Decentralized Learning in Markov Games

Peter Vrancx*, Katja Verbeeck, and Ann Nowé

October 31, 2007

Abstract

Learning Automata (LA) were recently shown to be valuable tools for designing Multi-Agent Reinforcement Learning algorithms. One of the principal contributions of LA theory is that a set of decentralized, independent learning automata is able to control a finite Markov Chain with unknown transition probabilities and rewards. In this paper we extend this result to the framework of Markov Games, a straightforward extension of single-agent Markov Decision Problems to distributed multi-agent decision problems. We put a simple learning automaton for every agent in each state and show that the problem can be viewed from 3 different perspectives; the single superagent view, in which a single agent is represented by the whole set of automata, the multi-agent view, in which each agent is represented by the automata it was associated with in each state and finally the LA-view, i.e. the view in which each automaton itself represents an agent. We show that under the same ergodic assumptions of the original theorem, the multi-agent and LA view can both be approximated by a limiting normal form game, that share the same pure equilibrium points. Moreover, if updating is done in sufficiently small steps, the LA will converge to one of these pure equilibrium points.

1 Introduction

Reinforcement Learning was originally developed for Markov Decision Problems (MDPs) [1]. It allows a single agent to learn a policy that maximizes a possibly delayed reward signal in a stochastic stationary environment. RL guarantees convergence to the optimal strategy as long as the agent can sufficiently explore and the environment in which it is operating has the Markov property[1].

However the MDP model does not allow multiple agents to act in the same environment. A straightforward extension of the MDP model to the multi-agent case is given by the framework of Markov Games [2]. In a Markov Game, actions

*P. Vrancx funded by a Ph.D grant of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT Vlaanderen).

[1]A system has the Markov property if the conditional probability distribution of future states given the present state, is independent of the whole history of past states, meaning that the system is memoryless.

are the result of the joint action selection of all agents and rewards and state transitions depend on these joint actions. When only one state is assumed, the Markov game is actually a repeated normal form game well known in game theory [3]. When only one agent is assumed, the Markov game is again an MDP.

Unfortunately, rewards are sensed for combinations of actions taken by different agents, and therefore agents are actually learning in a product or joint action space. Moreover, due to the individual reward functions of the agents, it generally is impossible to find policies which maximize the rewards for all agents. The latter is possible in the so-called team games or multi-agent MDPs (MMDPs). In this case, the Markov game is purely cooperative and all agents share the same reward function. In MMDPs the agents should learn to find and agree on the same optimal policy. In a general Markov game, an equilibrium point is sought; i.e. a situation in which no agent alone can change its policy to improve its reward when all other agents keep their policy fixed.

In addition, agents in a Markov game face the problem of incomplete information with respect to the action choice. One can assume that the agents get information about their own choice of action as well as that of the others. This is the case in what is called joint action learning, a popular way to address multi-agent learning [4–7]. Joint action learners are able to maintain models of the strategy of others and explicitely take into account the effects of joint actions. In contrast, *independent agents* only know their own action. The latter is often a more realistic assumption since distributed multi-agent applications are typically subject to limitations such as partial or non observability, communication costs, asynchronism and stochasticity.

Different approaches toward independent multi-agent learning exist, however most of them have limited applicability or lack theoretical results. In [8] an algorithm is proposed for learning cooperative MMDPs, but it is only suited for deterministic environments. In [9, 10] new multi-agent exploration mechanisms are developed, however only single-state problems or repeated normal form games are considered.

In this paper we will focus on how learning automata (LA) can tackle the problem of learning Markov Games. Learning automata are independent, adaptive decision making devices that were previously shown to be very useful tools for building new multi-agent reinforcement learning algorithms in general [11]. The main reason for this is that even in multi-automata settings they still exhibit nice theoretical properties. One of the principal contributions of LA theory is that a set of decentralized learning automata is able to control a finite Markov Chain with unknown transition probabilities and rewards. In this paper we extend this result to the framework of Markov Games. For each agent a simple learning automaton is put in each state. This setting is analysed from 3 different perspectives; *the single superagent view*, in which a single agent is represented by the whole set of automata, *the multi-agent view*, in which each agent is represented by the automata it was associated with in each state and finally *the LA-view*, i.e. the view in which each automaton represents an agent. We show that under the same ergodicity assumptions of the original theorem, the multi-agent and LA view can both be approximated by limiting normal form games,

that share the same pure equilibrium points. Combined with previous results on automata games we will be able to prove that a set of independent learning automata agents is able to reach a pure equilibrium point in Markov Games.

This paper is organized as follows: in the next section the definitions of MDPs, MMDPs and Markov games are given. In Section 3 learning automata theory is summarized. In Section 4, we discuss the existing LA algorithm for MDPs and introduce its extension to Markov games. We give three different views on the setting proposed. Next in Section 5, our theoretical results are shown. An illustrating example is given in Section 6. Finally, we discuss some related work and give directions for the future.

2 Definitions

2.1 MDP Definition

The problem of controlling a finite Markov Chain, called a Markov Decision Problem (MDP) for which transition probabilities and rewards are unknown can be stated as follows. Let $S = \{s_1, \ldots, s_N\}$ be the state space of a finite Markov chain $\{x_l\}_{l \geq 0}$ and $A^i = \{a^i_1, \ldots, a^i_{r_i}\}$ the action set available in state s_i. Each combination of starting state s_i, action choice $a^i \in A^i$ and next state s_j has an associated transition probability $T^{ij}(a^i)$ and reward $R^{ij}(a^i)$. The overall goal is to learn a policy α , or a set of actions, $\alpha = (a^1, \ldots, a^N)$ with $a^j \in A^j$ so that the expected average reward $J(\alpha)$ is maximized:

$$J(\alpha) \equiv lim_{l \to \infty} \frac{1}{l} E\left[\sum_{t=0}^{l-1} R^{x(t)x(t+1)}(\alpha)\right] \tag{1}$$

The policies we consider are limited to stationary, nonrandomized policies. Under the assumption that the Markov chain corresponding to each policy α is ergodic, it can be shown that the best strategy in any state is a pure strategy, independent of the time at which the state is occupied [12]. A Markov chain $\{x_l\}_{l \geq 0}$ is said to be ergodic when the distribution of the chain converges to a limiting distribution $\pi(\alpha) = (\pi_1(\alpha), \ldots, \pi_N(\alpha))$ with $\forall i, \pi_i(\alpha) > 0$ as $l \to \infty$. Thus, there are no transient states and the limiting distribution $\pi(\alpha)$ can be used to rewrite Equation 1 as:

$$J(\alpha) = \sum_{i=1}^{N} \pi_i(\alpha) \sum_{j=1}^{N} T^{ij}(\alpha) R^{ij}(\alpha) \tag{2}$$

2.2 Markov Game Definition

An extension of single agent Markov decision problems (MDPs) to the multi-agent case can be defined by Markov Games [2]. In a Markov Game, actions are the joint result of multiple agents choosing an action individually. Note that $A^i_k = \{a^i_{k1}, \ldots, a^i_{ki_r}\}$ is now the action set available in state s_i for agent

k, with $k : 1 \ldots n$, n being the total number of agents present in the system. Transition probabilities $T^{ij}(a^i)$ and rewards $R_k^{ij}(a^i)$ now depend on a starting state s_i, ending state s_j and a joint action from state s_i, i.e. $a^i = (a_1^i, \ldots a_n^i)$ with $a_k^i \in A_k^i$. The reward function $R_k^{ij}(a)$ is now individual to each agent k. Different agents can receive different rewards for the same state transition. Since each agent k has its own individual reward function, defining a solution concept is non-trivial. Again we will only treat non-randomized, stationary policies and we will assume that the Markov Game is ergodic in the sense that there are no transient states present and a limiting distribution on the joint policies exists. We can now use Equation 2 to define the expected reward for agent k, for a given joint policy α.

$$J_k(\alpha) = \sum_{i=1}^{N} \pi_i(\alpha) \sum_{j=1}^{N} T_k^{ij}(\alpha) R_k^{ij}(\alpha) \tag{3}$$

Due to the existence of different reward functions, it is in general impossible to find an optimal policy for all agents. Instead, equilibrium points are sought. In an equilibrium, no agent can improve its reward by changing its policy if all other agents keep their policy fixed.

In general a Markov game using the average reward criterion in Equation 1 does not necessarily have an equilibrium point in stationary policies. Examples of Markov games which only have equilibria in history dependent strategies can be found [13]. It can be shown however that Markov games, satisfying the ergodicity requirement stated above have at least one equilibrium in stationary policies [14]. This equilibrium does not need to consist of pure policies, however.

2.3 MMDP Definition

In a special case of the general Markov game framework, the so-called team games or multi-agent MDPs (MMDPs) [15] optimal policies still exist. In this case, the Markov game is purely cooperative and all agents share the same reward function. This specialization allows us to define the optimal policy as the joint agent policy, which maximizes the payoff of all agents. An MMDP can therefore also be seen as an extension of the single agent MDP to the multi-agent case.

Because the agents share the same transition and reward function, one can think of the collection of agents being a single super agent with joint actions at its disposal and whose goal is to learn the optimal policy for the joint MDP. Since the agents' individual action choices may be jointly suboptimal, the added problem in MMDPs is for the agents to learn to coordinate their actions so that joint optimality is achieved. The value of a joint policy $\alpha = (a^1, \ldots, a^N)$ with a^i the joint action of state s_i in $A_1^i \times \ldots \times A_n^i$, can still be defined by Equation 1.

Under the same assumption considered above, i.e. the Markov chain corresponding to each joint policy α is ergodic, it is sufficient to only consider joint policies in which the agents choose pure strategies. This can easily be seen, as the problem can be reduced to an ergodic MDP of Section 2.1 by considering the

single super agent view. Moreover, under this assumption the expected average reward of a joint policy α can also be expressed by Equation 2.

In this paper, we will address the single super agent perspective, the multi-agent perspective and also the local state perspective, which will be called the learning automaton perspective because for every agent we will put a learning automaton in every state. First we will briefly describe learning automata in the next subsection.

3 Learning Automata

A learning automaton formalizes a general stochastic system in terms of states, actions, action probabilities and environment responses (see [16]). In a variable structure stochastic automaton action probabilities are updated at every stage using a reinforcement scheme. It is defined by a quadruple $\{A, \beta, p, T\}$ for which A is the action or output set $\{a_1, a_2, \ldots a_r\}$ of the automaton, β is a random variable in the interval $[0, 1]$, p is a vector of the automaton's action probabilities and T denotes an update scheme. The output a of the automaton is actually the input to the environment. Whereas the input β of the automaton is the output of the environment, which is modeled through penalty probabilities c_i with $c_i = P[\beta \mid a_i], i : 1 \ldots r$.

Important examples of update schemes are linear reward-penalty, linear reward-inaction and linear reward-ϵ-penalty. The philosophy of these schemes is essentially to increase the probability of an action when it results in a success and to decrease it when the response is a failure. The general algorithm is given by:

$$\begin{aligned} p_m(t+1) &= p_m(t) + \lambda_1(1-\beta(t))(1-p_m(t)) \\ &\quad -\lambda_2\beta(t)p_m(t) \\ &\quad \text{if } a_m \text{ is the action taken at time } t \end{aligned} \tag{4}$$

$$\begin{aligned} p_j(t+1) &= p_j(t) - \lambda_1(1-\beta(t))p_j(t) \\ &\quad +\lambda_2\beta(t)[(r-1)^{-1} - p_j(t)] \\ &\quad \text{if } a_j \neq a_m \end{aligned} \tag{5}$$

The constants λ_1 en λ_2 are the reward and penalty parameters respectively. When $\lambda_1 = \lambda_2$ the algorithm is referred to as linear reward-penalty (L_{R-P}), when $\lambda_2 = 0$ it is referred to as linear reward-inaction (L_{R-I}) and when λ_2 is small compared to λ_1 it is called linear reward-ϵ-penalty ($L_{R-\epsilon P}$).

If the penalty probabilities c_i of the environment are constant, the probability $p(t+1)$ is completely determined by $p(t)$ and hence $p(t)_{t>0}$ is a discrete-time homogeneous Markov process. Convergence results for the different schemes are obtained under the assumptions of constant penalty probabilities, see [16].

3.1 Learning Automata Games

Automata games were introduced to see if automata could be interconnected in useful ways so as to exhibit group behavior that is attractive for either modeling or controlling complex systems.

A play $a(t) = (a^1(t) \dots a^n(t))$ of n automata is a set of strategies chosen by the automata at stage t, such that $a^j(t)$ is an element of the action set of the jth automaton. Correspondingly the outcome is now also a vector $\beta(t) = (\beta^1(t) \dots \beta^n(t))$. At every time-step all automata update their probability distributions based on the responses of the environment. Each automaton participating in the game operates without information concerning the number of other participants, their strategies, actions or payoffs.

In zero-sum games[2] the L_{R-I} scheme converges to the equilibrium point if it exist in pure strategies, while the $L_{R-\epsilon P}$ scheme can arbitrarily close approach a mixed equilibrium [17]. In general non zero-sum games [18,19] it is shown that when the automata use a L_{R-I} scheme and the game is such that a unique pure equilibrium point exists, convergence is guaranteed. In cases were the game matrix has more than one pure equilibrium, which equilibrium is found depends on the initial conditions. Summarized we have the following:

Theorem 1 [19] *When the automata game is repeatedly played with each player making use of the L_{R-I} scheme with a sufficiently small step size, then local convergence is established towards pure Nash equilibria.*

4 Learning in finite MDPs

It is well known that Reinforcement Learning techniques [1] are able to solve single-agent Markovian decision problems with delayed rewards. We focus here on how a set of interconnected LA is able to control an MDP [12,16]. Later on, we show how to extend this result to the multi-agent case in a very natural way.

The problem of controlling a Markov chain can be formulated as a network of automata in which control passes from one automaton to another. In this set-up every action state[3] in the Markov chain has a LA that tries to learn the optimal action probabilities in that state, using the learning scheme given in Equations (4,5). Only one LA is active at each time step and transition to the next state triggers the LA from that state to become active and take some action. LA LA^i active in state s_i is not informed of the one-step reward $R^{ij}(a^i)$ resulting from choosing action $a^i \in A^i$ in s_i and leading to state s_j. Only when state s_i is visited again, LA^i receives two pieces of data: the cumulative reward generated by the process up to the current time step and the current global time. From these, LA^i computes the incremental reward generated since this

[2] In a zero-sum game the winnings of a player are exactly balanced by the losses of the other(s), so that the sum of all rewards equal zero.

[3] A state is called an action state, when more than one action is present.

last visit and the corresponding elapsed global time. The environment response or the input to LA^i is then taken to be:

$$\beta^i(t_i+1) \quad = \quad \frac{\rho^i(t_i+1)}{\eta^i(t_i+1)} \tag{6}$$

where $\rho^i(t_i+1)$ is the cumulative total reward generated for action a^i in state s_i and $\eta^i(t_i+1)$ the cumulative total time elapsed. The authors in [12] denote the updating scheme as given in Equations (4,5) with the environment response as in (6) as learning scheme T1. The following results were proved:

Lemma 1 (Wheeler and Narendra, 1986) *The Markov chain control problem can be asymptotically approximated by an identical payoff game of N automata.*

Theorem 2 (Wheeler and Narendra, 1986) *Let for each action state s_i of an N state Markov chain, an automaton LA^i using learning scheme T1 and having r_i actions be associated with. Assume that the Markov Chain, corresponding to each policy α is ergodic. Then the decentralized adaptation of the LA is globally ϵ-optimal[4] with respect to the long-term expected reward per time step, i.e. $J(\alpha)$.*

Outline of the proof:
According to Lemma 1 the Markov Chain control problem under the assumptions above can be asymptotically approximated by an identicall payoff game of N automata. This game is shown to have a unique equilibrium. For the corresponding automata game with every automaton using an L_{R-I} updating scheme the above result is proved see Theorem 1. As long as the ratio of updating frequencies of any two controlles does not tend to zero, the result also holds for the asynchronous T1 updating scheme.

The principal result derived is that, without prior knowledge of transition probabilities or rewards, the network of independent decentralized LA controllers is able to converge to the set of actions that maximizes the long-term expected reward [16]. Moreover instead of one agent visiting all states and keeping a single Q-table for all the states in the system, as is the case in traditional RL algorithms such as Q-learning; in the LA learning model there are some non-mobile LA agents who do not move around the state space but stay in their own state waiting to get active and learn to take actions only in their own state. The intelligence of one mobile agent is now distributed over the states of the Markov chain, more precisely over the non-mobile LA agents in those states.

[4] A LA is said to behave ϵ-optimal when it approaches the optimal action probability vector arbitrarily close.

5 Learning in finite Markov Games

In a Markov Game the action chosen at any state is the joint result of individual action components performed by the agents present in the system. In this section we extend the LA-model of Section 4 to the framework of Markov Games just by putting a simple learning automaton for every agent in each state.

We start by considering the special case of MMDPs were all agents share the same payoff function. Three different perspectives emerge; the single superagent view, the agent view and the LA-game view. When appplying the technique to general Markov games we will lose the single agent perspective. We show that under the same ergodic assumptions of the original LA theorem, the agent and LA game view share the same pure equilibrium points. Moreover, if updating is done in sufficiently small steps, the LA will converge to an equilibrium point in this game.

5.1 The Model

Instead of putting a single learning automaton in each action state of the system, we propose to put an automaton LA_k^i in each action state s_i with $i : 1 \dots N$ and for each agent k, $k : 1 \dots n$. At each time step only the automata of one state are active; a joint action triggers the LA from that state to become active and take some joint action.

As before, LA LA_k^i active for agent k in state s_i is not informed of the one-step reward $R^{ij}(a^i)$ resulting from choosing joint action $a^i = (a_1^i, \dots, a_n^i)$ with $a_k^i \in A_k^i$ in s_i and leading to state s_j. When state s_i is visited again, all automata LA_k^i, $k : 1 \dots n$, receive two pieces of data: the cumulative reward generated by the process up to the current time step and the current global time. From these, all LA_k^i compute the incremental reward generated since this last visit and the corresponding elapsed global time. The environment response or the input to LA_k^i is exactly the same as in Equation 6.

5.2 MMDPs

Since the utility of a state in an MMDP is the same for all agents, the problem can be stated as an MDP in which the actions in each state are the joint actions of the agents. This means that a pure optimal policy in which all agents maximize their payoff can still be found, using the LA-model of section 4. The use of this model is possible only if we assume that the agents are of one mind and select a joint action together. However this assumption is far from realistic. In general, agents are independent and select their actions individually. This complicates learning considerably since individual agent actions may be jointly suboptimal. Therefore an important issue in learning MMDPs, is that of coordination.

We will explain the different views one can have on the model with the following example given in Fig. 1. The MMDP has 2 agents and 4 states, with only s_0 and s_1 being action states. In both states 4 joint actions are

present:$(0,0),(0,1),(1,0)$ and $(1,1)$. All transitions, except those leaving states s_2 and s_3 are deterministic, while the tranisitions leaving state s_2 or s_3 have uniform probability of going to one the other states or itself. Rewards are only given for the transitions (s_1, s_2) and (s_1, s_3).

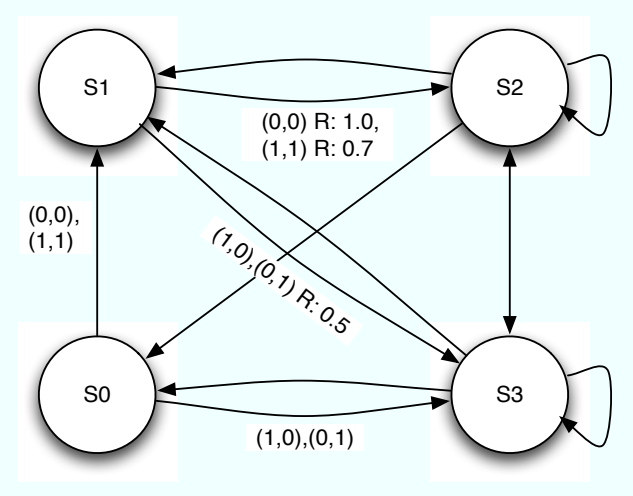


Figure 1: An example MMDP with 2 action states s_0 and s_1; each with 2 actions: 0 and 1. Joint actions and nonzero rewards (R) are shown. Transitions are deterministic; except in the non-action states s_2 and s_3 where the process goes to any state with equal probability (1/4).

5.2.1 Single Super Agent View

When we assume that the agents are of one mind, we can think of a single super agent being active. The joint actions of the MMDP of Example 1, are the actions of the super agent. In this case, the problem can be solved by the model of Section 4, i.e. put one learning automaton in each action state. According to Lemma 1 the control problem of Example 1 is then approximated by a 2 player identical payoff game with 4 actions, i.e. 2 player because there are 2 action states each having 4 actions. The rewards in the game are given by the long-term expected reward per time step, so in total 16 pure policies are possible. This game is given in Fig. 2. One can see that only optimal equilibria are present.

As proved in Theorem 2, the interconnected learning model of Section 4 will find a global optimal policy of the problem, i.e. the super agent chooses joint action $(0,0)$ in both s_0 and s_1, or joint action $(1,1)$ in s_0 and $(0,0)$ in s_1.

		$s1$			
joint	*actions*	$(0,0)$	$(0,1)$	$(1,0)$	$(1,1)$
s0	$(0,0)$	**0.2857**	0.1667	0.1667	0.2
	$(0,1)$	0.1429	0.0833	0.0833	0.1167
	$(1,0)$	0.1429	0.0833	0.0833	0.1167
	$(1,1)$	**0.2857**	0.1667	0.1667	0.2

Figure 2: An identical payoff game with 2 players and 4 actions that approximates the single agent view of the MMDP of Fig. 1.

5.2.2 The Multi-Agent View

In the multi-agent view, we no longer assume that agents jointly select their action in each state. This means that the underlying game played now depends on the agents' individual policies. Again this game is a 2-player identical payoff game with 4 actions. But here we have a 2-player game, because we have 2 agents, and we have 4 actions because each agent has 4 possible policies it can take, i.e $(0,0), (0,1), (1,0)$ and $(1,1)$. Note that here (a_1, a_2) denotes the policy instead of a joint action, i.e. the agent takes action a_1 in state s_0 and action a_2 in state s_1. In Fig. 3 the game matrix for the MMDP of Fig. 1 is given. Surprisingly, in this game 4 pure equilibria are present of which 2 are optimal and 2 are sub-optimal. While, as is shown in Theorem 2, in the super agent view only optimal equilibria can appear, this is clearly not the case in the multi-agent view.

		agent 2			
	policies	$(0,0)$	$(0,1)$	$(1,0)$	$(1,1)$
agent 1	$(0,0)$	**0.2857**	0.1667	0.1429	0.0833
	$(0,1)$	0.1667	**0.2**	0.0833	0.1167
	$(1,0)$	0.1429	0.0833	**0.2857**	0.1667
	$(1,1)$	0.0833	0.1167	0.1667	**0.2**

Figure 3: An identical payoff game with 4 actions that approximates the multi-agent view of the MMDP of Fig. 1. Equilibria are indicated in bold.

5.2.3 The Learning Automata View

The last view we have on the control problem is that of the lowest level. In this view we consider the game between all the learning automata that are present in the different action states, using the model of Section 5.1. For the MMDP of Fig. 1 this would give a 4 automata game with 2 actions for each automaton: 0 or 1. The complete game is shown in Fig. 4.

In the next section we will show that this view shares the same pure equilibria with the multi-agent view and that the interconnected model of Section 5.1 is able to find an equilibrium policy.

$(LA_0^0, LA_1^0, LA_0^1, LA_1^1)$	$J(\alpha)$	$(LA_0^0, LA_1^0, LA_0^1, LA_1^1)$	$J(\alpha)$
$(0,0,0,0)$	**0.2857**	$(1,0,0,0)$	0.1667
$(0,0,0,1)$	0.1429	$(1,0,0,1)$	0.0833
$(0,0,1,0)$	0.1429	$(1,0,1,0)$	0.0833
$(0,0,1,1)$	**0.2**	$(1,0,1,1)$	0.1167
$(0,1,0,0)$	0.1667	$(1,1,0,0)$	**0.2857**
$(0,1,0,1)$	0.0833	$(1,1,0,1)$	0.1429
$(0,1,1,0)$	0.0833	$(1,1,1,0)$	0.1429
$(0,1,1,1)$	0.1167	$(1,1,1,1)$	**0.2**

Figure 4: An identical payoff game between 4 players each with 2 actions, that approximates the LA view of the MMDP of Fig. 1. Equilibria are indicated in bold.

5.3 Markov Games

In this section we extend our approach to general Markov games. In these games each agent k has it own reward function R_k. The expected individual reward J_k of agent k can now be calculated by using Equation 3.

As an example we extend the MMDP of Section 5.2 so that agent 1 and agent 2 get different rewards for the state transitions. The resulting Markov game is shown in Fig. 5.

Since there is no longer a single reward function, it is not guaranteed that a single joint policy which maximizes the payoff for all agents exists. Therefore we cannot treat the problem as a single agent MDP anymore and the single super agent view of the previous section becomes impossible.

It is still possible, however, to approximate the Markov Game using the agent and learning automata views. The main difference with the MMDP section is that the resulting approximating games are no longer common interest, but rather conflicting interest. The resulting limiting game for the multi-agent view is shown in Fig. 6

In the resulting agent game, each agent can now get a different reward for a joint policy. In the automata game, all automata belonging to the same agent k get the same payoff J_k, while those belonging to different agents can get different rewards.

5.4 Theoretical Results

Note first that a policy for the LA-view is automatically also a policy for the agent view. The same notation can be used, only the interpretation is dif-

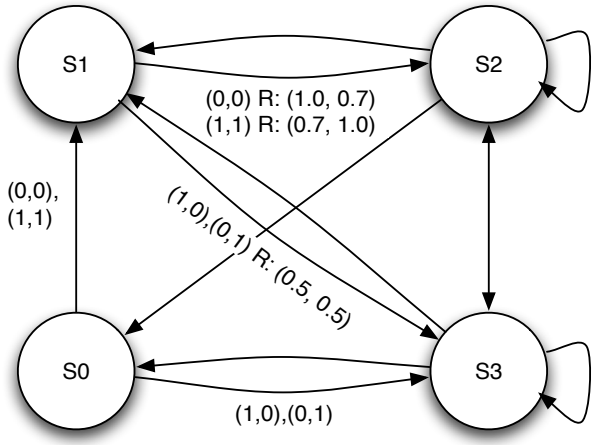


Figure 5: *An example Markov game with the same structure as Example 1, but with different individual rewards for both agents.*

ferent. A policy in the multi-agent view gives for each action state a set of individual actions, one for each agent: $\alpha_{multi-agent} = (a^1, \ldots, a^N)$ with $a^i = (a^i_1, \ldots, a^i_n)$ a joint action of state s_i in $A^i_1 \times \ldots \times A^i_n$. In the LA-view a policy is composed of an individual action for each LA in the system. As such $\alpha_{LA} = (a^0_1, \ldots, a^0_n, a^1_1, \ldots, a^1_n, \ldots, a^N_1, \ldots, a^N_n)$ which is just the expansion of $\alpha_{multi-agent}$.

We can now state the following:

Theorem 3 *Given a Markov Game with N action states. Assume that the Multi-agent Markov Chain, corresponding to each joint policy α is ergodic. When α is a pure equilibrium policy in the multi-agent view, α is also an pure equilibrium policy for the LA-view and vice versa.*

proof

First we note that an equilibrium in the agent game must always be an equilibrium in the LA game. This is easy to see, because a learning automaton switching its action corresponds to a single agent changing its policy in a single state. So in any situation were no agent alone can improve its expected reward by changing policies, it must also be impossible for a single learning automaton to improve its payoff.

Now suppose that α is an equilibrium for the LA game view but is not an equilibrium point for the agent game view. This means no single automaton on its own can improve its payoff by switching to another action. To get out of the equilibrium at least 2 LA should change their action choice. However

since we assumed that α is not an equilibrium point in the agent view, it is possible to find a single agent k that can improve its payoff simply by changing its strategy $\alpha_k = (a_k^1, \ldots, a_k^N)$. Because of the above, it has to do this by changing the actions of 2 or more different LA. For simplicity assume that agent k changes its policy in state i and state j with $i \neq j$ and call this new policy $\beta_k = (b_k^1, \ldots, b_k^N)$.[5] Then by construction we have that: $a_k^i \neq b_k^i$, $a_k^j \neq b_k^j$, $\forall l : 1 \ldots N, l \neq \ i, j : a_k^l = b_k^l$, $\beta_m = \alpha_m$ for all agents $m : 1 \ldots n, m \neq k$ and $J_k(\beta) > J_k(\alpha)$.

		agent 2			
	policies	$(0,0)$	$(0,1)$	$(1,0)$	$(1,1)$
agent 1	$(0,0)$	$(\mathbf{0.2857}, \mathbf{0.2})$	$(0.1667, 0.1667)$	$(0.1429, 0.1167)$	$(0.0833, 0.0833)$
	$(0,1)$	$(0.1667, 0.1667)$	$(\mathbf{0.2}, \mathbf{0.2857})$	$(0.0833, 0.0833)$	$(0.1167, 0.1429)$
	$(1,0)$	$(0.1429, 0.1167)$	$(0.0833, 0.0833)$	$(\mathbf{0.2857}, \mathbf{0.2})$	$(0.1667, 0.1667)$
	$(1,1)$	$(0.0833, 0.0833)$	$(0.1167, 0.1429)$	$(0.1667, 0.1667)$	$(\mathbf{0.2}, \mathbf{0.2857})$

Figure 6: A conflicting interest game with 4 actions that approximates the multi-agent view of the Markov game of Fig. 5. Equilibria are indicated in bold.

Let a_{-k}^l denote the joint action of policy α in state l without the action of agent k, and $(a_{-k}^l, a*^l)$ with $a*^l \in A_k^l$ the joint action in state l with all agents following policy α, except for agent k that takes action $a*^l$.

Analogous, define α_{-k} to be policy α for all agents except agent k and $(\alpha_{-k}, a*)$ with $a* = (a*^1, \ldots, a*^N) \in \mathbb{A}$ the policy for which all agents follow policy α and agent k follows policy $a*$.

We can now consider the following N-state Markov problem: $(N, \mathbb{A}, T, R_k)$ with actions set $\mathbb{A}$ being the action set of agent k : $\mathbb{A} = A_k^1 \times \ldots \times A_k^N$. Transitions and rewards are defined as follows: $T^{uv}(a*^u) = T^{uv}(a_{-k}^u, a*^u)$, $R^{uv}(a*^u) = R_k^{uv}(a_{-k}^u, a*^u)$ and $J(a*) = J_k(\alpha_{-k}, a*)$. Note that since we have assumed that the Markov chain for each joint policy in the full Markov game is ergodic, it follows that for this Markov problem the chain corresponding to each policy $a*$ will also be ergodic. Fom Theorem 2, we then know that this Markov problem can be approximated by a limiting game Γ which has only optimal equilibria.[6] In [12] it is shown that for each suboptimal policy in the game Γ, we can create a better policy which differs from the original policy in only one state of the MDP.

Consider now α_k, this policy cannot be an equilibrium point for Γ because :

$$J(\alpha_k) = J_k(\alpha_{-k}, \alpha_k) = J_k(\alpha) < J_k(\beta) = J_k(\alpha_{-k}, \beta_k) = J(\beta_k)$$

As α_k is not an equilibrium point, a better policy $a*$ can be found by changing

[5] The case where 3 or more automata should switch their action is analogous.

[6] Note that this does not mean that the game has a unique equilibrium. As can be seen in Example 1 of section 5.2, multiple equilibrium plays can exist, but they must achieve the same, optimal payoff.

α_k in only in one state. But since:

$$J_k(\alpha) = J_k(\alpha_k) < J_k(a*) = J_k(\alpha_{-k}, a*)$$

We have a policy $(\alpha_{-k}, a*)$ in the original agent game which differs from the original policy α only in the action of a single automaton LA_k^*, and receives a strictly higher payoff J_k. This means that a single automaton of agent k can change its action to receive a higher payoff and thus α cannot be an equilibrium in the full automata game. This contradicts our original assumption that α is an equilibrium for the automata game, but is not an equilibrium for the agent game view. Therefore we can conclude that both views share the same pure equilibria.

Corollary 1 *The Learning Automata model proposed in Section 5.1 is able to find an equilibrium in pure strategies in a Markov game that satisfies the above ergodic assumptions.*

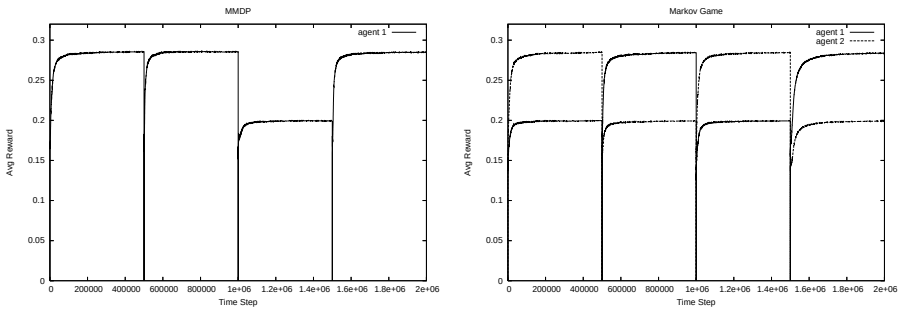
proof

According to Theorem 1, the automata will find a pure equilibrium point of the corresponding automata game. But because of Theorem 3 this point will also be an equilibrium point of the limiting agent game.

6 Experiments

We now demonstrate the behaviour of the LA learning model on the sample problems described above. Fig. 7a and Fig. 7b show the results of the LA algorithm from Section 5.1 on the sample MMDP and Markov game of Fig. 1 and Fig. 5, respectively. Since we are interested in the long term convergence we show a typical run, rather than an average over multiple runs. To demonstrate convergence to the different equilibria, we use a single very long run (2 million time steps) in which we give the automata 500000 steps to converge and then restart them. After every restart the automata are initialized with random probabilities to allow them to converge to different equilibria.

In Fig. 7a we see the common average reward for both agents in the MMDP of Fig. 1. The agents converge once to the suboptimal and three times to the optimal equilibrium point. After 500000 the time steps the average reward approaches the predicted values of 0.2 and 0.2857 very closely. In Fig. 7b we show the average reward for both agents on the Markov Game of Fig. 5, since each has its own reward function. Again the average rewards can be seen to closely approximate the predicted values, now with different payoffs for each agent.



(a) MMDP (b) Markov Game

Figure 7: Typical run of the LA learning model of Section 5.1 (a) Results on the MMDP of Fig. 1. (b) Results for the Markov game of Fig. 5. Both experiments used automata with the L_{R-I} update scheme and a learning rate of $\lambda_1 = 0.05$.

7 Related Work

Convergence results for decentralized learning in Markov Games are often lacking or are only applicable in very restrictive cases. In [8] an algorithm is proposed for learning cooperative MMDPs, but it is only suited for deterministic environments. In [9, 10] new multi-agent exploration mechanisms are developed, however only single-state problems or repeated normal form games are considered. In [7] an alternative view on Markov Games is taken, i.e. the game can be seen as a sequence of local normal form games. The algorithm the authors propose is Nash-Q-learning. Nash Q-values are Q-functions over joint actions. These are updated based on the Nash equilibrium behavior over the current Nash Q-values. The idea is to let these Q-values learn to reach the Nash Q-values of the equilibrium of the limiting game through repeated play. However only in very restrictive cases, this actually happens. Analytical results demand that for convergence to happen each intermediate normal form game has a unique equilibrium (or saddle point). Besides, the approach assumes that the agents have full knowledge of the system: the agents know their current joint state, the joint action played and the reinforcement signals all agents receive. In [20] team Markov Games or MMDPs are also approximated as a sequence of intermediate games. In team Markov Games, all agents get the same reward function and the agents should learn to select the same optimal equilibrium strategy. The authors present optimal adaptive learning and prove convergence to a Nash equilibrium of the limiting game, however agents know the joint actions played, they all receive the same reward and thus are able to build the game structure. In [21] the agents are not assumed to have a full view of the world. All agents contribute to a collective global reward function, but since domain knowledge is missing, independent agents use filtering methods in order to try to recover the underlying true reward signal from the noisy one that is observed. This approach seems to work well in the example domains shown.

Although we can show convergence to a pure equilibrium point of the limiting game with the learning model proposed here, we are still left with a selection problem. Indeed, often more than one equilibrium point is present, and the question left is then, which solution should the agents learn and how can they do this. In case of team Markov Games or MMDPs, an equilibrium which is optimal for all agents can easily be recognized. In [22] we show that the parametrized learning automata algorithm proposed in [23] allows us to achieve optimal convergence in team Markov games. In general Markov Games, we plan to extended some coordinated form of exploration such as ESRL in [9] to learn a front of Pareto optimal solutions[7], or a fair periodical policy, that switches between different equilibrium points [9].

Finally we also should consider problems with partial observability. For now we assume that each agents has knowledge of the current joint state. In spatial coordination problems however, it is more realistic to assume that the agent only knows its own location not and that of the others. As such the agent is not able to distinguish the true underlying joint state of the Markov Game. In [24, 25] we empirically and theoretically show that our result also holds for a slightly different version of the automa model in which agents do not have full view of the current state.

8 Conclusion

In this paper we study the problem of learning Markov Games with independent agents that only have knowledge of their own payoff, reward and the current state. We propose a model based on learning automata and analyze the setting from different perspectives: the superagent perspective (in case agents share the same reward function), an agent perspective and an automaton's perspective. We show that under a common ergodic assumption the latter perspectives can both be approximated by limiting normal form games, that share the same pure equilibrium points. Moreover, if updating is done in sufficiently small steps, the LA-model proposed will converge to one of these equilibrium points.

References

[1] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[2] L. Shapley, "Stochastic Games," *Proceedings of the National Academy of Sciences*, vol. 39, no. 10, pp. 1095–1100, 1953.

[3] J. Osborne and A. Rubinstein, *A Course in Game Theory*. Cambridge, MA: MIT Press, 1994.

[7] A solution is said to be Pareto Optimal if there is no other solution possible for which all players simultaneously do better and one player is doing strictly better.

[4] M. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Proceedings of the 11th International Conference on Machine Learning*, 1994, pp. 322 – 328.

[5] C. Claus and C. Boutilier, "The dynamics of reinforcement learning in cooperative multiagent systems," in *Proceedings of the 15th National Conference on Artificial Intelligence*, 1998, pp. 746 – 752.

[6] G. Chalkiadakis and C. Boutilier, "Coordination in multiagent reinforcement learning: A bayesian approach," in *Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems*, Melbourne, Australia, 2003, pp. 709 – 716.

[7] J. Hu and M. Wellman, "Nash q-learning for general-sum stochastic games," *Journal of Machine Learning Research*, vol. 4, pp. 1039 – 1069, 2003.

[8] M. Lauer and M. Riedmiller, "An algorithm for distributed reinforcement learning in cooperative multi-agent systems." in *Proceedings of the 17th International Conference on Machine Learning*, 2000, pp. 535 – 542.

[9] K. Verbeeck, A. Nowé, J. Parent, and K. Tuyls, "Exploring selfish reinforcement learning in repeated games with stochastic rewards," *Journal of Autonomous Agents and Multi-Agent Systems*, vol. 14, pp. 239–269, 2007.

[10] S. Kapetanakis, D. Kudenko, and M. Strens, "Learning to coordinate using commitment sequences in cooperative multi-agent systems." in *Proceedings of the 3rd Symposium on Adaptive Agents and Multi-agent Systems, (AISB03) Society for the study of Artificial Intelligence and Simulation of Behaviour*, 2003.

[11] A. Nowé, K. Verbeeck, and M. Peeters, "Learning automata as a basis for multi-agent reinforcement learning," *Lecture Notes in Computer Science*, vol. 3898, pp. 71–85, 2006.

[12] R. Wheeler and K. Narendra, "Decentralized learning in finite markov chains," *IEEE Transactions on Automatic Control*, vol. AC-31, pp. 519 – 526, 1986.

[13] D. Gillette, "Stochastic Games with Zero Stop Probabilities," *Ann. Math. Stud*, vol. 39, pp. 178–187, 1957.

[14] M. Sobel, "Noncooperative Stochastic Games," *The Annals of Mathematical Statistics*, vol. 42, no. 6, pp. 1930–1935, 1971.

[15] C. Boutilier, "Planning, learning and coordination in multiagent decision processes," in *Proceedings of the 6th Conference on Theoretical Aspects of Rationality and Knowledge*, Renesse, Holland, 1996, pp. 195 – 210.

[16] K. Narendra and M. Thathachar, *Learning Automata: An Introduction*. Prentice-Hall International, Inc, 1989.

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

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

[19] P. Sastry, V. Phansalkar, and M. Thathachar, "Decentralized learning of nash equilibria in multi-person stochastic games with incomplete information," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 24(5), pp. 769 – 777, 1994.

[20] X. Wang and T. Sandholm, "Reinforcement learning to play an optimal nash equilibrium in team markov games," in *Advances in Neural Information Processing Systems 15 (NIPS-2002.)*, 2002.

[21] Y.-H. Chang, T. Ho, and L. P. Kaelbling, "All learning is local: Multi-agent learning in global reward games," in *Advances in Neural Information Processing Systems 16*, S. Thrun, L. Saul, and B. Schölkopf, Eds. Cambridge, MA: MIT Press, 2004.

[22] P. Vrancx, K. Verbeeck, and A. Nowé, "Optimal convergence in multi-agent mdps." *Lecture Notes in Artificial Intelligence*, vol. 4694, pp. 107–114, 2007.

[23] M. Thathachar and P. Sastry, *Networks of Learning Automata: Techniques for Online Stochastic Optimization*. Kluwer Academic Publishers, 2004.

[24] P. Vrancx, K. Verbeeck, and A. Nowé, "Limiting games of multi-agent multi-state problems," in *Proceedings Workshop on Adaptive and Learning Agents 2007*, K. Turner, S. Sen, and L. Panait, Eds., Hawaii, USA, 2007.

[25] K. Verbeeck, P. Vrancx, and A. Nowé, "Networks of learning automata and limiting games." in *Proceedings Adaptive Learning Agents and Multi- Agent Systems Workshop 2007*, K. Tuyls and K. Verbeeck, Eds. Maastricht, The Netherlands: MICC/IKAT., 2007.