Autonomous Agents and Multi-Agent Systems, 12, 115–153, 2006

DOI: 10.1007/s10458-005-3783-9

An Evolutionary Dynamical Analysis of Multi-Agent Learning in Iterated Games

KARL TUYLS karl.tuyls@uhasselt.be
Theoretical Computer Science Group, Hasselt University, Belgium

PIETER JAN 'T HOEN hoen@cwi.nl
Centre for Mathematics and Computer Science, Amsterdam, The Netherlands

BRAM VANSCHOENWINKEL* bvschoen@vub.ac.be
Computational Modeling Lab, Vrije Universiteit Brussel, Belgium

Published online: 12 September 2005

Abstract. In this paper, we investigate Reinforcement learning (RL) in multi-agent systems (MAS) from an evolutionary dynamical perspective. Typical for a MAS is that the environment is not stationary and the Markov property is not valid. This requires agents to be adaptive. RL is a natural approach to model the learning of individual agents. These Learning algorithms are however known to be sensitive to the correct choice of parameter settings for single agent systems. This issue is more prevalent in the MAS case due to the changing interactions amongst the agents. It is largely an open question for a developer of MAS of how to design the individual agents such that, through learning, the agents as a collective arrive at good solutions. We will show that modeling RL in MAS, by taking an evolutionary game theoretic point of view, is a new and potentially successful way to guide learning agents to the most suitable solution for their task at hand. We show how evolutionary dynamics (ED) from Evolutionary Game Theory can help the developer of a MAS in good choices of parameter settings of the used RL algorithms. The ED essentially predict the equilibriums outcomes of the MAS where the agents use individual RL algorithms. More specifically, we show how the ED predict the learning trajectories of Q-Learners for iterated games. Moreover, we apply our results to (an extension of) the COllective INtelligence framework (COIN). COIN is a proved engineering approach for learning of cooperative tasks in MASs. The utilities of the agents are re-engineered to contribute to the global utility. We show how the improved results for MAS RL in COIN, and a developed extension, are predicted by the ED.

Keywords: multi-agent systems, iterated games, reinforcement learning, Evolutionary Game Theory, COllective INtelligence.

1. Introduction

One of the fundamental mathematical constructions of classical Game Theory (GT) [50] is the game. A game is a situation of strategic interaction, in which two or more players have choices from a strategy set to play and receive a payoff based on their own choice and that of the other players. Each player may

* Author funded by a doctoral grant of the institute for advancement of scientific technological research in Flanders (IWT).

desire to maximize his or her immediate payoff, long-term payoff, or may be concerned with more complex issues that relate to other players.

These mathematical games become even more interesting when they are iterated, reflecting many interesting contemporary technological problems (as for instance routing on the Internet, load balancing, niche selection), and reflecting social problems and conditions of decision making. Examples of these are dispersion games (DG) [12], iterated Prisoner's dilemma (PD), standard board games, the matching pennies game, and many more. Also over the last few years games have proved to be powerful tools to design autonomous agents, and to understand interactions in systems composed of many such agents, i.e. multi-agent systems (MASs). For this purpose GT has offered fundamental tools, as for instance the Nash Equilibrium (NE) concept, and many other interesting results. The *evolutionary* approach to GT (EGT) [23, 24], in which many of the strong assumptions of classical GT are relaxed, attracts ever more attention of researchers from different fields, such as economics, computer science and artificial intelligence [3, 5, 11, 27, 31, 34, 51].

However, the word evolution can be interpreted in different ways. Using the word evolution in the biological sense, means we are concerned with environments in which behavior is genetically determined. Strategy selection then depends on the reproductive success of their carriers, i.e. genes. Often, evolution is not intended to be understood in a biological sense but rather as a *learning process* which is called *cultural evolution* [6]. Of course it is *implicit* and intuitive that there is an analogy between biological evolution and learning. We consider this analogy at the individual level. An individual decision maker usually has many ideas or strategies in mind according to which he can behave. Which one of these ideas dominates, and which ones are given less attention depends on the experiences of the individual. We can regard such a set of ideas as a population of possible behaviors. The changes which such a population undergoes in the individual's mind can be very similar to biological evolution.

Learning at the individual level can also be cast as a reinforcement learning (RL) problem. Learning from interaction is a fundamental idea underlying nearly all theories of learning and intelligence. RL is learning what to do – how to map situations to actions – so as to maximize a numerical reward signal. The learner is not told which actions to take, as in forms of supervised machine learning, but instead must discover which actions yield the most reward by trying them. In the most interesting and challenging cases, actions may affect not only the immediate reward, but also the next situation and, through that, all subsequent rewards. These two characteristics – trial-and-error search and delayed reward – are the two most important distinguishing features of RL [37].

In this paper we discuss a strong connection between RL (at the individual level) and EGT explicitly. This work is for a great deal inspired by the work of Börgers and Sarin [5], who showed that in an appropriately constructed continuous time limit, the Cross learning model converges to the asymmetric, continuous time version of the replicator dynamics (RD). Therefore we briefly review their work in terms of dynamic behavior, i.e. the relation between their learning model and evolutionary dynamics (ED) from EGT. As a side result we also mention how Cross learning is related to the theory of learning automata

(LA) as derived in [46]. However we will not show the relation between LA en EGT explicitly as this would lead us too far from the actual topic of the paper, although we briefly outline the theory of LA as it pertains to Cross learning.

The main body of this work consists of extending the results of [5] to a more complex and popular RL model as Q-learning, by deriving the ED of Q-learning. We show how this explicit link between EGT and RL can be a novel approach for MASs, a growing area of research [1, 4, 13, 17, 19, 47]. In this work, the dynamics of the agents employing a RL process are visualized and analyzed from a perspective of ED. More specifically, we will demonstrate how ED from EGT can be used as a model for Q-learning in stochastic games. We show a one-to-one connection between RL and the ED from EGT. More precisely, we connect the RD and a mutation term mathematically with Q-learning. Analysis of the evolutionary stable strategies (ESS) and attractors of the ED aid in setting the parameters for the RL algorithms to achieve desired NE with high utility.

Lastly, we present work on the ED applied to (extensions of) the COllective INtelligence (COIN) framework [54]. COIN is used in cooperative MASs to induce incentives for actions by agents that promote the global utility of the system. This is in line with a growing collection of work where the goal is to establish conditions for cooperative MASs such that they are most likely to exhibit good emergent behavior [2, 4, 21]. Analyzing the COIN approach from the ED perspective is shown to give an indication, through the attractors for ED, as to what parameters to apply for COIN. Furthermore, the ED visualize the incentives in COIN for agents to strive for a high global utility using local learning. The ED hence offer the possibility to Analyze advanced MAS design methodologies.

The rest of this document is structured as follows. In Section 2 we recall the preliminaries from (E)GT and RL. Section 3 makes the relation between ED and RL explicit. More precisely, we start with the main result of Börgers and Sarin, i.e. that in an appropriately constructed continuous time limit, the Cross learning model converges to the asymmetric, continuous time version of the RD [5]. This is followed by our main result, i.e. we show how to extend the results of Börgers and Sarin to a more complex RL method as Q-learning. We end this section with an illustration of these results in iterated games. Section 4 applies these result to a real world application, i.e. load-balancing, represented by the the so-called DG. Section 5 presents the results on the use of EGT for the COIN framework of Wolpert et al. Section 6 discusses and concludes.

2. Preliminaries

In this section we summarize basic results from (Evolutionary) Game Theory (EGT) and RL. Where necessary we provide major references to the literature for the reader not familiar with either fields.

2.1. Concepts from GT

2.1.1. Introduction. The seminal work on GT is the 1944 book *Theory of Games and Economic Behavior* [50][1] by John von Neumann and Oskar Morgenstern. GT is an economical theory that models interactions between rational agents as games of two or more players that can choose from a set of strategies. It is a mathematical study of interactive decision making in the sense that the agents involved in the decisions take into account their own choices and those of others. Choices are determined by (1) stable preferences concerning the outcomes of their possible decisions, and (2) agents acting strategically. In other words, the agents take into account the relation between their own choices and the decisions of other agents. For different economical situations one tries to find the most rational strategy for the different players.

Despite the great usefulness of the NE concept, the assumptions traditional GT make, like rational players that correctly anticipate the other players in an equilibrium, made GT stagnate for quite some time [11, 34, 53]. A multitude of refinements of NE were developed (for instance trembling hand perfection), which made it hard to choose the appropriate equilibrium in a particular situation. Almost any NE could be justified in terms of some particular refinement. This made clear that the static Nash concept did not reflect the (dynamic) real world where people do not always act rationally.

This is were EGT originated. More precisely, John Maynard Smith adopted the idea of evolution from biology [23, 24]. He applied GT to Biology, which made him relax the premises behind GT. Under these biological circumstances, it becomes impossible to judge what choices are the most rational ones. The question now becomes how a player can learn to optimize its behavior and maximize its return. This learning process is the concept of evolution in Biology.

These new ideas led Smith and Price [24] to the concept of ESS, a special case of the Nash condition. In contrast to GT, EGT is descriptive and starts from more realistic views of the game and its players. Here the game is no longer played exactly once by rational players who know all the details of the game, like each others preferences over outcomes. Instead, EGT assumes that the game is played repeatedly by players randomly drawn from large populations, uninformed of the preferences of the opponent players. EGT offers a solid basis for rational decision making in an uncertain world.

In the next sections we briefly recall the elementary concepts from GT and EGT. The reader who is interested in a more thorough explanation of these game theoretic concepts we refer to [11, 53].

2.1.2. Strategic games. In this section we define n-player normal form games from GT as a conflict situation involving gains and losses between n players. In such a game n players interact with each other by all choosing an action (or strategy) to play. All players choose their strategy at the same time.

For reasons of exposition, we focus on a pure strategy set size of 2. For the same reasons, we present the examples from the perspective of 2 agents. Nevertheless, an extension to n-players k-actions games is workable, but examples

$$\mathrm{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \quad \mathrm{B} = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$$

Figure 1. The left matrix (A) defines the payoff for the row player, where the strategies are to be interpreted as rows and the right matrix (B) defines the payoff for the column player where strategies are to be interpreted as columns.

in the larger cases do not show the same illustrative strength as in the 2-player, 2-action cases.

A strategy is defined as a probability distribution over all possible actions. In the 2-pure strategies case, we have: $s_1 = (1, 0)$ and $s_2 = (0, 1)$. A mixed strategy s_m is then defined by $s_m = (x_1, x_2)$ with $x_1, x_2 \neq 0$ and $x_1 + x_2 = 1$. A game $G = (S_1, S_2, P_1, P_2)$ is defined by the payoff functions P_1, P_2 and their strategy sets S_1 for the first player and S_2 for the second player. In the 2-player 2-strategies case, the payoff functions $P_1 : S_1 \times S_2 \to \Re$ and $P_2 : S_1 \times S_2 \to \Re$ are defined by the payoff matrices, A for the first player and B for the second player; see the tables of Figure 1. The payoff tables A, B define the instantaneous rewards. Element a_{ij} is the reward the row-player (player 1) receives for choosing pure strategy s_i from set S_1 when the column-player (player 2) chooses the pure strategy s_j from set S_2. Element b_{ij} is the reward for the column-player for choosing the pure strategy s_j from set S_2 when the row-player chooses pure strategy s_i from set S_1.

The family of 2×2 games can be classified in three general subclasses characterized by the payoff matrix [33]. These three classes allow us to present illustrative experiments; one representative per class.

Class 1:
if $(a_{11} - a_{21})(a_{12} - a_{22}) > 0$ or $(b_{11} - b_{12})(b_{21} - b_{22}) > 0$, at least one of the two players has a dominant strategy, therefore there is just 1 strict equilibrium.

Class 2:
if $(a_{11} - a_{21})(a_{12} - a_{22}) < 0$, $(b_{11} - b_{12})(b_{21} - b_{22}) < 0$, and $(a_{11} - a_{21})(b_{11} - b_{12}) > 0$, there are two pure equilibriums and one mixed equilibrium.

Class 3:
if $(a_{11} - a_{21})(a_{12} - a_{22}) < 0$,$(b_{11} - b_{12})(b_{21} - b_{22}) < 0$, and $(a_{11} - a_{21})(b_{11} - b_{12}) < 0$, there is just one mixed equilibrium.

The first subclass includes those type of games where each player has a dominant strategy,[2], as for instance the PD illustrated in Figure 2.

$$\mathrm{A} = \begin{pmatrix} 1 & 5 \\ 0 & 3 \end{pmatrix} \quad \mathrm{B} = \begin{pmatrix} 1 & 0 \\ 5 & 3 \end{pmatrix}$$

Figure 2. Matrix (A) defines the payoff for the row player for the PD. Matrix (B) defines the payoff for the column player for the PD.

However it includes a larger collection of games since only one of the players needs to have a dominant strategy.

In the second subclass none of the players has a dominated strategy, e.g. battle of the sexes in Figure 3. But both players receive the highest payoff by both playing their first or second strategy. This is expressed in the condition $(a_{11} - a_{21})(b_{11} - b_{12}) > 0$.

$$\mathrm{A} = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix} \quad \mathrm{B} = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}$$

Figure 3. Matrix (A) defines the payoff for the row player for the BOF. Matrix (B) defines the payoff for the column player for the BOF.

The third subclass only differs from the second in the fact that the players do not receive their highest payoff by both playing the first or the second strategy as illustrated by the matching pennies game in Figure 4. This is expressed by the condition $(a_{11} - a_{21})(b_{11} - b_{12}) < 0$.

$$\mathrm{A} = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \quad \mathrm{B} = \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix}$$

Figure 4. Matrix (A) defines the payoff for the row player for the MG. Matrix (B) define the payoff for the column player for the MG.

2.1.3. Nash equilibrium. In traditional GT it is assumed that the players are rational, meaning that every player will choose the action that is best for him, given his beliefs about the other players' actions. A basic definition of a NE is stated as follows. If there is a set of strategies for a game with the property that no player can increase its payoff by changing his strategy while the other players keep their strategies unchanged, then that set of strategies and the corresponding payoffs constitute a Nash equilibrium.

Formally, a NE is defined as follows. When two players play the strategy profile $s = (s_i, s_j)$ belonging to the product set $S_1 \times S_2$ then s is a NE if $P_1(s_i, s_j) \geq P_1(s_x, s_j)\ \forall x \in \{1, ..., n\}$ and $P_2(s_i, s_j) \geq P_2(s_i, s_x)\ \forall x \in \{1, \ldots, m\}$

2.1.4. Pareto optimality. Intuitively a Pareto optimal solution of a game can be defined as follows: a combination of actions of agents in a game is Pareto optimal if there is no other solution for which all players do at least as well and where also at least one agent is strictly better off.

More formally we have: a strategy combination $s = (s_1, \ldots, s_n)$ for n agents in a game is Pareto optimal if there does not exist another strategy combination s' for which each player receives at least the same payoff P_i and where also at least one player j receives a strictly higher payoff than P_j.

2.2. *Concepts from Evolutionary GT*

2.2.1. Evolutionary Stable Strategies. The core equilibrium concept of EGT is that of an ESS. The idea of an ESS was introduced by Maynard Smith and Price in 1973 [24]. Imagine a population of agents playing the same strategy. Assume that this population is invaded by a different strategy, which is initially played by a small number of the total population. If the reproductive success of the new strategy is smaller than the original one, it will not overrule the original strategy and will eventually disappear. In this case we say that the original strategy is evolutionary stable against this new appearing strategy. More general, we say a strategy is an ESS if it is robust against evolutionary pressure from any appearing mutant strategy.

Formally an ESS is defined as follows. Suppose that a large population of agents plays the (mixed) strategy s, and suppose that this population is invaded by a small number of agents playing strategy s'. The population share of agents playing this mutant strategy is $\epsilon \in [0, 1]$. When an individual is playing the game against a random chosen agent, the chance that he is playing against a mutant is ϵ and against a non-mutant is $1-\epsilon$. The payoff for the first player, being a non-mutant is:

$$P(s, (1-\epsilon)s + \epsilon s'), \text{ i.e. } (1-\epsilon)P(s, s) + \epsilon P(s, s')$$

and when being a mutant is,

$$P(s', (1-\epsilon)s + \epsilon s'), \text{ i.e. } (1-\epsilon)P(s', s) + \epsilon P(s', s').$$

Now we can state that a strategy s is an ESS if $\forall\ s' \neq s$ there exists some $\delta \in [0, 1]$ such that $\forall \epsilon\colon\ 0 < \epsilon < \delta$,

$$P(s, (1-\epsilon)s + \epsilon s') > P(s', (1-\epsilon)s + \epsilon s')$$

holds. The condition $\forall \epsilon\colon\ 0 < \epsilon < \delta$ expresses that the share of mutants needs to be sufficiently small.

2.2.2. The relation between NE and ESS. This section explains how the core equilibriums concepts from classical and evolutionary GT relate to one another. We briefly recall the main result here. The reader who is interested in a complete discussion is referred to [11, 15, 29, 33, 53]. The set of ESSs for a particular game are contained in the set of NE for that same game,

$$\{ESS\} \subset \{NE\}.$$

The conditions for an ESS are stricter than the Nash condition. Intuitively this can be understood as follows: as defined above a NE is a best reply against the strategies of the other players. Now if a strategy s_1 is an ESS then it is also a best reply against itself, or optimal. If it was not optimal against itself there would have been a strategy s_2 that would lead to a higher payoff against s_1 than

s_1 itself. So, if the population share ϵ of mutant strategies s_2 is large enough then s_1 is not evolutionary stable because,

$$P(s_2, (1-\epsilon)s_1 + \epsilon s_2) > P(s_1, (1-\epsilon)s_1 + \epsilon s_2)$$

2.2.3. Population Dynamics. In this section we summarize the RD from EGT in a single and a multi-population setting. We discuss the relation with concepts as NE and ESS from GT.

One way in which EGT proceeds is by constructing a dynamic process in which the proportions of various strategies in a population evolve. Examining the expected value of this process gives an approximation which is called the RD [23, 34, 53].

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. The RD highlight the role of selection, it describes how systems consisting of different strategies change over time. They are formalized as a system of differential equations. Each replicator (or genotype in a more biological perspective) represents one (pure) strategy s_i. This strategy is inherited by all the offspring of the replicator. The general form of a replicator dynamic is the following:

$$\frac{dx_i}{dt} = [(A\mathbf{x})_i - \mathbf{x} \cdot A\mathbf{x}]x_i. \tag{1}$$

In Equation (1), x_i represents the density of strategy s_i in the population, A is the payoff matrix which describes the different payoff values each individual replicator receives when interacting with other replicators in the population. The state of the population ($\mathbf{x}$) is described as a probability vector $\mathbf{x} = (x_1, x_2, \ldots, x_J)$ which expresses the different densities of all the different types of replicators in the population. Hence $(A\mathbf{x})_i$ is the payoff which replicator s_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 $\frac{dx_i}{dt}/x_i$ of the population share using strategy s_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 [15, 53].

We now extend the study of population dynamics to more than one population. Games played by individuals of different populations are commonly called *evolutionary asymmetric games*. Here we consider a game to be played between the members of two different populations. As a result, we need two systems of differential equations: one for the row player (R) and one for the column player (C). This setup corresponds to a RD for asymmetric games. If $A = B^t$ (the transpose of B), Equation (1) would emerge again. Player R has a probability vector $\mathbf{p}$ over its possible strategies and player C a probability vector $\mathbf{q}$ over its strategies. This translates into the following replicator equations for the two populations:

$$\frac{dp_i}{dt} = [(A\mathbf{q})_i - \mathbf{p} \cdot A\mathbf{q}]p_i, \tag{2}$$

$$\frac{dq_i}{dt} = [(B\mathbf{p})_i - \mathbf{q} \cdot B\mathbf{p}]q_i . \tag{3}$$

As can be seen in Equations (2) and (3), the growth rate of the types in each population is now determined by the composition of the other population. Note that, when calculating the rate of change using these systems of differential equations, two different payoff matrices (A and B respectively) are used for the two different players.

2.2.4. Relating Nash, ESS and the RD. As being a system of differential equations, the RD have some rest points or equilibriums. An interesting question is how these RD equilibriums relate to the concepts of NE and ESS. We briefly summarize some known results from the EGT literature [11, 15, 29, 33, 53]. An important result is that every NE is an equilibrium of the RD. But the opposite is not true.

Dynamic equilibrium or stationarity alone is not enough to have a better understanding of the RD. For this reason the criterion of asymptotic stability was developed, where there is some kind of local[3] test of dynamic robustness. For a formal definition of asymptotic stability, we refer to [14]. Here we give an intuitive definition. An equilibrium is asymptotic stable if the following two conditions hold:

- Any solution path of the RD that is sufficiently close to the equilibrium remains arbitrarily close to it. This condition is called *Liapunov stability*.
- Any solution path that starts close enough to the equilibrium, converges to the equilibrium.

Now, if an equilibrium of the RD is asymptotically stable, then it is a NE. For a proof, the reader is referred to [33].

Another interesting result due to Hofbauer and Sigmund [15] is the following : If s is an ESS, then the population state $x = s$ is asymptotically stable in the sense of the RD. For a proof see [15, 33]. So, by this result we have some kind of refinement of the asymptotic stable rest points of the RD and it provides a way of selecting equilibriums from the RD that show dynamic robustness.

2.3. Reinforcement Learning

2.3.1. Introduction. The objective of a reinforcement learner is to discover a policy, i.e. a mapping from situations to actions, so as to maximize the reinforcement it receives. The reinforcement is a scalar value which is usually negative to express a punishment, and positive to indicate a reward. Unlike supervised learning techniques, RL methods do not assume the presence of a teacher who is able to judge the action taken in a particular situation. Instead the learner finds out what the best actions are by trying them out and by evaluating the consequences of the actions by itself. For many problems the consequences of an action do not become immediately apparent after performing the action, but only after a number of other actions have been taken. In other

words the selected action may not only affect the immediate reward/punishment the learner receives, but also the reinforcement it might get in subsequent situations, i.e. the delayed rewards and punishments.

One of the greatest contemporary RL challenges is the multi-agent RL problem. The extension of single-agent learning to multi-agent learning recently received a lot of attention. The original RL algorithms, were designed to be used in a single agent setting. When applied to Markovian decision problems most RL techniques are equipped with a proof stating that under very acceptable conditions they are guaranteed to converge to the optimal policy, for instance for Q-learning see [42]. However it is clear that the actions taken by one agent might affect the response characteristics of the environment. So we can no longer assume that the Markov property holds in a multi-agent setting. In the domain of LA, this is referred to as State Dependent Non-Stationarity [26].

When applying RL to a multi-agent case, typically two extreme approaches have been taken. The first one totally neglects the presence of the other agents, and agents are considered to be selfish reinforcement learners. The effects caused by the other agents also acting in that same environment are considered as noise. It is clear that for problems where agents have to coordinate in order to reach a preferable situation for all agents, this may not yield satisfactory results [16, 17].

The other extreme is the joint action space approach where the state and action space are respectively defined as the Cartesian product of the agent's individual state and action spaces. More precisely, if S is the set of states and $A_1, \ldots, A_n$ the action sets of the n different agents, the learning will be performed in the product space: $S \times A_1 \times \cdots \times A_n \rightarrow \Re$ This implies that the state information is shared amongst the agents and actions are taken and evaluated synchronously. It is obvious that this approach leads to very big state-action spaces, and assumes instant communication between the agents. Clearly this approach is in contrast with the basic principles of MASs: distributed control, asynchronous actions, incomplete information, cost of communication[4].

This work can be considered as another approach, i.e. the EG Theoretic one, taking benefit from the formalized relation between EGT and RL. More precisely, An EG Theoretic point of view towards the multi-agent learning problem enhances the understanding of this problem as the basic assumptions of both fields are very similar. Moreover, the formal relation between EGT and RL allows us to design and optimize the learning behavior of the different agents present in a MAS, as shown in Sections 3–5.

Before we continue with the formalization between EGT and RL, we summarize in the subsequent sections the RL-algorithms we consider in this work. We start to explain Cross learning as the work of Börgers and Sarin [5] is our starting point. We continue with a brief description of LA and conclude with explaining Boltzmann Q-learning.

2.3.2. Cross learning. The cross learning model is a special case of Bush and Mosteller's RL model [5, 7, 8, 10]. It originates from mathematical psychology. The model considers several agents playing the same normal form game repeatedly in discrete time. At each point in time, each player is characterized by a

probability distribution over his strategy set which indicates how likely he is to play any of his strategies. The probabilities change over time in response to experience. At each time step (indexed by t), a player chooses one of its strategies based on the probabilities which are related to each isolated strategy. In response to an action, a player receives a payoff, expressing how good the chosen action was. For more details on the formal specification we refer to [7, 8, 10].

Here we will consider a normal-form game with two players to simplify the representation. In case of a 2-player game with payoff matrix A and B, the row player is associated with payoff table A and the column player is associated with payoff table B. The row player can choose from s strategies, each corresponding to a row in the tables, while the column player can choose from r strategies, each corresponding to a column in the payoff tables. The row player gets payoff a_{ij} when he chooses strategy i and the column player chooses strategy j and the column player receives in that case payoff b_{ij}. It is assumed that $0 < a_{i,j}, b_{i,j} < 1$(Figure 5).

$$\mathrm{A} = \begin{pmatrix} a_{11} & \ldots & a_{1j} & \ldots & a_{1r} \\ \ldots & & & & \ldots \\ a_{s1} & \ldots & a_{sj} & \ldots & a_{sr} \end{pmatrix} \qquad \mathrm{B} = \begin{pmatrix} b_{11} & \ldots & b_{1j} & \ldots & b_{1r} \\ \ldots & & & & \ldots \\ b_{s1} & \ldots & b_{sj} & \ldots & b_{sr} \end{pmatrix}$$

Figure 5. (a) Matrix (A) defines the payoff for the row player. (*b*) Matrix (B) defines the payoff for the column player.

Players do not observe each others strategies and payoffs and play the game repeatedly. After making their observations, at each stage they update their probability vector. For the row player we note this by vector **p**:

$$\mathbf{p}(t) = (p_1(t), \ldots, p_s(t))$$

And for the column player this is vector q:

$$\mathbf{q}(t) = (q_1(t), ..., q_r(t)).$$

The vectors are updated according to,

$$p_i(t+1) = a_{ij} + (1 - a_{ij})\, p_i(t), \tag{4}$$

$$p_{i'}(t+1) = (1 - a_{ij})\, p_{i'}(t). \tag{5}$$

Equation (4) expresses how the probability of the selected strategy (i) is updated and Equation (5) expresses how all the other strategies $i' \neq i$, which where not selected, are corrected for the row player. If this player p played strategy i in the t-th repetition of the game, and if he received payoff a_{ij}, then he updates his state by taking a weighted average of the old state, and of the unit vector which puts all probability on strategy i (see Equation (4)).

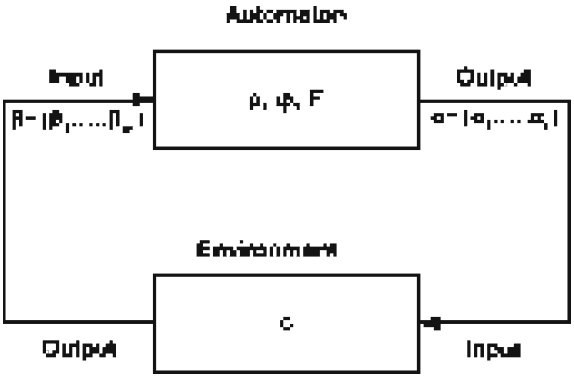

Figure 6. The feedback connection of the learning automaton – Environment pair.

The probability vector of $q(t)$ is updated in an analogous manner. Börgers and Sarin show that in an appropriately constructed continuous time limit, the model converges to the asymmetric, continuous time version of the RD, [5].

2.3.3. Learning automata's. The term LA was for the first time introduced in the work of Narendra and Thathacher in 1974 [25]. Since then there has been a large amount of development in the field and a number of survey papers and books around this topic have appeared: to name a few [26, 41].

In Figure 6 a learning automaton is illustrated in its most general form. The automaton tries to determine an optimal action out of a set of possible actions to perform. A learning automaton is defined by a quintuple $\{\alpha, \beta, F, \mathbf{p}, T\}$ for which α is the action or output set $\{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ of the automaton, β is a random variable in the interval $[0, 1]$, F is the state transition function $F : \phi \times \beta \rightarrow \phi$ mapping the current state and input into the next state, $\mathbf{p}$ is the action probability vector of the automaton or agent and T denotes an update scheme. The output α of the automaton is actually the input to the environment. The input β of the automaton is the output of the environment, which is modeled through penalty probabilities c_i with $c_i = P[\beta \,|\, \alpha_i], i = 1 \cdots r$. The automaton can be either stochastic or deterministic.

The function T is the reinforcement algorithm which modifies the action probability vector p with respect to the performed action and the received response. The new probability vector can therefore be written as:

$$p(t+1) = T\{\alpha, \beta, p(t)\}$$

with t the timestep.

Next we summarize the different update schemes T. The most important examples of update schemes are *linear reward-penalty*, *linear reward-inaction* and *linear reward-ϵ-penalty*. The philosophy of those 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 update algorithm is given by

$$p_i(t+1) \leftarrow p_i(t) + a(1-\beta(t))(1-p_i(t)) - b\beta(t)p_i(t)$$
$$\text{if } \alpha_i \text{ is the action taken at time t} \tag{6}$$

$$p_j(t+1) \leftarrow p_j(t) - a(1-\beta(t))p_j(t) + b\beta(t)[(r-1)^{-1} - p_j(t)] \text{ if } \alpha_j \neq \alpha_i \tag{7}$$

The constants a and b in $[0, 1]$ are the reward and penalty parameters respectively. When $a = b$ the algorithm is referred to as linear reward-penalty (L_{R-P}), when $b = 0$ it is referred to as linear reward-inaction (L_{R-I}) and when b is small compared to a it is called linear reward-ϵ-penalty ($L_{R-\epsilon P}$).

If the penalty probabilities c_i of the environment are constant, the probability $p(n+1)$ is completely determined by $p(n)$ and hence $p(n)_{n>0}$ is a discrete-time homogeneous Markov process. Convergence results for the different schemes are obtained under the assumptions of constant penalty probabilities, see [26].

2.3.4. Q-learning. Common RL methods, which can be found in [37] are structured around estimating value functions. A value of a state or state-action pair, is the total amount of reward an agent can expect to accumulate over the future, starting from that state. One way to find the optimal policy is to find the optimal value function. If a perfect model of the environment as a Markov decision process is known, the optimal value function can be learned with an algorithm called value iteration. Q-learning is an adaptive value iteration method [52], which bootstraps its estimate for the state-action value $Q_{t+1}(s, a)$ at time $t+1$ upon its estimate for $Q_t(s', a')$ with s' the state where the learner arrives after taking action a in state s:

$$Q_{t+1}(s, a) \leftarrow (1-\alpha)Q_t(s, a) + \alpha(r + \gamma \, max_{a'} Q_t(s', a')) \tag{8}$$

With α the usual step size parameter, γ a discount factor and r the immediate reinforcement. The Q-function is a quality function, expressing how good it is to take action a in state s. The reader who is interested in more details, we refer to [42].

3. Relating evolution and reinforcement learning

At this point, we have summarized the main concepts from (evolutionary) GT in order to be able to present our first contribution. In this section we will show the dynamics of Cross learning and Q-learning and illustrate them on different examples [40, 45, 48]. First we discuss the main result of Börgers and Sarin [5], i.e. that in appropriately constructed continuous time limit, the Cross' learning model converges to the asymmetric, continuous time version of the RD. The continuous time limit is constructed in such a manner that each time interval sees many iterations of the game, and that the adjustments that the players (or

Cross learners) make between two iterations of the game are very small. If the limit is constructed in this manner the (stochastic) learning process becomes in the limit deterministic. This limit process satisfies the system of differential equations which characterizes the RD.

Furthermore as a new result we show that Cross learning is a special case of LA. A more extensive discussion of this result and its relation to EGT can be found in [46].

Next we continue with a novel derivation of the Q-learning dynamics. Although not shown, a similar approach can be applied to the derivation of the Cross dynamics [5].

3.1. The Cross and LA dynamics

In their paper, *Learning through Reinforcement and RD*, Börgers and Sarin proved a most interesting link between EGT and RL [5]. More precisely they considered a version of R.R. Bush and F. Mosteller's [7, 8] stochastic learning theory in the context of games and proved that in a continuous time limit the learning model converges to the asymmetric continuous time replicator equations of EGT. With this result they provided a formalization of the relation between learning at the individual level and biological evolution.

The version of the learning model of Bush and Mosteller is called Cross learning and has been explained in detail in Section 2.3.2. It is important to note that each time interval has to see many iterations of the game, and that the adjustments which players make between two iterations of the game are very small. If the limit is constructed in this manner, then the learning process converges to the RD. Important to understand is that this result refers to arbitrary, finite points in time. The result does not hold if infinite time is considered. The asymptotic behavior for time tending to infinity of the discrete time learning can be quite different from the asymptotic behavior of the continuous time RD. The reader who is interested in the mathematical proof is referred to [5].

If it is assumed that $b = 0$ and $a = 1$ in Equations (6) and (7), the relation between (4) and (5) with (6) and (7) becomes apparent. In this association, when the reward penalty term $a = 1$, the feedback from the environment $(1 - \beta(n))$ equals the game reward A_{ij}. Hence the equations become equivalent. As a result, the conditions and implications from the relation between the Cross learning model and RD carry over to LA under restricted conditions [46].

3.2. The Q-learning dynamics

In this section we derive mathematically the relation between Q-learning and the RD[5]. More precisely we construct a continuous time limit of the Q-learning model, where Q-values are interpreted as Boltzmann probabilities for the action selection. For simplicity we consider games between two players. Each agent (or player) has a probability vector over his action set, more precisely $x_1, \ldots, x_n$ over action set $a_1, \ldots, a_n$ for the first player and $y_1, \ldots, y_m$ over $b_1, \ldots, b_m$ for

the second player. Formally the Boltzmann distribution is described by

$$x_i(k) = \frac{e^{\tau Q_{a_i}(k)}}{\sum_{j=1}^{n} e^{\tau Q_{a_j}(k)}}$$

$$\frac{dx_i(t)}{dt} = \frac{d}{dt}\frac{e^{\tau Q_i(t)}}{\sum_j e^{\tau Q_j(t)}}$$
$$= \tau x_i(t)\left[\frac{dQ_i(t)}{dt} - \sum_j x_j(t)\frac{dQ_j(t)}{dt}\right],$$

where $x_i(k)$ is the probability of playing strategy i at time step k and τ[6] is the temperature. Now we can find an expression for, $x_i(k+1)$.

$$\frac{x_i(k+1)}{x_i(k)} = \frac{e^{\tau Q_{a_i}(k+1)}\sum_j e^{\tau Q_{a_j}(k)}}{e^{\tau Q_{a_i}(k)}\sum_j e^{\tau Q_{a_j}(k+1)}}$$
$$= \frac{e^{\tau Q_{a_i}(k+1)}e^{-\tau Q_{a_i}(k)}\sum_j e^{\tau Q_{a_j}(k)}}{\sum_j e^{\tau Q_{a_j}(k+1)}}$$
$$= \frac{e^{\tau \Delta Q_{a_i}(k)}}{\sum_j x_j e^{\tau \Delta Q_{a_j}(k)}}.$$

From which follows,

$$x_i(k+1) = x_i(k)\frac{e^{\tau \Delta Q_{a_i}(k)}}{\sum_j x_j e^{\tau \Delta Q_{a_j}(k)}}.$$

If we consider the difference equation for x_i we have,

$$x_i(k+1) - x_i(k) = \frac{x_i(k)e^{\tau \Delta Q_{a_i}(k)}}{\sum_j x_j(k)e^{\tau \Delta Q_{a_j}(k)}} - x_i(k)$$
$$= x_i(k)\left(\frac{e^{\tau \Delta Q_{a_i}(k)} - \sum_j x_j(k)e^{\tau \Delta Q_{a_j}(k)}}{\sum_j x_j(k)e^{\tau \Delta Q_{a_j}(k)}}\right).$$

For the continuous time version we suppose that the amount of time that passes between two repetitions of the game is given by δ with $0 < \delta \leq 1$. The variable $x_i(k\delta)$ describes the x-values at time $k\delta = t$. Under these assumptions we have,

$$\frac{x_i(k\delta+\delta) - x_i(k\delta)}{\delta} = \frac{x_i(k\delta)}{\delta\sum_j x_j(k\delta)e^{\tau \Delta Q_{a_j}(k\delta)}}$$
$$\times(e^{\tau \Delta Q_{a_i}(k\delta)} - \sum_j x_j(k\delta)e^{\tau \Delta Q_{a_j}(k\delta)}).$$

We are interested in the limit with $\delta \to 0$. We find the state of the limit process at some time $t \geq 0$ (which we keep fixed) by taking the limit of $x_i(k\delta)$ with $\delta \to 0$ and $k\delta \to t$.

$$\begin{aligned}\lim_{\delta\to 0}\frac{\Delta x_i(k\delta)}{\delta} &= \lim_{\delta\to 0}\frac{x_i(k\delta)}{\delta\sum_j x_j(k\delta)e^{\tau\Delta Q_{a_j}(k\delta)}} \\ &\quad\times(e^{\tau\Delta Q_{a_i}(k\delta)} - \sum_j x_j(k\delta)e^{\tau\Delta Q_{a_j}(k\delta)}). \\ &= \lim_{\delta\to 0}\frac{x_i(k\delta)}{\sum_j x_j(k\delta)e^{\tau\Delta Q_{a_j}(k\delta)}} \\ &\quad\times\lim_{\delta\to 0}\left(\frac{e^{\tau\Delta Q_{a_i}(k\delta)}}{\delta} - \frac{\sum_j x_j(k\delta)e^{\tau\Delta Q_{a_j}(k\delta)}}{\delta}\right).\end{aligned}$$

The first limit,

$$\lim_{\delta\to 0}\frac{x_i(k\delta)}{\sum_j x_j(k\delta)e^{\tau\Delta Q_{a_j}(k\delta)}}$$

equals $x_i(t)$ abbreviated to x_i, because the exponent term becomes 1 ($\Delta Q_{a_j}(k\delta)$ becomes zero) and $\sum_j x_j(k\delta)$ equals 1(sum of all probabilities equals 1). Therefore,

$$\lim_{\delta\to 0}\frac{\Delta x_i(k\delta)}{\delta} = x_i \underbrace{\times\lim_{\delta\to 0}\left(\frac{e^{\tau\Delta Q_{a_i}(k\delta)}}{\delta} - \frac{\sum_j x_j(k\delta)e^{\tau\Delta Q_{a_j}(k\delta)}}{\delta}\right)}_{T_2}.$$

The second limit is undefined (we have a $\frac{0}{0}$ situation), which allows us to use the rule of the l'hôspital. The second term now equals (abbreviated to T_2),

$$\begin{aligned}T_2 &= \lim_{\delta\to 0}\frac{\tau\Delta Q_{a_i}(k\delta)e^{\tau\Delta Q_{a_i}(k\delta)}}{\delta} \\ &\quad - \sum_j x_j(k\delta)\times\lim_{\delta\to 0}\left(\tau\Delta Q_{a_j}(k\delta)\frac{e^{\tau\Delta Q_{a_j}(k\delta)}}{\delta}\right) \\ &= \tau\frac{dQ_{a_i}(t)}{dt} - \sum_j x_j(t)\tau\frac{dQ_{a_j}(t)}{dt}.\end{aligned}$$

The total limit now becomes,

$$\frac{\frac{dx_i}{dt}}{x_i} = \tau\left(\frac{dQ_{a_i}}{dt} - \sum_j \frac{dQ_{a_j}}{dt}x_j\right). \tag{9}$$

We now have derived the continuous time model of the Q-learning process. As you can see in Equation (9) we need an expression for $\frac{dQ_{a_i}(t)}{dt}$. We can derive the differential equation for the Q-function by performing the following steps.

The Q-learning update rule for the first player can be written as follows,

$$Q_{a_i}(k+1) = Q_{a_i}(k) + \alpha(r_{a_i}(k+1) + \gamma \max_{a_i} Q - Q_{a_i}(k))$$

which implies,

$$\Delta Q_{a_i}(k) = \alpha(r_{a_i}(k+1) + \gamma \max_{a_i} Q - Q_{a_i}(k)).$$

This expression is the difference equation for the Q-function. If we make this equation infinitesimal, going from discrete steps to a continuous version, we suppose that the amount of time that passes between two repetitions of updates of the Q-values is given by δ with $0 < \delta \leq 1$. The variable $Q_{a_i}(k\delta)$ describes the Q-values at time $k\delta$. Now, we get,

$$\Delta Q_{a_i}(k\delta) = \alpha(r_{a_i}((k+1)\delta) + \gamma \max_{a_i} Q - Q_{a_i}(k\delta)) \times ((k+1)\delta - k\delta)$$

which is the same as writing,

$$\Delta Q_{a_i}(k\delta) = \alpha(r_{a_i}((k+1)\delta) + \gamma \max_{a_i} Q - Q_{a_i}(k\delta))\delta.$$

We are interested in the limit $\delta \to 0$. We find the state of the limit process at some time $t \geq 0$ (which we keep fixed) by taking the limit of $Q_{a_i}(k\delta)$ with $\delta \to 0$ and $k\delta \to t$. If we now bring δ to the left side, and take the limit for $\delta \to 0$, we have

$$\frac{dQ_{a_i}}{dt} = \alpha(r_{a_i} + \gamma \max_{a_i} Q - Q_{a_i}). \tag{10}$$

Now, we can substitute Equation (10) in Equation (9), yielding,

$$\begin{aligned}\frac{\frac{dx_i}{dt}}{x_i} &= \tau \left(\alpha r_{a_i} + \alpha\gamma \max_{a_i} Q_{a_i} - \alpha Q_{a_i} \right. \\ &\qquad \left. - \sum_j x_j \alpha(r_{a_j} + \gamma \max_{a_i} Q_{a_i} - Q_{a_j}) \right) \\ &= \tau\alpha \left(r_{a_i} - \sum_j x_j r_{a_j} - Q_{a_i} + \sum_j Q_{a_j} x_j \right)\end{aligned}$$

because $\sum_j x_j = 1$, this expression becomes,

$$\begin{aligned}\frac{\frac{dx_i}{dt}}{x_i} &= \tau\alpha \left(r_{a_i} - \sum_j x_j r_{a_j} - Q_{a_i} \sum_j x_j + \sum_j Q_{a_j} x_j \right) \\ &= \tau\alpha \left(r_{a_i} - \sum_j x_j r_{a_j} + \sum_j x_j (Q_{a_j} - Q_{a_i}) \right)\end{aligned}$$

because $\frac{x_j}{x_i}$ equals $\frac{e^{\tau Q_{a_j}}}{e^{\tau Q_{a_i}}}$, we have

$$\alpha \sum_j x_j ln\left(\frac{x_j}{x_i}\right) = \alpha\tau \sum_j x_j (Q_{a_j} - Q_{a_i})$$

which gives us,

$$\frac{\frac{dx_i}{dt}}{x_i} = \alpha\tau \left(r_{a_i} - \sum_j x_j r_{a_j} \right) + \alpha \sum_j x_j ln\left(\frac{x_j}{x_i}\right)$$

Now suppose that we have payoff matrices A and B for the two players. Now we can write r_{a_i} as $\sum_j a_{ij} y_j$. This results in,

$$\frac{dx_i}{dt} = x_i \alpha\tau ((A\mathbf{y})_i - \mathbf{x} \cdot A\mathbf{y}) + x_i \alpha \sum_j x_j ln\left(\frac{x_j}{x_i}\right) \tag{11}$$

analogously for the second player, we have,

$$\frac{dy_i}{dt} = y_i \alpha\tau ((B\mathbf{x})_i - \mathbf{y} \cdot B\mathbf{x}) + y_i \alpha \sum_j y_j ln\left(\frac{y_j}{y_i}\right) \tag{12}$$

Equations (11) and (12) express the dynamics of both Q-learners in terms of Boltzmann probabilities. We remark that Equations (11) and (12) make explicit that each player takes into account the other player for its immediate reward. This is usually implicit in RL. In the next subsection we discuss the derived equations, followed by an illustration in the last subsection.

3.3. Discussion of Q-learning dynamics

In this subsection we analyze the evolutionary model expressed by (11) and (12), both in an evolutionary perspective as in a RL perspective.

3.3.1. Selection – mutation perspective. Comparing (11) or (12) with the RD in (1), we see that the first term of (11) or (12) is exactly the RD and thus takes care of the selection mechanism, see [53]. The mutation mechanism for Q-learning is therefore left in the second term, i.e.

$$x_i \alpha \sum_j x_j ln\left(\frac{x_j}{x_i}\right)$$

and can be rewritten as:

$$x_i \alpha \left(\sum_j x_j (ln(x_j) - ln(x_i)) \right) \tag{13}$$

In Equation (13) we recognize two entropy terms, one over the entire probability distribution x, and one over strategy x_i. We can describe the entropy terms as follows:

$$S_i = -x_i ln(x_i)$$

and

$$S_n = -\sum_j x_j ln(x_j)$$

S_i describes the information we have concerning strategy i, S_n describes the information concerning the entire distribution. Therefore (13) can be written as

$$-(\alpha x_i S_n - \alpha S_i) \tag{14}$$

In [15] the mutation equation was derived as a difference between the old state of x_i and the new one,

$$\sum_j \epsilon_{ij}(x_j - x_i),$$

where ϵ_{ij} explicitly represents the mutation rate from strategy j to strategy i, and $\epsilon_{ij} \geq 0 \ \forall j = 1, \ldots, n$ and $\sum_j \epsilon_{ij} = 1$. In the Q-learning dynamics mutation is expressed analogously with entropy expressing the state or information of a strategy.

Relating entropy and mutation is an established result. It is a well known fact [35, 36] that mutation increases entropy. In [36], it is stated that the concepts are familiar with thermodynamics in the following sense: the selection mechanism is analogous to *energy* and mutation to *entropy*. So generally speaking, mutations tend to increase entropy. In Figure 7 we show the entropy function S_n in the $[0, 1]$ interval for binomial strategies. Entropy reaches its maximum at 0.5, which is normal because x is then an equiprobable distribution, meaning there is a lot of uncertainty about which strategy will be played.

The dynamic equations for Q-learning reveal that selection happens as expressed in a RD, i.e. a strategy is favored according to the pay-off it receives, relative to it's opponent. As seen in the formula a mutation effect is also present. This is calculated by the difference in entropy of a strategy compared to the whole population. In the next section we discuss how selection and mutation relate to the concepts of exploitation and exploration from RL.

3.3.2. Exploration – exploitation perspective. One of the great challenges of RL is the trade-off between exploration and exploitation. To collect enough reward an agent should select one of the actions that return a big payoff, but to discover such actions the agent should try actions he didn't try before, or possibly recently. Mapping exploration and exploitation to *selection* and *mutation* we obtain a biological interpretation of the exploration-exploitation concept in RL. More precisely, the RD of (11), i.e. the first term

$$x_i \alpha \tau((A\mathbf{y})_i - \mathbf{x} \cdot A\mathbf{y})$$

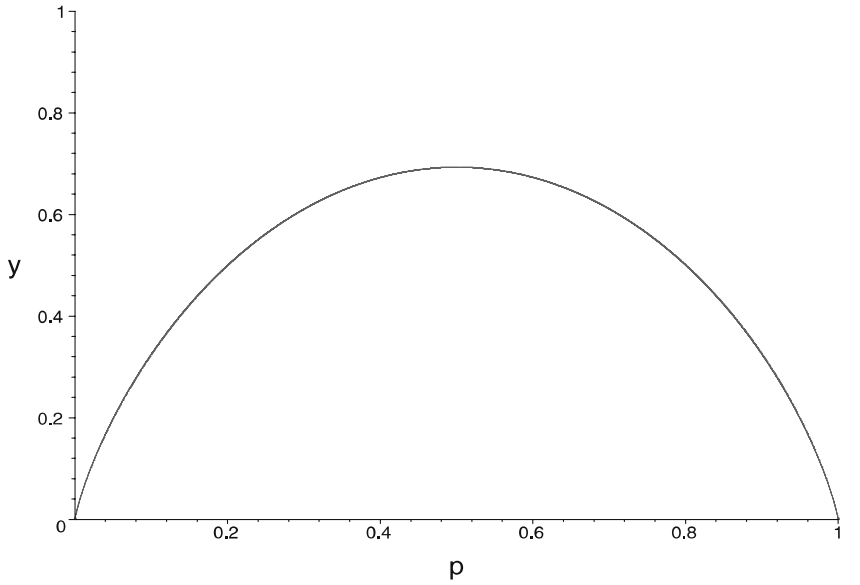

Figure 7. The entropy function for a binomial distribution.

takes care of selecting better strategies over time, which maps perfectly on the concept of exploitation (being greedy). The mutation term of (11), i.e. the second term

$$x_i \alpha \sum_j x_j ln(x_j) - \alpha x_i ln(x_i)$$

increases entropy, but at the same time it provides variety. Exploration can be considered as the mutation concept, as both concepts take care of providing variety. In the dynamic model of RL (see Equation (11)) variation is measured by entropy. In the algorithmic version of RL (see Equation 8) it is a matter of correctly setting the exploration parameter. The exploration-exploitation tradeoff is normally a difficult fine tuning process of the necessary parameters in the learning process. With the selection-mutation model of Equations (11) and (12), we tackle this problem as illustrated by the experiments in Section 4.

3.4. Illustration of the Q-learning dynamics in Iterated Games

In this section we describe some experiments that illustrate the mathematical derivation of the previous sections. The experiments are conducted in the general three types of game classification introduced in Section 2.1.2. We plot the derived RD equations derived above and the learned strategies of the agents based on the Q-values from the experiments to illustrate the one-to-one mapping between the two.

$$A = \begin{pmatrix} 1 & 5 \\ 0 & 3 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 0 \\ 5 & 3 \end{pmatrix}$$

Figure 8. PD: The left matrix (A) defines the payoff for the row player, the right one (B) for the column player.

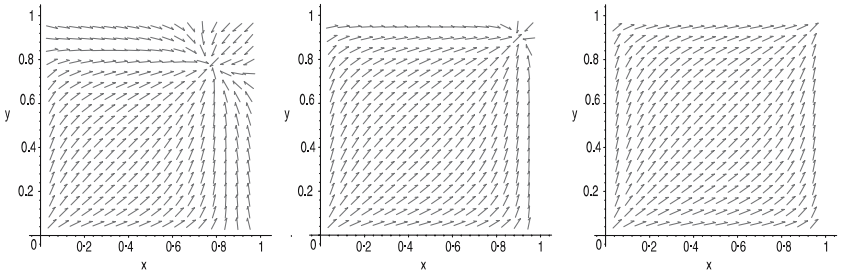

Figure 9. The direction field plots of the PD (subclass 1) game with $\tau = 1, 2, 10$. The x-axis represents the probability with which the first player plays *defect*, the y-axis represents the probability with which the second player plays *defect*.

3.4.1. Subclass 1. For this subclass we consider the PD game [11, 53]. In this game both players have the same strategy set containing one dominant strategy, more precisely *defect*. The payoff tables for this game are defined as follows (Figure 8),

In Figure 9 the direction field plot of the differential equations of the PD game is plotted. These equations can be derived by filling in the payoff matrices A and B, α and τ in Equations (11) and (12). The x-axis expresses the probability with which player 1 plays strategy 1 (*defect*), and the y-axis expresses the probability with which player 2 plays strategy 1 (*defect*). The equations are plotted for three values of τ, more precisely $1, 2, 10$. Only the last setting of the dynamics attain the NE (the only attractor in the last plot) for the game at the coordinates $(1, 1)$. In the first 2 plots τ is not chosen large enough to reach the NE, which is a focus, or attractor.

In Figure 10 we also plotted the Q-learning process for the same game with the same settings as for the system of differential equations.

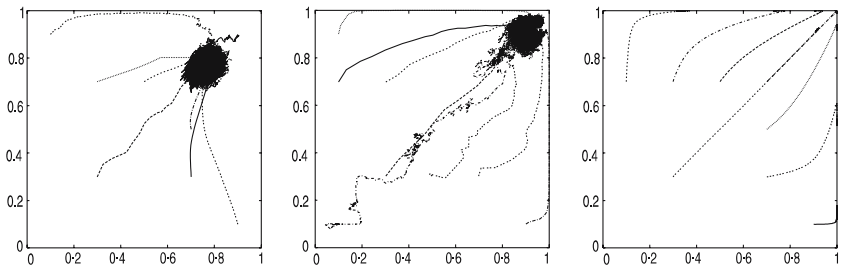

Figure 10. The Q-learning plots of the PD (subclass 1) game with $\tau = 1, 2, 10$. The x-axis represents the probability with which the first player plays *defect*, the y-axis represents the probability with which the second player plays *defect*.

$$A = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}$$

Figure 11. Battle of the sexes: The left matrix (A) defines the payoff for the row player, the right one (B) for the column player.

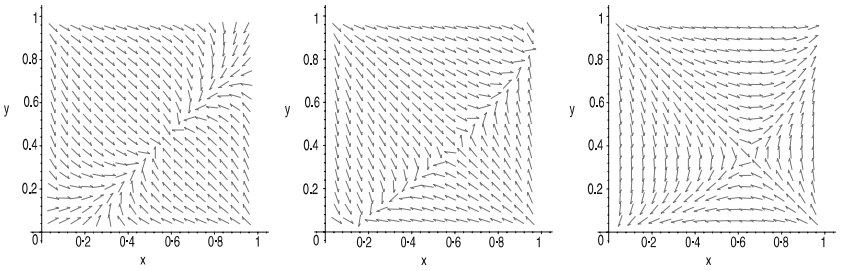

Figure 12. The direction field plots of the battle of the sexes (subclass 2) game with $\tau = 1, 2, 10$. The x-axis represents the probability with which the first players plays its first strategy, the y-axis represents the probability with which the second players plays its first strategy.

As starting points for the Q-learning process we chose a grid of 25 points from which we plotted the results for the following 8 points, {(0.1; 0.7), (0.3; 0.3), (0.3; 0.7), (0.5; 0.7), (0.7; 0.3), (0.7; 0.5), (0.7; 0.7), (0.9; 0.1)}.[7] In every point a learning path, i.e. the joint strategies over time, starts and converges to a particular point. If you compare the plots with the direction field plots for the same value of τ you can see that the sample paths of the learning process are approximated by the paths of the differential equations. This allows us to better understand and predict the Q-learning process with our evolutionary model of Equations (11) and (12).

3.4.2. Subclass 2. For the second subclass we considered the battle of the sexes game [11, 53] defined by the payoff matrices of Figure 11. In Figure 12 the direction field plot of the differential equations of this game is plotted. Again the direction field of the equations are plotted for three values of τ, more precisely $1, 2, 10$. In the first 2 plots τ is not big enough to reach one of the three NE. Only in the last one the dynamics attain the NE (the three attractors in the last plot) for the game at the coordinates $(1, 1)$, $(0, 0)$ and $(\frac{2}{3}, \frac{1}{3})$. The mixed equilibrium though is very unstable. Any small perturbation away from this equilibrium will typically lead the dynamics to one of the two pure equilibriums.

In Figure 13 we also plotted the Q-learning process for the same game with the same settings as for the system of differential equations. As starting points for the Q-learning process we chose the same grid of points used for the experiments under subclass 1. Again in every point a learning path starts and converges to a particular point. If you compare the plots with the direction field plots for the same value of τ you can see that the sample paths of the learning process approximates the paths of the differential equations. The instability of

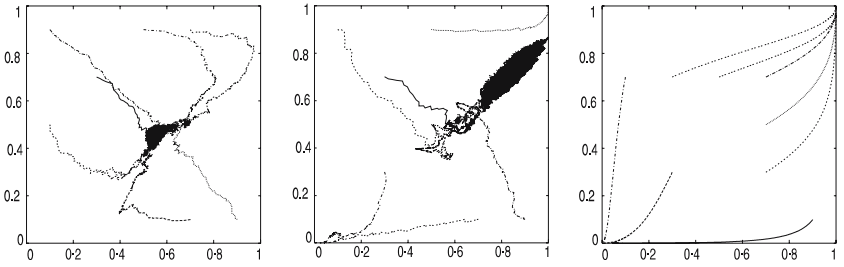

Figure 13. The Q-learning plots of the battle of the sexes (subclass 2) game with $\tau = 1, 2, 10$. The x-axis represents the probability with which the first players plays its first strategy, the y-axis represents the probability with which the second players plays its first strategy.

$$\mathrm{A} = \begin{pmatrix} 2 & 3 \\ 4 & 1 \end{pmatrix} \quad \mathrm{B} = \begin{pmatrix} 3 & 1 \\ 2 & 4 \end{pmatrix}$$

Figure 14. The left matrix (A) defines the payoff for the row player, the right one (B) for the column player. The x-axis represents the probability with which the first players plays its first strategy, the y-axis represents the probability with which the second players plays its first strategy.

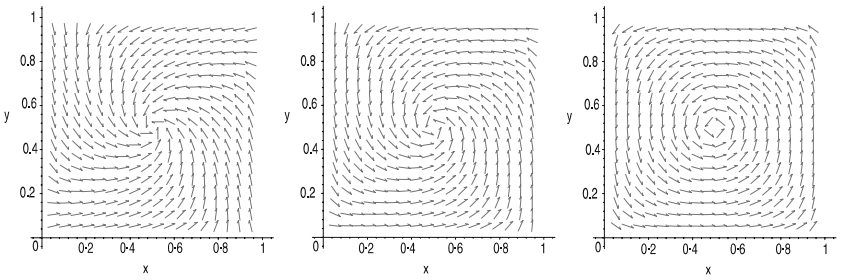

Figure 15. The direction field plots of subclass 3 with $\tau = 1, 2, 10$. The x-axis represents the probability with which the first players plays its first strategy, the y-axis represents the probability with which the second players plays its first strategy.

the mixed equilibrium is the reason why this equilibrium does not emerge from the learning process.

3.4.3. Subclass 3. The third class consists of games with a unique mixed equilibrium. For this category we used the game defined by the matrices in Figure 14. Typical for this class of games is that the interior trajectories define closed orbits around the equilibrium point. You can see this in Figure 15 for the RD and the Q-learning trajectories in Figure 16. For this experiment we used the same grid of starting points as we used in the previous two subclasses. In the last plot of Figure 16 we only used three starting points for reasons of clarity. Again in every point a learning path starts and converges to a particular point. If you compare the plots with the direction field plots for the same value of τ you can see that the sample paths of the learning process approximates the paths of the differential equations.

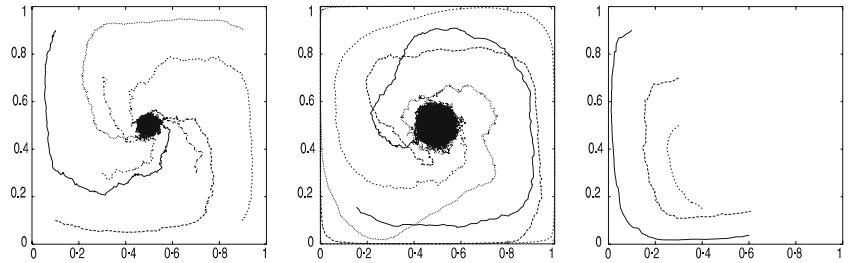

Figure 16. The Q-learning plots of subclass 3 with $\tau = 1, 2, 10$. The x-axis represents the probability with which the first player plays its first strategy, the y-axis represents the probability with which the second player plays its first strategy.

4. Applying the Q-learning dynamics to DG

In this section we apply the Q-learning dynamics to stochastic DG [12]. DG are of interest as they model natural problems in a number of different domains. Perhaps the most natural application is presented by the much studied load balancing problem [30, 49]. Load balancing is the problem of dividing the amount of work that a computer has to do between two or more computers so that more work gets done in the same amount of time and, in general, all users get served faster. Typically, load balancing is the main reason for computer server clustering. This problem can be modeled as a DG in which the agents are the users, the possible actions are the resources, and the equilibriums of the game are the outcomes in which agents are maximally dispersed. The DGs also lend themselves well to scaling MAS learning experiments. In Section 5, we present DG up to with 2500 agents. We here focus fist on the DG with two players.

In Figure17a a stochastic game is defined. This is the full DG for two agents where one agent is called the row player and the other the column player. The *i-th* row for a row player represents the choice for execution of task t_i, and similarly the i-th column for the column player. The highest possible global utility is achieved by the two players if (and only if) different tasks are chosen. Otherwise, the overly chosen task is randomly assigned to either the row or column player and only half of the full global utility is achieved in the DG.

The stochastic payoff Figure17a has four possible states that each represent one of the possible outcomes. These are summarized in payoff Figure 18:

For each of these different states we can now apply the RD Equations (11) and (12) by filling in the payoff tables for A and B for the row and column player. Doing this allows us to plot the direction field for each state. This can

(a)

0.5 | 1,0 **0.5** | 0,1	1,1
1,1	**0.5** | 1,0 **0.5** | 0,1

(b)

0.5	1,1
1,1	0.5

Figure 17. Example of a stochastic (multi-state) game.

state 1	state 2	state 3	state 4
1,0 | 1,1 1,1 | 1,0	0,1 | 1,1 1,1 | 1,0	0,1 | 1,1 1,1 | 0,1	1,0 | 1,1 1,1 | 0,1

Figure 18. The stochastic game of Figure 17 implicitly defines four states with a different payoff table.

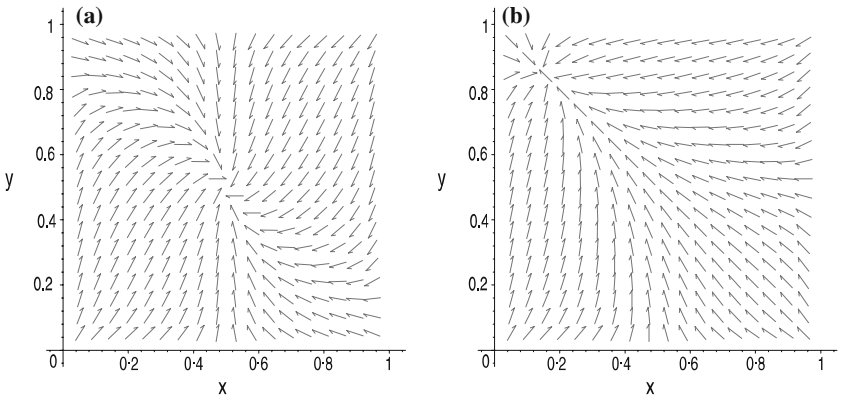

Figure 19. The direction field plots of the DG with $\tau = 2$ for the row player states 1 and 2.

be seen in Figures 19 and 20 for a temperature $\tau = 2$ and 10. The plots show the preferences of the choice of strategies for the row player for the representative states one and two of Figures 19, 20 respectively. The direction fields of the other two states (not shown) are the complement of the fields for states 1 and 2. The preference the row player has for choosing task t_1 is shown along

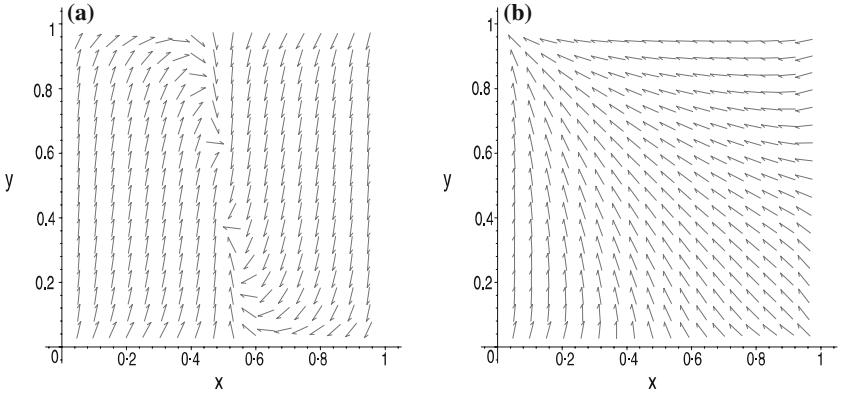

Figure 20. The direction field plots of the DG with $\tau = 10$ for the row player states 1 and 2.

the x axis and the preference the column player has for the same task is shown along the y axis. A point (p_1, p_2) in a graph is hence the joint probability for choosing task t_1 and it also determines the complement for choosing task $t_2(1 - p_1, 1 - p_2)$. The RD predict the behavior of both players.

In Figure 19a, for state 1, the row player receives a reward of 1 independently of the choice of task of the column player. There is hence no incentive for the row player to focus on one of the tasks. The resulting attractor in the RD is a mixed strategy where the choice of task execution is entirely random. If however there is a bias in payoff like in Figure 19b for state 2, there is a shift in bias of the preference of the row player. In the second case, task t_1 gives an average higher reward and the row player will converge to this task. Conversely, for the mirrored scenario of state 4 from table Figure 18. (not shown), the row player will converge to task t_2. These phenomena are more strongly pronounced if the temperature is increased as can be seen in Figure 20 for a temperature τ of 10 instead of 2. The RD hence give an indication of the strength of the attractors for the possible states and show what parameter setting to use to produce strong attractors.

Figure 17b gives the average expected payoff for each of the players joint actions. Figure 21 shows the RD of this averaged payoff for $\tau = 10$. This figure is equivalent to the RD of Figure 17a for equiprobable choice of states from Figure 18. The row player, and conversely also the column player, initially play an equilibrium strategy where both tasks are equally likely if the system is started with all preferences of all agents for all tasks identical, this is point $(0.5, 0.5)$ in Figure 21. A slight bias in preference by one of the agents for either task will however likely push the system away from this weak equilibrium as one of the agents, say the row player, chooses more than average one of the tasks, say t_1. This means that for the column player, any shift towards a preference for task t_2 will result in a strengthening of this shift. This is also evident from the RD in Figure 21. As the row player moves along one of the diagonals away from the mixed equilibrium strategy of $(0.5, 0.5)$, towards 0 or 1, the column player will have an incentive to move towards 1 or 0 respectively. More precisely, if the row player gets a bias towards task t_1, and thus towards a preference of 1 for task t_1, the column player will get a bias towards task t_2, and thus towards a preference of 0 for task t_1 (or a preference of 1 for task t_2) and the dynamics end in point $(1, 0)$ of Figure 21.

The DG for two agents hence has three NE $\{(1, 0), (0, 1)\}$, $\{(0, 1), (1, 0)\}$, and $\{(0.5, 0.5), (0.5, 0.5)\}$. However, only the first two are stable as can be seen from the plot. The mixed equilibrium is very unstable as any small perturbation away from this equilibrium will typically lead the dynamics to one of the two pure equilibriums.

In Figure 22 we plot a typical Q-learning process for the above game with the same settings as for the system of differential equations with $\tau = 10$. As starting points for the Q-learning process we chose a grid of 25 points.

These 25 representative points are again chosen so as to keep the plots clear and well-organized. In every point a learning path starts and converges to a particular point. If you compare the plots with the direction field plots for the value of $\tau = 10$ you can see that the sample paths of the learning process

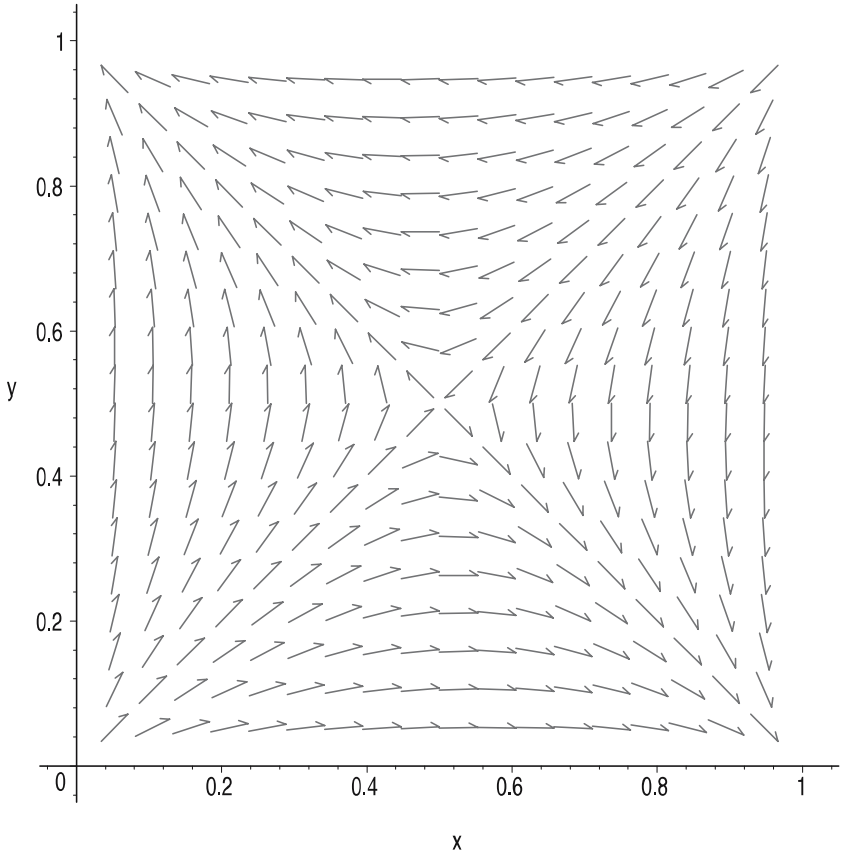

Figure 21. The direction field plots of the DG for the stochastic payoff.

approximates the paths of the RD in Figure 21. The instability of the mixed equilibrium is the reason why this equilibrium doesn't emerge from the sampled learning process.

5. COIN and ED

In this section we show that the explicit relation between RL and EGT can be beneficial for COIN. More precisely we present work on COIN used in cooperative MASs to induce incentives for actions by agents that promote the global utility of the system. This is in line with a growing collection of work where the goal is to establish conditions for MASs such that they are most likely to exhibit good emergent behavior [2, 4, 21]. The effectiveness of the COIN top-down approach has been demonstrated by applying the COIN framework to a number of example problems: network routing [54], increasingly difficult versions of the El Farol Bar problem [39], Braess' paradox [43], and complex token retrieval tasks [38].

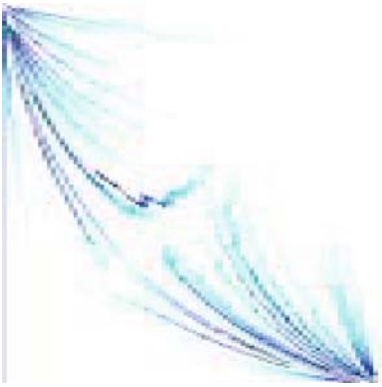

Figure 22. Trace of the Q-learner for the row player. The x-axis represents the probability with which the first player plays its first strategy, the y-axis represents the probability with which the second player plays its first strategy.

The COIN framework by Wolpert et al. defines how to engineer (or *modify*) the rewards an agents receives for its actions in *private utility functions*. Optimization of each agent's private utility here leads to increasingly effective emergent behavior of the collective, while discouraging agents from working at cross-purposes.

Translating the COIN approach to EGT is shown to give an indication through the attractors for RD as to what parameters to apply for COIN. Furthermore, the RD visualize the incentives in COIN for agents to strive for a high global utility using local learning.

For RL, we show that a MAS using (extensions of) COIN is able to efficiently solve the full DG with up to 2500 agents. The RD hence offer the possibility to analyze advanced MAS design methodologies. The above is also an engineering approach to induce ESS that meet a global fitness criteria for the population. This is a first example of engineering of an incentive for an optimal ESS in EGT.

In the next subsection we briefly summarize the COIN framework. Then we continue with the application of the ED to COIN.

5.1. COllective INtelligence

In this section, we briefly outline the theory of COIN as developed by Wolpert et al. [54, 55]. Broadly speaking, COIN defines the conditions that an agent's private utility function has to meet to increase the probability that learning to optimize this function leads to increased performance of the collective of agents, i.e. the *world utility*. Thus, the challenge is to define suitable private utility functions for the individual agents, given the performance of the collective.

In particular, the work by Wolpert et al. explores the conditions sufficient for effective emergent behavior for a collective of independent agents, each employing, for example, RL for optimizing their private utility. These conditions relate to (i) the learnability of the problem each agent faces, as obtained through each

individual agent's private utility function, (ii) the relative "alignment" of the agents' private utility functions with the utility function of the collective (the *world utility*), and lastly (iii) the learnability of the problem. Whereas the latter factor depends on the considered problem, the first two in COIN are translated into conditions on how to shape the private utility functions of the agents such that the world utility is increased when the agents improve their private utility.

Formally, let ζ be a vector representing the joint actions of all agents. A function $G(\zeta)$ provides the utility of the collective system, the *world utility*, for a given ζ. The goal is to find a ζ that maximizes $G(\zeta)$. Each individual agent μ has a private utility function g_μ that relates the reward obtained by the collective to the reward that the individual agent collects. Each agent will act such as to improve its own reward. The challenge of designing the collective system is to find private utility functions such that when individual agents optimize their payoff, this leads to increasing the world utility G, while the private function of each agent is at the same time also easily learnable. In this work, ζ represents the choice of which of the n tasks each of the n agent chooses to execute and the challenge is to find a private function for each agent such that optimizing the local payoffs optimizes the total task execution.

Following a mathematical description of this issue, Wolpert et al. propose the Wonderful Life Utility (WLU) as a private utility function that is both *learnable* and *aligned* with G, and that can also be easily calculated.

$$WLU_\mu(\zeta) = G(\zeta) - G(CL_{S_\mu^{\text{eff}}}(\zeta)). \tag{15}$$

The function $CL_{S_\mu^{\text{eff}}}(\zeta)$ as classically applied[8] "clamps" or suspends the choice of task by agent μ and returns the utility of the system without the effect of agent μ on the remaining agents $\hat{\mu}$ with which it possibly interacts. For our problem domain of DG, the clamped effect set are those agents $\hat{\mu}$ that are influenced in their utility by the choice of task of agent μ. Hence $\text{WLU}_\mu(\zeta)$ for agent μ is equal to the value of all the tasks executed by all the agents minus the value of the tasks executed by the other agents $\hat{\mu}$ if agent μ had not been in the system.

If agent μ picks a task T, which is not chosen by the other agents, then μ receives a reward of $V(T)$, where V assigns a value to a task T. If this task is however also chosen by any of the other agents, then the first term $G(\zeta)$ of Equation (15) is unchanged while the second term drops with the value of $V(T)$ as agent μ competes for completion of the task. Agent μ then receives a penalty $-V(T)$ for competing for a task targeted by one of the other agents $\hat{\mu}$. The WLU hence has a built in incentive for agents to find an unfulfilled task and hence for each agent to strive for a high global utility in its search for maximizing its own rewards.

5.2. *ED for COIN*

The full DG has $n!$ stable equilibriums where each of the n agents each fulfills one unique task. These NE are very robust as any deviation by one agent η immediately results in a drop in reward from 1 to an average of at most 0.5 as

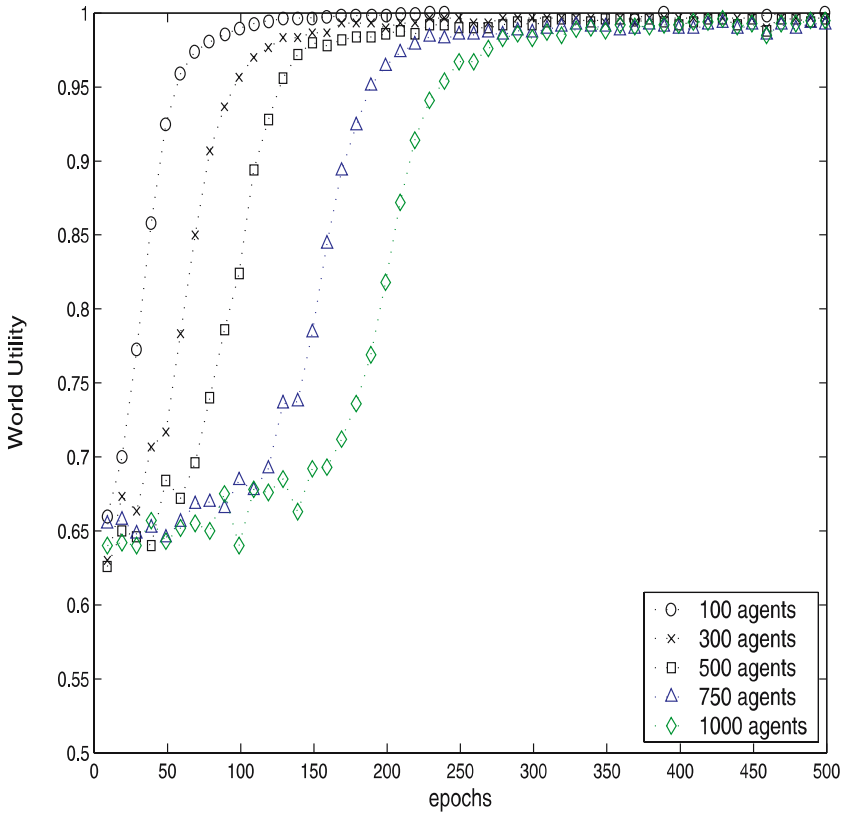

Figure 23. DG for the WLU.

it now competes with another agent for the same task. These $n!$ NE coincide with the states of the system that provide full utility. These states are unfortunately increasingly difficult to find in the RL exploration phase as the full state space of actions of the agents is n^n. Straightforward use of RL leads to a performance of ≈ 0.8 as agents prematurely converge to a subset of the tasks and ignore the remaining 20%. As can be seen in Figure 23, the COIN framework with the WLU is however able to efficiently solve the assignment problem where classical RL fails. The y-axis shows the utility of the system (1 represents execution of all tasks) and the x-axis shows the number of required epochs for the Q-learners.

In Figure 24a, we show the payoff table for the agents used in the case where agents in the MAS use an interpretation of the WLU. The Q-learner update rules of Section 2.3.4 are not changed and the agents still act as if they optimize their immediate (discounted) reward. An implementation of a distributed RL system can hence be reused.

To implement the WLU, a penalty must effectively be incurred if both agents choose the same task. This is realized for a 2 agent full DG by computing the payoff Figure 17a using Equation 15. Payoffs for the choice of an identical task

(a)

0.5 | -1,-1 **0.5** | -1,-1	1,1
1,1	**0.5** | -1,-1 **0.5** | -1,-1

(b)

0.5 | **1**,-1 **0.5** | -1,**1**	1,1
1,1	**0.5** | -1,**1** **0.5** | **1**,-1

Figure 24. Payoff tables for the WLU and WLUm.

by the agents in the system are no longer stochastic but are to be interpreted as a penalty. Behavior where agents do not interfere in the pursuit of global utility, for example the row agent chooses task t_1 and the column agent chooses task t_2, however receives the original full payoff. This approach allows for domain knowledge to be introduced into the system, much like the work of [13] allows for penalties for undesired actions in the coordination graphs. The WLU can also be computed as discussed in [38] without modifying the payoffs of the agents if this is too cumbersome or even possibly unknown and must be discovered by the MAS Figure 23.

The added value of the COIN framework can be explained by a closer study of the RD for two agents. In Figure 23 we show the Δ with respect to the original RD of the stochastic payoff of Figure 21. This vector field shows where the dynamics lead to and the size of the individual vectors show how large the differences are. The differences for just two agents are slight, but indicative of the increasing added value as the MAS is scaled. The penalties imposed by the WLU push the MAS away from states where agents choose to execute the same tasks. The RD indicate that then the agents are on route to desired, optimal and stable NE. Straightforward RL only improves marginally on the initial random behavior of the system as the number of considered agents is increased beyond 10.

Furthermore, the Δ of the WLU as compared to the standard RL-RD was found to increase as the temperature of the system was increased. This sensitivity analysis, along with the found RD and corresponding attractors give an indication of what settings for the Q-learners will lead to a MAS with a high global utility for the full DG. As such, the RD are a possible venue to investigate to acquire an indication of viable parameters for COIN and RL, in this case Q-learners.

To further improve on convergence of the COIN framework, we introduce a new extension of COIN based on the observation that the WLU as defined in Section 5.1 and as translated to the payoff matrix in Figure 24a is symmetric. If, for example, two agents a_1 and a_2 both choose task t_i, then both agents, according to Equation (15), receive a penalty when calculating $\mathrm{WLU}_{a_1}(\zeta)$ and $\mathrm{WLU}_{a_2}(\zeta)$ respectively. The intuition is that this however can lead to slower convergence of the MAS as a whole as *both* agents then may be pushed to target different tasks while only *one* of the agents need choose a different task. This slower convergence becomes more dramatic as more than one agent, say $l > 2$ agents, focuses on the same task and $l - 1$ "too many" agents need to switch. This phenomenon partially explains the trend in slower convergence of the WLU for an increasing number of agents in Figure 23. Agents for longer periods compete for tasks in their early

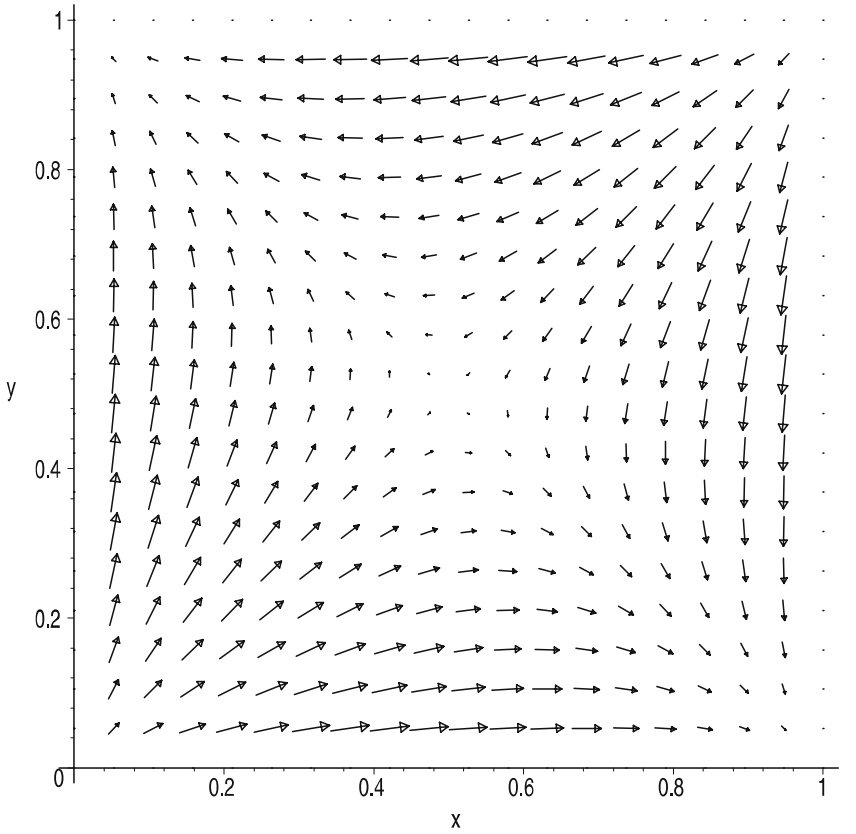

Figure 25. The ΔRD of the WLU.

exploratory behavior. The issued penalties force the agents to keep looking for a task for which they are the sole executor. But until an agent is successful in finding its own task, it can not yet push unsuccessful agents to unfulfilled tasks. Study of the preferences by agents for tasks show that the steep convergence curves of Figure 23 occur when a majority of the agents has converged on a task sufficiently to remain "determined" in their choice even under the influence of an occasional penalty as the remaining roaming agents in the system look for their own unique task (Figure 25).

We break the symmetry in the penalties by assigning the reward to the agent that is most likely *a priori* to choose task t. We reward the agent η with the highest Q-value for this task t and penalize all other agents that made the same choice for this same task (ties are broken at random). We name this adapted WLU the WLUm from most likely. In Figure 24b we give the modified payoff table for the case that agent 1 has a higher chance to pick task 1 and symmetrically agent 2 has a higher chance to pick task 2 than agent 1. Slight biases in the preference of the agents for tasks are quickly reinforced for fast convergence to final, good solutions. Figure 26 shows the RD for agent 1 for this case.

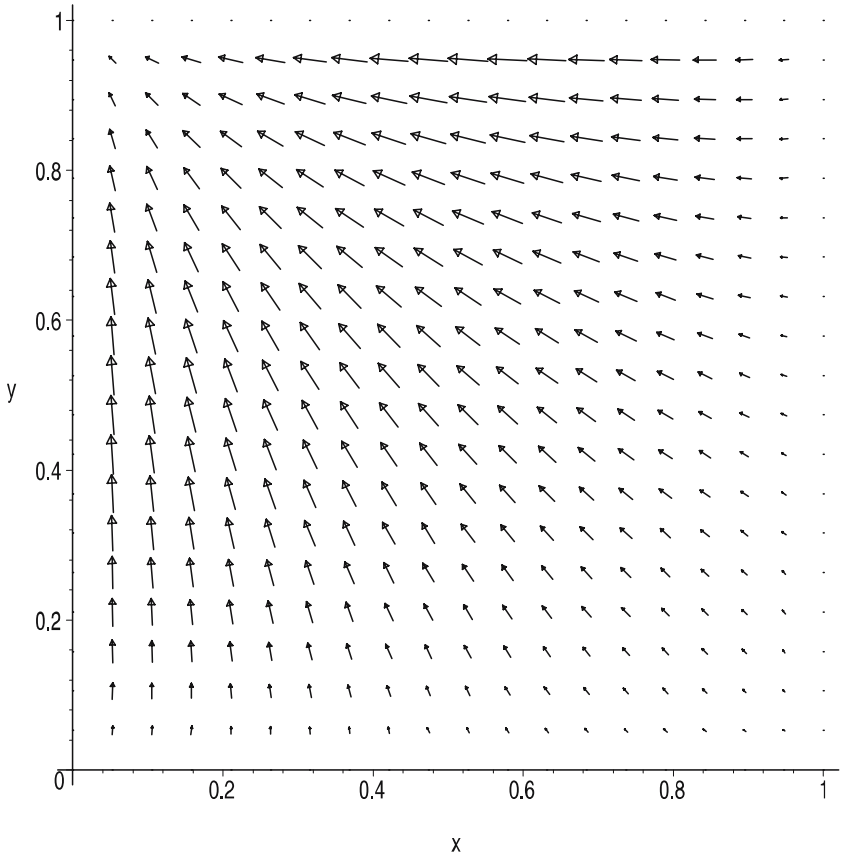

Figure 26. A RD of WLUm.

The WLUm in the case of more than two agents can be likewise represented or simply calculated. The WLUm for one agent a_i with choice of task t_j is calculated by counting the number of agents other than a_i that have chosen in one epoch to execute the same task t_j. If this number is zero, then the reward is 1. Otherwise, a penalty is incurred if a_i is not most likely to choose to execute this task from the set of contending agents, i.e. he does not have the highest Q-value for this task.

In Figure 27 we show typical results for the new WLUm for 2500 agents.[9] The WLUm converges dramatically faster than the classic WLU, especially for a large number of agents. Agents using the WLUm can most quickly converge to a task and drive other agents to choose another task. The mixed equilibrium where execution of any tasks by each of the agents is equiprobable, or a close approximation of this state, becomes extremely unstable using the WLUm. Any bias of an agent in its choice of tasks is quickly exploited.

Note that this adaption of the WLU's still only involves local use of information per task in the problem domain and no global information is used while the WLU and WLUm are competitive with the hand-tailored strategies of [12]. If the agents do not wish to even share information about their respective Q-values, assigning the

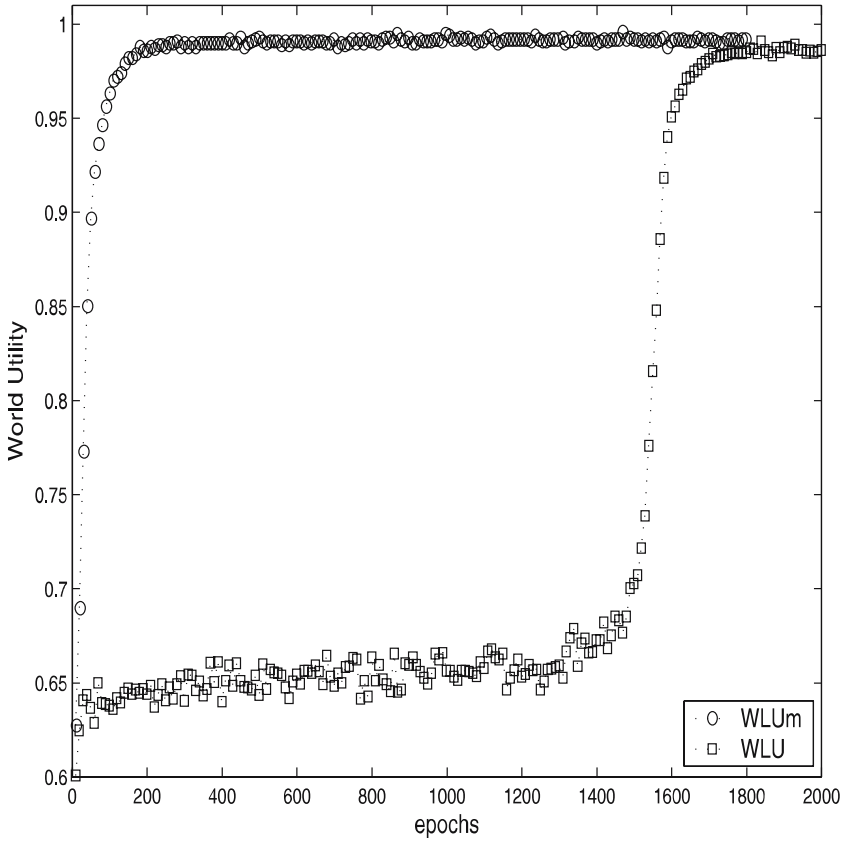

Figure 27. Improved convergence results WLUm for 2500 agents.

positive reward to a random competitor for a task was shown to have only a slightly worse performance to WLUm, and a similar analysis from the RD perspective.

To conclude this section we show the dynamics for the WLUm for three agents which have to choose from three tasks in Figure 28. This figure is a simplex plot of 3D data samples. Each 3D point (x_1, x_2, x_3), with x_i representing the probability with which agent a_i chooses task t_1, is transformed to a 2D (y_1, y_2) point in the simplex.[10] It becomes clear from Figure 28 that the mixed equilibrium, situated at the centroid, is very unstable and there is a push away from the center to the corners. Each corner represents one agent choosing task t_1 with probability 1 and the other two agents choosing this task with probability 0. To conclude we can derive from this plot that the WLUm handles the 3-agent 3-tasks situation very well as predicted by the RD.

6. Conclusion

In this work, we presented a novel approach for RL in MASs. Learning in MAS is still a difficult problem and a growing area of research. In this

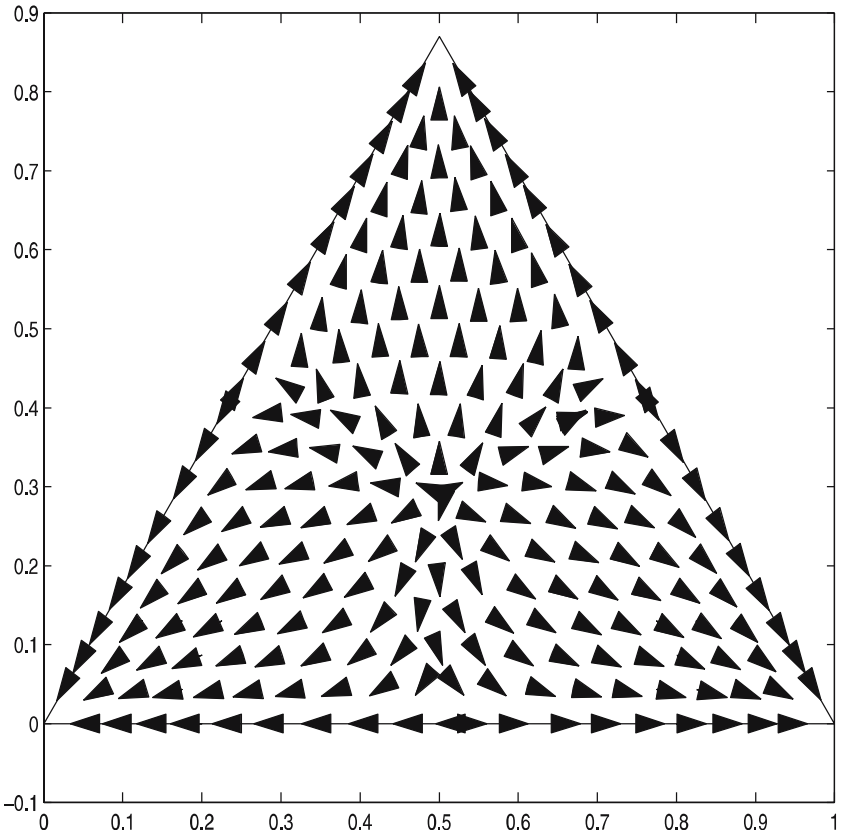

Figure 28. The direction field for three agents, three tasks for the WLUm.

paper we take an Evolutionary Game Theoretic point of view toward this problem.

In their work [5], Börgers and Sarin proved that in an appropriately constructed time limit, a simple RL model, i.e. the Cross learning model, converges to the RD from EGT. This result was the first step in describing the triangular relation between MASs, RL and EGT.

In this paper we extended these relations to a more general RL model, i.e. Q-learning. Moreover, we showed how the Evolutionary Game Theoretic approach opens a new perspective toward solving multi-agent RL problems.

We have argued in this work that this approach toward the multi-agent learning problem provides new insights towards understanding, analyzing and designing RL algorithms for MAS. These new insights can be summarized as follows:

– First of all we showed the clear similarities between EGT and MAS. EGT is a descriptive theory which starts from realistic views of the world. Players are not fully informed about the environment and the other players. Typical

for both fields is their uncertain character. Important to understand is that a MAS is very dynamic in nature. In contrast to classical GT, EGT is also dynamic. Different models exist in EGT to describe dynamical processes. In this work we consider the RD. These clear similarities gave a strong indication that EGT can contribute to learning in MAS.
– Secondly, we showed that the formal relation between EGT and RL could be extended from Cross learning to LA and Q-learning. This link shows us how learning in a MAS can be understood. More precisely, it gives us a nice tool to visualize the dynamics of the learning process and to show us where strong attractors as NE and ESS lie. In this manner, the RD can guide us in correctly fine tuning the parameters of the learning process. Analysis of the ESS and attractors of the derived RD from the RL application predict the impact of the parameters on the learning trajectories of the RL algorithms.
– Finally, we showed the added value of the COIN framework of Wolpert et al. by visualizing the incentives provided in COIN towards cooperative behavior. The link between EGT and RL is therefore also potentially applicable to analyze advanced MAS RL design methodologies.

The above work has presented a formal link between the RD of EGT and a specific form of Q-learning. Recently, [32] has formulated new criteria for evaluating (RL) multi-agent learning algorithms. The authors extensively empirically test their newly developed algorithm against a suite of opponents, and for a large number of different types of games using the GAMUT [28] framework. A major extension of this paper would be a systematic derivation of the RD equations of the used algorithms of [32] as done in Section 3. This would allow for a more analytical study of the used algorithms and their relative performance for the testbed games of GAMUT.

Notes

1. We do not intend to ignore previous game theoretic results as for instance the theorem of Zermelo which asserts that chess is strictly determined. We only state that this is the first book under the name GT assembling many different results.
2. A strategy is dominant if it is always better than any other strategy, regardless of what the opponent may do.
3. Local is defined in the sense of minimal perturbations.
4. Note that we can find examples which try to overcome the drawbacks of the joint action approach [9, 18 22, 27]. There has also been quite some effort to extend these RL techniques to Partially Observable Markovian decision problems and non-Markovian settings [20].
5. The reader who is interested in the complete derivation, we refer to [44]
6. Note that in the literature the temperature parameter τ appears both in the nominator as in the denominator.
7. The main reason for this is to keep the plots clear and well-organized.
8. Ongoing work investigates more general clamping functions.
9. We did not explore settings with more agents due to memory restrictions with the current implementation.
10. To do this we need a matrix A so that $y = A.x$. This is straightforward to solve.

References

1. B. Banerjee and J. Peng, "Adaptive poilcy gradient in multiagent learning," in *Proceedings of the Third International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2003.
2. A. Barto and S. Mahadevan, "Recent advances in hierarchical reinforcement learning," *Discrete-Event Syst. J.* vol. 13, pp. 41–77, 2003.
3. A. L. C. Bazzan, *A game-theoretic approach to coordination of traffic signal agents*, PhD thesis, University of Karlsruhe, 1997.
4. R. Becker, S. Zilberstein, V. Lesser, and C. V. Goldman, "Transition independent decentralized Markov decision problem," in *Proceedings of the Third International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2003.
5. T. Börgers and R. Sarin, "Reinforcement and replicator dynamics," *J. Econ. Theory*, vol. 77, no.1, pp. 1–14, 1997.
6. R. Boyd and P. J. Richerson, *Culture and the Evolutionary Process*, The University of Chicago Press, 1985.
7. R. R. Bush and F. Mosteller, "A Mathematical Model for Simple Learning," *The Psychol. Rev.* vol. 58, pp. 15–18, 1951.
8. R. R. Bush and F. Mosteller, *Stochastic Models for Learning*, Wiley: New York, 1955.
9. C. Claus and G. Boutilier, "The Dynamics of Reinforcement Learning in Cooperative Multi-Agent Systems," in *Proceedings of the 15th International Conference on Artificial Intelligence*, pp. 746–752, 1998.
10. J. G. Cross, "A stochastic learning model of economic behaviour," *Quart. J. Econ.*, vol. 87, no.5, pp. 239–266, 1973.
11. C. M. Gintis, *Game Theory Evolving*, Princeton University Press, 2000.
12. T. Grenager, and R. Powers, and Y. Shoham, "Dispersion games: general definitions and some specific learning results," in *Proceedings of the Eighteenth National Conference on Artificial Intelligence AAAI 02*, 2002.
13. C. Guestrin, D. Koller, C. Gearhart, and N. Kanodia, "Generalizing plans to new environments in relational MDPs," in *International Joint Conference on Artificial Intelligence (IJCAI-03)*, 2003.
14. M. W. Hrisch and S. Smale, *Differential Equation, Dynamical Systems and Linear Algebra*, Academic Press, Inc 1974.
15. J. Hofbauer and K. Sigmund, *Evolutionary Games and Population Dynamics*, Cambridge University Press, 1998.
16. J. Hu and M. P. Wellman, "Multiagent reinforcement learning in stochastic games," in *Internal Report from the Laboratory for Information and Decision Systems and the Operation Research Center*, 1999.
17. P. Huang and K. Sycara, "Multi-agent Learning in Extensive Games with complete information," in *Proceedings of the Third International Conference on Autonomous Agents and Multi-agent Systems (AAMAS)*, 2003.
18. C. Jafari, A. Greenwald, D. Gondek, and G. Ercal, "On no-regret learning fictitious play and nash equilibrium," in *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*, Cambridge University Press, pp. 223–226, 2001.
19. H. Jung and M. Tambe, "Performance model for large scale multiagent systems" in *Proceedings of the Third International Conference Autonomous Agents and Multiagent Systems (AAMAS)*, 2003.
20. L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement Learning: A Survey," *J. Artif. Intell. Res.* vol. 4, pp. 237–285, 1996.
21. M. Lauer and M. Riedmiller, "An algorithm for distributed reinforcement learning in cooperative multi-agent systems," in *Proc. 17th International Conf. on Machine Learning* Morgan Kaufmann: San Francisco, CA, pp. 535–542, 2000.
22. M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Proceedings of the Eleventh International Conference on Machine Learning*," Cambridge University Press, pp. 157–163, 1994.
23. J. Maynard Smith, *Evolution and the Theory of the Games*, Cambridge University Press, 1982.

24. J. Maynard Smith, and G. R. Price, "The logic of animal conflict," *Nature*, vol. 146, no. 2, pp. 15–18, 1973.
25. K. Narendra and M. Thathachar, "Learning automata: A survey," *IEEE Trans. Syst. Man Cybernet*. vol. 14, no.5, pp. 323–334, 1974.
26. K. Narendra and M. Thathachar, *Learning Automata: An Introduction*, Prentice-Hall, 1989.
27. A. Nowé, J. Parent, and K. Verbeeck, "Social agents playing a periodical policy," in *Proceedings of the 12th European Conference on Machine Learning*, Volume 2176 of *Lecture Notes in Artificial Intelligence*, Springer, pp. 382–393, 2001.
28. E. Nudelman, J. Wortman, K. Leyton-Brown, and Y. Shoham, "Run the GAMUT: A comprehensive approach to evaluating game-theoretic algorithms, algorithms," in *Proceedings of the Fourth International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2004.
29. M. J. Osborne and A. Rubinstein, *A Coruse in Game Theory*, MIT Press, 1994.
30. J. Parent, K. Verbeeck, A. Nowé, K. Steenhaut, J. Lemeire, and E. Dirkx, "Adaptive load balancing of parallel applications with social reinforement learning on heterogeneous sysems," *J. Sci. Program*. 2004. to appear.
31. S. Phelps, S. Parsons, and P. McBurney, "An evolutionary game-theoretic comparison of two double-action market designs," in *Workshop on Agent Medicated Electronic commerce VI: Theories for Engineering of Distributed Mechanisms and Systems (AMEC'04)*, Volume 2531 of *Lecture Notes in Artificial Intelligence*, Springer, pp. 109–118, 2004.
32. R. Powers and Y. Shoham, "New criteria and a new algorithm for learning in multi-agent system," in *Proceedings of Eighteenth Annual Conference on Neural Information Processing Systems (NIPS)*, 2004.
33. F.V. Redondo, *Game Theory and Economics*, Cambridge University Press, 2001.
34. L. Samuelson, *Evolutionary Games and Equilibrium Selection*, MIT Press: Cambridge, MA, 1997.
35. T. D. Schneider, "Evolution of biological information," *J. Nucl. Acid Res*. vol. 28, no. 14, pp. 2794–2799, 2000.
36. D. Stauffer, *Life Love and Death: Models of Biological Reproduction and Aging*, Institute for Theoretical physics: Köln Euroland, 1999.
37. R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction.* MIT Press: Cambridge MA, 1998.
38. P. J. 't Hoen and S. M. Bohte, "COllective INtelligence with sequence of actions," in *14th European conference on Machine Learning*, Volume 2837 of *Lecture Notes in Articifical Intelligence*, Springer, 2003.
39. P. J. 't Hoen and S. M. Bohte, "COllective INtelligence with task assignment," in *proceedings of the Workshop on Collectives and the Design of Complex Systems (CDOCS03), forthcoming. Also available as Technical Rapport SEN-E0315, Lecture Notes in Artificial Intelligence*, Springer, 2003.
40. P. J. 't Hoen and K. Tuyls, "Analyzing multi-agent reinforcement learning using evolutionary dynamics," in *Proceedings of the 15th European Conference on Machine Learning (ECML), Lecture Notes in Artificial Intelligence*, Springer, 2004.
41. P. S. Sastry and M. A. L. Thathacher, "Varieties of Learning Automata: An Overview," *IEEE Trans. Sys. Man Cybernet*, vol. 32, no. 6, pp. 323–334, 2002.
42. J. N. Tsitsiklis, "Asynchronous stochastic approximation and Q-learning," *Machine Learn*, vol. 16, pp. 185–202, 1994.
43. K. Tumer and D. Wolpert, "COllective INtelligence and Braess' Paradox," in *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, Austin, pp. 104–109, August, 2000.
44. K. Tuyls, *Learning in Multi-Agent Systems, An Evolutionary Game Theoretic Approach*, Ph.D. dissertation, Computational Modeling Lab, Vrije Universiteit Brussel, Belgium, 2004.
45. K. Tuyls, D. Heytens, A. Nowé, and B. Manderick, "Extended Replicator Dynamics as a Key to Reinforcement Learning in Multi-Agent Systems," in *Proceedings of the 14th European Conference on Machine Learning (ECML)*, Volume 2837, of *Lecture Notes in Artificial Intelligence*, Springer, 2003.
46. K. Tuyls, T. Lenaerts, K. Verbeeck, S. Maes and B. Manderick, "Towards a relation between learning agents and evolutionary dynamics", in *Proceedings of the Belgian-Dutch Conference on Artificial Intelligence (BNAIC 2002)*, Cambridge University Press, pp. 223–226, 2002.

47. K. Tuyls, A. Nowe, T. Lenaerts, and B. Manderick, "An evolutionary game theoretic perspective on learning in multi-agent systems," in *Synthese, Section Knowledge, Rationality and Action*, Kluwer Academic Publishers, 2004, vol. 139, no. 2, pp. 297–330.
48. K. Tuyls, K. Verbeeck, T. Lenaerts, "A Selection-Mutation model for Q-learning in Multi-Agent Systems," in *Proceedings of the Third Interational conference on Autonomous Agents and Multi-agent Systems (AAMAS)*, The ACM International Conference Proceedings Series, 2003.
49. K. Verbeeck, A. Nowé, and J. Parent, "Homo egualis reinforcement learning agents for load balancing," in *Proceedings of the 1st NASA Workshop on Radical Agent Concepts*, Volume 2564 of *Lecture Notes in Artificial Intelligence*, Springer, pp. 109–118, 2002.
50. J. von Neumann and O. Morgenstern, *Theory of Games and Economic Behavior*, Princeton University Press, 1944.
51. W. E. Walsh, R. Das, G. Tesauro, and J. O. Kephart, "Analyzing complex strategic interactions in multi-agent games," in *Proceedings of the The Eighteenth National Conference on Artificial Intelligence (AAAI-02) Workshop on Game Theoretic and Decision Theoretic Agents, Lecture Notes in Artificial Intelligence*, Springer, pp. 109–118, 2002.
52. C. Watkins and P. Dayan, "Q-learning", *Machine Learn.*, vol. 8, pp. 279–292, 1992.
53. J. W. Weibull, *Evolutionary Game Theory*, MIT Press, 1996.
54. David H. Wolpert, Kagan Tumer, and Jeremy Frank, "Using COllective INtelligence to route internet traffic," in *Advances in Neural Information Processing Systems-11*, Denver, pp. 952–958, 1998.
55. David H. Wolpert, Kevin R. Wheler, and Kagan Tumer, "General Principles of learning-based multi-agent systems", in Oren Etzioni and Jörg P. Müller and Jeffrey M. Bradshaw (ed.), *Proceedings of the Third International Conference on Autonomous Agents (Agents'99)*, ACM Press: Seattle, WA, USA, pp. 77–83, 1999.