Switching Dynamics of Multi-Agent Learning

Peter Vrancx *
Vrije Universiteit Brussel, Belgium
pvrancx@vub.ac.be

Karl Tuyls
Eindhoven Technical University, The Netherlands
ktuyls@gmail.com

Ronald Westra
Maastricht University, The Netherlands
westra@micc.unimaas.nl

ABSTRACT

This paper presents the dynamics of multi-agent reinforcement learning in multiple state problems. We extend previous work that formally modelled the relation between reinforcement learning agents and replicator dynamics in stateless multi-agent games. More precisely, in this work we use a combination of replicator dynamics and switching dynamics to model multi-agent learning automata in multi-state games. This is the first time that the dynamics of problems with more than one state is considered with replicator equations. Previously, it was unclear how the replicator dynamics of stateless games had to be extended to account for multiple states. We use our model to visualize the basin of attraction of the learning agents and the boundaries of switching dynamics at which an agent possibly arrives in a new dynamical system. Our model allows to analyze and predict the behavior of the different learning agents in a wide variety of multi-state problems. In our experiments we illustrate this powerful method in two games with two agents and two states.

Categories and Subject Descriptors

I.2.6 [**Learning**]; I.2.11 [**Distributed Artificial Intelligence**]

General Terms

Algorithms, Theory

Keywords

Multi-Agent Learning, Piecewise Replicator Dynamics

1. INTRODUCTION

Learning in Multi-Agent Systems is a complex and cumbersome task. The theoretical foundation of single agent learning implies that as long as the environment an agent experiences is stationary, and the agent can experiment enough, Reinforcement Learning (RL) guarantees convergence to the optimal strategy [6]. This is no longer valid in the multi-agent case because there are now multiple agents learning in the same environment, facing unobservable actions and rewards of other agents and non-stationarity of the environment. All these complicating properties of multi-agent systems make it hard to engineer learning algorithms capable of finding optimal solutions.

Recent debate in the Multi-Agent Learning (MAL) community gave direction to a new research agenda for the field [5]. An important problem of MAL that stands out is the lack of a theoretical framework such as exists for the single agent case. As discussed in previous work we employ an evolutionary game theoretic approach to this problem and the international research agenda [8]. We do this by analyzing the relation between RL and replicator dynamics (RD). More precisely, in [9, 10, 4] the authors derived a formal link between the replicator equations of Evolutionary Game Theory (EGT) and reinforcement learning techniques as Q-learning and Learning Automata. In particular this link showed that in the limit these learning algorithms converge to a certain form of the RD. This allows to establish equilibria using the RD that tell us what states a given learning system will settle into over time and what intermediate states it will go through.

In previous work it was shown that there are a number of benefits to exploiting this link: one, the model predicts desired parameters to achieve Nash equilibriums with high utility, two, the intuitions behind a specific learning algorithm can be theoretically analysed and supported by using the basins of attraction, three, it was shown how the framework could easily be adapted and used to analyze new MAL algorithms, such as for instance lenient Q-learning [4].

The major limitation of using the RD as a model of MAL is that it has only been used in stateless repeated games. In this work, however, we take the next step. We show how the link between EGT and RL can be exploited in multiple state problems for multiple agents using Learning Automata as RL technique, while maintaining the above mentioned benefits. We do this by introducing a combination of switching dynamics and RD, which we call Piecewise Replicator Dynamics, to describe the learning processes over the multiple states. We calculate a new average reward game for each state, which takes into account the rewards that are obtained in the other states. This game can then be studied using the replicator dynamics. The resulting dynamic works under the assumption that agents are only learning in a single state. In reality, however, agents update their action probabilities in all states, and these probabilities will

*funded by a Ph.D grant of the Institute for the Promotion of Innovation through Science and Technology in Flanders.

Cite as: Switching Dynamics of Multi-Agent Learning, Peter Vrancx, Karl Tuyls, Ronald Westra and Ann Nowé, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp. 307-314.

all change in parallel. Changes in action probabilities in one state will cause the average reward games to change in other states. The dynamics observed for an average reward state game are only a snapshot of the true learning dynamics. To account for these changes we model the system as a switching dynamical system. Based on the qualitative changes in the dynamic, we partition the state space in a number of cells which correspond to the different possible attractors in the state games. We assume that each cell has its own fixed replicator dynamic. The entire system can then be modeled by updating the current action probabilities in each state, according to the replicator dynamics of the current cell. When the action probabilities leave this cell the equilibria of the state game change, and we get a new replicator dynamic that drives the system. In this way we follow the trajectory in all states through multiple cells and dynamics until an equilibrium is reached.

The remainder of this paper is structured as follows. In Section 2 we elaborate on the necessary background to understand the further developments of this paper. Section 3 introduces piecewise replicator dynamics for modeling multi-learning in multi-state problems. Section 4 demonstrates our approach with some experiments. Finally, we conclude in section 5.

2. BACKGROUND

In this section we start by introducing the multi-agent learning setting we consider. We continue with a concise summary of Learning Automata and we end this section with a description of the Replicator Dynamics.

2.1 Multi-agent learning setting

In this paper we adopt the formal setting of Markov games (also called stochastic games). Markov games are a straightforward extension of single agent Markov decision problems (MDPs) to the multi-agent case. A Markov game consists of a set of states S and a set of agents N. In each state s_i $A_k^i = \{a_{k1}^i, \ldots, a_{ki_r}^i\}$ is the action set available for agent k, with $k : 1 \ldots |N|$. Actions in the game are the joint result of multiple agents choosing an action independently. The transition function $T(s_i, a^i)$ and reward function $R_k(s_i, a^i)$, determine the probability of moving to another state and the reward for each agent k, depending on the current state s_i and the joint action in this 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 $R_k(s, a)$ can be individual to each agent k, meaning that different agents can receive different rewards for the same state transition.

The objective for each agent in the game is to find a policy which maps each state to a strategy, in order to maximize its reward. In this paper we consider the limit average reward, meaning that agents try to maximize their average reward over time. For a joint policy α consisting of a policy for each agent in the system, the limit average reward to agent k is defined as:

$$J_k(\alpha) \equiv lim_{l\to\infty} \frac{1}{l} E\left[\sum_{t=0}^{l-1} R_k(s(t), a_1(t), \ldots, a_{|N|}(t))\right] \quad (1)$$

In the remainder of this paper we will assume that the Markov chain of system states under every joint policy is ergodic. 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 $l \to \infty$. This assumption allows us to rewrite the average reward to agent k under a given joint policy α as:

$$J_k(\alpha) = \sum_{i=1}^{|S|} \pi_i(\alpha) E_\alpha\left[R_k(s_i, a^i)\right] \quad (2)$$

.

where $E_\alpha\left[R_k(s_i, a^i)\right]$ is the expected reward for agent k in state s_i under joint policy α.

2.2 Learning Automata

Learning Automata are simple reinforcement learners which attempt to learn an optimal action, based on past actions and environmental feedback. Formally, the automaton is described by a tuple $\{A, \beta, p, U\}$ where $A = \{a_1, \ldots, a_r\}$ is the set of possible actions the automaton can perform, p is the probability distribution over these actions, β is a random variable between 0 and 1 representing the evironmental response, and U is a learning scheme used to update p.

A single automaton is connected in a feedback loop with its environment. Actions chosen by the automaton are given as input to the environment and the environmental response to this action serves as input to the automaton. Several automaton update schemes with different properties have been studied. In this paper we use the so called Linear Reward Inaction (L_{R-I}) scheme:

$$p_m(t+1) = p_m(t) + \alpha_r \beta(t)(1 - p_m(t)) \quad (3)$$
$$\text{if } a_m \text{ is the action taken at time } t$$
$$p_j(t+1) = p_j(t) - \alpha_r \beta(t) p_j(t) \quad (4)$$
$$\text{if } a_j \neq a_m$$

Where $\alpha_r \in [0, 1]$ is a constant called the reward parameter or learning rate.

Groups of learning automata can be interconnected by letting them play in a repeated game. In such a game multiple automata interact with the same environment. A play $a(t) = (a_1(t) \ldots a_n(t))$ of n automata is a set of strategies chosen by the automata at stage t. Correspondingly, the response is now a vector $\beta(t) = (\beta_1(t) \ldots \beta_n(t))$, specifying a payoff for each automaton.

At every instance, 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 participants, their strategies, their payoffs or actions.

In general sum games it can be shown that when all automata use the L_{R-I} scheme with a sufficiently small learning rate, and the game is such that a unique pure equilibrium point exists, convergence to this point is guaranteed [7] . In cases where the game matrix has more than one pure equilibrium, which equilibrium is found depends on the initial conditions.

LA can also be used in more complex, multi-state problems. We now explain an automata based algorithm, capable of finding pure equilibria in Markov games [12]. The algorithm is an extension of an LA algorithm for solving MDPs, originally proposed by Wheeler and Narendra [14].

The main idea behind the algorithm is that agent k associates a different learning automaton LA_k^i with each state s_i. The agents then defer the actual action selection in each state to the automaton they have associated with that state.

Each time step each agent k in the system activates LA_k^i that it associates with the current system state s_i. The joint action a^i consisting of the actions of all automata associated with s_i, then triggers a transition to the next system state s_j and an individual reward $R_k^i(s_i, a^i)$ for each agent. The agents then repeat the process in state s_j.

Automata in the system are not informed of the immediate reward that their joint action triggers. Instead each agent keeps track of the cumulative reward it has gathered up to the current time step. When the system returns to a state s_i, that was previously visited, each agent k computes the time Δt^i that has passed since the last visit and the reward Δr_k^i that it has gathered since. Automaton LA_k^i then updates the action it took last time using feedback [1]:

$$\beta_k^i = \frac{\Delta r_k^i}{\Delta t^i} \tag{5}$$

The interactions between all automata in the system described above, can be approximated by a repeated automata game [12] . In this game a play of all automata corresponds to a pure joint policy for the Markov game. The payoff that each automaton LA_k^i receives for such a play is exactly $J_k(\alpha)$, the expected average reward for agent k, as defined in Equation 2. In [12] it is shown that the pure equilibria of this game correspond to equilibria between the pure agent policies. This means that if all automata use the L_{R-I} scheme with a sufficiently small learning rate, the algorithm will converge to a pure equilibrium between agent policies, if such a point exists.

2.3 Replicator Dynamics

Evolutionary Game Theory (EGT) has two central concepts, i.e., evolutionary stable strategies (ESS) and the replicator dynamics (RD). ESS is a refinement of the Nash equilibrium from classical Game Theory. We will not discuss it further in this paper.

The RD are formalized as a system of differential equations. Each replicator represents one (pure) strategy available to a player. EGT assumes that players will gradually adjust their strategy over time in response to repeated observations of their own and others' payoffs. The RD control this learning, specifying the frequency with which different pure strategies should be played depending on the mix of strategies played by the remainder of the population of agents playing the game. Simply stated, an abstraction of an evolutionary process usually combines two basic elements: selection and mutation. Selection favors some varieties over others, while mutation provides variety in the population. RD in its most elementary form highlights the role of selection. More precisely, strategies that gain above-average payoff become more likely to be played (or selected), and the RD models a process in which agents switch to strategies that appear to be more successful. Thus RD are a system of differential equations describing how a population of different strategies evolves through time.

The general form of an RD is the following:

$$\frac{dx_i}{dt} = [(A\mathbf{x})_i - \mathbf{x} \cdot A\mathbf{x}]x_i \tag{6}$$

[1]In this paper we made a small modification to the algorithm. The original authors updated automata using a feedback calculated over all visits to the same state.

In equation (6), x_i represents the density of strategy i in the population, and A is the payoff matrix that describes the different payoff values that each individual replicator receives when interacting with other replicators in the population. The state of the population ($\mathbf{x}$) can be described as a probability vector $\mathbf{x} = (x_1, x_2, ..., x_J)$ which expresses the different densities of all the different types of replicators in the population. Hence $(A\mathbf{x})_i$ is the payoff that replicator i receives in a population with state x and $\mathbf{x} \cdot A\mathbf{x}$ describes the average payoff in the population. The growth rate $\frac{\frac{dx_i}{dt}}{x_i}$ of the population share using strategy i equals the difference between the strategy's current payoff and the average payoff in the population. For further information we refer the reader to [3, 13].

When we consider multiple agents that learn concurrently we need more systems of differential equations. For simplicity, we restrict the discussion to only two such learning agents. As a result, we need two systems of differential equations: one for the row agent (P) and one for the column agent (Q). This setup corresponds to a RD for asymmetric games. If B is the payoff matrix that describes the payoff values received by the second agent, and if $A = B^t$, then equation (6) would emerge again to characterize the dynamics of the second learner.

This translates into the following replicator equations for the two populations:

$$\frac{dp_i}{dt} = [(A\mathbf{q})_i - \mathbf{p} \cdot A\mathbf{q}]p_i \tag{7}$$

$$\frac{dq_i}{dt} = [(B\mathbf{p})_i - \mathbf{q} \cdot B\mathbf{p}]q_i \tag{8}$$

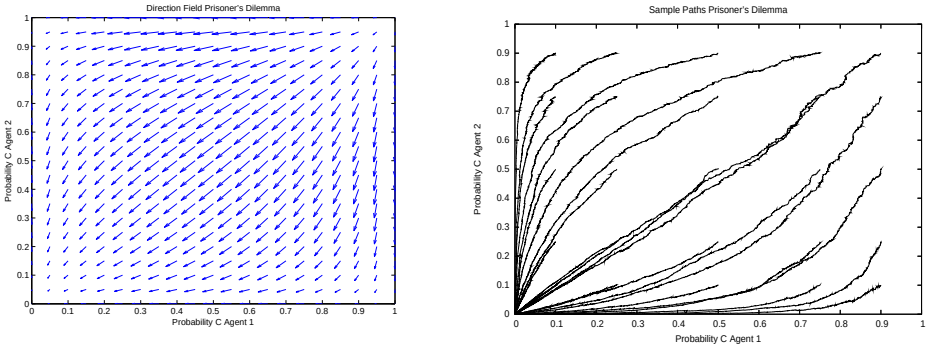
As can be seen in equation (7) and (8), the growth rate of the types in each population is additionally determined by the composition of the other population, in contrast to the single learner case described by equation (6).

As an example for the application of RD in stateless games, we illustrate the dynamics of the well known Prisoner's Dillema (PD) game. Figure 1(a) shows the direction field obtained for this game, Figure 1(b) plots the action probability trajectories generated by LA playing the repeated game.

3. PIECEWISE REPLICATOR DYNAMICS

We first describe the general approach of piecewise linear dynamics for modeling dynamic interactions between agents and their environment, after which we apply this approach to MAL with the replicator dynamics from EGT as piecewise approximation method.

The prevailing approach to modeling dynamical interactions is by representing them as a set of ordinary differential equations (ODEs). In many cases this will represent a statistical average over the entire ensemble of possible configurations, involving e.g. a mean field, steady state, or quasi equilibrium assumption, rather than a fundamental law of Nature [11]. So, these ODEs are statistical approximations that – under certain conditions – predict the average evolution of the system. Let us for the moment forsake the numerous problems with regard to these conditions, and consider the general dynamics of multi-agent switching systems. When we assume a stochastic differential equation as model for the dynamics of the interaction, the relation can



(a) (b)

Figure 1: Dynamics for the stateless Prisoner's Dilemma game (a) Direction field for the replicator equations on this game. (b) Sample paths showing the evolution action probabilities in a repeated automata game. Both automata use the L_{R-I} update with learning rate 0.001.

be expressed as:

$$\frac{dx}{dt} = f(x, u|\theta) + \xi(t) \tag{9}$$

Here $x(t)$, called the state-vector, denotes the N parameters at time t fully describing the agents – possibly involving higher order time derivatives. $u(t)$ denotes the P controlled inputs to the system. $\xi(t)$ denotes a stochastic Gaussian white noise term. This expression involves a parameter vector θ, that contains the coupling constants between agents and inputs. We can consider this system as being represented by the state vector $x(t)$ that wanders through the (at least) N-dimensional space of all possible configurations. In the formalism of dynamic systems theory, eventually x will enter an area of attraction, and become subject to the influence of an attractor. An attractor here can be an uniform convergent attractor, a limit cycle, or a 'strange attractor'. We can understand the entire space as being partitioned into cells, where such attractors – or their antagonists so-called repellers – reign. Thus, the behavior of x can be described by motion through this collection of cells, swiftly moving through cells of repellers, until they enter the basin of attraction of an attractor. Under the effects of external agents via the vector $u(t)$ or by stochastic fluctuations via $\xi(t)$ they can leave this cell, and start wandering again, thereby repeating the process. Now, a vital assumption is that in each cell the behavior is governed by specific (un)stable equilibrium points (possibly outside this cell), and therefore it is possible to make a linear approximation of equation 9 in the cell with index l as:

$$\frac{dx}{dt}(t) = F_l x(t) + G_l u(t) \tag{10}$$

In case of a uniform attractor the largest eigen-value of F_l will be negative, and in case of a uniform repeller the smallest eigen-value will be positive.

We can now formalize the qualitative behavioral dynamics of complex interactions as predominantly linear behavior near the stable equilibria – called the steady states, interrupted by abrupt transitions where the system quickly relaxes to a new steady state, either externally induced or by process noise. This approach corresponds to the piecewise linear models introduced by Glass and Kauffman [2], and the qualitative piecewise linear models described by de Jong et al. [1]. In biology such behavior is frequently observed, as for instance in embryonic growth where the organism develops by transitions through a number of well-defined 'check points'. Within each such checkpoint the system is in relative equilibrium.

We will follow the reasoning of *piecewise linear behavior* (also known more appropriately as piecewise *affine* behavior), with the revision that in the context of Evolutionary Game Theory the atomic mode of propagation is not so much a linear function but the Replicator Dynamics (RDs). Following above line of reasoning, we therefore propose to approximate the systems dynamics as a collection of piecewise RDs as:

$$\frac{dx}{dt}(t) = RD_l(x(t), u(t)) \tag{11}$$

where RD_l is the prevailing replicator dynamics in cell l, as defined according to Equations 7 and 8, with $x(t) = p(t)$ and $u(t) = q(t)$.

Analyzing the learning dynamics becomes significantly more complex when we move from stateless games to multi-state problems. As the agents have independent action probabilities for each state, the result is a very high dimensional problem. In order to deal with this high dimensionality we present an approach to analyze the dynamics per state.

For each state of the Markov game, we define an average reward state game. This game gives the expected reward for each joint action in the state, under the assumption that the agents play a fixed strategy in all other states. When we assume that the action probabilities in the other states remain fixed, we can use Formula 2 to calculate the expected average rewards for each joint action in a state. The game obtained by these rewards can then be studied using the replicator dynamics, exactly as was described in the previous section.

The main problem with analyzing the dynamic on an average reward state game, is that it assumes that agents are only learning in a single state, and are keeping the action probabilities in other states fixed. In reality agents update their action probabilities in all states, and these probabilities will all change in parallel. As the probabilities in other states change, the state game and corresponding dynamic will also change.

To account for these changes we model the system as piecewise dynamical system. This means that for each state, we partition the space of all action probabilities in all states into a number of discrete cells. Each cell corresponds to a different set of attractors in the average reward state game. More precisely, for each state we examine the boundaries were the replicator dynamics of the state game change qual-

	2 State PD						Common Interest Game					
	State 1			State 2			State 1			State 2		
		C	D		C	D		b1	b2		b1	b2
Rewards	C	0.3, 0.3	0, 1	C	0.4, 0.4	0, 1	a1	0.5	0.6	a1	0	1
	D	1, 0	0.2, 0.2	D	1, 0	0.1, 0.1	a2	0.6	0.7	a2	0.5	0
Transitions	(C,C)→(0.9,0.1)			(C,C)→(0.1,0.9)			(a1,b1)→(0.1,0.9)			(a1,b1)→(0.1,0.9)		
	(C,D)→(0.1,0.9)			(C,D)→(0.9,0.1)			(a1,b2)→(0.1,0.9)			(a1,b2)→(0.1,0.9)		
	(D,C)→(0.1,0.9)			(D,C)→(0.9,0.1)			(a2,b1)→(0.1,0.9)			(a2,b1)→(0.1,0.9)		
	(D,D)→(0.9,0.1)			(D,D)→(0.1,0.9)			(a2,b2)→(0.9,0.1)			(a2,b2)→(0.9,0.1)		
	(a)						(b)					

Table 1: Two example Markov games with 2 states and 2 agents with 2 actions in each state. Rewards for joint actions in each state are given in the first row as matrix games. The second row specifies the transition probabilities to both states under each joint action. (a) Conflicting interest game in which the immediate rewards in both states have the same structure as the Prisoner's Dillemma game. (b) Common Interest Markov game in which both agents receive identical immediate rewards.

itatively. We do this by looking for points where equilibria disappear or new equilibria appear. It is important to note that we focus on qualitative changes of the dynamic system. Within each region quantative changes of the dynamic can still occur as the payoffs in the game change, but the same attractor points remain present. When the action probabilities cross a cell boundary, however, they will cause a radical change in the dynamic in the corresponding state. Inside each cell we assume that the probabilities for that state evolve according to a fixed replicator dynamic.

This method can then be used to analyze the full dynamics as follows. When we initialize the learning algorithm with action probabilities, these probabilities define an average reward state game and corresponding replicator dynamic for each state. We then assume that the system follows this dynamic, until the action probabilities cross one of the cell boundaries. When this happens the attractors in the corresponding state change and we get new equilibria in the state game with a new replicator dynamic. This dynamic then drives the dynamics in that state until another boundary is crossed. In this way we can follow the trajectory in all states through multiple cells and dynamics until an equilibrium is reached. In the next section we demonstrate this approach on 2 example Markov games.

4. EXPERIMENTS

We first demonstrate our approach on the example 2 state Markov game in Table 1(a). This problem is a 2 agents, 2 state system. In each state the agents play a Prisoner's Dilemma type game. When the agents both play the same action (i.e joint action (D,D) or (C,C)) the system has a 0.9 probability of staying in the same state and a 0.1 probability of moving to the other state. When the agents play different actions (i.e. joint actions (C,D) or (D,C)) these probabilities are reversed.

As the rewards in each state have the same structure as the PD repeated game of the previous section, one might assume that the agents will converge to the equilibrium point (D,D) in both states. The only pure equilibria in the multi-state example, however, are the points where one agent plays *defect*(D) in state 1 and *cooperate*(C) in state 2, and the other agent does exactly the opposite. This means that instead of mutual defection, the agents converge to a situation, where an agent is exploited in one state, but exploits the other agent in the other state. This is an important change from the stateless game. Table 2 gives an average reward game obtained for state 1 of the 2 state PD game, when agent 1 and agent 2 have a fixed probability of 0.7 and 0.2 respectively, to play action cooperate in state 2. The corresponding direction field is shown in 3(d).

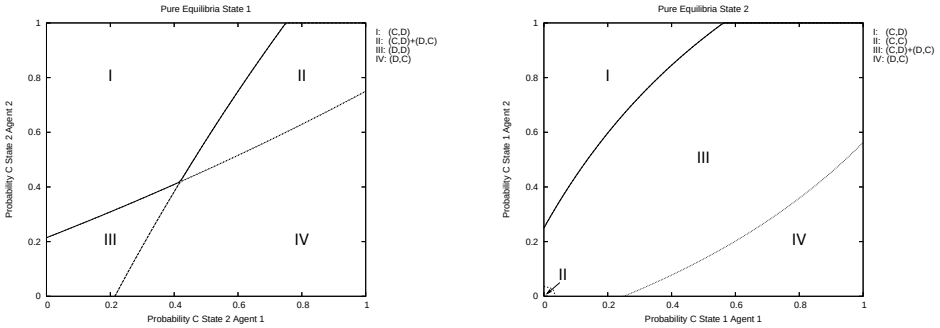
	C	D
C	0.28, 0.35	0.08, 0.78
D	0.48, 0.39	0.19, 0.26

Table 2: Average reward game for state 1 of the 2 state PD, when the agents 1 and 2 play action cooperate in state 2 with probabilities 0.7 and 0.2, respectively.

In Figure 2 we show how the average reward games for both states change as a function of the current action probabilities. Since we have only 2 states, the average reward game in state 1 is completely determined by the strategies in state 2, and vice versa. Figures 2 (a) and (b) show the cells corresponding to different equilibria for state 1 and 2, respectively. For both states we get 4 possible regions that correspond to different dynamics. Figures 3 (a)-(d) give direction fields for state 1 for each of the 4 regions in state 2. In Figure 3 (a) we see the direction field for state 1 when action probabilities in state 2 are in region I. The result is a single equilibrium at joint action (C,D). Figures 3 (c) and (d) give the dynamics corresponding to regions III and IV. Both regions result in a single equilibrium at joint actions (D,D) and (D,C), respectively. Figure 3 (b) shows the dynamics for region II, where we have 2 pure equilibria at (D,C) and (C,D) and an additional mixed equilibrium.

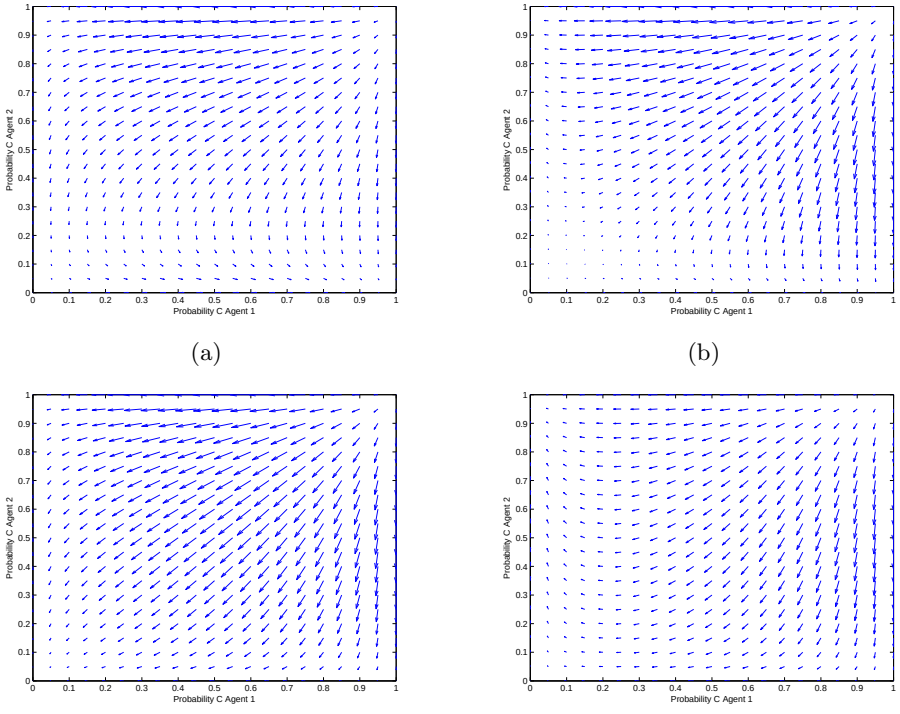
The first experiment demonstrates that the average reward state games indeed approximates the dynamics of the multi-state learning problem. Figure 3(d) shows the direction field obtained by applying the replicator dynamic to the game in Table 2. Figure 4 plots sample paths of the action probabilities in state 1, corresponding to the situation in this game. These sample paths were obtained by running the automata algorithm from Section 2.2 on the example 2 state PD Markov Game. During these experiments the agents only updated their action probabilities in state 1. The action probabilities in state 2 were kept fixed. It is clear from the plot that the resulting paths closely mimic the dynamic predicted by the direction fields of 4(a).

In the next experiment we allow the agents to update all action probabilities in all states. This means that action probabilities in state 1 and 2 change in parallel. The boundaries in the plots mark where an update will cause a change in the dynamics for the other state. Figure 5 (a) shows that the action probabilities in state 1 stay in region I the entire run. This means that the equilibria in state 2 will not switch and the dynamic drives the probabilities to the equilibrium point (C,D). As the probabilities come closer



(a) (b)

Figure 2: Possible pure equilibria in the average reward state games of the 2-state PD problem.(a) Possible pure equilibria in state 1 as a function of state 2 action probabilities.(b) Possible pure equilibria in state 2 as a function of state 1 action probabilities.



(a) (b)

(c) (d)

Figure 3: Example direction fields for state 1 of the 2 state PD. Each figure corresponds to one of the 4 regions shown in Figure 2(a). (a) Region I: (C,D) is the only pure equilibrium. (b) Region II: (C,D) and (D,C) are both equilibria.(c) Region III: (D,D) is the only pure equilibrium.(d) Region IV: (D,C) is the only pure equilibrium.

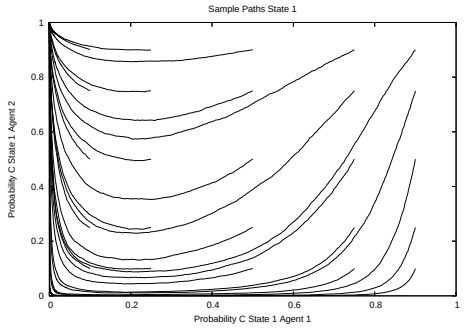


Figure 4: Sample paths generated by the LA algorithm in state 1 when agents use fixed strategy of Table 2 in state 2. (learning rate: 0.0001).

to this point, however, the paths cross 2 region boundaries moving first to region III and then into region IV. This means that as the system evolves the dynamic in state 1 will change from those shown in Figure 3(a) to those shown in Figure 3(c) and eventually to those shown in Figure 3(d). The points where these changes in attractors occur are indicated in Figure 5(a). The crosses indicate the move to region III, while the asterisks indicate the switch to region IV. As can be seen in the plot, the switch has little visible effect on the dynamics. This is not surprising as both dynamics behave very similarly along these paths. When the second transition takes place, however, the driving dynamic completely reverses. This has a very noticeable effect on the evolution of the probabilities as they suddenly change from moving towards (D,D) to moving towards (C,D). This change is clearly visible in Figure 5(b).

In a final experiment we show results for another Markov game, shown in Table 1(b). This game is a common interest game, with both agents receiving identical payoffs. In both states, both agents have a choice between 2 actions: a_1 and a_2 for agent 1, b_1 and b_2 for agent 2. Two pure equilibria exist in this game. In the first the agents play (a2,b1) in state 1 and (a1,b2) in state 2, while in the second they play (a1,b2) in state 1 and again (a1,b2) in state 2. For the experiment we completed multiple runs with different initial probabilities in state 1, and using the same starting point in state 2. From this point the dynamics in state 2 always drive the probabilities to the equilibrium point (a1,b2). As the trajectory comes closer to this point, it crosses the boundary indicated in Figure 6(b). When this happens the dynamics in state 1 switch. In Figure 6(b) we see that the trajectories in state 1 completely reverse as this boundary is crossed. From following the dynamic shown in Figure 7(a) towards (a2,b2), the trajectories follow the dynamic in Figure 7(b) which takes the probabilities in very different directions towards (a1,b2) or (a2,b1).

5. CONCLUSIONS

In this paper we introduced a new method for analyzing the dynamics of multi-agent learning in multi-state problems. By combining piecewise linear dynamic systems with the replicator equations from evolutionary game theory, we obtained a new modeling system which we call Piecewise Replicator Dynamics. In this system the dynamics for each system state are modeled as a set of independent replicator dynamics, between which the system switches based on the current strategies in other states. This method allows us to move from stateless to multi-state games, while retaining the powerful methods EGT offers. More precisely, we are still able to predict the learning system's trajectories and attractors by studying the piecewise replicator dynamics.

Even in rather abstract 2-state problems, such as the ones experimented on for this paper, our methodology already goes beyond the state-of-the-art, since thus far, MAL could only be studied in stateless games using RD. Moreover, even these 2-state problems have shown to be complex enough to emphasize hard challenges for multi-agent reinforcement learning algorithms. In future research we will scale our experiments to systems with more than 2 states.

6. ADDITIONAL AUTHORS

Ann Nowé, Vrije Universiteit Brussel, Belgium. E-mail: ann.nowe@vub.ac.be

7. REFERENCES

[1] H. de Jong, J. Gouze, C. Hernandez, M. Page, T. Sari, and J. Geiselmann. Qualitative simulation of genetic regulatory networks using piecewise-linear models. *Bull Math Biol.*, 66(2):301–340, 2004.

[2] L. Glass and S. Kauffman. The logical analysis of continuous non-linear biochemical control networks,. *J.Theor.Biol.*, 39(1):103–129, 1973.

[3] J. Hofbauer and K. Sigmund. *Evolutionary Games and Population Dynamics*. Cambridge University Press, 1998.

[4] L. Panait and K. Tuyls. Theoretical advantages of lenient q-learners: An evolutionary game theoretic perspective. In *AAMAS'07*, Honolulu, Hawai, 2007.

[5] Y. Shoham, R. Powers, and T. Grenager. If Multi-Agent Learning is the Answer, What is the Question? *Artificial Intelligence*, 171(7):365–377, 2007.

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

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

[8] K. Tuyls and S. Parsons. What evolutionary game theory tells us about multiagent learning. *Artificial Intelligence*, 171(7):406–416, 2007.

[9] K. Tuyls, P. 't Hoen, and B. Vanschoenwinkel. An evolutionary dynamical analysis of multi-agent learning in iterated games. *Autonomous Agents and Multi-Agent Systems*, 12:115–153, 2006.

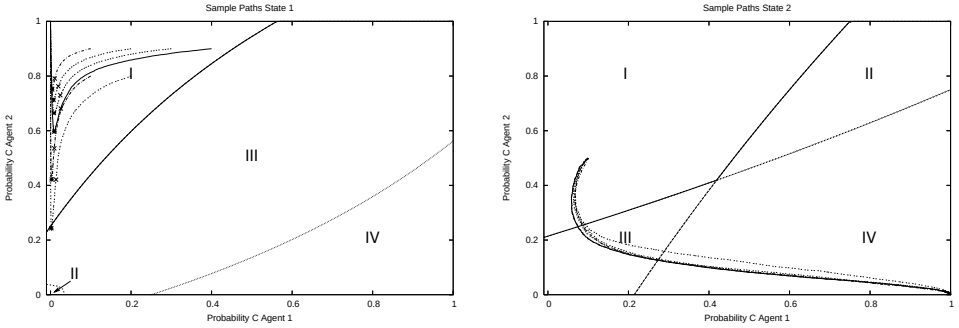
[10] K. Tuyls, K. Verbeeck, and T. Lenaerts. A Selection-Mutation model for Q-learning in Multi-Agent Systems. In *AAMAS'03, Melbourne, Australia*, 2003.

[11] N. van Kampen. *Stochastic Processes in Physics and Chemistry,*. Elsevier Science B. V, Amsterdam, the Netherlands, 1992.

[12] P. Vrancx, K. Verbeeck, and A. Nowé. Networks of learning automata and limiting games. *Alamas 2007, LNAI*, 4865, 2007.

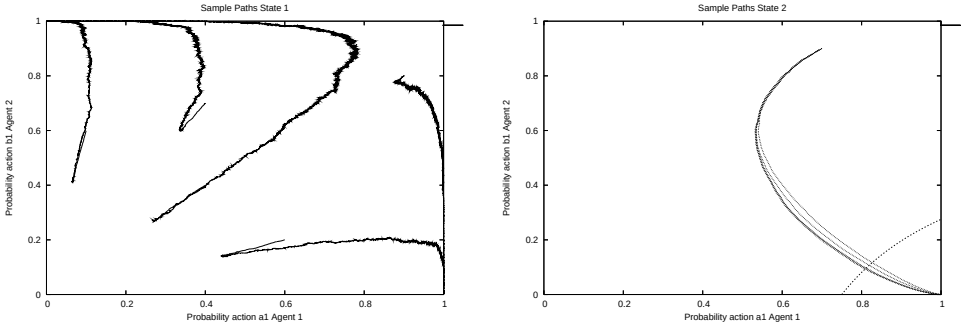
[13] J. W. Weibull. *Evolutionary Game Theory*. MIT Press, 1996.

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



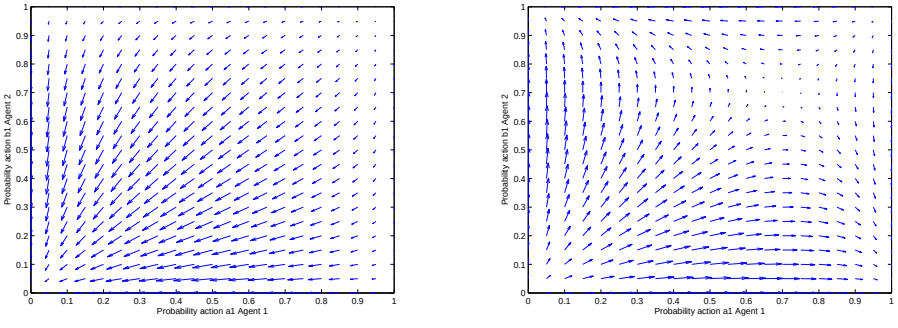
(a) (b)

Figure 5: Sample paths for both states of the 2 state PD, generated by automata using the L_{R-I} scheme with learning rate 0.0001. (a) State 1. (b) State 2.



(a) (b)

Figure 6: Sample paths for both states of the common interest Markov game, generated by automata using the L_{R-I} scheme with learning rate 0.0001. (a) State 1. (b) State 2.



(a) (b)

Figure 7: Two direction fields for state 1 of the common interest Markov game.