A New Local Rule for Convergence of ICLA to Compatible Point

Hossein Morshedlou[1]
Soft Computing Lab
Department of Computer Engineering
AmirKabir University of Technology, Tehran, Iran

Mohammad Reza Meybodi
Soft Computing Lab
Department of Computer Engineering
AmirKabir University of Technology, Tehran, Iran

Abstract

Many problems in the modern world have a decentralized and distributed nature. Irregular Cellular Learning Automata is a powerful mathematical model for decentralized problems and applications. Convergence of ICLA to a compatible point is very important because this convergence can provide efficient solutions for the problems. The local rule of ICLA can play a key role in this convergence. A local rule that simply rewards or punishes learning automata just based on the response of environment and actions of neighbors doesn't guarantee convergence of ICLA to compatible point. In this paper, we present a new local rule that guarantees convergence to a compatible point. Formal proofs for the convergence are provided and results of the conducted experiments support our theoretical findings.

Keywords: *Irregular Cellular Learning Automaton, Compatible Point, Learning Automata, Convergence.*

1. Introduction

There are many problems that have a decentralized and distributed nature. Irregular Cellular Learning Automata (ICLA) which is introduced formally in [14] is a powerful mathematical model for decentralized applications. ICLA is an extension of CLA [6] in which the restriction of regular structure is removed. An ICLA can be defined as a graph in which each vertex represents a cell and there are one or multiple learning automata in each cell [7]. Each edge in this graph represents neighborhood relation between two cells (two learning automata). There is a local rule in ICLA that determines the reinforcement signal to any particular learning automaton (LA). This rule uses response of environment and the actions selected by the neighboring LAs to generate the reinforcement signal. In an iterative process, each LA updates the state of its cell using the received reinforcement signal and ICLA evolves this way until the convergence or desired result is obtained. Concept of compatible point, introduced in [6], is an equilibrium point for CLA and this concept can be extended to ICLA as well. Each LA in ICLA learns to choose an action which maximizes the received reward (positive reinforcement) from environment. Due to dependency of the reward to the selected actions of the neighbors, choosing same action by a LA in different rounds probably doesn't conduce to the same reward. In such conditions, each LA should reach equilibrium (i.e. an agreement on action selection) with its neighbors to increase its rewards. The equilibrium should be such that no learning automaton in ICLA has any reason to change its selected action. In the other words, unilateral deviation from the equilibrium should not be profitable. The mentioned equilibrium point in ICLA is called *compatible point* or *compatible configuration.* Reaching the equilibrium is called *convergence of ICLA* to compatible point. Convergence of ICLA to compatible point is of great importance because it can conduce to efficient solutions for the problems and applications. Examples from such applications are graph coloring [3], clustering the wireless sensor networks [4] or channel assignment in cellular networks [5]. Because there is no any direct interaction between learning automata, each LA just perceives reinforcement signal or response of environment and observes the selected actions of its neighbors. Under these conditions it's needed to find a way such that ICLA be able to converge to compatible point. For this purpose the local rule of ICLA can play a key role. The ordinary local rule which simply maps response (reward or penalty) of environment to the perceived reinforcement signal by LA doesn't guarantee convergence of ICLA to a compatible point. In this paper we aim to present a new local rule that guarantees this convergence. Formal proofs for the convergence are provided and results of the conducted experiments support our theoretical findings. The rest of this paper is organized as follow: in section 2 we briefly introduce preliminary concepts. Section 3 contains related work. We present the new local rule and the related proofs in section 4. In section 5, results of the conducted experiments are given. Section 6 concludes the paper.

[1] Corresponding Author: Email: morshedlou@aut.ac.ir, Tel: +982164545120

2. Preliminary Concepts

In this section, stochastic environment, learning automaton, Cellular Learning Automata (CLA), and Irregular CLA (ICLA) are introduced. The contents of these introductions are from [3, 6]. After that, ICLA game is explained and concept of compatible point will be presented formally.

2.1. Stochastic Environment and Learning Automaton

The environment can be described by a triple $E=\{\alpha,\beta,c\}$, where $\alpha=\{\alpha_1,\alpha_2,...,\alpha_r\}$ represents the finite set of the inputs, $\beta=\{\beta_1,\beta_2,...,\beta_m\}$ indicates set of the values that are responses of environment and $c=\{c_1,c_2,...,c_r\}$ denotes the set of the penalty probabilities, where the element c_i is associated with the given input α_i. We call the process of generating reinforcement (penalty or reward) signal as the local rule.

A learning automaton is represented by a triple $<\underline{\beta},\underline{\alpha},T>$, where $\underline{\beta}$ is the set of inputs, $\underline{\alpha}$ is the set of actions, and T is learning algorithm. Actions of learning automata are inputs to environments. Let $\alpha_i(k)\in\underline{\alpha}$ and $\underline{p(k)}$ respectively denote the selected action by learning automaton and probability vector defined over the action set at round k. Let a and b indicate the reward and penalty parameters and determine the amount of increases and decreases of the action probabilities. r is number of actions can be taken by learning automaton. At each round, the action probability vector $\underline{p(k)}$ is updated by the linear learning algorithm given in Eq. (1), if the selected action $\alpha_i(k)$ is rewarded by the environment, and it is updated as given in Eq. (2) if the selected action is penalized.

$$p_j(k+1)=\begin{cases} p_j(k)+a[1-p_j(k)] & j=i \\ (1-a)\,p_j(k) & \forall j\neq i \end{cases} \tag{1}$$

$$p_j(k+1)=\begin{cases} (1-b)\,p_j(k) & j=i \\ (\frac{b}{r-1})+(1-b)\,p_j(k) & \forall j\neq i \end{cases} \tag{2}$$

If $a=b$, the recurrence Eq. (1) and (2) are called linear reward-penalty (L_{R-P}) algorithm. If $a>>b$ the given equations are called linear reward-ε penalty ($L_{R-\varepsilon P}$), and finally if $b = 0$ they are called linear reward-Inaction (L_{R-I}).

Note: All unit probability vectors $\{\underline{p}\mid \underline{p}=(\underline{p}_1,\underline{p}_2,...,\underline{p}_r),\forall i: p_i\in\{0,1\},\sum_{i=1}^{r}p_i=1\}$ are called absorbing states [26] for learning automaton when the learning algorithm is L_{R-I}. This is for the reason that action probability vector of learning automaton remains unchanged when the taken action is penalized by the environment. So when learning automaton enters an absorbing state, it can not leave it again.

2.2. Cellular Learning Automata

Cellular learning automata, which is a combination of cellular automata (CA) and learning automaton, is a powerful mathematical model for many decentralized problems and phenomena. A CA is composed of a regular grid of cells, each in one of a finite number of states. With binary states CA can be considered as a specific kind of Boolean Networks [11]. The grid can be in any finite number of dimensions. For each cell, a set of cells called its neighborhood is defined relative to the specified cell. The basic idea of CLA, which is a subclass of stochastic CA, is to use learning automata to adjust the state transition probability of stochastic CA. CLA consists of a large number of simple components which have the learning capability, and cooperate to produce complicated behavioral patterns. The learning automaton residing in a particular cell determines its state (action) on the basis of its action probability vector. Like CA, there is a rule that CLA operates under it. The rule of CLA and the actions selected by the neighboring LAs of any particular LA determine the reinforcement signal to the LA residing in that cell. In CLA, the

neighboring learning automata of any particular learning automaton constitute its local environment. The action probability vector of a neighboring learning automaton varies during the evolution of the CLA, and so the local environment is non-stationary. The operation of CLA could be described as follows: At the first step, the internal state of every cell is specified. The state of every cell is determined on the basis of action probability vector of the learning automaton residing in that cell. The initial value of this state may be chosen on the basis of past experience or at random. In the second step, the rule of cellular automata determines the reinforcement signal to each learning automaton residing in that cell. Finally, each learning automaton updates its action probability vector on the basis of supplied reinforcement signal and the chosen action. This process continues until the desired result is obtained. A d-dimensional CLA is given formally in below:

Definition1. CELLULAR LEARNING AUTOMATA [6]: A d-dimensional cellular learning automata is a structure $\Lambda = (Z^d, \Phi, A, N, f)$, where.

1. Z^d is a lattice of d-tuples of integer numbers.
2. Φ is a finite set of states.
3. A is the set of LAs each of which is assigned to each cell of the CA.
4. $N = \{\overline{\chi}_1, \overline{\chi}_1, ..., \overline{\chi}_{\overline{m}}\}$ is a finite subset of Z^d called neighborhood vector, where $\overline{\chi}_i \in Z^d$.
5. $f : \underline{\Phi^{\overline{m}}} \to \underline{\beta}$ is the local rule of the CLA, where $\underline{\beta}$ is the set of values that the reinforcement signals can take. It gives the reinforcement signal to each LA from the current actions selected by its neighboring LAs.

In what follows, we consider CLA with n cells and the neighborhood function $\overline{N}(i)$. The learning automaton A_i, which has a finite action set $\underline{\alpha_i}$, is associated to cell *i* (for *i* = 1,..., n) of the CLA. Let the cardinality of $\underline{\alpha_i}$ be r_i. The state of the CLA represented by $\underline{p} = (\underline{p}_1, \underline{p}_2, ..., \underline{p}_n)$ where $\underline{p}_i = (p_{i1}, ..., p_{ir_i})$ is the action probability vector of A_i. At round or iteration *k,* each learning automaton chooses an action. Let $\alpha_i \in \underline{\alpha_i}$ be the action chosen by A_i. Then all learning automata receive a reinforcement signal. Let $\beta_i \in \underline{\beta}$ be the reinforcement signal received by A_i, This reinforcement signal is produced by the application of local rule $F^i(\alpha_{i+\overline{x}_1}, \alpha_{i+\overline{x}_2}, ..., \alpha_{i+\overline{x}_{\overline{m}_i}}) \to \beta_i$. The value of β_i shows that the chosen action of A_i will receive reward ($\beta_i = 1$) or penalty ($\beta_i = 0$). For the sake of simplicity in presentation as follows the rule $F^i(\alpha_{i+\overline{x}_1}, \alpha_{i+\overline{x}_2}, ..., \alpha_{i+\overline{x}_{\overline{m}_i}})$ is denoted by $F^i(\alpha_1, \alpha_2, ..., \alpha_{\overline{m}_i})$.

2.3. Irregular Cellular Learning Automata

An Irregular Cellular Learning Automata (ICLA) [14] is a CLA in which there is not the restriction for rectangular grid structure (Z^d). This generalization is expected because there are many applications that cannot be adequately modeled with rectangular grids. An ICLA is defined as an undirected graph in which, each vertex represents a cell which is equipped with a learning automaton. The learning automaton residing in a particular cell determines its state (action) on the basis of its action probability vector. The structure of ICLA is similar to Random Boolean Networks (RBN). Except for its two-valued variables in RBN, an RBN is a truly generalized CA and maybe it's possible to imagine ICLA as a combination of LA with specific kind of Boolean Networks. For more details about Boolean Networks see [11, 12, 19]. Like CLA, there is a rule that the ICLA operates under. The rule of the ICLA and the actions selected by the neighboring LAs of any particular LA determines the reinforcement signal to the LA residing in a cell. The neighboring LAs of any particular LA constitute the local environment of that cell. The local environment of a cell is non-stationary because the action probability vectors of the neighboring LAs vary during the evolution of the ICLA.

2.4. ICLA Game

ICLA game is an extended form of stochastic game of learning automata [30]. Consider a game with N players. Every iteration the players select an action from their action sets. Then they receive stochastic payoffs from the

environment. The payoff each player receives from environment depends on joint actions of all players. These payoffs can be dissimilar for different players. The stationary probability distributions which establish payoffs of joint actions are unknown for all players. Now assume a new game in which payoffs of player *i* and player *j* are independent from the selected actions of each others. Assume there exist a sequence of players $i, i_1, ..., i_m, j$ for $m \geq 1$ such that each two successive players in the sequence care about each others' strategy. Note that, the payoffs of players *i* and *j* do not affect the selected action of each other instantly but in long term, they would be influenced from each other's strategy indirectly and through other players who are in the sequence. Now if in former game and in latter game we represent each player by a learning automaton (LA), then we have a game of learning automata in the former case and an ICLA game in the latter case. Actions of a player constitute the action set of LA. Hence at any given instant, the probability vector of a LA is equivalent to the mixed strategy of a player. Since all parts of Definition 1 (except part 1) are valid for ICLA as well, so we can use them also for describing ICLA game. The action set of each LA determines its state set (see Definition1-(2)). All LAs that LA *i* care about their strategy are in LA *i*'s neighborhood vector (see Definition1-(4)). Local rule of ICLA game (see Definition1-(5)) establishes stochastic payoff of each learning automaton.

Definition 2. A configuration of ICLA at iteration *t* is denoted by $\underline{p}^t = (\underline{p}_1^t, \underline{p}_2^t, ..., \underline{p}_n^t)$ where $\underline{p}_i^t$ is the action probability vector of learning automaton A_i.

Definition 3. The set of all probabilistic configurations K in ICLA are

$$\mathrm{K} = \{ \underline{p} \mid \underline{p} = (\underline{p}_1, \underline{p}_2, ..., \underline{p}_n), \underline{p}_i = (p_{i1}, ..., p_{ir_i}),\ \forall y, i: p_{iy} = 0 \text{ or } 1,\ \forall i: \sum_y p_{iy} = 1 \} \quad (3)$$

Definition 4. The evolution of ICLA from a given initial configuration $\underline{p}^0$ is a sequence of configurations $\{\underline{p}^t\}_{t \geq 0}$ where $\underline{p}^{t+1} = G(\underline{p}^t)$. $G: K \to K$ is a mapping that describes global behavior or dynamics of ICLA.

Definition 5. The average reward for action *r* of automaton A_i for configuration $\underline{p}$ is defined as

$$d_{ir}(\underline{p}) = \sum_{\alpha_2} ... \sum_{\alpha_{\overline{m}}} F^i(r, \alpha_2, ..., \alpha_{\overline{m}_i}) \prod_{\substack{l \in N(i) \\ l \neq i}} p_{l\alpha_l} \quad (4)$$

and the average reward for learning automaton A_i is defined as

$$D_i(\underline{p}) = \sum_r d_{ir}(\underline{p}) \times p_{ir} \quad (5)$$

The above definition implies that if the learning automaton A_j is not a neighboring learning automaton for A_i, then $d_{ir}(\underline{p})$ does not depend on $\underline{p}_j$.

Definition 6. In ICLA game, a configuration $\underline{p}$ is compatible if

$$\sum_r d_{ir}(\underline{p}) \times p_{ir} \geq \sum_r d_{ir}(\underline{p}) \times q_{ir} \quad (6)$$

for all configurations $\underline{q} \in \mathrm{K}$ and all learning automaton A_i. In other words, compatible configuration in an ICLA game is equivalent to the Nash equilibrium of a game of learning automata. Therefore in a compatible configurations unilateral deviation of a LA is not profitable. In rest of this paper, we use compatible point instead of compatible configuration.

3. Related Work

Potentials of a single LA to operate in a stochastic environment, made researchers to extend its idea to a team of LAs as a group operating in a game setting. Common payoff [26, 33] and zero-sum [17, 33] games are studied in the

literature. An important result is that learning automata reach a game-theoretic equilibrium point as the penalty parameter goes to zero [17, 28]. In [9] a cooperative game-theoretic approach to model a multi-level decision making process is presented. Players of this game are learning automata which decide about profitability of group constitution. The reported results illustrate that delayed information, both types of penalties (asymmetric and symmetric) lead to chaos but with different Lyapunov exponents. These results are confirmed by [8, 10] as well. The games of LAs have also been used in many applications such as multiple access channel selection [38], congestion avoidance in wireless networks [25], channel selection in radio networks [34], clustering in wireless ad hoc networks [1], power system stabilizers [16], backbone formation in ad hoc wireless networks [2], spectrum allocation in cognitive networks [18], and solving various NP-complete problems [31, 32]. These applications show capabilities of LA when multiple learning automata interact with each other. Another class of games, which uses capabilities of learning automata, is Markov games category. The idea of using LAs in Markov games originates from the ability of LAs to manage and control a Markov Chain [35, 37] without knowledge about transition probabilities in an independent and non-centralized way. In [36] it is illustrated that a group of independent LAs are able to converge to equilibrium of a limiting game, even with limited observations. Furthermore in [20], some learning automata based methods for finding optimal policies in Markov games are suggested.

In all the mentioned works, all LAs affect from each other directly but this may not be true for some applications. There are many applications that players don't influence directly each other. The mathematical framework presented in [6] is appropriate for modeling such situations. Also the introduced concept of compatible point in [6] can be used as an equivalent equilibrium point to Nash. However capabilities and applications of CLA are not restricted to stochastic games with transitive influences. CLA has been found to perform well in many application areas such as image processing [21], rumor diffusion [22], modeling of commerce networks [23], channel assignment in cellular networks [5], clustering the wireless sensor networks [4], mobile wireless sensor network [15], image processing [24] and solving NP-hard problems [13], to name just a few.

4. Convergence of ICLA to a Compatible Point

In this section, we propose a new local rule for convergence of ICLA to a compatible point. Before presenting the new local rule first we show two preliminary lemmas which are required to prove convergence of ICLA to a compatible point.

4.1. Preliminary Lemmas

In this section, first we prove existence of compatible point in ICLA and then a preliminary lemma will be presented.

$n_i^t(a_j)$ shows how many times, LA_i selected action a_j during epoch t.

L = length of epoch (number of times LA_i chooses an action)

$t = 1;$

$$\underline{\hat{p}}_{ij}^{1} = \frac{n_i^1(a_j)}{L}$$

while (*True*) {

$$\underline{\hat{p}}_{ij}^{t+1} = \frac{\underline{\hat{p}}_{ij}^{t} + \frac{n_i^t(a_j)}{L}}{2}$$

$t = t + 1;$

}

Fig 1- Pseudo code of Algorithm 1 for estimation of neighbor's strategy

Lemma1: In ICLA game there is at least one compatible point.

Outline of Proof: For the proof first we define a particular mapping *T* over K ($T : K \to K$) and we show that *T* has at least one fixed point. Then we prove that a point is a fixed point if and only if it be a compatible point. For complete proof see Appendix A.

Lemma 2: In algorithm 1, If *t* approaches infinity then $\underline{\hat{p}}_i^t$ converges to $\underline{p}_i^t$.

Outline of Proof: To prove $\underline{\hat{p}}_i^t$ is an accurate estimation of $\underline{p}_i^t$ in infinity, we show when ICLA has converged to a point such as $\underline{p}^*$ (probability vector of LA_i ($\underline{p}_i^t$) evolved to $\underline{p}_i^*$), we have $\lim_{t\to\infty}(\underline{\hat{p}}_i^t - \underline{p}_i^*) = 0$. This means $\underline{\hat{p}}_i^t$ is an accurate estimation at infinity. For complete proof see Appendix B.

4.2. The Proposed Local Rule

Let $\underline{p}_i^c$ denotes the current probability vector of LA_i and the best response probability vector of LA_i (with respect to the probability vectors of its neighbors) is indicated by $\underline{p}_i^{br}$. Algorithm 2 (see figure 2) illustrates our proposed local rule to generate reinforcement signal (β). According to definition 6, for each LA unilaterally deviation is not profitable in compatible point. As a result when ICLA is converged to a compatible point the probability vector of every learning automaton is the best response probability vector with respect to the probability vectors of its neighbors. Considering this fact, the concept behind the proposed local rule is as below:
In an iterative process, if each learning automaton selects its action according to the best response probability vector (respect to the observed and estimated probability vectors of its neighbors) then with accurate estimations, a non-compatible point can't be a stable point. This is for the reason that in a non-compatible point there is at least one learning automaton that unilateral deviation will be profitable for it. To estimate the best response probability vector for learning automaton *i* we use the expression in line 5 in the proposed local rule. The proposed local rule tries to reduce difference ($\Delta\underline{R}_i^{RS}$) between the current probability vector ($\underline{p}_i^c$) and the best response probability vector ($\underline{p}_i^{br}$). The average reward for learning automaton *i* (d_i) is determined such that action *j* to be rewarded proportional to $\Delta\underline{R}_i^{RS}(j)$ to reduce the difference. For computation of $\underline{p}_i^{br}$ we need $\underline{\hat{p}}^t$ and Q_t^i. $\underline{\hat{p}}^t$ can be estimated using algorithm 1, but for Q_t^i, we should maintain a table of Q-values $Q^i(a^1,...,a^{\overline{m}_i})$ for each joint action of $(a^1,...,a^{\overline{m}_i})$. By inspiration from Q-learning, we define the optimal Q-values for LA_i with respect to its neighbors as Eq. (7).

$$Q_*^i(a^1,...,a^{\overline{m}_i}) = Er_i(a^1,...,a^{\overline{m}_i}) + \beta.v^i(\underline{p}_1^*, \underline{p}_2^*,..., \underline{p}_i^{br}(\underline{p}^*, Q_*^i),..., \underline{p}_{\overline{m}_i}^*) \quad (7)$$

Where $v^i(\underline{p}_1^*, \underline{p}_2^*,..., \underline{p}_i^{br}(\underline{p}^*, Q_*^i),..., \underline{p}_{\overline{m}_i}^*)$ is total discounted reward of LA_i over infinite iterations when probability vectors of the neighbors are $(\underline{p}_1^*, \underline{p}_2^*,..., \underline{p}_{i-1}^*, \underline{p}_{i+1}^*,..., \underline{p}_{\overline{m}_i}^*)$ and $\underline{p}_i^{br}(\underline{p}^*, Q_*^i)$ is the best response probability vector with respect to the probability vectors of the neighbors. Er_i is the expected value of the responses of the environment that learning automaton *i* has received until iteration *t*. The ordinary local rule uses these responses to simply reward or penalize the selected action of learning automaton. Since environment is stochastic and information is incomplete, we have no idea about Q-values and we should find a way to estimate them. Initially, we propose to assign a fixed value (e.g. zero) to Q-values and then update them using the Eq. (8).

1 $\underline{p}_i^c$ = current probability vector of LA_i

2 $\underline{p}_i^{br}$ = best response probability vector accuired from Eq.(8)

3 d_i = vector of reward probabilites

4 *Begin*

5 $\underline{p}_i^{br}(\underline{\hat{p}}^t, Q_t^i) = \arg\max_{\underline{p}_i} \sum_{a^1} \sum_{a^2} ... \sum_{a^{\overline{m}_i}} \underline{p}_i(\underline{\hat{p}}^t, Q_t^i)(a^i) \prod_{\substack{j=1 \\ j \neq i}}^{\overline{m}_i} \underline{\hat{p}}_j^t(a^j).Q_t^i(a^1,...,a^{\overline{m}_i});$

6 $\Delta \underline{p}_i^{RS} = \underline{p}_i^{br} - \underline{p}_i^c;$

7 $j_{\max} = \arg\max_j \Delta \underline{p}_i^{RS}(j);$

8 $j_{\min} = \arg\min_j \Delta \underline{p}_i^{RS}(j);$

9 $d_i = \left(\frac{\Delta \underline{p}_i^{RS}(1) - \Delta \underline{p}_i^{RS}(j_{\min})}{\Delta \underline{p}_i^{RS}(j_{\max}) - \Delta \underline{p}_i^{RS}(j_{\min})}, ..., \frac{\Delta \underline{p}_i^{RS}(\overline{m}_i) - \Delta \underline{p}_i^{RS}(j_{\min})}{\Delta \underline{p}_i^{RS}(j_{\max}) - \Delta \underline{p}_i^{RS}(j_{\min})} \right)$

10 if (selected action $== a^k \in \{a^1,...,a^{\overline{m}_i}\}$) then

11 if ($\Delta \underline{p}_i^{RS}(k) \geq 0$) then

12 set $\beta = 1$ with probability $d_i(k)$ and $\beta = 0$ with probability $(1 - d_i(k))$;

13 else if ($\Delta \underline{p}_i^{RS}(k) < 0$) then

14 set $\beta = 0$ with probability $d_i(k)$ and $\beta = 1$ with probability $(1 - d_i(k))$;

15 update Q - values using Eq. (8) ;

16 return β;

17 End

Fig 2- Algorithm 2 that shows the proposed local rule

$$Q_{t+1}^i(a^1,...,a^{\overline{m}_i}) = (1-\alpha) \times Q_t^i(a^1,...,a^{\overline{m}_i}) + \alpha \times [Er_i^t(a^1,...,a^{\overline{m}_i}) + \beta . \sum_{a^1} \sum_{a^2} ... \sum_{a^{\overline{m}_i}} \underline{p}_i^{br}(\underline{\hat{p}}^t, Q_t^i)(a^i) \prod_{\substack{j=1 \\ j \neq i}}^{\overline{m}_i} \underline{\hat{p}}_j^t(a^j).Q_t^i(a^1,...,a^{\overline{m}_i})] \quad (8)$$

Where $\underline{p}_i^{br}(\underline{\hat{p}}^t, Q_t^i)$ is the best response probability vector against $(\underline{\hat{p}}_1^t,..., \underline{\hat{p}}_{i-1}^t, \underline{\hat{p}}_{i+1}^t,..., \underline{\hat{p}}_{\overline{m}_i}^t)$. Now we point out a corollary from [29] that is the basis of our proof and then we show that using Eq. (8) Q_t^i will converge to Q_*^i when $t \to \infty$.

Corollary 1[29]: *Consider the process generated by iteration of Equation (9), where* $0 \leq f_t(x) \leq 1$.

$$V_{t+1}(x) = (1 - f_t(x)) \times V_t(x) + f_t(x) \times [P_t V_t](x) \quad (9)$$

If the process defined by Eq. (10) converges to v^* *w.p.1.*

$$U_{t+1}(x) = (1 - f_t(x)) \times U_t(x) + f_t(x) \times [P_t v^*](x) \quad (10)$$

and the following conditions hold:

- *There exist number* $0 < \gamma < 1$ *and a sequence* $\lambda_t \geq 0$ *converging to zero w.p.1 such that* $\| P_t V - P_t v^* \| \leq \gamma \| V - v^* \| + \lambda_t$ *holds for all* $V \in \{V\}$.
- $0 \leq f_t(x) \leq 1,\ t \geq 0$ *and* $\sum_{t=1}^{n} f_t(x)$ *converges to infinity uniformly in x as* $n \to \infty$.

Then, the iteration defined by Equation (9) converges to v^* *w.p.1.*

Before presenting the convergence proof for the proposed local rule, we first explain some notations. We put $f_t(x) = \alpha$ and use Q instead of V. thus $\{V\}$ is replaced by $\{Q\}$ which is a set of all Q-functions ($Q : A_1 \times ... \times A_{\overline{m}} \to R$) and P_t is a mapping from $\{Q\}$ to $\{Q\}$. Also $\|Q\|$ is equal to $\max_{(a^1,...,a^{\overline{m}})} |Q(a^1,...,a^{\overline{m}})|$ which is a finite value.

Lemma 3. Using Eq. (8) to update Q-values, the Q-values converge to optimal Q-values of Eq. (7) w.p.1.
Outline of Proof: Corollary 1 is the base of this proof. Let Q_*^i to be the optimal Q-values of Eq. (7). We define a mapping $P_t : \{Q\} \to \{Q\}$ such that Eq. (8) can be rewritten in form of Eq. (9). Then we show that using P_t we can write Eq. (11).

$$\| P_t^i Q_t^i(a^1,...,a^{\overline{m}_i}) - P_t^i Q_*^i(a^1,...,a^{\overline{m}_i}) \| \leq \beta . \| Q_t^i(a^1,...,a^{\overline{m}_i}) - Q_*^i(a^1,...,a^{\overline{m}_i}) \| + \lambda_t \quad (11)$$

Then using Eq. (11) and Eq. (12) which are results of lemma 2 and corollary 1 we show that all requirements of corollary 1 are satisfied and accordingly updating of Q-values using Eq. (8) will cause convergence to optimal Q-values of Eq. (7) w.p.1 and proof is completed. For detailed proof see Appendix C.

$$\lim_{t \to \infty} P(| \hat{\underline{p}}_i^t - \underline{p}_i^* | > \varepsilon) = 0 \quad (12)$$

Theorem 1: Using the local rule of algorithm 2, ICLA converges to a compatible point.
Outline of Proof: In a compatible point every learning automaton has the best response configuration with respect to configurations of the other learning automata. We use result of lemma 2 and 3 to show that $\underline{p}_i^{br}$ will be the best response configuration with respect to configurations of the others in infinity (Considering that $\hat{\underline{p}}_i^t$ is an accurate estimation of $\underline{p}_i^t$ (lemma 2) and Q-values converge to optimal Q-values (lemma 3)). We also show algorithm 2 tries to diminish difference between current probability vector of learning automaton ($\underline{p}_i^c$) and $\underline{p}_i^{br}$ in each iteration ($\Delta \underline{p}_i^{RS} = \underline{p}_i^{br} - \underline{p}_i^c$) and ICLA converges to a compatible point as a consequence of it. For detailed proof see Appendix D.

4.3. Complexity Analysis of the Proposed Local Rule

As described before, each learning automaton in ICLA needs to maintain Q-values for each combination of joint actions of itself and its neighbors. These Q-values are maintained internally by the learning automaton, assuming that it can observe actions of its neighbors. The learning automaton i updates $Q^i(a^1,...,a^{\overline{m}_i})$, where a^j ($j = 1,...,\overline{m}_i$) is the selected action of neighbor j th (assume the neighbors are labeled by index j). $\overline{m}_i$ is

number of the neighbors of learning automaton i in ICLA. Let $|A^i|$ be the size of action space of learning automaton i assuming $\overline{m} = \max_i \overline{m}_i$ and $|A| = \max_i |A^i|$, $n \times |A|^{\overline{m}}$ is an upper bound for total space requirement for Q-values where n is total number of learning automata in ICLA. In addition, there is a similar space requirement to maintain $Er_i(a^1,...,a^{\overline{m}_i})$ values. Therefore the proposed local rule in terms of space complexity is linear in the number of learning automata (size of ICLA), polynomial in the number of actions of learning automata, but exponential in the number of neighbors of learning automata. Since reinforcement learning algorithms, like Q-learning, compute values for each joint-action or state-action pair, this leads to an exponential computational complexity. Running time of algorithm 2 is dominated by the computation of $\underline{p}_i^{br}(\underline{\hat{p}}^t, Q_t^i)$. Computational complexity of this computation is $|A|^{\overline{m}}$. It is polynomial in the number of actions of learning automata but exponential in the number of neighbors of learning automata.

Note: If we have multiple learning automata in each cell of ICLA (for example see [7]) then each learning automaton in a cell will be a neighbor for every learning automaton which is located in that cell or its neighboring cells.

5. Experiments and Results

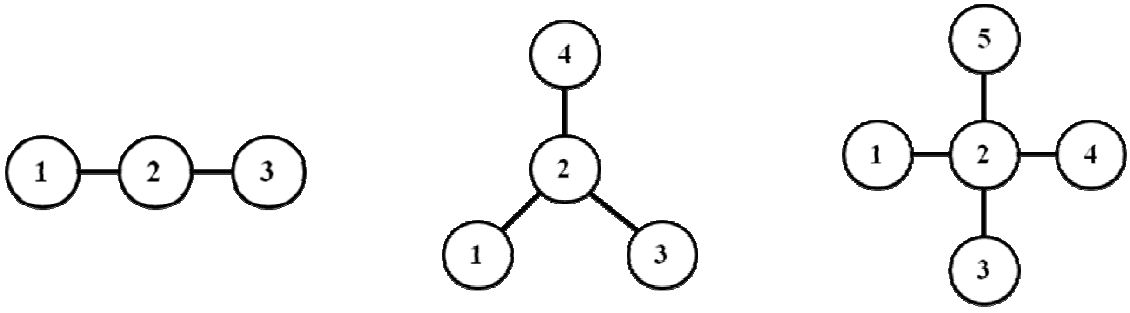
This section contains results of conducted experiments. First we present two numerical experiments and then an example for application of ICLA in channel assignment problem is given.

5.1. Numerical Experiments

The numerical experiments are given to illustrate the superiority and effectiveness of the proposed local rule. These results are compared with results of the ordinary local rule. In the experiments $ICLA_{m,n}$ refers to an ICLA with m cells and n actions for each learning automaton located in a cell. Because the notation is destitute of neighboring details so a neighboring matrix is also needed to specify a unique ICLA. Learning algorithm of all learning automata is L_{R-I}.

5.1.1. Numerical Experiment 1

The aim of this experiment is to compare convergence rate of ICLA using the proposed local rule versus the ordinary local rule. For this propose, we investigate convergence rate in $ICLA_{3,2}$, $ICLA_{4,2}$ and $ICLA_{5,2}$ which is illustrated by figure 3. The edges in the graphs show neighborhood relation. Here response of environment is generated using a response matrix. This matrix contains probability of rewarding the selected actions of learning automata in ICLA. For example, when selected joint actions by LAs is (*Low, High, High*) and the corresponding entry in the response matrix is (0.91, 0.88, 0.91) (see e.g. Table 1) then the reinforcement signals to LA1, LA2 and LA3 is reward with probability 0.91, 0.88 and 0.91 respectively. Tables 1, 2 and 3 show the reinforcement matrices for $ICLA_{3,2}$, $ICLA_{4,2}$ and $ICLA_{5,2}$.



(a) (b) (c)

Fig 3- ICLA Structures for (a) $ICLA_{3,2}$, (b) $ICLA_{4,2}$ and (c) $ICLA_{5,2}$

Table 1- Response Matrix for $ICLA_{3,2}$

a_1	a_3	$a_2 = Low$	$a_2 = High$
Low	***Low***	(0.54, 0.59, 0.72)	(0.91, 0.68, 0.81)
	High	(0.54, 0.82, 0.66)	**(0.91,0.88, 0.91)**
High	***Low***	(0.48, 0.75 , 0.72)	(0.84, 0.70, 0.81)
	High	(0.48, 0.59, 0.66)	(0.84, 0.77, 0.91)

Table 2- Response Matrix for $ICLA_{4,2}$

		$a_2 = Low$		$a_2 = High$	
a_1	a_3	$a_4 = Low$	$a_4 = High$	$a_4 = Low$	$a_4 = High$
Low	***Low***	(0.54, 0.59, 0.72, 0.62)	(0.54, 0.71, 0.72, 0.48)	(0.91, 0.68, 0.81, 0.95)	(0.91, 0.68, 0.81, 0.78)
	High	(0.54, 0.82, 0.66, 0.62)	(0.54, 0.69, 0.66, 0.48)	**(0.91,0.88, 0.91, 0.95)**	(0.91,0.76, 0.91, 0.78)
High	***Low***	(0.48, 0.75, 0.72, 0.62)	(0.48, 0.61 , 0.72, 0.48)	(0.84, 0.70, 0.81, 0.95)	(0.84, 0.70, 0.81, 0.78)
	High	(0.48, 0.59, 0.66, 0.62)	(0.48, 0.79, 0.66, 0.48)	(0.84, 0.77, 0.91, 0.95)	(0.84, 0.77, 0.91, 0.78)

Table 3- Response Matrix for $ICLA_{5,2}$

			$a_2 = Low$		$a_2 = High$	
a_1	a_3	a_5	$a_4 = Low$	$a_4 = High$	$a_4 = Low$	$a_4 = High$
Low	***Low***	***Low***	(0.54, 0.59, 0.72, 0.62, 0.57)	(0.54, 0.71, 0.72, 0.48, 0.57)	(0.91, 0.68, 0.81, 0.95, 0.88)	(0.91, 0.68, 0.81, 0.78,)
		High	(0.54, 0.67, 0.72, 0.62, 0.68)	(0.54, 0.81, 0.72, 0.48, 0.68)	(0.91, 0.77, 0.81, 0.95, 0.75)	(0.91, 0.78, 0.81, 0.78,)
	High	***Low***	(0.54, 0.82, 0.66, 0.62, 0.57)	(0.54, 0.69, 0.66, 0.48, 0.57)	**(0.91,0.88, 0.91, 0.95, 0.88)**	(0.91,0.76, 0.91, 0.78,)
		High	(0.54, 0.65, 0.66, 0.62, 0.68)	(0.54, 0.77, 0.66, 0.48, 0.68)	(0.91,0.72, 0.91, 0.95, 0.75)	(0.91,0.69, 0.91, 0.78,)
High	***Low***	***Low***	(0.48, 0.75, 0.72, 0.62, 0.57)	(0.48, 0.61 , 0.72, 0.48, 0.57)	(0.84, 0.70, 0.81, 0.95, 0.88)	(0.84, 0.70, 0.81, 0.78,)
		High	(0.48, 0.83, 0.72, 0.62, 0.68)	(0.48, 0.70 , 0.72, 0.48, 0.68)	(0.84, 0.69, 0.81, 0.95, 0.75)	(0.84, 0.80, 0.81, 0.78,)
	High	***Low***	(0.48, 0.59, 0.66, 0.62, 0.57)	(0.48, 0.79, 0.66, 0.48, 0.57)	(0.84, 0.77, 0.91, 0.95, 0.88)	(0.84, 0.77, 0.91, 0.78,)
		High	(0.48, 0.68, 0.66, 0.62, 0.68)	(0.48, 0.81, 0.66, 0.48, 0.68)	(0.84, 0.35, 0.91, 0.95, 0.75)	(0.84, 0.75, 0.91, 0.78,)

Figure 4 shows convergence rate with different amounts of reward parameter *a* (see Eq. (1)) in L_{R-I} learning algorithms using the proposed local rule and the ordinary local rule. The results is obtained from 1000 times running of learning process by starting from preliminary configuration of ((0.5, 0.5), (0.5, 0.5), (0.5, 0.5)) for (LA_1, LA_2, LA_3) in $ICLA_{3,2}$. Figures 5 and 6 show the same diagrams for $ICLA_{4,2}$ and $ICLA_{5,2}$.

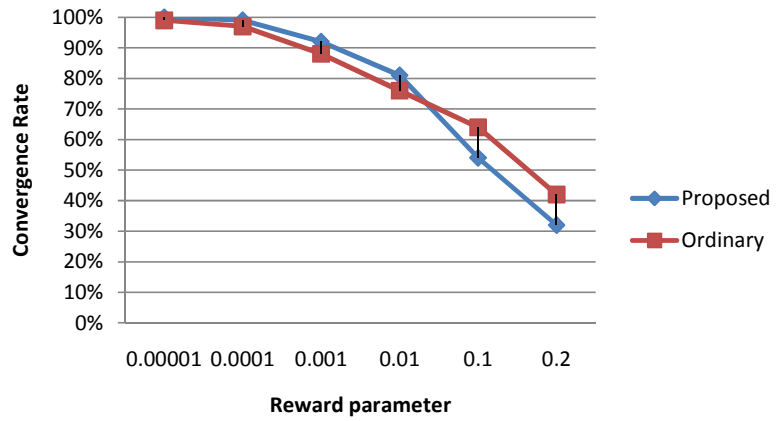


Fig 4- Convergence Rate using different reward parameters in $ICLA_{3,2}$

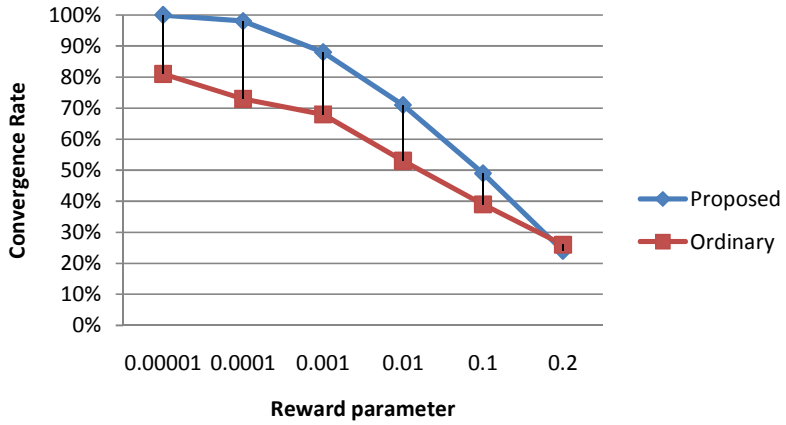


Fig 5- Convergence Rate using different reward parameters in $ICLA_{4,2}$

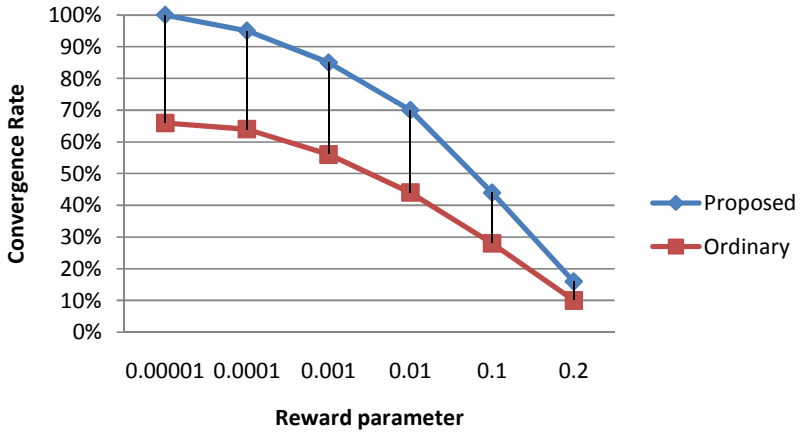


Fig 6- Convergence Rate using different reward parameters in $ICLA_{5,2}$

As illustrated in figure 4, 5 and 6, by bigger reward parameters, we have lower convergence rate by both the local rules. Problem of catching in absorbing states (see the note in section 2.1) is a major reason for lower convergence rate here. Except for big reward parameter values in $ICLA_{3,2}$, the proposed local rule always reaches better convergence rate than the ordinary local rule. The difference in convergence rate is because of the reason that the ordinary local rule uses just the response of the environment in current round to generate reinforcement signal, while the proposed local rule uses a history of the responses (using Q-values). Although the better convergence rate is obtained using the proposed local rule but by increasing reward parameter, drop in convergence rate using the proposed local rule is more sensible than drop using the ordinary local rule. This is for the reason that changes of ICLA configuration won't be negligible in two successive round using big reward parameter values. This causes drop in convergence rate because algorithm 2 assumes the best response in iteration *t* (see $\underline{p}_i^{br}$ in algorithm 2) is too close to the best response in iteration *t+1*.

Using table 1 as the response matrix to generate environment response, $ICLA_{3,2}$ converges to (*Low, High, High*) which is a compatible point. Figures 7 plots the evolution of the probability vector of the learning automaton of cell 2 in $ICLA_{3,2}$ using the ordinary (a) and the proposed local rule (b). As illustrated in this figure by using the proposed local rule, ICLA needs further iterations to converge.

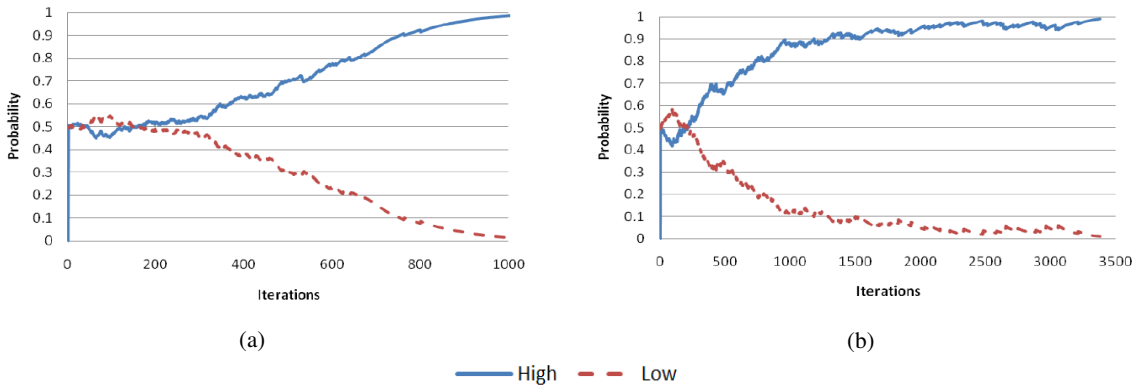


Fig 7- Evolution of probability vector of LA 2 in $ICLA_{3,2}$ using (a) the ordinary (b) the proposed local rule

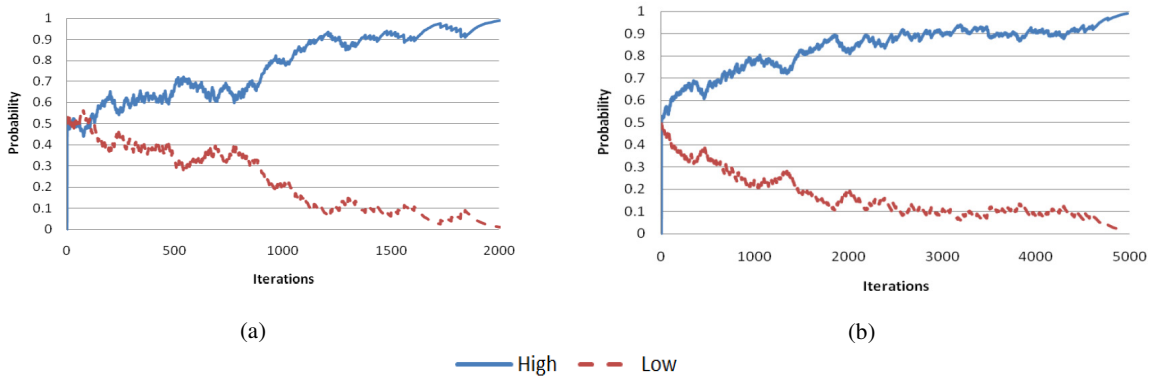


Fig 8- Evolution of probability vector of LA 2 in $ICLA_{4,2}$ using (a) the ordinary (b) the proposed local rule

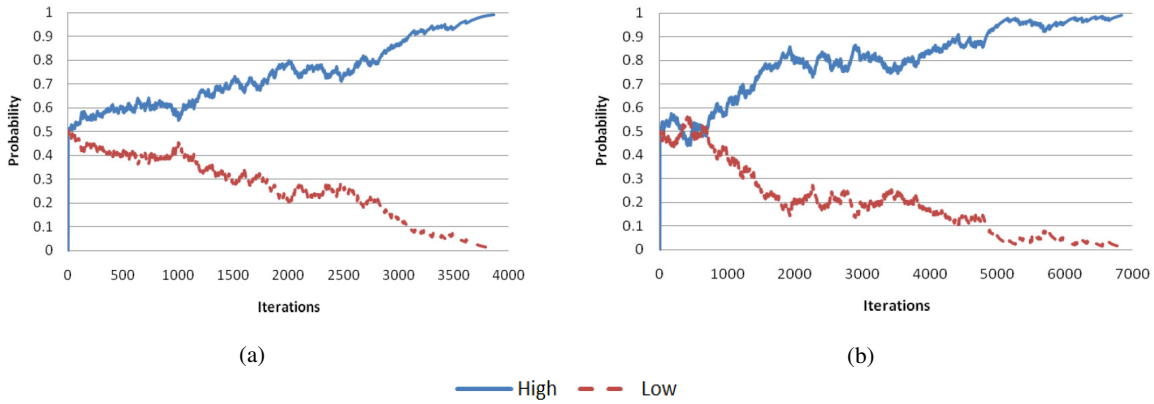


Fig 9- Evolution of probability vector of LA 2 in $ICLA_{5,2}$ using (a) the ordinary (b) the proposed local rule

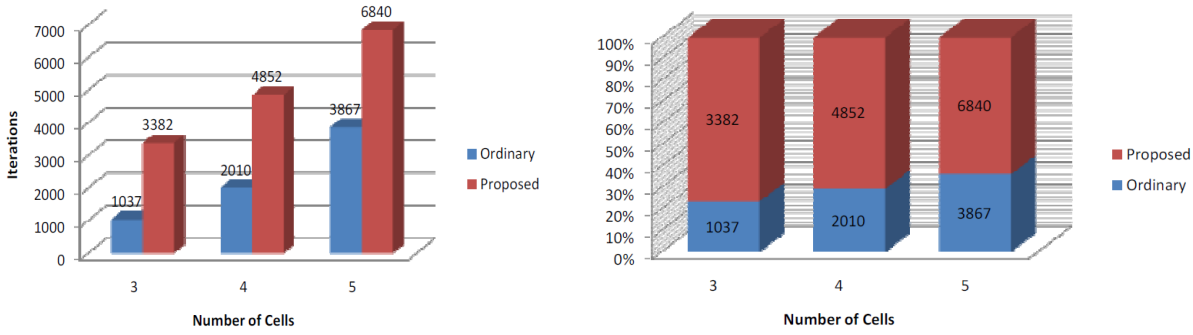


Fig 10- Comparing number of necessary iterations for convergence of ICLA using the proposed and ordinary local rule

Figure 8 and 9 show that similar to figure 7 the proposed local rule needs more iterations to convergence in $ICLA_{4,2}$ and $ICLA_{5,2}$. Now let $L_{nm} = \frac{I_{nm}(\text{Proposed})}{I_{nm}(\text{Ordinary})}$ where $I_{nm}(R)$ denotes the number of iterations $ICLA_{n,m}$ needs for convergence using the local rule *R*. According to the results illustrated in figure 10, we have $L_{32} = 3.6$, $L_{42} = 2.4$ and $L_{52} = 1.7$. For $n \geq 9$, L_{n2} is less than 1. So by increasing number of cells in $ICLA_{n,2}$, it is expected that less iterations to be needed for convergence by the proposed local rule in comparison with the ordinary local rule. Figure 10 (b) demonstrates that by using the ordinary local rule the necessary iterations for convergence in $ICLA_{3,2}$, $ICLA_{4,2}$ and $ICLA_{5,2}$ are 23%, 29% and 36% of ($I_{n2}(\text{Proposed}) + I_{n2}(\text{Ordinary})$) for n = 3, 4 and 5 respectively.

5.1.2. Numerical Experiment 2

The aim of this numerical experiment is to study the convergence of ICLA when it starts learning from different initial configurations. For this experiment we use an $ICLA_{3,3}$ with L_{R-I} learning algorithm. Structure of $ICLA_{3,3}$ is similar to $ICLA_{3,2}$ in figure 3(a). Responses of environment are generated using Table 4. Table 5 contains the initial configurations for this ICLA. For example, configuration 3 shows that the initial probability vector of the learning automaton in cell 2 is [0.2, 0.7, 0.1]. Using table 4 to generate responses of environment, ((0,1,0), (0,0,1), (0,0,1)) is a compatible configuration for $ICLA_{3,3}$. Figure 11 shows convergence rate of ICLA using different values of *a*. As illustrated in figure 11 by having different initial configurations the proposed local rule still has a better convergence rate in all the cases. Moreover using the proposed local rule, the difference between convergence rates by starting from different initial configurations are smaller than the difference in case of using the ordinary local rule. This means convergence rate is less dependent to initial configuration using the proposed local rule. Figure 12 compares this dependency. The best and worst convergence rates are obtained by starting from configuration 2 and configuration 3 respectively. This is for the reason that configuration 2 spatially is nearer to a compatible point than other configurations and configuration 3 is the farthest configuration among the others.

Table 4- Response Matrix for $ICLA_{3,3}$

Reward Probability for (a_1, a_2 , a_3)		***a_2=Low***	***a_2= Normal***	***a_2= High***
a_1=Low	*a_3= Low*	(0.51, 0.81, 0.38)	(0.36, 0.25, 0.84)	(0.53, 0.44, 0.32)
	a_3= Normal	(0.51, 0.76, 0.52)	(0.36, 0.47, 0.39)	(0.53, 0.25, 0.57)
	a_3= High	(0.51, 0.45, 0.77)	(0.36, 0.66, 0.71)	(0.53, 0.36, 0.90)
a_1=Normal	*a_3= Low*	(0.87, 0.32, 0.38)	(0.65, 0.90, 0.84)	(0.92, 0.51, 0.32)
	a_3= Normal	(0.87, 0.61, 0.52)	(0.65, 0.54, 0.39)	(0.92, 0.75, 0.57)
	a_3= High	(0.87, 0.71, 0.77)	(0.65, 0.82, 0.71)	**(0.92, 0.95, 0.90)**
a_1=High	*a_3= Low*	(0.44, 0.37, 0.38)	(0.73, 0.61, 0.84)	(0.71, 0.65, 0.32)
	a_3= Normal	(0.44, 0.49, 0.52)	(0.73, 0.37, 0.39)	(0.71, 0.51, 0.57)
	a_3= High	(0.44, 0.56, 0.77)	(0.73, 0.80, 0.71)	(0.71, 0.73, 0.90)

Table 5- Initial Configurations

	Configuration 1			**Configuration 2**		
	Low	**Normal**	**High**	**Low**	**Normal**	**High**
LA in Cell 1	0.33	0.33	0.34	0.1	0.7	0.2
LA in Cell 2	0.33	0.33	0.34	0.2	0.2	0.6
LA in Cell 3	0.33	0.33	0.34	0.05	0.05	0.9
	Configuration 3			**Configuration 4**		
	Low	**Normal**	**High**	**Low**	**Normal**	**High**
LA in Cell 1	0.6	0.2	0.1	0.3	0.5	0.2
LA in Cell 2	0.2	0.7	0.1	0.6	0.2	0.2
LA in Cell 3	0.35	0.5	0.15	0.2	0.1	0.7

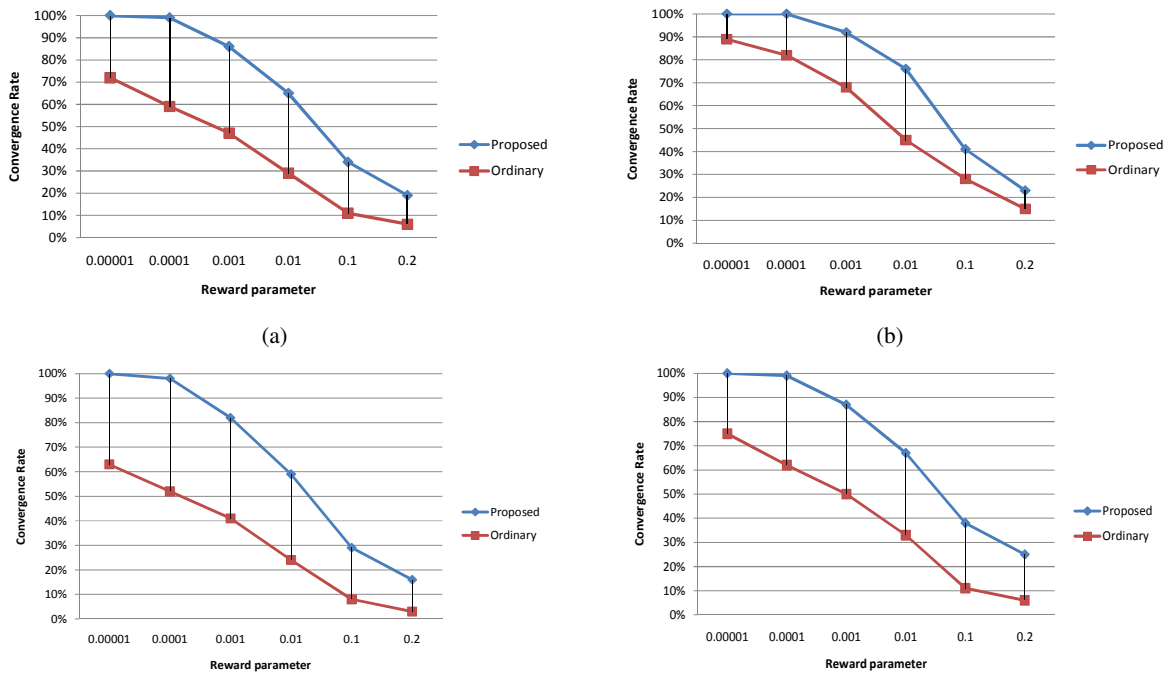


Fig 11- Convergence rate using the Proposed and Ordinary local rule by starting from (a) Configuration 1 (b) Configuration 2 (c) Configuration 3 (d) Configuration 4

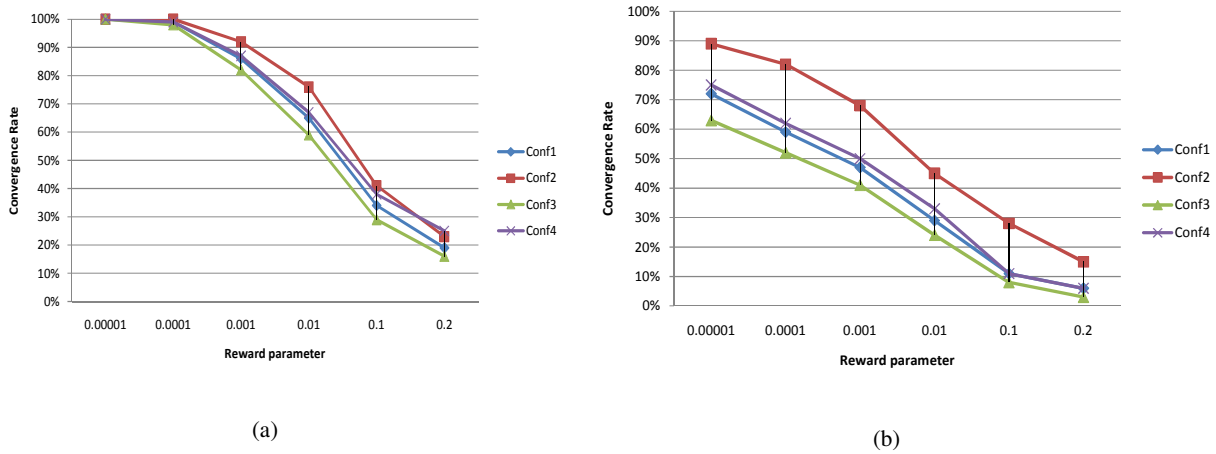


Fig 12- Comparing Convergence Rate by starting from different initial configurations using (a) the proposed local rule (b) the ordinary local rule

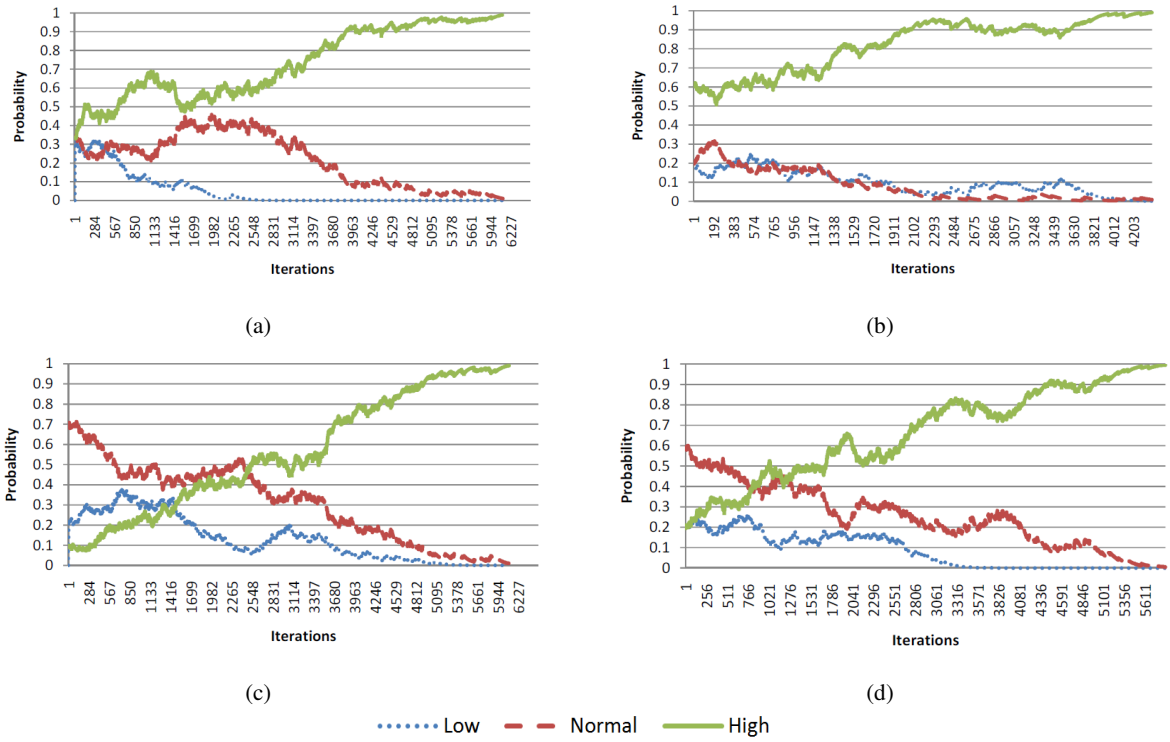


Fig 13- Evolution of probability vector of LA 2 in $ICLA_{3,3}$ by starting from (a) Configuration 1 (b) Configuration 2 (c) Configuration 3 (d) Configuration 4

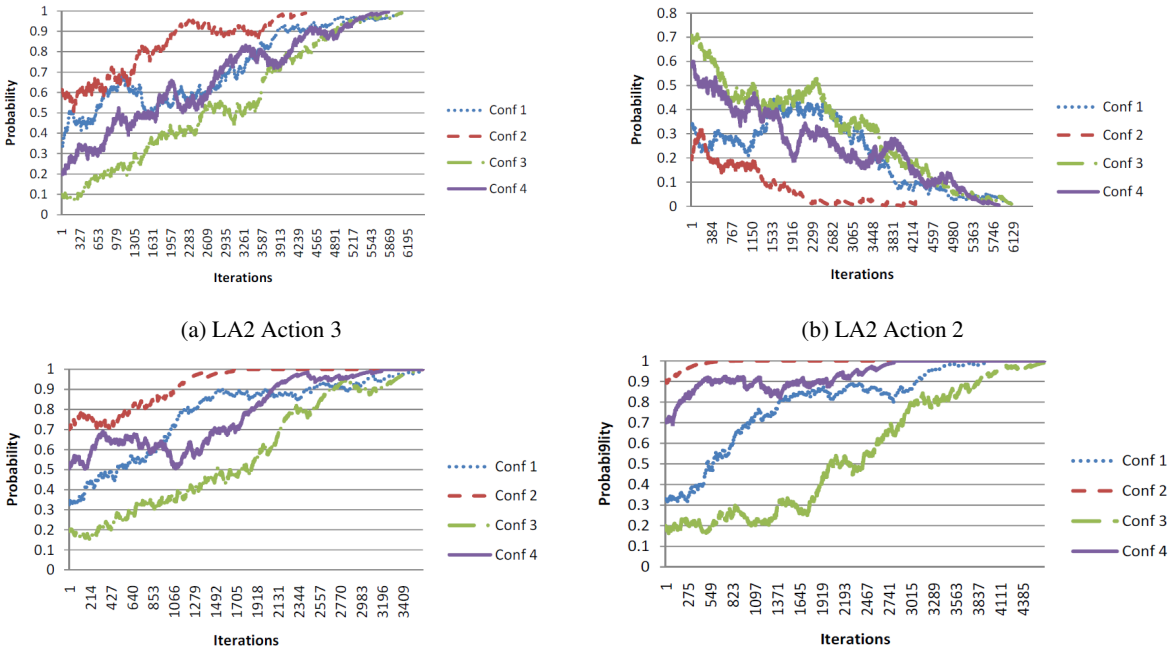
Figure 13 plots the evolution of the probability vector of learning automaton in cell 2 of $ICLA_{3,3}$ by starting from (a) Configuration 1, (b) Configuration 2, (c) Configuration 3 and (d) Configuration 4. Figure 14 plots the evolution of the action probability for selected actions of learning automata using the four mentioned initial configurations of table 5. As it can be seen from figure 14, regardless of what the initial configuration is, ICLA converges to same configuration. Figures 14 (a), (c) and (d) show that the configuration, ICLA is converged to it, is ((0, 1, 0), (0, 0, 1), (0, 0, 1)) which is compatible point of table 4.

5.2. Channel Assignment Problem

In this section, we consider application of the proposed local rule in ICLA for channel assignment problem. First a brief introduction to this problem is provided and then mapping the problem to ICLA is explained. Finally results of the conducted experiment are given to show superiority of the proposed local rule.

5.2.1. Cellular Networks

A cellular network as shown in figure 15 is constituted from multiple small regions called cells. These cells form a covered area by the network. There is base station (BS) located in center of each cell which serve the area covered by the cell. There is a switching center which connects the BSs to each other. This switching center also operates as a gateway that connects the network to the wired networks. A node uses wireless connection to communicate with other nodes of the network through the BS of its cell. A dedicated channel is needed for every connection. If a channel is used by two or more connections at the same time in the same cell or in the neighboring cells, the signals of connections will interfere with each others. These interferences are called co-channel interference. However two non-adjacent cells that their signals do not interfere with each other can use same channel at the same time.



(a) LA2 Action 3

(b) LA2 Action 2

(c) LA1 Action 2

(d) LA3 Action 3

Fig 14- Evolution of probability of (a) action 3 of LA2 (b) action 2 of LA 2 (c) action 2 of LA 1 (d) action 3 of LA 3 by starting from different initial configurations when (*Normal, High, High*) which is equivalent to configuration ((0,1,0),(0,0,1),(0,0,1)) is compatible point

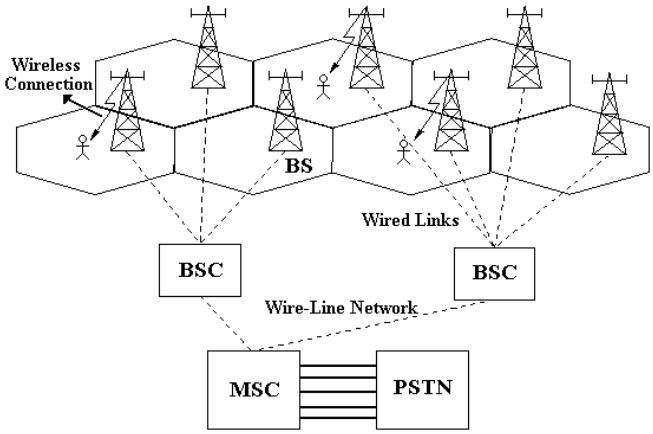


Fig 15- Cellular Network

Assignment of channels to connections is called channel assignment problem. A constraint matrix C is used to illustrate the required gap between assigned channels of cells to have interference-less assignment. For example element *c(u, v)* shows the minimum required gap between assigned channels of cells *u* and *v*. Co-channel reuse distance is the minimum distance at which a channel can be reused without interference. A cluster is set of all neighboring cells that are in co-channel interference range of each other. To have interference-less assignment at most one connection in a cluster can use a channel. There are three common approaches for channel assignment

problem: fixed channel assignment (FCA), dynamic channel assignment (DCA), and hybrid channel assignment (HCA). ICLA can be employed to offer FCA, DCA or HCA solutions [7] but in this paper for simplicity purposes we consider just a FCA solution.

5.2.2. Mapping channel assignment problem to ICLA

FCA allocates a set of channels to each cell permanently. Note that these channels can be used in another cell with enough distance again. When a request arrives at any cell, the BS of the cell allocates a free channel to the request. If all channels of this cell are busy, then the BS blocks the request. To offer a FCA approach, first it should be determined how many channels per cell are needed to support incoming requests. This can be done by calculation of the expected traffic load of the network. Then the required channels must be assigned to each cell such that it avoids interference among the neighboring cells. Considering the features of the channel assignment problem, ICLA is a good candidate for solving this problem in cellular networks. In our proposed approach, ICLA evolves until it reaches a compatible point which is a FCA solution for the channel assignment problem. It is assumed that estimation for demand is given *a priori.* Let $\hat{d}_v$ denote the expected number of required channels for cell *v*. Now assume a cellular network with *n* cells and total of *m* available channels. This network can be modeled as an ICLA with *n* cells, where there are $\hat{d}_v$ learning automata in cell *v*. Each learning automaton has *m* actions to choose (one action per available channel) and the learning algorithm is L_{R-I}. All the cells in interference area of a cell are considered to be neighboring cells of that cell in ICLA. Let to put a label with superscript *j* ($j = 1,...,\overline{m}_{vi}$) on all neighbors of the *i* th learning automaton located in cell *v* (LA_{vi}). $\overline{m}_{vi}$ is total number of learning automatons which are located in cell *v* or neighboring cells to *v*. Furthermore let (a^1 ,..., a^j ,..., $a^{\overline{m}_{vi}}$) denotes joint actions of LA_{vi} and its neighbors. To reward or punish the selected action of LA_{vi} according to the proposed local rule we need value of $Er_{vi}^t(a^1,...,a^{\overline{m}_{vi}})$ (see Eq. (8)). To calculate $r_{vi}^t(a^1,...,a^{\overline{m}_{vi}})$ we have used the constraint matrix C as illustrated by Eq. (13). $C_{\max}$ is maximum value among the elements of the constraint matrix C.

$$r_{vi}^t(a^1,...,a^{\overline{m}_{vi}}) = \sum_{k=1}^{\overline{m}_{vi}-1} \sum_{j=k+1}^{\overline{m}_{vi}} (C_{\max} - C(a^k, a^j)) \qquad (13)$$

In the following section, we repeat the experiments of [7] for FCA approach and compare results of the proposed local rule with the results of the proposed local rule in FCA algorithm in [7]. In following the local rule of this algorithm is called the ordinary local rule.

5.2.3. Simulation Results

To illustrate the superiority of the proposed local rule, we give its results for a simplified version of Philadelphia problem. The Philadelphia problem is a channel assignment problem based on a realistic cellular mobile network covering this city. We use a FCA algorithm which is proposed in [7]. Figure 16 shows this algorithm. To apply the proposed local rule to this algorithm we must use the reward (step 7) to calculate Er_i in Eq. (7). Then reward or punishment for the action j is obtained using algorithm 2. In the following we compare the results of the proposed local rule with results of the ordinary local rule in this algorithm (see line 6 and 7).

```
1   Initialize the ICLA.
2   While interference for assignment is not found do
3     for every cell in the ICLA concurrently do
4       for every LA i in cell u concurrently do
5         choose an action. Let j be chosen action of this LA.
6         if channel j doesn't interfere with channels used in the neighboring cells then
7           reward the action j of LA i in cell u
8         end if
9       end for
10    end for
11  end while
```

Fig 16- FCA Algorithm

In simplified version of Philadelphia problem, the interference $c(i, j)$ is defined as follows:

$$c(i,j)=\begin{cases}2, & if \;\; i=j\\ 1, & if \;\; d(i,j)=1\\ 0, & otherwise\end{cases} \tag{14}$$

$c(i, j) = 0$ means there is no interference between two cells *i* and *j*. Covering network in Philadelphia problem is similar to a regular grid with 21 cells as shown in figure 17. If we map this grid to an ICLA then each cell and its six neighboring cells, which constitute a cluster, define neighboring cells in ICLA. Moreover when there are multiple learning automata in each cell, all those are neighbors of each other as well. Similar to [7] we consider two demand vectors which is given in Table 6.

Figure 18 illustrates the number of interfering channels for demand vector 1 using the proposed local rule versus ordinary local rule. Due to randomness of the demands for connections, conducting the described experiment in [7] every time may lead to different number of interfering channels. Therefore for the purpose of better evaluation, we repeated the mentioned experiments for demand vector 1 and 2 thousand times and compared the average results. Figure 19 show average number of interfering channels for demand vector 1. As illustrated in figure 19 (a), by existence of 4 channels, average number of interfering channels using the proposed local rule is less than what it is using the ordinary local rule. Figures 19 (b), 19 (c) and 19 (d) support the idea of superiority of the proposed local rule as well. Moreover, using the proposed local rule ICLA converges faster than what it converges using the ordinary local rule. For example in fig 19 (a) by using the proposed local rule, convergence is reached before iteration 170000 while it takes 20000 iterations by using the ordinary local rule. Faster convergence of ICLA using the proposed local rule is illustrated by figures 19 (b), 19 (c) and 19 (d) as well. Comparing four diagrams of figure 19, it is obvious that convergence of ICLA in parts (b), (c) and (d) is before iteration 26000 which is too faster than what illustrated in 19(a). This is for the reasons that existence of 4 channels is nearer to minimum number of channels (3 channels) required for reaching interfere less assignment. Having fewer channels causes longer time to be needed for convergence. Figures 20, 21 and 22 show the result of the experiment for demand vector 2. The diagrams show faster convergence and smaller average number of interfering channels using the proposed local rule. These results support idea of superiority of the proposed local rule.

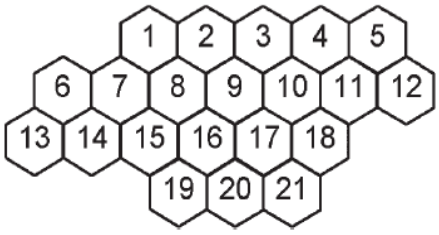
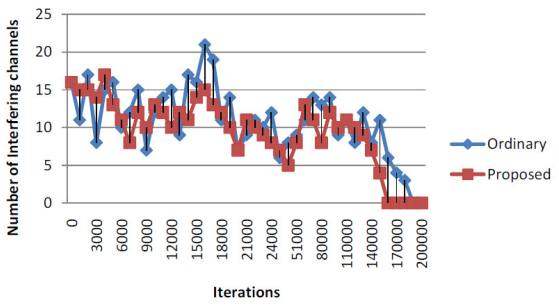


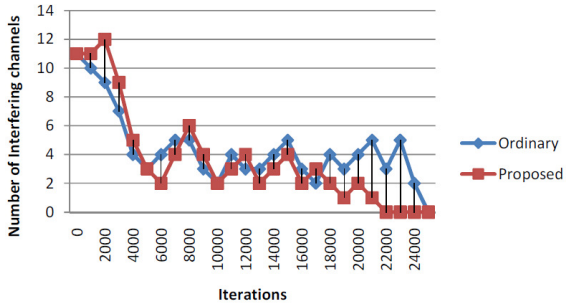
Fig 17- Regular grid of Philadelphia problem

Table 6- Demand Vectors for Philadelphia problem

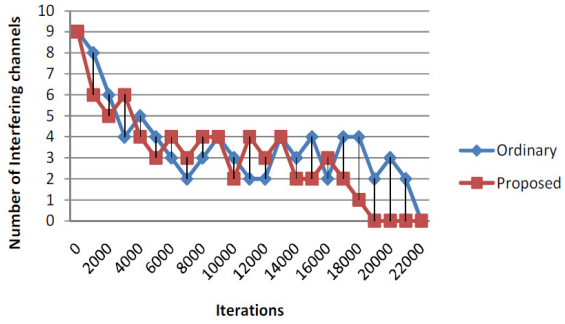
Cell	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*	*16*	*17*	*18*	*19*	*20*	*21*
Modified Vector 1	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*
Modified Vector 2	*1*	*1*	*1*	*1*	*1*	*2*	*2*	*2*	*2*	*2*	*2*	*2*	*2*	*2*	*1*	*1*	*2*	*2*	*2*	*2*	*2*



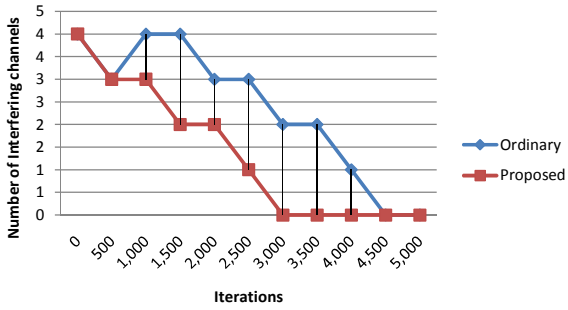
(a)



(b)



(c)



(d)

Fig 18- Comparing interference for demand vector 1 with (a) 4 channels (b) 5 channels (c) 6 channels (d) 7 channels using the proposed local rule and ordinary local rule

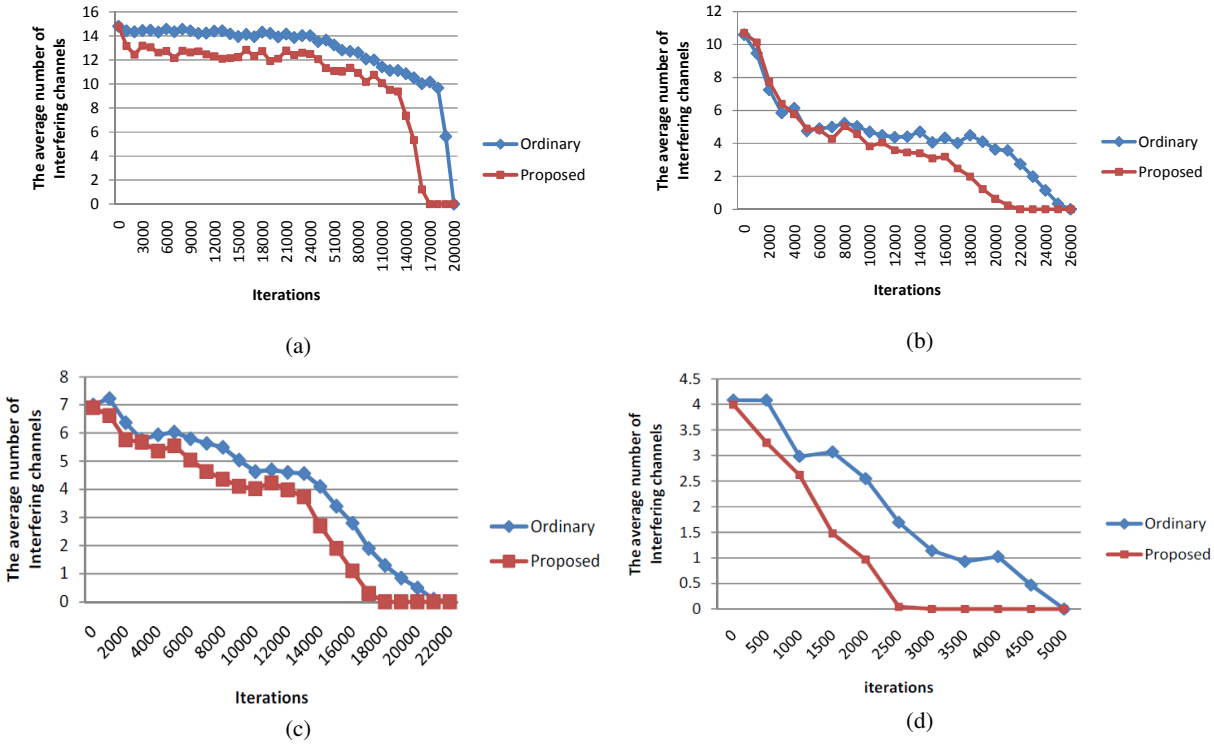


Fig 19- Comparing average number of interference over 1000 times for demand vector 1 with (a) 4 channels (b) 5 channels (c) 6 channels (d) 7 channels using the proposed local rule and ordinary local rule

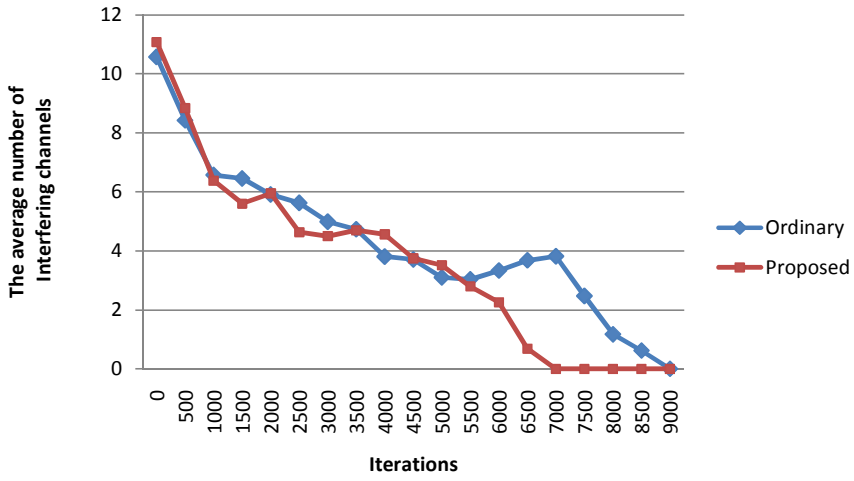


Fig 20- Comparing average of interference over 1000 times for demand vector 2 with 16 channels using the proposed and ordinary local rules

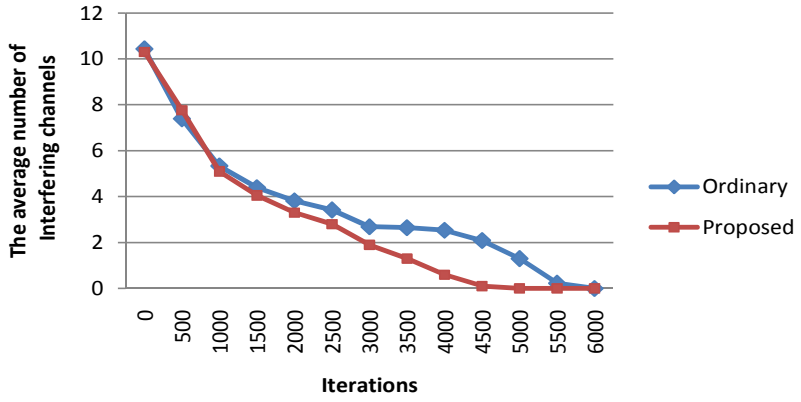


Fig 21- Comparing average of interference over 1000 times for demand vector 2 with 17 channels using the proposed and ordinary local rules

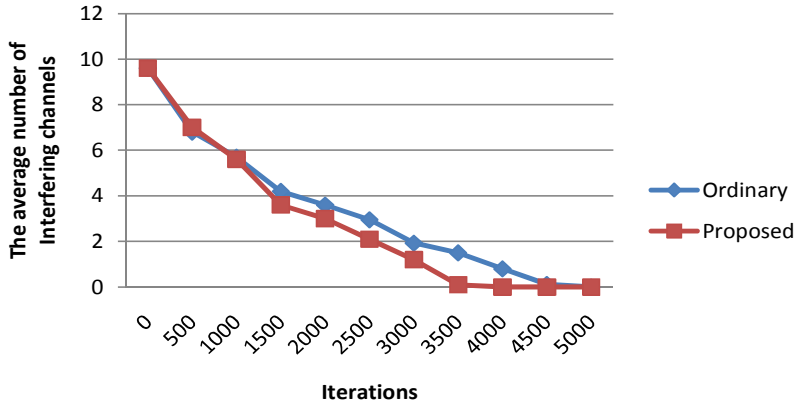


Fig 22- Comparing average of interference over 1000 times for demand vector 2 with 18 channels using the proposed and ordinary local rules

6. Conclusion

ICLA is a powerful mathematical model for decentralized applications. Convergence of ICLA to a compatible point has great importance in studying behavior of ICLA. With a simple local rule which rewards or punishes LAs just based on response of environment and selected actions of neighbors, convergence of ICLA to a compatible point is not guaranteed and ICLA shows an unpredictable behavior. In this paper, we proposed a new local rule which guarantees convergence of ICLA to a compatible point. Formal proofs for existence of a compatible point and convergence are provided. We provided different experiments to show usefulness and superiority of the proposed local rule. The obtained results from experiments support our idea about superiority of the proposed local rule against the ordinary local rule which simply maps response of environment to rewards or punishments. For future works, we aim to study and analysis behavior of ICLA using the proposed local rule under different conditions such as existence of multiple compatible points. Having multiple compatible points, existence of an optimal compatible point, and convergence of ICLA to it is an appealing subject. A compatible point is optimal if the obtained reward by each learning automaton in ICLA is the maximum reward that can be obtained among all the other compatible points. Unlike compatible points, existence of an optimal compatible point is not guaranteed always and its existence is application-dependent.

References

[1] Akbari Torkestani, J., & Meybodi, M. R. Clustering the wireless Ad Hoc networks: A distributed learning automata approach. Journal of Parallel and Distributed Computing, 70(4), 394-405 (2010).

[2] Akbari Torkestani, J., & Meybodi, M. R. An intelligent backbone formation algorithm for wireless ad hoc networks based on distributed learning automata. Computer Networks, 54(5), 826-843 (2010).

[3] Akbari Torkestani, J., & Meybodi, M. R. A cellular learning automata-based algorithm for solving the vertex coloring problem. Expert Systems with Applications, 38(8), 9237-9247 (2011).

[4] Asnaashari, M., & Meybodi, M. R. Irregular cellular learning automata and its application to clustering in sensor networks. In Proceedings of 15th Conference on Electrical Engineering (15th ICEE), Volume on Communication, Telecommunication Research Center, Tehran, Iran (2007).

[5] Beigy, H., & Meybodi, M. R. A self-organizing channel assignment algorithm: A cellular learning automata approach. In Intelligent Data Engineering and Automated Learning (pp. 119-126). Springer Berlin Heidelberg (2003).

[6] Beigy, H., & Meybodi, M. R. A mathematical framework for cellular learning automata. Advances in Complex Systems, 295-319 (2004).

[7] Beigy, H., & Meybodi, M. R. Cellular learning automata with multiple learning automata in each cell and its applications. IEEE Transactions on Systems, Man, and Cybernetics, Part B, 40(1), 54-65 (2010).

[8] Billard, E. A. Chaotic behavior of learning automata in multi-level games under delayed information. IEEE International Conference on Systems, Man, and Cybernetics, Vol. 2, pp. 1412-1417, (1997).

[9] Billard, E. A. Asymmetry in learning automata playing multi-level games. IEEE International Conference on Systems, Man, and Cybernetics, Vol. 3, pp. 2202-2206, (1998).

[10] Billard, E. A. Instabilities in learning automata playing games with delayed information. IEEE International Conference on Systems, Man, and Cybernetics, Vol. 2, pp. 1160-1165, (1994).

[11] Cheng, D., Qi, H., & Li, Z. Analysis and control of Boolean networks: a semi-tensor product approach. Springer Science & Business Media, (2010).

[12] Cheng, D., & Zhao, Y. Identification of Boolean control networks. Automatica, 47(4), 702-710, (2011).

[13] Eraghi, A. E., Torkestani, J. A., & Meybodi, M. R. Cellular learning automata-based graph coloring problem. International Conference on Machine Learning and Computing (ICMLC 2009), Perth, Australia (2011).

[14] Esnaashari, M. & Meybodi, M. R., Irregular Cellular Learning Automata, IEEE Transactions on Cybernetics, Issue 99, (2014).

[15] Esnaashari, M., & Meybodi, M. R. Deployment of a mobile wireless sensor network with k-coverage constraint: a cellular learning automata approach. Wireless Networks, 1-24 (2012)

[16] Kashki, M., Abido, M. A., & Abdel-Magid, Y. L. Pole placement approach for robust optimum design of PSS and TCSC-based stabilizers using reinforcement learning automata. Electrical Engineering, 91(7), 383-394 (2010)

[17] Lakshmivarahan, S., & Narendra, K. S. Learning algorithms for two-person zero-sum stochastic games with incomplete information. Mathematics of Operations Research, 6(3), 379-386 (1981)

[18] Liu, L., Hu, G., Xu, M., & Peng, Y. Learning automata based spectrum allocation in cognitive networks. IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS), pp. 503-508, (2010).

[19] Ma, H., Wang, D., Qi, H., & Fu, M. An approach of Bayesian filtering for stochastic Boolean dynamic systems. In Proceedings of the 26th Chinese Control and Decision Conference (2014 CCDC), pp. 4335-4340, (2014).

[20] Masoumi, B., & Meybodi, M. R. Learning automata based multi□agent system algorithms for finding optimal policies in Markov games. Asian Journal of Control, 14(1), 137-152 (2012).

[21] Meybodi, M. R., Beigy, H. & Taherkhani, M. Application of cellular learning automata to image processing, In Proceedings of First Conference in Mathematics and Communication, Iranian Telecommunication Research Center, pages 23.1–23.10, Tehran, Iran (2000)

[22] Meybodi, M. R., & Taherkhani, M. Application of cellular learning automata to modeling of rumor diffusion. In Proceedings of 9th Conference on Electrical Engineering, Power and Water institute of Technology, Tehran, Iran, 102-110 (2001)
[23] Meybodi, M. R., & Khojasteh, M. R. Application of cellular learning automata in modeling of commerce networks. In Proceedings of 6th Annual International Computer Society of Iran Computer Conference CSICC2001, Isfahan, Iran, 284-295 (2001)
[24] Meybodi, M. R., & Kharazmi, M. R. Application of cellular learning automata to image processing. Journal of Amirkabir, 14(56A), 1101-1126 (2004)
[25] Misra, S., Tiwari, V., & Obaidat, M. LACAS: learning automata-based congestion avoidance scheme for healthcare wireless sensor networks. IEEE Journal on Selected Areas in Communications, 27(4), 466-479 (2009).
[26] Narendra, K. S., & Thathachar, M. A. Learning automata: an introduction. Englewood Cliffs, NJ: Prentice-Hall (1989).
[27] Nisan, N., Schapira, M., Valiant, G., Zohar, A., Best-Response Mechanisms, in: ICS, 2011, pp. 155-165.
[28] Sastry, P. S., Phansalkar, V. V., & Thathachar, M. A. L. Decentralized learning of Nash equilibria in multi-person stochastic games with incomplete information. IEEE Transactions on Systems, Man and Cybernetics, 24(5), 769-777 (1994).
[29] Szepesvári, C., & Littman, M. L. A unified analysis of value-function-based reinforcement-learning algorithms. Neural computation, 11(8), 2017-2060 (1999)
[30] Thathachar, M. A., & Sastry, P. S. Networks of learning automata: Techniques for online stochastic optimization. Springer (2004)
[31] Thierens, D. An adaptive pursuit strategy for allocating operator probabilities. In Proceedings of conference on Genetic and evolutionary computation (pp. 1539-1546). ACM (2005).
[32] Tilak, O., Mukhopadhyay, S., Tuceryan, M., & Raje, R. A novel reinforcement learning framework for sensor subset selection. International Conference on Networking, Sensing and Control (ICNSC), pp. 95-100, (2010).
[33] Tilak, O., Martin, R., & Mukhopadhyay, S. Decentralized indirect methods for learning automata games. IEEE Transactions on Systems, Man, and Cybernetics: Part B, 41(5), 1213-1223 (2011).
[34] Tuan, T. A., Tong, L. C., & Premkumar, A. B. An adaptive learning automata algorithm for channel selection in cognitive radio network. IEEE International Conference on Communications and Mobile Computing (CMC), (Vol. 2, pp. 159-163), (2010).
[35] Vrancx, P., Verbeeck, K., & Nowé, A. Decentralized learning in markov games. IEEE Transactions on Systems, Man, and Cybernetics, Part B, 38(4), 976-981 (2008).
[36] Vrancx, P., Verbeeck, K., & Nowé, A. Networks of learning automata and limiting games. In Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning (pp. 224-238). Springer Berlin Heidelberg (2008).
[37] Wheeler Jr, R., & Narendra, K. Decentralized learning in finite Markov chains. IEEE Transactions on Automatic Control, 31(6), 519-526 (1986).
[38] Zhong, W., Xu, Y., & Tao, M. Precoding strategy selection for cognitive mimo multiple access channels using learning automata. IEEE International Conference on Communications (ICC), (pp. 1-5), (2010).

Appendix A

Proof of Lemma 1: The proof for ICLA is somewhat similar to lemma 2 in [6]. Let' us define a mapping $T:\mathrm{K}\to\mathrm{K}$ as Eq. (A-1).

$$\forall i,\forall r\in\{1,...,m_i\}\qquad \overline{p}_{ir}=\frac{p_{ir}+\phi_{ir}}{1+\sum_{j=1}^{m_i}\phi_{ij}} \tag{A-1}$$

Where $\psi_{ir}=d_{ir}(\underline{p})-D_i(\underline{p})$ and $\phi_{ir}(\underline{p})=\max\{\psi_{ir}(\underline{p}),0\}$.

Because T is continuous and K is closed, bounded and convex so based on Brouwer's fixed point theorem, T has at least one fixed point. We claim that a point will be a fixed point if and only if it be a compatible point of ICLA. To show this, assume that $\underline{p}$ is a fixed point of T. So for all r and i: $\overline{p}_{ir}=p_{ir}$. With respect to this fact and Eq. (A-1), we have:

$$\forall i,\forall r\in\{1,...,m_i\}\qquad p_{ir}=\frac{p_{ir}+\phi_{ir}}{1+\sum_{j=1}^{m_i}\phi_{ij}} \tag{A-2}$$

From Eq. (A-2) we can conclude Eq. (A-3).

$$\sum_{j=1}^{m_i}\phi_{ij}=\frac{\phi_{ir}}{p_{ir}} \tag{A-3}$$

Eq. (A-3) results in Eq. (A-4):

$$\frac{\phi_{i1}}{p_{i1}}=\frac{\phi_{i2}}{p_{i2}}=...=\frac{\phi_{ir}}{p_{ir}} \tag{A-4}$$

Being $\phi_{ir}>0$ for all ϕ_{ir} is not possible, so the only case that satisfy Eq. (A-4) is the case that

$$\forall i,m\qquad \phi_{ir}=0 \tag{A-5}$$

Now assume that $y_i(1\le i\le m_i)$ is the action that reaches maximum value for $d_{iy_i}(\underline{p})$ among the others. Consider another action, $y_j(j\neq i)$ that it's $d_{iy_j}(\underline{p})$ is smaller than $d_{iy_i}(\underline{p})$. Assume that probability distribution $\underline{p}$ puts a positive probability value on y_j, so we will have $D_i(\underline{p})<d_{iy_i}(\underline{p})$ that means $\phi_{iy_i}(\underline{p})>0$. This is in contradiction with Eq. (A-5) so the assumption is not valid and probability distribution $\underline{p}$ puts zero probability values on all action that their $d_{iy_j}(\underline{p})$ is smaller than $d_{iy_i}(\underline{p})$. According to this fact we have Eq. (A-6):

$$\forall \underline{q}\in\mathrm{K}\qquad \sum_r d_{ir}(\underline{p})\times p_{ir}\ge\sum_r d_{ir}(\underline{p})\times q_{ir} \tag{A-6}$$

So $\underline{p}$ is a compatible point. Conversely if $\underline{p} \in \mathrm{K}$ is a compatible point, then Eq. (A-6) will be valid for all *i*. configuration $\underline{q}$ also contains $\underline{q} = (\underline{p}_1,...,\underline{e}_{r_i},...,\underline{p}_n)$ for fixed *i*. Since $d_{ir_i}(\underline{p})$ is independent of $\underline{p}_i$, we have $\psi_{ir_i}(\underline{p}) \le 0$. Therefore $\phi_{ir_i}(\underline{p}) = 0$ for all *i* and all $r_i = 1,...,m_i$. So $\underline{p}$ is a fixed point of T. □

Appendix B

Proof of Lemma 2:

Notations:

$\underline{p}_i^t$ is the probability vector of LA_i at iteration *t* and is not known to the other learning automata

$\underline{\hat{p}}_i^t$ is estimation of $\underline{p}_i^t$ by observing the chosen actions of LA_i during L.

Proof: When ICLA reaches a stable point, in that point there exist a $\underline{p}^*$ such that for every $\varepsilon > 0$ we have

$$\lim_{t\to\infty} P(|\underline{p}^t - \underline{p}^*| > \varepsilon) = 0 \quad \text{(B-1)}$$

It's obvious that $\lim_{t\to\infty} P(|\underline{p}_i^t - \underline{p}_i^*| > \varepsilon) = 0$ (B-2)

Now if we can prove (B-3) the proof will be completed.

$$\lim_{t\to\infty} P(|\underline{\hat{p}}_i^t - \underline{p}_i^*| > \varepsilon) = 0 \quad \text{(B-3)}$$

In algorithm 1, $n_i^t(a_j)$ can be replaced by summation of X^t random variables where X^t defined as:

$$X^t = \begin{cases} 1 & \text{if action } a_j \text{ was chosen during epoch t} \\ 0 & \text{otherwise} \end{cases}$$

Because changes of strategies in each epoch are negligible so the probability to choose action a_j during epoch *t* is a fixed value like as $\underline{p}_{ij}^t$. Then we have:

$$E(X^t) = 1\times\underline{p}_{ij}^t + 0\times(1-\underline{p}_{ij}^t) = \underline{p}_{ij}^t \quad \text{(B-4)}$$

So $E(n_i^t(a_j)) = E(\frac{X_1^t + ... + X_L^t}{L}) = \frac{1}{L}\times(E(X_1^t) + ... + E(X_L^t)) = \frac{1}{L}\times L\times\underline{p}_{ij}^t = \underline{p}_{ij}^t$ (B-5)

Now using algorithm 1 we can see that $\underline{\hat{p}}_{ij}^{t+1} = \frac{1}{2}\times\underline{\hat{p}}_{ij}^t + \frac{1}{2}\times\frac{n_i^t(a_j)}{L}$.

So $\underline{\hat{p}}_{ij}^{t+1} = \frac{1}{2^t}\times\underline{\hat{p}}_{ij}^1 + \frac{1}{2^{t-1}}\times\frac{n_i^2(a_j)}{L} + \frac{1}{2^{t-2}}\times\frac{n_i^3(a_j)}{L} + ... + \frac{1}{2}\times\frac{n_i^t(a_j)}{L}$

$$E(\underline{\hat{p}}_{ij}^{t+1}) = E(\frac{1}{2^t}\times\frac{n_i^1(a_j)}{L} + ... + \frac{1}{2}\times\frac{n_i^t(a_j)}{L}) \Rightarrow E(\underline{\hat{p}}_{ij}^{t+1}) = \frac{1}{2^t}\times\underline{p}_{ij}^1 + ... + \frac{1}{2}\times\underline{p}_{ij}^t$$

$$\lim_{t\to\infty} E(\underline{\hat{p}}_{ij}^{t+1}) = \lim_{t\to\infty}(.\frac{1}{2^t}\times\underline{p}_{ij}^1 + ... + \frac{1}{2}\times\underline{p}_{ij}^t) = ... + 0 + 0 + ... + \frac{1}{2^k}\times\lim_{t\to\infty}(\underline{p}_{ij}^{t-k+1}) + ... + \frac{1}{2}\times\lim_{t\to\infty}(\underline{p}_{ij}^t) \quad \text{(B-6)}$$

$$\lim_{t\to\infty} E(\underline{\hat{p}}_{ij}^{t+1}) = 0 + 0 + 0 + ... + \frac{1}{2^2}\times\underline{p}_{ij}^* + \frac{1}{2}\times\underline{p}_{ij}^* = \underline{p}_{ij}^*(\sum_{t=1}^{\infty}\frac{1}{2^t}) = \underline{p}_{ij}^* \quad \text{(B-7)}$$

So in general case, we have

$$\lim_{t\to\infty} E(\underline{\hat{p}}_i^t) = \underline{p}_i^* \quad \text{(B-8)}$$

Since $\underline{p}_i^*$ is the expected value of $\underline{\hat{p}}_i^t$ from the central limit theorem we get:

$$\lim_{t\to\infty} P(|\underline{\hat{p}}_i^t - \underline{p}_i^*| > \varepsilon) = 0 \quad \text{(B-9)}$$

So the proof is completed. □

Appendix C

Proof of Lemma 3: Let' us to define a mapping $P_t : \{Q\} \to \{Q\}$ as Eq. (C-1).

$$P_t^i Q_t^i(a^1,...,a^{\overline{m}}) = Er_i^t(a^1,...,a^{\overline{m}}) + \beta.\sum_{a^1}\sum_{a^2}...\sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{\hat{p}}^t, Q_t^i)(a^i)\prod_{\substack{j=1\\ j\neq i}}^{\overline{m}} \underline{\hat{p}}^t{}_j(a^j).Q_t^i(a^1,...,a^{\overline{m}}) \tag{C-1}$$

We have also Eq. (C-2).

$$P_t^i Q_*^i(a^1,...,a^{\overline{m}}) = Er_i^t(a^1,...,a^{\overline{m}}) + \beta.\sum_{a^1}\sum_{a^2}...\sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{p}^*, Q_*^i)(a^i)\prod_{\substack{j=1\\ j\neq i}}^{\overline{m}} \underline{p}^*{}_j(a^j).Q_*^i(a^1,...,a^{\overline{m}}) \tag{C-2}$$

It's straight forward that

$$\| P_t^i Q_t^i(a^1,...,a^{\overline{m}}) - P_t^i Q_*^i(a^1,...,a^{\overline{m}}) \| = \| Er_i^t(a^1,...,a^{\overline{m}}) + \beta.\sum_{a^1}\sum_{a^2}...\sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{\hat{p}}^t, Q_t^i)(a^i)\prod_{\substack{j=1\\ j\neq i}}^{\overline{m}} \underline{\hat{p}}^t{}_j(a^j).Q_t^i(a^1,...,a^{\overline{m}}) -$$

$$Er_i^t(a^1,...,a^{\overline{m}}) - \beta.\sum_{a^1}\sum_{a^2}...\sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{p}^*, Q_*^i)(a^i)\prod_{\substack{j=1\\ j\neq i}}^{\overline{m}} \underline{p}^*{}_j(a^j).Q_*^i(a^1,...,a^{\overline{m}}) \| \tag{C-3}$$

Since result of (C-3) is independent of $Er_i^t(a^1,...,a^{\overline{m}})$, so we have Eq. (C-4)

$$\| P_t^i Q_t^i(a^1,...,a^{\overline{m}}) - P_t^i Q_*^i(a^1,...,a^{\overline{m}}) \| = \beta.| \sum_{a^1}\sum_{a^2}...\sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{\hat{p}}^t, Q_t^i)(a^i)\prod_{\substack{j=1\\ j\neq i}}^{\overline{m}} \underline{\hat{p}}^t{}_j(a^j).Q_t^i(a^1,...,a^{\overline{m}}) -$$

$$- \sum_{a^1}\sum_{a^2}...\sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{p}^*, Q_*^i)(a^i)\prod_{\substack{j=1\\ j\neq i}}^{\overline{m}} \underline{p}^*{}_j(a^j).Q_*^i(a^1,...,a^{\overline{m}}) | \tag{C-4}$$

Let to put $\underline{\hat{p}}_i^t(a_j) = \underline{p}_i^*(a_j) + \varepsilon_i^t(a_j)$, so we have Eq. (C-5):

$$\beta.| \sum_{a^1}...\sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{\hat{p}}^t, Q_t^i)(a^i)\prod_{\substack{j=1\\ j\neq i}}^{\overline{m}} \underline{\hat{p}}^t{}_j(a^j).Q_t^i(a^1,...,a^{\overline{m}}) - \sum_{a^1}...\sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{p}^*, Q_*^i)(a^i)\prod_{\substack{j=1\\ j\neq i}}^{\overline{m}} \underline{p}^*{}_j(a^j).Q_*^i(a^1,...,a^{\overline{m}}) | \leq$$

$$\beta.| \sum_{a^1}...\sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{p}^*, Q_t^i)(a^i)\prod_{\substack{j=1\\ j\neq i}}^{\overline{m}} \underline{p}^*{}_j(a^j).Q_t^i(a^1,...,a^{\overline{m}}) - \sum_{a^1}...\sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{p}^*, Q_*^i)(a^i)\prod_{\substack{j=1\\ j\neq i}}^{\overline{m}} \underline{p}^*{}_j(a^j).Q_*^i(a^1,...,a^{\overline{m}}) | +$$

$$\beta.| \sum_{a^1}...\sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{p}^*, Q_t^i)(a^i)\prod_{\substack{j=1\\ j\neq i}}^{\overline{m}} \underline{\varepsilon}^t{}_j(a^j).Q_t^i(a^1,...,a^{\overline{m}}) | \tag{C-5}$$

Using Eq. (B-9), we have:

$$\lim_{t\to\infty} | \sum_{a^1}...\sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{p}^*, Q_t^i)(a^i)\prod_{\substack{j=1\\ j\neq i}}^{\overline{m}} \underline{\varepsilon}^t{}_j(a^j).Q_t^i(a^1,...,a^{\overline{m}}) | = 0 \tag{C-6}$$

Now let the following notations:

$$EQ_t = \sum_{a^1} ... \sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{p}^*, Q_t^i)(a^i) \prod_{\substack{j=1 \\ j \neq i}}^{\overline{m}} \underline{p}^*_j(a^j).Q_t^i(a^1,...,a^{\overline{m}}) \tag{C-7}$$

$$EQ_* = \sum_{a^1} ... \sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{p}^*, Q_*^i)(a^i) \prod_{\substack{j=1 \\ j \neq i}}^{\overline{m}} \underline{p}^*_j(a^j).Q_*^i(a^1,...,a^{\overline{m}}) \tag{C-8}$$

$$\lambda_t = \beta \times | \sum_{a^1} ... \sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{p}^*, Q_t^i)(a^i) \prod_{\substack{j=1 \\ j \neq i}}^{\overline{m}} \underline{\varepsilon}^t_j(a^j).Q_t^i(a^1,...,a^{\overline{m}}) | \tag{C-9}$$

Using Eq. (C-4) and Eq. (C-5) we have Eq. (C-10).

$$\| P_t^i Q_t^i(a^1,...,a^{\overline{m}}) - P_t^i Q_*^i(a^1,...,a^{\overline{m}}) \| \leq \beta.| EQ_t - EQ_* | + \lambda_t \tag{C-10}$$

From Eq. (B-9), we have $\underline{p}_i^{br}(\underline{p}^*, Q_t^i) \to \underline{p}^*_i(a^i)$ and $\underline{p}_i^{br}(\underline{p}^*, Q_*^i) \to \underline{p}^*_i(a^i)$ when $t \to \infty$. Now if $EQ_t \geq EQ_*$ then we have:

$$| EQ_t - EQ_* | = EQ_t - EQ_* = \sum_{a^1} ... \sum_{a^{\overline{m}}} \underline{p}^*_i(a^i) \prod_{\substack{j=1 \\ j \neq i}}^{\overline{m}} \underline{p}^*_j(a^j).Q_t^i(a^1,...,a^{\overline{m}}) -$$

$$\sum_{a^1} ... \sum_{a^{\overline{m}}} \underline{p}^*_i(a^i) \prod_{\substack{j=1 \\ j \neq i}}^{\overline{m}} \underline{p}^*_j(a^j).Q_*^i(a^1,...,a^{\overline{m}}) = \sum_{a^1} ... \sum_{a^{\overline{m}}} \prod_{j=1}^{\overline{m}} \underline{p}^*_j(a^j)(Q_t^i(a^1,...,a^{\overline{m}}) - Q_*^i(a^1,...,a^{\overline{m}}))$$

$$\leq \sum_{a^1} ... \sum_{a^{\overline{m}}} \prod_{j=1}^{\overline{m}} \underline{p}^*_j(a^j) \times \max_{(a^1,...,a^{\overline{m}})} (Q_t^i(a^1,...,a^{\overline{m}}) - Q_*^i(a^1,...,a^{\overline{m}})) = 1 \times \max_{(a^1,...,a^{\overline{m}})} (Q_t^i(a^1,...,a^{\overline{m}}) - Q_*^i(a^1,...,a^{\overline{m}}))$$

$$\leq \max_{(a^1,...,a^{\overline{m}})} | Q_t^i(a^1,...,a^{\overline{m}}) - Q_*^i(a^1,...,a^{\overline{m}}) | = \| Q_t^i(a^1,...,a^{\overline{m}}) - Q_*^i(a^1,...,a^{\overline{m}}) \| \tag{C-11}$$

So using Eq. (C-10) and Eq. (C-11) we have:

$$\| P_t^i Q_t^i(a^1,...,a^{\overline{m}}) - P_t^i Q_*^i(a^1,...,a^{\overline{m}}) \| \leq \beta.\| Q_t^i(a^1,...,a^{\overline{m}}) - Q_*^i(a^1,...,a^{\overline{m}}) \| + \lambda_t \tag{C-12}$$

If we have the other case ($EQ_t < EQ_*$) then:

$$| EQ_t - EQ_* | = EQ_* - EQ_t = \sum_{a^1} ... \sum_{a^{\overline{m}}} \prod_{j=1}^{\overline{m}} \underline{p}^*_j(a^j)(Q_*^i(a^1,...,a^{\overline{m}}) - Q_t^i(a^1,...,a^{\overline{m}}))$$

$$\leq \sum_{a^1} ... \sum_{a^{\overline{m}}} \prod_{j=1}^{\overline{m}} \underline{p}^*_j(a^j) \times \max_{(a^1,...,a^{\overline{m}})} (Q_*^i(a^1,...,a^{\overline{m}}) - Q_t^i(a^1,...,a^{\overline{m}}))$$

$$\leq \max_{(a^1,...,a^{\overline{m}})} | (Q_*^i(a^1,...,a^{\overline{m}}) - Q_t^i(a^1,...,a^{\overline{m}})) | = \max_{(a^1,...,a^{\overline{m}})} | Q_t^i(a^1,...,a^{\overline{m}}) - Q_*^i(a^1,...,a^{\overline{m}}) |$$

$$= \| Q_t^i(a^1,...,a^{\overline{m}}) - Q_*^i(a^1,...,a^{\overline{m}}) \| \tag{C-13}$$

So Eq. (C-12) is valid for the case $EQ_t < EQ_*$ as well. This result in (based on corollary 1, Eq. (B-9) and Eq. (C-12)) that updating of Q-values using Eq. (8) will cause convergence to optimal Q-values of Eq. (7) w.p.1 and proof is completed. □

Appendix D

Proof of Theorem 1: In lemma 3 we proved that updating of Q-values using Eq. (8) causes convergence to optimal Q-values of Eq. (7) w.p.1. So when $t \to \infty$, Eq. (8) will approach to (D-1)

$$Q_*^i(a^1,...,a^{\overline{m}}) = (1-\alpha)\times Q_*^i(a^1,...,a^{\overline{m}}) + \alpha\times[Er_i^t(a^1,...,a^{\overline{m}}) + \beta.\sum_{a^1}\sum_{a^2}...\sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{p}^*, Q_*^i)(a^i)\prod_{\substack{j=1\\ j\neq i}}^{\overline{m}} \underline{p}^*_j(a^j).Q_*^i(a^1,...,a^{\overline{m}})] \quad \text{(D-1)}$$

It's obvious that

$$Q_*^i(a^1,...,a^{\overline{m}}) = Er_i^t(a^1,...,a^{\overline{m}}) + \beta.\sum_{a^1}\sum_{a^2}...\sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{p}^*, Q_*^i)(a^i)\prod_{\substack{j=1\\ j\neq i}}^{\overline{m}} \underline{p}^*_j(a^j).Q_*^i(a^1,...,a^{\overline{m}}) \quad \text{(D-2)}$$

Comparing Eq. (7) and Eq. (D-2) results in that

$$v^i(\underline{p}^*_1, \underline{p}^*_2,..., \underline{p}_i^{br}(\underline{p}^*, Q_*^i),..., \underline{p}^*_{\overline{m}}) = \sum_{a^1}\sum_{a^2}...\sum_{a^{\overline{m}}} \underline{p}_i^{br}(\underline{p}^*, Q_*^i)(a^i)\prod_{\substack{j=1\\ j\neq i}}^{\overline{m}} \underline{p}^*_j(a^j).Q_*^i(a^1,...,a^{\overline{m}}) \quad \text{(D-3)}$$

In algorithm 2, we specify the best response strategy using RHS of Eq. (D-3) which is equal to $\underline{p}_i^{br}(\underline{p}^*, Q_*^i)$ in $v^i(\underline{p}^*_1, \underline{p}^*_2,..., \underline{p}_i^{br}(\underline{p}^*, Q_*^i),..., \underline{p}^*_{\overline{m}})$. So algorithm 2 specifies a best response strategy to $(\underline{p}^*_1,..., \underline{p}^*_{i-1}, \underline{p}^*_{i+1},..., \underline{p}^*_{\overline{m}_i})$. Based on concepts in chapter 4 of [26], this local rule leads the probability vector of LA to the best response configuration. Since every *LA* in ICLA applies this rule, therefore probability vectors of all of them are best response configuration. So according to best response strategy playing concepts [27], ICLA converges to compatible point. □