problem one is given

1) a set of objects, $\mathbf{O} = \{O_1, \ldots, O_N\}$,
2) a set of labels, $\mathbf{\Lambda} = \{1, \ldots, M\}$,
3) a neighbour relation over $\mathbf{O} \times \mathbf{O}$ which specifies which pairs of objects constrain each other, and
4) a set of Compatibility functions, $r_{ij} : \mathbf{\Lambda} \times \mathbf{\Lambda} \rightarrow \mathbf{R}$

The objective is to assign labels to objects 'satisfying' the constraints imposed by the compatibility functions. Formalising the notion of satisfying the compatibility functions, we define what are called consistent labellings.

An unambiguous labelling is a function from $\mathbf{O}$ to $\mathbf{\Lambda}$, i.e., an assignment of a unique label to each object.

Definition 5.1 An unambiguous labelling $(\lambda_1, \ldots, \lambda_N)$, which assigns label λ_i to object i, is said to be consistent if for each i, $1 \leq i \leq N$,

$$\sum_j r_{ij}(\lambda_i, \lambda_j) \geq \sum_j r_{ij}(\lambda, \lambda_j), \forall \lambda. \tag{29}$$

$r_{ij}(\lambda_i, \lambda_j)$ can be thought of as the degree of compatibility for label λ_i on object i with label λ_j on obect j. Thus, at a consistent labelling, if we change the label on any one object the 'net consistency' at that object decreases. Throughout

this section we assume, without loss of generality [6], that $r_{ij}(.,.) \in [0,1]$. We also assume that $r_{ii} = 0$.

Instead of assigning labels unambiguously, we may choose to assign labels probabilistically. Let $\mathbf{p}_i = (p_{i1} \ldots p_{iM})$ be a probability vector such that we assign label q to object i with probability p_{iq}. Then the N-tuple of probability vectors given by $P = (\mathbf{p}_1, \ldots, \mathbf{p}_N)$ is called a probabilistic label assignment. It is easy to see that $P \in \mathbf{K}$, defined by (5) with $m_1 = m_2 = \cdots = m_N = M$. Similarly, unambiguous labellings correspond to points in $\mathbf{K}^*$.

Definition 5.2 A probabilistic label assignment $P = (\mathbf{p}_1, \ldots, \mathbf{p}_N) \in \mathbf{K}$ *is said to be consistent if for each* i,

$$\sum_j \sum_{\lambda,\lambda'} r_{ij}(\lambda,\lambda') p_{j\lambda'} p_{i\lambda} \geq \sum_j \sum_{\lambda,\lambda'} r_{ij}(\lambda,\lambda') p_{j\lambda'} v_{i\lambda} \quad \forall\, V = (\mathbf{v}_1, \ldots, \mathbf{v}_N) \in \mathbf{K} \tag{30}$$

We now establish the connection between consistent labellings and Nash equilibria. Let each object be represented by a player and let the set of actions (or pure strategies) of each player be the label set. Let the payoff functions (defined by (1)) be given by

$$d^i(\lambda_1, \ldots, \lambda_N) = \frac{1}{N} \sum_j r_{ij}(\lambda_i, \lambda_j) \tag{31}$$

Now from Definitions (2.1, 2.2, 5.1, 5.2) it is clear that consistent unambiguous labellings are Nash equilibria in pure strategies and other consistent labellings are Nash equilibria in mixed strategies.

We can use the automata algorithm given in Section III for learning the consistent labellings. We associate an automaton with each object and the action set will be the label set. We start with some initial label probabilities $P_i(0) = (\mathbf{p}_1(0) \ldots \mathbf{p}_N(0))$. At each instant each of the automata chooses a label at random. We update the label probabilities using (9). So, all that remains to be specified is how to obtain the reaction, r_i, for each automaton. We will make $r_i = \frac{1}{N} \sum_j r_{ij}(\lambda_i, \lambda_j)$ where the current choice of label for object j is λ_j, $1 \leq j \leq N$. Since, by (31), we have $E[r_i | (\lambda_1, \ldots, \lambda_N)] = d^i(\lambda_1, \ldots, \lambda_N)$, it is clear that our algorithm will now learn Nash equilibria which are the consistent labellings. We thus get a parallel stochastic algorithm for solving the labelling problem. It may be noted here that our algorithm can solve the labelling problem even when the r_{ij} functions are unknown if we have access to signals r_i that satisfy $E[r_i | (\lambda_1, \ldots, \lambda_N)] = \frac{1}{N} \sum_j r_{ij}(\lambda_i, \lambda_j)$.

This is the algorithm proposed in [6]. It is proved in [6] that for this algorithm all unambiguous consistent labellings are locally asymptotically stable and all other corners of $\mathbf{K}$ are unstable. Now, in view of Theorem 3.2, which assures that all non-Nash stationary points are unstable, we can say that the algorithm will never converge to an inconsistent labelling.

We now give a complete convergence proof for the algorithm, using Theorem 3.3, if the compatibility functions satisfy some condition.

Definition 5.3 The compatibility functions are said to be symmetric if we have

$$r_{ij}(\lambda_i, \lambda_j) = r_{ji}(\lambda_j, \lambda_i) \; \forall i, j, \lambda_i, \lambda_j.$$

Define a function $F : \mathbf{R}^{NM} \to \mathbf{R}$ by

$$F(P) = \sum_i \sum_j \sum_\lambda \sum_{\lambda'} r_{ij}(\lambda,\lambda') p_{i\lambda} p_{j\lambda'} \tag{32}$$

Since the payoff to the i^{th} automaton, r_i, is given by $r_i = \frac{1}{N} \sum_j r_{ij}(\lambda_i, \lambda_j)$ where λ_j is the action currently chosen by j, $1 \leq j \leq N$, the g^i functions (defined by (2)) will now be

$$g^i(P) = \frac{1}{N} \sum_j \sum_\lambda \sum_{\lambda'} r_{ij}(\lambda,\lambda') p_{i\lambda} p_{j\lambda'} \tag{33}$$

Similarly, the h_{is} function defined by (7) will now be

$$h_{\text{is}}(P) = \frac{1}{N} \sum_j \sum_{\lambda'} r_{ij}(s,\lambda') p_{j\lambda'} \tag{34}$$

From (32) and (34) we get, for the case of symmetric compatibility functions,

$$\frac{\partial F}{\partial p_{iq}}(P) = 2 \sum_j \sum_\lambda r_{ij}(q,\lambda) p_{j\lambda} = 2N h_{iq}(P) \; \forall i, q, P \in \mathbf{K} \tag{35}$$

Now we can use Theorem 3.3 to prove the convergence of the algorithm.

Theorem 5.1 Consider the automata algorithms for solving the labelling problem with symmetric compatibility functions. The algorithm (for sufficiently small value of the parameter b *in (9)) will always converge to a consistent labelling.*

Proof: Under (31), the $N-$person game is such that all consistent labellings are Nash equilibria and vice-versa. Hence the proof is immediate using Theorem 3.2, Theorem 3.3 and (35). □

Remark 5.1 Theorem 5.1 establishes convergence of the automata algorithm to a consistent labelling for the class of problems where compatibility functions are symmetric. The new results given by this theorem which are not contained in the analysis presented in [6] are (i) all stationary points of the ODE that do not represent consistent labellings are unstable, (ii) there are no limit cycles and the algorithm always converges to a stationary point of the ODE. This analysis is important in view of the fact that the labelling problem is a very useful framework for many computer vision tasks [16], [17], [26], [28]. The assumption of symmetric compatibility functions is not very restrictive because, in fact, they are symmetric in most applications. In our treatment we considered only pairwise constraints for defining compatibility functions. This can easily be extended. If, for example, three objects jointly constrain the labels on them, then, we can define r_{ijk} instead of r_{ij}. The analysis goes through by substituting r_{ijk} in place of r_{ij} and increasing the summation indices accordingly. The algorithm can be implemented in a parallel distributed fashion, e.g., on an SIMD machine [29].

VI. Discussion

In this paper we have considered an N-person stochastic game with incomplete information. We have given an algorithm for each player to modify his current mixed strategy using the random outcome of successive plays of the game.

The algorithm is decentralized and each player needs to know only the action chosen by him and his payoff at that instant in order to update his mixed strategy. If the game is repeatedly played with each player making use of our learning algorithm with small step size, then we established local convergence of our algorithm around Nash equilibria.

We have considered two special cases of this N-person game, namely, a common payoff game and the consistent labelling problem. In both cases, it is established that the algorithm will converge to a Nash equilibrium which is the needed solution. The automata model for these special cases has been successfully employed in Pattern Recognition and Computer Vision [1], [4], [16], [17].

There are other N-person games such as stochastic prisoner's dilemma [12, 13] where the $L_{R\text{-}I}$ algorithm analyzed in this paper was found useful. Our convergence result subsumes the results known for these special cases.

The $L_{R\text{-}I}$ algorithm for learning Nash equilibria in N-person stochastic games has been empirically investigated by many researchers. Hence we do not report any simulation studies to support the theoretically predicted performance. Instead we point to various works where such simulations are available.

For the common payoff game it is observed that if the game matrix is unimodal then the algorithm always converges to it. However, if there is more than one mode then the algorithm converges to one of the modes depending on the initial condition. This is seen from simulation studies reported in [1, Sec. 8.5.6]. For the labelling problem, simulations on a simple problem are reported in [6] where it is seen that depending on the initial condition the team will converge to different consistent labellings. Similar performance of the algorithm in specific applications, namely, stereopsis and 2D object recognition, was reported in [16], [17]. Empirically observed performance of the algorithm in stochastic prisoner's dilemma and other 2-person non-zero sum games is reported in [12], [13]. All these results conform to the theoretically expected performance based on the analysis presented in this paper.

Our algorithm can be used for learning the Nash equilibrium even if the game is deterministic and the game matrix is known. We then simply make r_i, payoff to the i^{th} player, equal to (suitably normalized) $d^i(a_1, \ldots, a_N)$ which is the game matrix entry corresponding to the actions played. An example of this is the consistent labelling problem where the functions r_{ij} are known. For many problems in Computer Vision we found this algorithm (which works even where r_{ij} are unknown) to give better performance than other deterministic algorithms [6]. Whether this is an efficient algorithm for obtaining Nash equilibria for general deterministic games with known payoff functions needs to be investigated. (There are distributed algorithms for computation of Nash equilibria in deterministic games which need every player to know his payoff function [30], [31].

The convergence results proved in this paper are valid if all the players use the $L_{R\text{-}I}$ algorithm. (It may be noted that even if different players use different values of the learning parameter, b, the proofs go through). It should be possible to extend the results of this paper to the case where different players may use different absolutely expedient algorithms (see [1, Ch. 4] for definition of absolutely expedient algorithms). The model presented in the paper, wherein all players use the same algorithm, is acceptable for the applications we have considered in the previous two sections where the game model is utilized as a parallel algorithm to solve certain optimization and constraint satisfaction problems. In such cases we are free to choose the algorithm for each agent in the decentralized model.

But in a truly competitive situation a player can choose what he does; but he has no control over the other players. So, a relevant question is: if only one player is using the $L_{R\text{-}I}$ algorithm then how does he perform with respect to his payoff? It is easy to see from our analysis that if we assume that all other players are employing fixed but unknown mixed strategies, then the player using $L_{R\text{-}I}$ algorithm will maximize his payoff. This is true of any subset of players also if the rest of the players are employing fixed but unknown mixed strategies.

A particular case of competitive games is the 2-person zero-sum game. Here, in each play, if one player gains then the other loses by an equal amount. Nash equilibria are called saddle points in zero-sum games. It is known that if the game has a unique saddle point in pure strategies then the automata algorithm presented here will converge to it [7]. Theorem 3.2 assures us that even if there are multiple saddle points in pure strategies, each one of them is locally asymptotically stable.

Our analysis does not establish the stability or otherwise of Nash equilibria in mixed strategies for the general N-person game. It is known that in any stationary random environment $L_{R\text{-}I}$ algorithm always converges to a unit vector [1] and hence it might be the case that even the decentralized team cannot converge to an interior point. It may be possible to establish convergence to interior points using the $L_{R\text{-}\epsilon P}$ algorithm [1]. This needs to be investigated further.

VII. Conclusion

In this paper, we have considered N-person stochastic games with incomplete information. We presented a decentralized learning algorithm and proved a general convergence result for the algorithm. This result subsumes many known results regarding learning optimal strategies in different types of games. The special cases we have considered are games with common payoff and the consistent labelling problem. The automata model for these cases has been employed for many applications in pattern recognition and computer vision. We provided complete convergence results for the algorithm in these special cases.

The learning algorithm is completely decentralized and can be thought of as a parallel distributed network similar to neural network models. As a matter of fact, the automata model for the labelling problem can be thought of as a type of stochastic extension to the Hopfield model [32]. This algorithm was also used for combinatorial optimization problem [33]. An interesting open question is the utility of such game models as general learning algorithms for neural networks.

REFERENCES

[1] K. S. Narendra and M. A. L. Thathachar, "Learning Automata: An Introduction," Englewood Cliffs: Prentice Hall, 1989.

[2] J. C. Harsanyi, "Games with incomplete information played by bayesian player—I," *Management Sciences,* vol. 14, pp. 159–182, Nov. 1967.

[3] Binmore, "Essays on Foundations of Game Theory," Basil Blackwell, 1990.

[4] M. A. L. Thathachar and P. S. Sastry, "Learning optimal discriminant functions through a coperative game of automata," *IEEE Transaction on Systems, Man and Cybernetics,* vol. 17, no. 1, pp. 73–85, 1987.

[5] P. R. Srikanta Kumar and K. S. Narendra, "A Learning Model for Routing in Telephone Networks," *SIAM Journal of Control and Optimisation,* vol. 20, pp. 34–57, Jan. 1982.

[6] M. A. L. Thathachar and P. S. Sastry, "Relaxation labelling with learning automata," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 8, pp. 256–268, March 1986.

[7] S. Lakshmivarahan and K. S. Narendra, "Learning algorithms for two-person zero-sum stochastic games with incomplete information," *Mathematics of Operations Research,* vol. 6, pp. 379–386, Nov. 1981

[8] S. Lakshmivarahan and K. S. Narendra, "Learning algorithms for two-person zero-sum stochastic games with incomplete information: A unified approach," *SIAM Journal of Control and Optimisation,* vol. 20, pp. 541–552, July 1982.

[9] M. A. L. Thathachar and K. R. Ramakrishnan, "A cooperative game of a pair of automata," *Automatica,* vol. 20, pp. 797–801, June 1984.

[10] R. M. Wheeler, Jr. and K. S. Narendra, "Decentralized learning in finite markov chains," *IEEE Transaction on Automatic Control,* vol. 31, no. 6, pp. 519–526, 1986.

[11] S. Mukhopadhyay and M. A. L. Thathachar, "Associative learning of boolean functions," *IEEE Transaction on Systems, Man and Cybernetics,* vol. 19 pp. 1008–1015, Sep. 1989.

[12] T. Vishwanatha Rao, "Learning Solutions to Stochastic Non-Cooperative Games," *ME Thesis*, Dept. of Electrical Engineering, Indian Institute of Science, Bangalore, India, 1984.

[13] M. A. L. Thathachar and P. S. Sastry, "Learning automata in stochastic games with incomplete information," *Systems and Signal Processing,* (R. N. Madan, N. Viswanadham and R. L. Kashyap, eds.), (New Delhi), pp. 417–434, Oxford and IBH, 1991.

[14] P. Billingsley, "Convergence of Probability Measures," New York, John Wiley, 1968.

[15] H. J. Kushner, *Approximation and Weak Convergence Methods for Random Processes,* Cambridge, MA: MIT Press, 1984.

[16] P. S. Sastry, S. Banerjee and K. R. Ramakrishnan, "A local cooperative processing model for low level vision," *Systems and Signal Processing,* (R. N. Madan and N. Viswanadham and R. L. Kashyap, eds.), (New Delhi), Oxford and IBH, 1991.

[17] S. Banerjee, "Stochastic Relaxation Paradigms for Low Level Vision," *Ph.D. Thesis*, Dept. of Electrical Engineering, Indian Institute of Science, Bangalore, India, 1989.

[18] J. W. Friedman, "Oligopoly and the Theory of Games," North Holland, New York, 1977.

[19] T. Basar and G. J. Olsder, "Dynamic Noncooperative Game Theory," Academic Press, New York, 1982.

[20] W. Jianhua, "The Theory of Games," Clarendon Press, Oxford, 1988.

[21] Albert Beneveniste, Michel Metivier and Pierre Priouret, "Adaptive Algorithms and Stochastic Approximations," Springer Verlag, NewYork, 1987.

[22] K. S. Narendra and A. Annaswamy, "Stable Adaptive Systems," Englewood Cliffs: Prentice Hall, 1989.

[23] P. S. Sastry, K. Rajaram and Sohan Rashmi Ranjan, "Learning optimal conjunctive concepts using stochastic automata," To appear in *IEEE Trans.on Systems, Man and Cybernetics.*

[24] V. V. Phansalkar, "Learning Automata Algorithms for Connectionist systems—local and global convergence," *Ph.D Thesis*, Dept. of Electrical Engineering, Indian Institute of Science, 1991.

[25] V. V. Phansalkar and M. A. L. Thathachar, "Global convergence of feedforward networks of learning automata," *Proc. Int. Joint Conf. Neural Networks,* (Baltimore), June 1992.

[26] R. A. Hummel and S. W. Zucker, "On the foundations of relaxation labelling processes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 5, pp. 267–287, May 1983.

[27] F. Ricci, "Equilibrium theory and constrained networks," in *Constraint Directed Reasoning Workshop*, 1990.

[28] L. S. Davis and A. Rosenfeld, "Cooperative processes in low level vision: A Survey," *Artificial Intelligence,* vol. 17, pp. 245–265, Aug. 1981.

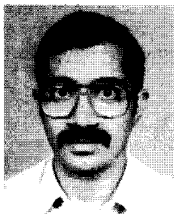
[29] K. Banerjee, P. S. Sastry, K. R. Ramakrishnan and Y. V. Venkatesh, "An simd machine for low level vision," *Information Sciences,* vol. 44, pp. 19–50, 1988.

[30] S. Li and T. Basar, "Distributed algorithms for the computation of noncooperative equilibria," *Automatica,* vol. 23, pp. 523–533, 1987.

[31] T. Basar, "Relaxation techniques and asynchronous algorithms for on line compution of noncooperative equilibria," *Journal of Economic Dynamics and Control,* 1988.

[32] P. S. Sastry, "Stochastic networks for constraint satisfaction and optimisation," *Sadhana,* vol. 15, pp. 251–262, Dec. 1990.

[33] S. Muralidharan, "A Study of Parallel Distributed Processing Schemes for Combinatorial Optimisation," *M.Sc Thesis*, Dept. of Electrical Engineering, Indian Institute of Science, Bangalore, India, 1989.

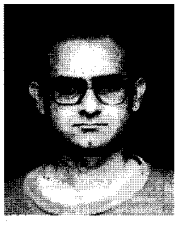


P. S. Sastry (S'82–M'85) received the B.Sc. (Hons.) degree in Physics from the Indian Institute of technology, Kharagpur, in 1978, the B.E. degree in electronics and electrical communication engineering and the Ph.D. degree in electrical engineering from the Indian Institute of Science, Bangalore, in 1981 and 1985, respectively.

Currently, he is an Assistant Professor at the Department of Electrical Engineering, Indian Institute of Science. His research interests include Learning Systems, Neural Networks and Artificial Intelligence.

M. A. L. Thathachar (SM'79–F'91) was born in 1939 in Mysore City, India. He obtained the Bachelor's degree in Electrical Engineering from the University of Mysore in 1959, Master's degree in Power Engineering and the Ph.D. degreein Control Systems from the Indian Institute of Science, Bangalore, in 1961 and 1968, respectively.

He was a member of the faculty of the Indian Institute of Technology, Madras, from 1961–64. Since 1964, he has been with the Indian Institute of Science, where he is currently Professor in the Department of Electrical Engineering and Chairman of the Division of Electrical Sciences. He has held visiting faculty positions at Yale University, Concordia University and Michigan State University. Dr. Thathachar is a fellow of the Indian National Science Academy, the Indian Academy of Sciences and the Indian National Academy of Engineering. His current research interests include Learning Automata, Neural Networks and Fuzzy Systems.



V. V. Phansalkar received the B.Tech. in EE and the M.Tech. in Systems and Control from IIT-Bombay in 1983 and 1985, respectively, and the Ph.D. in EE from IISc, Bangalore, in 1992.

Currently, he is a Project Associate in the Department of Electrical Engineering, Indian Institute of Science. His current research interests include Learning Systems, Neural Networks and Applications of Probability Theory.