ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/220254307

Evolutionary game theory and multi-agent reinforcement learning

Article *in* The Knowledge Engineering Review · March 2005

DOI: 10.1017/S026988890500041X · Source: DBLP

CITATIONS
63

READS
1,277

2 authors:

Karl Tuyls
DeepMind and University of Liverpool
277 PUBLICATIONS **2,237** CITATIONS
SEE PROFILE

Ann Nowe
Vrije Universiteit Brussel
374 PUBLICATIONS **2,293** CITATIONS
SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Project OPTIMIZATION OF HEAVILY CONSTRAINED HYBRID-FLEXIBLE FLOWSHOP PROBLEMS USING A MULTI-AGENT REINFORCEMENT LEARNING APPROACH View project

Project Game Theory View project

All content following this page was uploaded by Ann Nowe on 19 May 2014.

The user has requested enhancement of the downloaded file.

Evolutionary Game Theory and Multi-Agent Reinforcement Learning

Karl Tuyls[1][1] and Ann Nowé[2]

[1] *Theoretical Computer Science Group, Hasselt University, Diepenbeek, Belgium*
E-mail: karl.tuyls@uhasselt.be or k.tuyls@cs.unimaas.nl
[2] *Computational Modeling Lab, Vrije Universiteit Brussel, Brussels, Belgium*
E-mail: asnowe@info.vub.ac.be

Abstract

In this paper we survey the basics of Reinforcement Learning and (Evolutionary) Game Theory, applied to the field of Multi-Agent Systems. This paper contains three parts. We start with an overview on the fundamentals of Reinforcement Learning. Next we summarize the most important aspects of Evolutionary Game Theory. Finally, we discuss the state-of-the-art of Multi-Agent Reinforcement Learning and the mathematical connection with Evolutionary Game Theory.

1 Introduction

In this paper we describe the basics of Reinforcement Learning and Evolutionary Game Theory, applied to the field of Multi-Agent Systems. The uncertainty inherent to the Multi-Agent environment implies that an agent needs to learn from, and adapt to, this environment to be successful. Indeed, it is impossible to foresee all situations an agent can encounter beforehand. Therefore, learning and adaptiveness become crucial for the successful application of Multi-agent systems to contemporary technological challenges as for instance routing in telecom, e-commerce, robocup, etc. Reinforcement Learning (RL) is already an established and profound theoretical framework for learning in stand-alone or single-agent systems. Yet, extending RL to multi-agent systems (MAS) does not guarantee the same theoretical grounding. As long as the environment an agent is experiencing is Markov[2], and the agent can experiment enough, RL guarantees convergence to the optimal strategy. In a MAS however, the reinforcement an agent receives, may depend on the actions taken by the other agents present in the system. Hence, the Markov property no longer holds. And as such, guarantees of convergence do no longer hold.

In the light of the above problem it is important to fully understand the dynamics of reinforcement learning and the effect of exploration in MAS. For this aim we review Evolutionary Game Theory (EGT) as a solid basis for understanding learning and constructing new learning algorithms. The Replicator Equations will appear to be an interesting model to study learning in various settings. This model consists of a system of differential equations describing how a population (or a probability distribution) of strategies evolves over time, and plays a central role in biological and economical models.

In Section 2 we summarize the fundamentals of Reinforcement Learning. More precisely, we discuss policy and value iteration methods, RL as a stochastic approximation technique and some

[1]Note that as of October 1st 2005, the first author will move to the University of Maastricht, Institute for Knowledge and Agent Technology (IKAT), The Netherlands. His corresponding adress will change into k.tuyls@cs.unimaas.nl

[2]The Markov property states that only the present state is relevant for the future behavior of the learning process. Knowledge of the history of the process does not add any new information.

convergence issues. We also discuss distributed RL in this section. Next we discuss basic concepts of traditional and evolutionary game theory in Section 3. We provide definitions and examples of the most basic concepts as Nash equilibrium, Pareto optimality, Evolutionary Stable Strategies and the Replicator Equations. We also discuss the relationship between EGT and RL. Section 4 is dedicated to Multi-Agent Reinforcement Learning. We discus some possible approaches, their advantages and limitations. More precisely, we will describe the joint action space approach, independent learners, informed agents and an EGT approach. Finally, we conclude in Section 5.

2 Fundamentals of Reinforcement Learning

Reinforcement learning (RL) finds its roots in animal learning. It is well known that we can teach an animal to respond in a desired way by rewarding and punishing it appropriately. For example we can train a dog to detect drugs in people's luggage at customs by rewarding it each time it responds correctly and punishing it otherwise. Based on this external feedback signal the dog adapts to the desired behavior. More general, 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, reinforcement learning 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 become not 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. Originally, RL was considered to be single agent learning. All events the agent has no control over, are considered to be part of the environment. In this section we consider the single agent setting, in Section 4 we discuss different approaches to multi-agent RL.

2.1 RL and it's relationship to Dynamic Programming

From a mathematical point of view RL is closely related to Dynamic Programming (DP). DP is a well known method to solve Markovian Decision Problems (MDP) [Ber76]. An MDP is a multi-stage decision problem for which an optimal policy must be found, i.e. a policy that optimizes the expected long term reward. Usually the expected discounted cumulative return is considered. However other measures do exist [Bel62]. The Markovian property assures that the learner can behave optimal observing it's current state only, i.e. there is no need to keep track of the history, so the learner does not need to know how it got there. The DP techniques are usually classified into two approaches: the policy iteration approach and the value iteration approach. The same classification is used in RL, and each RL approach can be viewed as an asynchronous, model free approach of it's DP counter part. Before we present the two RL classes, we briefly introduce the DP counter parts. DP is a model based approach, so it assumes that a model of the environment is available. In general the environment is stochastic, and it's response is described by a transition matrix. This matrix gives the probability with which the next state s_{t+1} will be reached and a reward r_{t+1} will be received, given the current state is s_t, and the action taken is a_t. This is represented by

$$P^a_{ss'} = P\{s_{t+1} = s' | s_t = s, a_t = a\} \tag{1}$$

which are called the transition probabilities. Now, given any state and action s and a, together with any next state s', the expected value of the next reward is,

$$R^a_{ss'} = E\{r_{t+1} | s_t = s, a_t = a, s_{t+1} = s'\} \tag{2}$$

Important here to note is that we assume that the Markov property is valid. This allows us to determine the optimal action based on the observation of the current state only. Below we introduce the two approaches in DP: policy iteration and value iteration, and introduce their RL counter parts.

2.1.1 Policy iteration in DP

The policy iteration approach considers a current policy π, and tries to locally improve the policy based on the state-values that correspond to the current policy π. More formally

$$\begin{aligned} V^{\pi}(s) &= E_{\pi}\{r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + ...|s_t = s\} && (3) \\ &= E_{\pi}\{r_{t+1} + \gamma V^{\pi}(s_{t+1})|s_t = s\} && (4) \end{aligned}$$

It is well known that the $V^*_\pi(s)$, with π^* the optimal policy, are the solutions of the Bellman optimality equation given below:

$$V^*(s) = max_a \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V^*(s')] \tag{5}$$

The $V_\pi(s)$ can be calculated using successive approximation as follows:

$$\begin{aligned} V_{k+1}(s) &= E_{\pi}\{r_{t+1} + \gamma V_{k+1}(s_{t+1})|s_t = s\} && (6) \\ &= \sum_a \pi(s,a) \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V_k(s')] && (7) \end{aligned}$$

To locally improve the policy in a given state s, the best action a is looked for based on the current state values $V_k(s)$. So π is improved in state s, by updating $\pi(s)$ into the action that maximises the right hand side of equation 7, yielding an updated policy π. The policy iteration algorithm is given below:

Algorithm 1 Policy Iteration

```
Choose a policy $\pi'$ arbitrarily
- loop
-    $\pi := \pi'$
-    Compute the value function $V$ of policy $\pi$:
-          solve the linear equations:
-             $V^{\pi}(s) = \sum_{s'} P^{\pi(s)}_{ss'}[R^{\pi(s)}_{ss'} + \gamma V^{\pi}(s')]$
-    Now improve the policy at each state:
(I)           $\pi'(s) = argmax_a \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V^{\pi}(s')]$
- until $\pi = \pi'$
```

2.1.2 Policy iteration in RL

Because for a reinforcement learning approach we usually assume that the model is not available, we cannot improve the policy locally using equation (I) from Algorithm 1. Instead Policy iteration RL techniques build up their own internal evaluator or critic. Based on this internal critic the appropriateness of an action for a state is evaluated. An architecture realising this type of learning is given in Figure 1.

It was first proposed in [Bar83] and generalised in [And87]. Since then it has been adopted by many others. The scheme in Figure 1 contains two units, the evaluation unit and the action unit. The former is the internal evaluator, while the latter is responsible for determining the actions which look most promising according to the internal evaluation. The evaluation unit yields an estimate of the current $V_\pi(s)$ values. In [Bar83] this component is called the adaptive critic element. Based on the external reinforcement signal r_t, and its own prediction, the evaluation

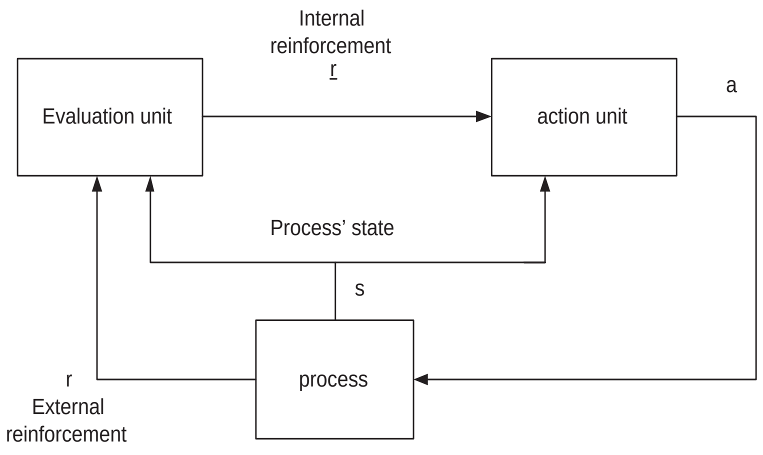

Figure 1 An architecture for policy iteration reinforcement learning.

unit adjusts its prediction on-line as follows:

$$V(s_t) = V(s_t) + \zeta(r_t + \gamma V(s_{t+1}) - V(s_t)) \tag{8}$$

Where ζ is a positive constant determining the rate of change. This updating rule is the so called temporal difference, TD(0), method of [Sut88]. As stated above, the goal of the evaluation unit is to transform the environmental reinforcement signal r into a more informative internal signal $\underline{r}$. To generate the internal reinforcement, differences in the predictions between two successive states are used. If the process moves from a state with a prediction of lower reinforcement into a state with a prediction of higher reinforcement, the internal reinforcement signal will reward the action that caused the move. In [Bar83] it is proposed to use the following internal reinforcement signal:

$$\underline{r}(t) = r(t) + \gamma V(s_{t+1}) - V(s_t) \tag{9}$$

Given the current state, the action unit produces the action that will be applied next. Many different approaches exist for implementing this action unit. If the action unit contains a mapping from states to actions, the action that will be applied to the system can be generated by a two step process. In the first step the most promising action is generated, this is that action to which the state is mapped. This action is then modified by means of a stochastic modifier S. This second step is necessary to allow exploration of alternative actions. Actions that are "close" to the action that was generated in the first step, are more likely to be the outcome of this second step. This approach is often used if the action set is a continuum. If an action that was applied to the system turned out to be better than expected, i.e. the internal reinforcement signal is positive, then the mapping will be "shifted" towards this action. If the action set is discrete, a table representation can be used. Then the table maps states to probabilities of actions to be selected for a particular state, and the probabilities are updated directly. Below we discuss a simple mechanism to update these probabilities.

2.1.3 Learning Automata

Learning Automata have their origins in the research labs of the former USSR. More precisely it started with the work of Tsetlin in the 1960s [Tse62, Tse73].

In those early days, Learning Automata were deterministic and based on complete knowledge of the environment. Later developments came up with uncertainties in the system and the environment and lead to the stochastic automaton. More precisely, the stochastic automaton tries to provide a solution for the reinforcement learning problem without having any information on the optimal action initially. It starts with equal probabilities on all actions and during the learning process these probabilities are updated based on responses from the environment. We consider LA to be a method for solving RL problems in a policy iterative fashion.

The term Learning Automaton was introduced for the first time in the work of Narendra and Thathacher in 1974 [Nar74]. Since then there has been a lot of development in the field and a number of survey papers and books on this topic have been published: to cite a few [Tha02, Nar89, Nar74].

In Figure 2 a Learning Automaton is illustrated in its most general form.

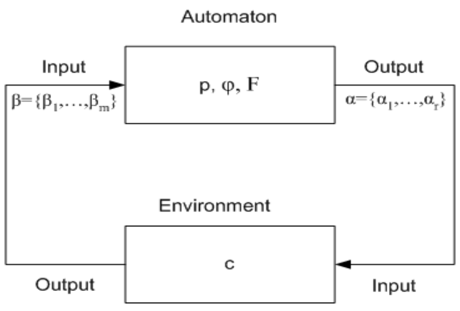
Automaton
Input
$\beta=\{\beta_1,\ldots,\beta_m\}$
p, φ, F
Output
$\alpha=\{\alpha_1,\ldots,\alpha_r\}$
Environment
c
Output
Input

Figure 2 The feedback connection of the Learning Automaton - Environment pair.

The automaton tries to determine an optimal action out of a set of possible actions to perform.

Let us now first zoom in to the environment part of Figure 2. This part is illustrated in Figure 3. The environment responds to the input action α by producing an output β. The output also belongs to a set of possible outcomes, i.e. $\{0, 1\}$, which is probabilistically related to the set of inputs through the environment vector c.

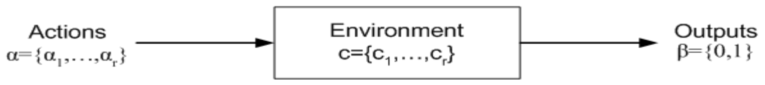

Figure 3 Zooming in to the environment of the Learning Automaton

The environment is represented by a triple $\{\alpha, c, \beta\}$, where α represents a finite action set, β represents the response set of the environment, and c is a vector of penalty probabilities, where each component c_i corresponds to an action α_i.

The response β from the environment can take on 2 values β_1 or β_2. Often they are chosen to be 0 and 1, where 1 is associated with a penalty response (a failure) and 0 with a reward (a success).

Now, the penalty probabilities c_i can be defined as

$$c_i = P(\beta(n) = 1|\alpha(n) = \alpha_i) \tag{10}$$

Consequently, c_i is the probability that action α_i will result in a penalty response. If these probabilities are constant, the environment is called *stationary*.

Several models are recognized by the response set of the environment. Models in which the response β can only take 2 values are called P-models. Models which allow a finite number of values in the fixed interval $[0, 1]$ are called Q-models. When β is a continuous random variable in the fixed interval $[0, 1]$, the model is called S-model.

Now we considered the environment of the LA model of Figure 2, we will now zoom in at the automaton itself of Figure 2. More precisely, Figure 4 illustrates this.

The automaton is represented by a set of states $\phi = \{\phi_1, ..., \phi_s\}$. As opposed to the environment, β becomes the input and α the output. This implicitly defines a function $F : \phi \times \beta \rightarrow \phi$, mapping the current state and input into the next state, and a function $H : \phi \times \beta \rightarrow \alpha$, mapping the current state and current input into the current output. In this text we will use p as the

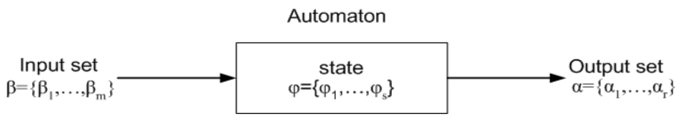

Figure 4 Zooming in to the automaton part of the Learning Automaton

probability vector over the possible actions of the automaton which corresponds to the function H.

Summarizing, this brings us to the definition of a Learning Automaton. More precisely, it is defined by a quintuple $\{\alpha, \beta, F, p, T\}$ for which α is the action or output set $\{\alpha_1, \alpha_2, \dots \alpha_r\}$ of the automaton , β is a random variable in the interval $[0, 1]$, F is the state transition function, 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 $c_i = P[\beta \mid \alpha_i]$, $i = 1 \dots r$ over the actions.

The automaton can be either stochastic or deterministic, the former's output function H being composed of probabilities based on the environment's response, whilst the latter having a fixed mapping function between the internal state and the function to be performed.

Further sub-division of classification occurs when considering the transition or updating function F which determines the next state of the automaton given its current state and the response from the environment. If this is fixed then the automaton is a *fixed structure deterministic* or a *fixed structure stochastic* automaton.

However if the updating function is variable, allowing for the transition function to be modified so that choosing the operations or actions changes after each iteration, then the automaton is a *variable structure deterministic* or a *variable structure stochastic* automaton. In this paper we are mainly concerned with the variable structure stochastic automata, which have the potential of greater flexibility and therefore performance. Such an automaton A at timestep t is defined as:

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

where we have an action set α with r actions, an environment response set β and a probability set p containing r probabilities, each being the probability of performing every action possible in the current internal automaton state. 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.

The most important 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:

$$\begin{aligned} p_i(t+1) &\leftarrow p_i(t) + a(1-\beta(t))(1-p_i(t)) - b\beta(t)p_i(t) && (11)\\ & \quad if\ \alpha_i\ is\ the\ action\ taken\ at\ time\ t \\ p_j(t+1) &\leftarrow p_j(t) - a(1-\beta(t))p_j(t) + b\beta(t)[(r-1)^{-1} - p_j(t)] && (12)\\ & \quad if\ \alpha_j \neq \alpha_i \end{aligned}$$

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 fully 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 [Nar89]. LA belong to the Policy Iteration (PI) approach since action probabilities are updated directly. In the Value Iteration (VI) approach the quality of an action is determined, and the action with the highest quality is part of the optimal policy.

2.1.4 Value iteration in DP

The value iteration approach turns the Bellman equation (equation 5) into an update rule. Because the $V^*(s)$ are the unknowns, an estimate of these values is used at the right hand side, these estimates $V_k(s)$ are iteratively updated using the update rule:

$$\begin{aligned} V_{k+1}(s) &= max_a E\{r_{t+1} + \gamma V_k(s_{t+1}) | s_t = s, a_t = a\} && (13) \\ &= max_a \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V_k(s')] && (14) \end{aligned}$$

Since for all states $k \to V_k(s)$ is a contraction mapping, the $V_k(s)$ values converge in the limit to the optimal values $V^*(s)$. In practice the updating is stopped, when the changes become very small, and the corresponding optimal policy doesn't changes any more. Since value iteration in DP assumes a transition model of the system is available, the optimal policy π^* can be obtained using the equation below:

$$\pi^*(s) = argmax_a \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V(s')] \qquad (15)$$

2.1.5 Value iteration in RL

The best known counter part of value iteration of DP in RL is Q-learning [Wat92]. Since RL is model-free the optimal policy can not be retrieved from equation 15. Therefore, Q-learning stores explicitly the **Q**uality of each action for each state. These values are called Q-values, denoted by $Q^*(s,a)$. The relationship between the $V^*(s)$ values and the $Q^*(s,a)$ values is:

$$V^*(s) = max_a Q^*(s,a) \qquad (16)$$

The $Q^*(s,a)$ are equal to the expected return of taking action a in state s, and from then on behaving according to the optimal policy π^*, i.e.

$$Q^*(s,a) = \sum_{s'} P^a_{ss'} E[R^a_{ss'} + \gamma V^*(s')] \qquad (17)$$

and therefore we also have that

$$Q^*(s,a) = \sum_{s'} P^a_{ss'} E[R^a_{ss'} + \gamma max_a Q^*(s',a)] \qquad (18)$$

The same way as in value iteration of DP, the Q-values are iteratively updated. But since in RL we don't have a model of the environment, we don't know the $P^a_{ss'}$, nor the $E[R^a_{ss'}]$, therefore stochastic approximation is used, yielding the well known Q-learning updating rule:

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r + \gamma \max_{a'} Q(s',a') - Q(s,a)) \qquad (19)$$

where α is a learning factor.

2.2 Some convergence issues

Not all RL techniques come with a proof of convergence. Especially the policy iteration approaches often lack such a proof. The Learning Automata, introduced above as an example of one stage policy iteration, do have a proof of convergence. A single LA that uses the reward-inaction updating scheme is guaranteed to converge [Nar89], the same is true for a set of independent LA (see Section 4).

The value iteration approach, Q-learning is also proved to converge if applied in a Markovian environment, and provided some very reasonable assumptions apply such as appropriate settings for α (see Section 2.2.2). The Markovian property is really crucial, as soon as this not longer holds, the guarantee for convergence is lost. This however does not mean that RL can not be applied in non-Markovian environments but care has to be taken.

2.2.1 Partially Observable Markov Decision Processes

RL has been successfully applied to Partially Observable MDP's (POMDP's). Here the states are only partially observable such that the learner can not distinguish sufficiently between the states to let the Markovian property hold. This might e.g. be because a continuous state problem is translated into a discrete problem, and the discretization is too coarse. A denser discretization can restore the Markovian property, but leads to an increased size of the state space. Another reason to not have the Markovian property is because an agent might miss a particular component of the state description, due to the lack of a sensor that can measure that component. E.g. if temperature is a critical component to have a Markovian view on the environment, and the agent cannot measure this, the environment looks non-Markovian to the agent. Or if the agent only has a local view of the environment, like a mobile robot, then it is possible that two different locations result in the same sensor input to the agent. A technique that is often used to tackle the non-Markovianism is to guide the exploration. If the agent can actively take part in it's exploration, the exploration can be steered such that e.g. two states that are indistinguishable, become distinguishable because they result in different consequences when the same action is applied, by doing so a limited history is introduced. Since we do not believe that this kind of guided exploration is the way to go to solve the non-Markovian environments where MAS operate in, we do not go in to more detail in this issue here. For more details see [Per02, Cas94, Kae96].

2.2.2 The convergence of Q-learning

Amongst others, Tsitsiklis [Tsi93] has proved that under very reasonable assumptions Q-learning is guaranteed to converge to the optimal policy. The proof of Tsitsiklis is very interesting because it considers Q-learning as a stochastic approximation technique, and can be adapted to prove all kinds of variants to Q-learning. See e.g. the distributed version in the next section. We will not discuss the proof in detail here, but it is worthwhile to have a look at the assumptions that have an impact on the setting of the learning process. A first assumption says that the learning parameter in the Q-learning updating rule (i.e. α in Section 2.1.5) must be decreased in time such that, $\sum_{t=0}^{\infty} \alpha_{(s,a)}(t) = \infty$ and $\sum_{t=0}^{\infty} \alpha_{(s,a)}^2(t) < \infty$. This allows that Q-values are updated in an asynchronous way.

Another assumption states that it is no problem that past experience is used to guide the exploration, so the agent can actively take part in the exploration.

A last interesting assumption is that the agent can use outdated information as long as old information is eventually discarded, i.e. in the updating rule of Q-learning, the update of the $Q_{k+1}(s,a)$ values can be done based on $Q_{k-i}(s,a)$ values, with i between 0 and k. It becomes clear that Q-learning can be considered as a stochastic approximation method by rewriting the Q-learning update rule as follows:

$$Q_{k+1}(s,a) = Q_k(s,a) + \alpha_{(s,a)}(k)[Q_k(s,a)(Q_k) - Q_k(s,a) + W_k(s,a)] \qquad (20)$$

With $Q_k(s,a)(Q_k) = E[r(s,a) + \gamma \sum_{s'} P^a_{ss'} max_{a'} Q_k(s',a')]$ and (Q_k) is the vector containing all $Q_k(s,a)$ values. $\alpha_{(s,a)}(k) = 0$ if $Q_{k+1}(s,a)$ is not updated at time step $k+1$, otherwise $\alpha_{(s,a)}(k) \in]0,1]$ obeying the restrictions stated above.

And

$$W_k(s,a) = (r_k(s,a) + \gamma max_{a'} Q_k(s',a')) - E[r(s,a) + \gamma \sum_{s'} P^a_{ss'} max_{a'} Q_k(s',a')] \tag{21}$$

Q is a contraction mapping, meaning that it is monotonously converging, and W is proved to behave as white noise [Tsi93].

In the next section we briefly introduce a first MAS setting of RL, however we prefer to refer to it as a simple distributed approach.

2.3 Distributed RL

Since the proof of Tsitsiklis allows that Q-values are updated asynchronously and based on outdated information, it is rather straightforward to come up with a parallel or distributed version of Q-learning.

Assume we subdivide the state space in different regions. In each region an agent gets the responsibility of updating the corresponding Q-values by exploring his region. Agents can explore their own region, and make updates in their copy of the table of the Q-values to the Q-values that belong to their own region. As long as they make transitions in their own region, they can apply the usual updating rule. If they make however a transition to another region, with Q-values that belong to the responsibility of another agent, they should not directly communicate to that other agent and ask for the particular Q-value, but use the information they have in their own copy table, i.e. use out-dated information, and steer the exploration so as to get back to their own region. Since out-date information needs to be updated from time to time, the agents should communicate from time to time and distribute the Q-values of which they have the responsibility. Since we do not put the Markovian property in to danger by this approach, the prove of Tsitsiklis can be applied, and convergence is still assured. While this approach can be considered as a MAS, we prefer to refer to it as a distributed or parallel version of Q-learning. The approach has been successfully applied to the problem of Call Admission Control in telecommunications [Ste97][3].

3 Evolutionary Game Theory

3.1 Introduction

Originally, Game Theory was launched by John von Neumann and Oskar Morgenstern in 1944 in their book *Theory of Games and Economic Behavior* [Neu44][4].

Game theory 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 and the corresponding preferences. It is the 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 act strategically, in other words, they take into account the relation between their own

[3]In this paper we focus on genuine model-free RL. However many RL techniques incorporate domain knowledge. If this knowledge is available then it is a good idea to use it one way or the other. The nice thing about RL, is that if the domain knowledge seems not to be perfect or even totally incorrect, still the RL techniques will converge, at the end, to the optimal policy. The incorporation of domain knowledge can e.g. be done by initialing the Q-values based on the knowledge, or by hypothetical moves, so updates are based on the model.

[4]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 Game Theory assembling many different results.

choices and the decisions of other agents. Different economical situations lead to different rational strategies for the players involved.

When John Nash discovered the theory of games at Princeton, in the late 1940's and early 1950's, the impact was enormous. The impact of the developments in Game Theory expressed itself especially in the field of economics, where its concepts played an important role in for instance the study of international trade, bargaining, the economics of information and the organization of corporations. But also in other disciplines such as social and natural sciences the importance of Game Theory became clear, examples are: studies of legislative institutions, voting behavior, warfare, international conflicts, and evolutionary biology.

However, von Neumann and Morgenstern had only managed to define an equilibrium concept for 2-person zero-sum games. Zero-sum games correspond to situations of pure competition, whatever one player wins, must be lost by another. John Nash addressed the case of competition with mutual gain by defining best-reply functions and using Kakutani's fixed point-theorem[5]. The main results of his work were the development of the *Nash Equilibrium* and the *Nash Bargaining Solution* concept.
Despite the great usefulness of the Nash equilibrium concept, the assumptions traditional game theory make, like hyper rational players that correctly anticipate the other players in an equilibrium, made game theory stagnate for quite some time [Wei96, Gin00, Sam97]. A lot of refinements of Nash equilibria came along (for instance *trembling hand perfection*), which made it hard to choose the appropriate equilibrium in a particular situation. Almost any Nash equilibrium 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 make decisions under hyper-rationality assumptions.

This is where evolutionary game theory originated. More precisely, John Maynard Smith adopted the idea of evolution from biology [May73, May82]. He applied Game Theory (GT) to Biology, which made him relax some of the premises of 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 core of evolution in Biology.
These new ideas led Smith and Price to the concept of Evolutionary Stable Strategies (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, such as 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.
Evolutionary Game Theory offers a solid basis for rational decision making in an uncertain world, it describes how individuals make decisions and interact in complex environments in the real world. Modeling learning agents in the context of Multi-agent Systems requires insight in the type and form of interactions with the environment and other agents in the system. Usually, these agents are modeled similar to the different players in a standard game theoretical model. In other words, these agents assume complete knowledge of the environment, have the ability to correctly anticipate the opposing player (hyper-rationality) and know that the optimal strategy in the environment is always the same (static Nash equilibrium). The intuition that in the real world people are not completely knowledgeable and hyper-rational players and that an equilibrium can change dynamically led to the development of evolutionary game theory.

Before introducing the most elementary concepts from (Evolutionary) Game Theory we summarize some well known examples of strategic interaction in the next section.

[5]Kakutani's fixpoint theorem goes as follows. Consider X a nonempty set and F a point-to-set map from X to subsets of X. Now, if F is continuous, X is compact and convex, and for each x in X, $F(x)$ is nonempty and convex, F has a fixed point. Applying this theorem (and thus checking its conditions) to the best response function proves the existence of a Nash equilibrium.

3.2 Examples of Strategic interaction

3.2.1 The Prisoner's Dilemma

As the first example of a strategic game we consider the prisoner's dilemma game [Gin00, Wei96].

In this game, 2 prisoners, who committed a crime together, have a choice to either collaborate with the police (to defect) or work together and deny everything (to cooperate). If the first criminal (row player) defects and the second one cooperates, the first one gets off the hook (expressed by a maximum reward of 5) and the second one gets the most severe punishment (reward 0). If they both defect, they get the second most severe punishment one can get (expressed by a payoff of 1). If both cooperate, they both get a small punishment (reward 3).

The rewards are summarized in payoff Tables 1 and 2. The first table has to be read from a row perspective and the second one from a column perspective. For instance if the row player chooses to Defect (D), the payoff has to be read from the first row. It then depends on the strategy of the column player what payoff the row player will receive.

A=

D	1	5
C	0	3

Table 1 Matrix (A) defines the payoff for the row player for the Prisoner's dilemma. Strategy D is Defect and strategy C is Cooperate.

B=

D	C
1	0
5	3

Table 2 Matrix (B) defines the payoff for the column player for the Prisoner's dilemma. Strategy D is Defect and strategy C is Cooperate.

3.2.2 The Battle of the Sexes

The second example of a strategic game we consider is the battle of the sexes game [Gin00, Wei96].

In this game, a married couple loves each other so much they want to do everything together. One evening the husband wants to see a football game and the wife wants to go to the opera. If they both choose their own preference and do their activities separately they receive the lowest payoff. This situation is described by the payoff matrices of Tables 3 and 4.

A=

F	2	0
O	0	1

Table 3 Matrix (A) defines the payoff for the row player for the Battle of the sexes. Strategy F is choosing Football and strategy O is choosing the Opera.

B=

F	O
1	0
0	2

Table 4 Matrix (B) defines the payoff for the column player for Battle of the sexes. Strategy F is choosing Football and strategy O is choosing the Opera.

3.2.3 Matching Pennies

The third example of a strategic game we consider is the matching pennies game. [Gin00, Wei96].

In this game two children hold both a pennie and independently choose which side of the coin to show (Head or Tails). The first child wins if both coins show the same side, otherwise child 2 wins. This is an example of a zero-sum game as can be seen from the payoff Tables 5 and 6. Whatever is lost by one player, must be won by the other player.

A=

H	1	-1
T	-1	1

Table 5 Matrix (A) defines the payoff for the row player for the matching pennies game. Strategy H is playing Head and strategy T is playing Tail.

B=

H	T
-1	1
1	-1

Table 6 Matrix (B) defines the payoff for the column player for the matching pennies game. Strategy H is playing Head and strategy T is playing Tail.

3.3 Elementary concepts

In this section we review the key concepts of GT and EGT and its mutual relationships. We start by defining strategic games and concepts as Nash equilibrium, Pareto optimality and evolutionary stable strategies. Then we discuss the relationships between these concepts and provide some examples.

3.3.1 Strategic games

An n-player normal form games models 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 without being informed about the choices of the others. For reasons of simplicity, we limit the pure strategy set of the players to 2 strategies. 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 $x_1 + x_2 = 1$.

Defining a game more formally we restrict ourselves to the 2-player 2-action game. Nevertheless, an extension to n-players n-actions games is straightforward, but examples in the n-player case do not show the same illustrative strength as in the 2-player case. 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 $P_1 : S_1 \times S_2 \rightarrow \Re$ and $P_2 : S_1 \times S_2 \rightarrow \Re$ are defined by the payoff matrices, A for the first player and B for the second player, see Table 7. 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 is usually classified in three subclasses, as follows [Red01],

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

Table 7 The left matrix (A) defines the payoff for the row player, the right matrix (B) defines the payoff for the column player

Subclass 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 2 players has a dominant strategy, therefore there is just 1 strict equilibrium.

Subclass 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 2 pure equilibria and 1 mixed equilibrium.

Subclass 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 1 mixed equilibrium.

The first subclass includes those type of games where each player has a dominant strategy[6], as for instance the prisoner's dilemma. 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). But both players receive the highest payoff by both playing their first or second strategy. This is expressed in the condition $(a_{11} - a_{21})(b_{11} - b_{12}) > 0$. 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 (e.g. matching pennies game). This is expressed by the condition $(a_{11} - a_{21})(b_{11} - b_{12}) < 0$.

3.3.2 Nash equilibrium

In traditional game theory it is assumed that the players are hyperrational, 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 Nash equilibrium 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 Nash equilibrium is defined as follows. When 2 players play the strategy profile $s = (s_i, s_j)$ belonging to the product set $S_1 \times S_2$ then s is a Nash equilibrium if $P_1(s_i, s_j) \geq P_1(s_x, s_j)$ $\forall x \in \{1, ..., n\}$ and $P_2(s_i, s_j) \geq P_2(s_i, s_x)$ $\forall x \in \{1, ..., m\}$ [7].

3.3.3 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 at least one agent is strictly better off.

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

Another related concept is that of Pareto Dominance: An outcome of a game is Pareto dominated if some other outcome would make at least one player better off without hurting any other player. That is, some other outcome is weakly preferred by all players and strictly preferred by at least one player. If an outcome is not Pareto dominated by any other, than it is Pareto optimal.

3.3.4 Evolutionary Stable Strategies

The core equilibrium concept of Evolutionary Game Theory is that of an Evolutionary Stable Strategy (ESS). The idea of an evolutionarily stable strategy was introduced by John Maynard Smith and Price in 1973 [May73]. 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

[6]A strategy is dominant if it is always better than any other strategy, regardless of what the opponent may do.
[7]For a definition in terms of best reply or best response functions we refer the reader to [Wei96].

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 strategy is evolutionary stable against this new appearing strategy. More generally, we say a strategy is an Evolutionary Stable strategy 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 is programmed to play 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, chances that he is playing against a mutant are ϵ and against a non-mutant are $1-\epsilon$. The expected payoff for the first player, being a non mutant is:

$$P(s,(1-\epsilon)s+\epsilon s^{'}) = (1-\epsilon)P(s,s)+\epsilon p(s,s^{'})$$

and being a mutant is,

$$P(s^{'},(1-\epsilon)s+\epsilon s^{'})$$

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 $\forall\,\epsilon :\ 0<\epsilon<\delta$,

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

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

3.3.5 The relation between Nash equilibria and ESS

This section explains how the core equilibria concepts from classical and evolutionary game theory relate to one another. The set of Evolutionary Stable Strategies for a particular game are contained in the set of Nash Equilibria for that same game,

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

The condition for an ESS is more strict than the Nash condition. Intuitively this can be understood as follows: as defined above a Nash equilibrium 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, and as such 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.

$$P(s_2 s_1) > P(s_1,s_1)$$

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

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

Another important property for an ESS is the following. If s_1 is ESS and s_2 is an alternative best reply to s_1, then s_1 has to be a better reply to s_2 than s_2 to itself. This can easily be seen as follows, because s_1 is ESS, we have for all s_2

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

i.e.

$$P(s_2,s_2) = P(s_1,s_2)$$

If s_2 does as well against itself as s_1 does, then s_2 earns at least as much against $(1-\epsilon)s_1+\epsilon s_2$ as s_1 and then s_1 is no longer evolutionary stable. To summarize we now have the following 2 properties for an ESS s_1,

1. $P(s_2,s_1) \leq P(s_1,s_1)\ \ \forall\ s_2$
2. $P(s_2,s_1) = P(s_1,s_1) \Longrightarrow P(s_2,s_2) < P(s_1,s_2)\ \ \forall\ s_2 \neq s_1$

3.3.6 Examples

In this section we provide an example for each class of game described in Section 3.3.1) and illustrate the Nash equilibrium concept and Evolutionary Stable Strategy concept as well as Pareto optimality.

For the first subclass we consider the prisoner's dilemma game. The strategic setup of this game has been explained in Section 3.2. The payoffs of the game are repeated in table 8. As one can see both players have one dominant strategy, more precisely *defect.*

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

Table 8 Prisoner's dilemma: The left matrix (A) defines the payoff for the row player, the right one (B) for the column player.

For both players, defecting is the dominant strategy and therefore always the best reply toward any strategy of the opponent. So the Nash equilibrium in this game is for both players to defect. Let's now determine whether this equilibrium is also an evolutionary stable strategy. Suppose $\epsilon \in [0, 1]$ is the number of cooperators in the population. The expected payoff of a cooperator is $3\epsilon + (1 - 0\epsilon)$ and that of a defector is $5\epsilon + (1 - 1\epsilon)$. Since for all ϵ,

$$5\epsilon + 1(1 - \epsilon) > 3\epsilon + 0(1 - \epsilon)$$

defect is an ESS. So the number of defectors will always increase and the population will eventually only consist of defectors. In Section 3.4 this dynamical process will be illustrated by the replicator equations.

This equilibrium which is both Nash and ESS, is not a Pareto optimal solution. This can be easily seen if we look at the payoff tables. The combination ($defect, defect$) yields a payoff of $(1, 1)$, which is a smaller payoff for both players than the combination (*cooperate*, *cooperate*) which yields a payoff of $(3, 3)$. Moreover the combination (*cooperate*, *cooperate*) is a Pareto optimal solution. However, if we apply the definition of Pareto optimality, then also ($defect$, *cooperate*) and (*cooperate*, $defect$) are Pareto optimal. But both these Pareto optimal solutions do not Pareto dominate the Nash equilibrium and therefore are not of interest to us. The combination (*cooperate*, *cooperate*) is a Pareto optimal solution which Pareto dominates the Nash equilibrium.

For the second subclass we considered the battle of the sexes game [Gin00, Wei96]. In this game there are 2 pure strategy Nash equilibria, i.e. ($football$, $football$) and (*opera*, *opera*), which both are also evolutionary stable (as demonstrated in Section 3.4.4). There is also 1 mixed nash equilibrium, i.e. where the row player (the husband) plays $football$ with 2/3 probability and *opera* with 1/3 probability and the column player (the wife) plays *opera* with 2/3 probability and $football$ with 1/3 probability. However, this equilibrium is not an evolutionary stable one.

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

Table 9 Battle of the sexes: The left matrix (A) defines the payoff for the row player, the right one (B) for the column player.

The third class consists of the games with a unique mixed equilibrium $((1/2, 1/2), (1/2, 1/2))$. For this category we used the game defined by the matrices in Table 10, i.e. maching pennies . This equilibrium is not an evolutionary stable one. Typical for this class of games is that the interior trajectories define closed orbits around the equilibrium point.

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

Table 10 The left matrix (A) defines the payoff for the row player, the right one (B) for the column player.

3.4 Population Dynamics

In this section we discuss the Replicator Dynamics in a single and a multi population setting. We discuss the relation with concepts as Nash equilibrium and ESS and illustrate the described ideas with some examples.

3.4.1 Single Population Replicator Dynamics

The basic concepts and techniques developed in EGT were initially formulated in the context of evolutionary biology. [May82, Wei96, Sam97]. In this context, the strategies of all the players are genetically encoded (called genotype). Each genotype refers to a particular behavior which is used to calculate the payoff of the player. The payoff of each player's genotype is determined by the frequency of other player types in the environment.

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. 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 replicator dynamics 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) 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:

$$\frac{dx_i}{dt} = [(A\mathbf{x})_i - \mathbf{x} \cdot A\mathbf{x}]x_i \tag{22}$$

In equation (22), 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 ($\mathbf{x}$) can be described as a probability vector $\mathbf{x} = (x_1, x_2, ..., x_J)$ which expresses the different densities of all the different types of replicators in the population. Hence $(A\mathbf{x})_i$ is the payoff 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 $\frac{\frac{dx_i}{dt}}{x_i}$ 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 [Wei96, Hof98].

3.4.2 Multi-Population Replicator Dynamics

So far the study of population dynamics was limited to a single population. However in many situations interaction takes place between 2 or more individuals from different populations. In this section we study this situation in the 2-player multi-population case for reasons of simplicity. 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 (22) would result again. Player R has a probability vector p over its possible strategies and player C a probability vector q over its strategies.

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

$$\frac{dp_i}{dt} = [(A\mathbf{q})_i - \mathbf{p} \cdot A\mathbf{q}]p_i \tag{23}$$

$$\frac{dq_i}{dt} = [(B\mathbf{p})_i - \mathbf{q} \cdot B\mathbf{p}]q_i \tag{24}$$

As can be seen in equation (23) and (24), 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) are used for the two different players.

3.4.3 Relating Nash, ESS and the RD

As being a system of differential equations, the RD have some rest points or equilibria. An interesting question is how these RD-equilibria relate to the concepts of Nash equilibria and ESS. We briefly summarize some known results from the EGT literature [Wei96, Gin00, Osb94, Hof98, Red01]. An important result is that every Nash equilibrium is an equilibrium of the RD. But the opposite is not true. This can be easily understood as follows. Let us consider the vector space or simplex of mixed strategies determined by all pure strategies. Formally the unit simplex is defined by,

$$\Delta = \{x \in \Re^m_+ : \sum_{i=1}^{m} x_i = 1\}$$

where x is a mixed strategy in m-dimensional space (there are m pure strategies), and x_i is the probability with which strategy s_i is played. Calculating the RD for the unit vectors of this space (putting all the weight on a particular pure strategy), yields zero. This is simply due to the properties of the simplex Δ, where the sum of all population shares remains equal to 1 and no population share can ever turn negative. So, if all pure strategies are present in the population at any time, then they always have been and always will be present, and if a pure strategy is absent from the population at any time, then it always has been and always will be absent[8]. So, this means that the pure strategies are rest points of the RD, but depending on the structure of the game which is played these pure strategies do not need to be a Nash equilibrium. Hence not every rest point of the RD is a Nash equilibrium. So the concept of dynamic equilibrium or stationarity alone is not enough to have a better understanding of the RD.
For this reason the criterion of asymptotic stability came along, where you have some kind of local test of dynamic robustness. Local in the sense of minimal perturbations. For a formal definition of asymptotic stability, we refer to [Hir74]. Here we give an intuitive definition. An equilibrium is asymptotic stable if the following two conditions hold:

- Any solution path of the RD that starts 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 (i.e. being robust to local perturbations) then it is a Nash equilibrium. For a proof, the reader is referred to [Red01]. An interesting result due to Sigmund ans Hofbauer [Hof98] 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 [Hof98, Red01]. 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 equilibria from the RD that show dynamic robustness.

[8]Of course a solution orbit can evolve toward the boundary of the simplex as time goes to infinity, and thus in the limit, when the distance to the boundary goes to zero, a pure strategy can disappear from the population of strategies. For a more formal explanation, we refer the reader to [Wei96]

3.4.4 Examples

In this section we continue with the examples of Section 3.2 and the classification of games of Section 3.3.1. We start over with the Prisoner's Dilemma game (PD). In Figure 5 we plotted the direction field of the replicator equations applied to the PD. A Direction field is a very elegant and excellent tool to understand and illustrate a system of differential equations. The direction fields presented here consist of a grid of arrows tangential to the solution curves of the system. Its a graphical illustration of the vector field indicating the direction of the movement at every point of the grid in the state space. Filling in the parameters for each game in equations 23 and 24, allowed us to plot this field.

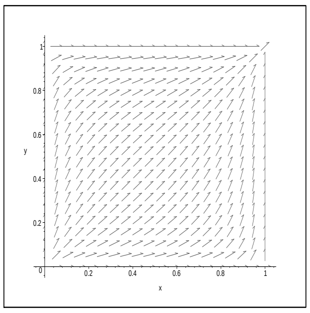

Figure 5 The direction field of the RD of the prisoner's dilemma using payoff Table 8.

The x-axis represents the probability with which the first player will play defect and the y-axis represents the probability with which the second player will play defect. So the Nash equilibrium and the ESS lie at coordinates $(1, 1)$. As you can see from the field plot all the movement goes toward this equilibrium.

Figure 6 illustrates the direction field diagram for the battle of the sexes game. As you may recall from Section 3.3.6 this game has 2 pure Nash equilibria and 1 mixed Nash equilibrium. These equilibria can be seen in the figure at coordinates $(0, 0), (1, 1), (2/3, 1/3)$. The 2 pure equilibria are ESS as well. This is also easy to verify from the plot, more precisely, any small perturbation away from the equilibrium is led back to the equilibrium by the dynamics.
The mixed equilibrium, which is Nash, is not an asymptotic stable strategy, which is obvious from the plot. From Section 3.3.6, we can now also conclude that this equilibrium is not evolutionary stable either.

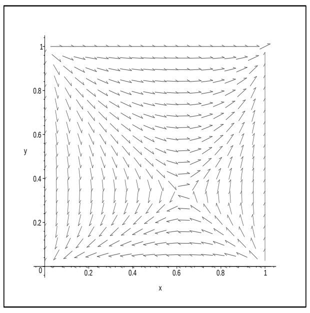

Figure 6 The direction field of the RD of the Battle of the sexes game using payoff Table 9.

3.5 The role of EGT in MAS

In this section we discuss the most interesting properties that link the fields of EGT and MAS. These properties make clear that there exists an important role for EGT in MAS.

Traditional Game theory 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 and the corresponding preferences. It is the 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,
2. agents act strategically, in other words, they take into account the relation between their own choices and the decisions of other agents.

Typical for the traditional game theoretic approach is to assume perfectly rational players (or hyperrationality) who try to find the most rational strategy to play. These players have a perfect knowledge of the environment and the payoff tables and they try to maximize their individual payoff. These assumptions made by classical game theory just do not apply to the real world and Multi-Agent settings in particular.

In contrast, EGT is descriptive and starts from more realistic views of the game and its players. A game is not played only once, but repeatedly with changing opponents. Moreover, the players are not completely informed, sometimes misinterpret each others' actions, and are not completely rational but also biologically and sociologically conditioned. Under these 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. For this learning process, mathematical models are developed, e.g. replicator equations.

Summarizing the above we can say that EGT treats agents' objectives as a matter of fact, not logic, with a presumption that these objectives must be compatible with an appropriately evolutionary dynamic [Gin00]. Evolutionary models do not predict self-interested behaviour. It describes how agents can make decisions in complex environments, in which they interact with other agents. In such complex environments software agents must be able to learn from their environment and adapt to its non-stationarity.

The basic properties of a Multi-Agent System correspond exactly with that of EGT. First of all, a MAS is made up of interactions between two or more agents, who each try to accomplish a certain, possibly conflicting, goal. No agent has the guarantee to be completely informed about the other agents intentions or goals, nor has it the guarantee to be completely informed about the complete state of the environment. Of great importance is that EGT offers us a solid basis to understand dynamic iterative situations in the context of strategic games. A MAS has a typical dynamical character, which makes it hard to model and brings along a lot of uncertainty. At this stage EGT seems to offer us a helping hand in understanding this typical dynamical processes in a MAS and modeling them in simple settings as iterative games of two or more players.

4 Multi-Agent Reinforcement Learning

Coordination is an important problem in multi-agent reinforcement learning research (MARL). For cooperative multi-agent environments several algorithms and techniques exist. However they are usually defined for one state problems or games, only few of them are suited for the multi-stage case, moreover restrictions on the structure of the multi-stage case are imposed. Below we give an overview of the most important one stage techniques, and the results of a comparative study where we tested the techniques on a variety of settings, even for situations for which some of the algorithms were originally not developed.

We shed light on some useful characteristics and strengths of each algorithm studied. For learning in a multi-agent system, two extreme approaches can be recognized. On the one hand, the presence of other agents, who are possibly influencing the effects a single agent experiences, can be completely ignored. Thus a single agent is learning as if the other agents are not around.

On the other hand, the presence of other agents can be modeled explicitly. This results in a joint action space approach which recently received quite a lot of attention [Cla98, Hu99, Lit94].

4.1 The Joint Action Space Approach

In the joint action space technique, learning happens in the product space of the set of states S, and the collections of action sets $A_1, ...A_n$ (one set for every agent). The state transition function $T : S \times A_1 \times \ldots \times A_n \rightarrow P(S)$ maps a state and an action from each agent onto a probability distribution on S and each agent receives an associated reward, defined by the reward function R_i : $S \times A_1 \times \ldots \times A_n \rightarrow P(\Re)$. This is the underlying model for the stochastic games, also referred to as Markov games in [Hu99, Lit94].

The joint action space, is a safe technique in the sense that the influence of an agent on every other agent can be modeled in such a manner that the Markov property still holds. This combined with a unique solution concept such as the Stackelberg equilibrium, allows to bootstrap as is usually done in RL techniques [Kon04]. However the joint action space approach violates the basic principles of multi-agent systems : distributed control, asynchronous actions, incomplete information, cost of communication etc.

4.2 Independent Reinforcement learners

Independent RL are trying to optimize their behavior without any form of communication with the other agents, they only use the feedback they receive from the environment. These independent RL might use traditional reinforcement learning algorithms, created for stationary, one-agent settings. Since the feedback coming from the environment is in general dependent on the combination of action taken by the agents, and not just the action of a single agent, the Markovian property no longer holds, and the problem becomes non-stationary state dependent [Nar89]. It is shown in [Nar89] that if the agents use a reward-inaction updating scheme they are able to converge to a pure Nash equilibrium state. In case no pure NE exists, we need to extend the RL algorithm as discussed in the last section of this paper.

In our study, we also consider optimistic independent agents, introduced in [Lau00]. Optimistic independent agents will only do their update when the performance they receive from the new experience is better than was expected until then.

4.3 Informed Reinforcement learners

Informed agents use communication through *revelation schemes*, in order to coordinate on the optimal joint action, [Muk01]. Revelation comes in different 2-player schemes, ranging from not allowing agents to communicate actions to each other (no revelation), to allowing agents in turn to reveal their actions (alternate revelation), and to allowing agents to reveal their actions simultaneously (simultaneous revelation). The agents, who are allowed to communicate, decide for themselves whether they reveal their action or not.

4.4 Exploration-exploitation schemes for independent Reinforcement learners

The last group of techniques used for the comparison of Section 4.5 tries extended exploration-exploitation schemes to push independent agents toward their part of the optimal joint-action. Two algorithms have been considered in this study. The Frequency Maximum Q technique described in [Kap02] adds an heuristic value to the Q-value of an action in the Boltzmann exploration strategy. This FMQ heuristic takes into account how frequently an action produces its maximum corresponding reward. In [Ver02] a new exploration technique is used for coordination games. It is based on exploring, selfish reinforcement learning agents (ESRL), playing selfish for a period of time and then excluding actions from their private action space, so that the joint action space gets considerably smaller and the agents are able to converge to a NE of the remaining subgame. By repeatedly excluding action, the agents are able to figure out the Pareto front and decide on which combination of actions is preferable.

4.5 Comparison of techniques

The 2 player games we used as a test-bed were respectively 2 revelation games, the penalty game and the climbing game. The revelation games were extracted from [Muk01]. The first one was constructed in such a way that each agent has a preferred action no matter what the other agent does. However the joint combination of these choices is not optimal. The second revelation game has a Pareto optimal joint action however this is not a Nash equilibrium. The other 2 games were originally used in [Cla98]. The challenge in the climbing game is to reach the optimal joint action when it is surrounded by heavy penalties. The penalty game has the additional difficulty of having two Pareto optimal solutions on which the agents should agree. A complete overview of the results can be found in [Pee03]. We summarize the most important conclusions, flaws and strong points here.

The independent learners produced the most remarkable result in one of the revelation games. They were able to perform better than the revelation agents. This shows that providing more information to an agent may result in a worse performance. However the other games showed that acting independently is not always enough to guarantee convergence to the Pareto Optimal NE. Independent learners are also very dependent on the settings of the learning rate and the exploration temperature. The optimistic assumption is useful in driving the agents to the Pareto Optimal solution, however this technique will not work in stochastic environments.

The revelation learners were originally not created for the climbing and penalty game, and the penalties in these games turned out to be too difficult to overcome. The revelation learners would probably behave better in an auction environment. The modeling of other agents might give them an advantage and the concept of revelation might lead to interesting results (perhaps extending the agents to give them the ability to lie).

The FMQ learners have a convergence rate of 100% in identical payoff games (i.e the games the algorithm is designed for). For the other games convergence is not always assured, but if it does converge the convergence time is very good. When we play games where the optimal group utility differs from the agents' personal utility, FMQ learners fail to reach the optimal solution. Also at this stage, FMQ learners have problems with genuine stochastic games.

The ESRL learners reach the Pareto optimal solution in every game we played under the assumption that they were allowed enough time to converge to a NE. A downside is that this time is to be set in advance. The time they need to converge is usually longer than for FMQ learners, because playing in periods and visiting all equilibria is a slow process. However, because of the agents playing in periods of time and therefore allowing them to sample enough information on joint actions, exclusion learners are able to work in stochastic games.

4.6 An Evolutionary Game Theoretic perspective on Multi-Agent Learning

Relating RL and EGT is not new. Börgers and Sarin were the first[9] to mathematically connect RL with EGT [Bör97]. Their work is located in the field of economics, where also other researchers are very active on the link between RL and EGT, as for instance [Red01]. However this relation has received far less attention in the field of AI, and more specifically in the field of Multi-Agent Systems. Börgers and Sarin have made the formal connection between the Replicator Equations and Cross Learning (a simple RL model) explicit in their work [Bör97]. The evolutionary approach to game theory attracts ever more attention of researchers from different fields, such as economics, computer science and artificial intelligence [Bör97, Gin00, Wei96, Sam97, Baz97, Now99a]. The

[9]As far as we know.

possible successful application of evolutionary game theoretic concepts and models in these different fields becomes more and more apparent.

If the word evolution is used in the biological sense, then this 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 we call cultural evolution [Bjo95]. Of course it is implicit and intuitive that there is an analogy between biological evolution and learning. We can now look at this analogy at two different levels. First there is the individual level. An individual decision maker usually has many ideas or strategies in his 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. Secondly, it is possible that individual learning behavior is different from biological evolution. An example is best response learning where individuals adjust too rapidly to be similar to evolution. However, then it might be the case that at the population level, consisting of different individuals, a process is operating analogous to biological evolution. In this paper we describe the similarity between biological evolution and learning at the individual level in a formal and experimental manner.

In this section we discuss or merely point out the results in making this relation between Multi-Agent Reinforcement Learning and EGT explicit. Börgers and Sarin have shown how the two fields are related in terms of dynamic behaviour, i.e. the relation between Cross learning and the replicator dynamics. The replicator dynamics postulate gradual movement from worse to better strategies. This is in contrast to classical Game Theory, which is a static theory and does not prescribe the dynamics of adjustment to equilibrium. The main result of Börgers and Sarin is that in an appropriately constructed continuous time limit, the Cross' learning model converges to the asymmetric, continuous time version of the replicator dynamics. 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 replicator dynamics. For more details see [Bör97]. We illustrate this result with the prisoners dilemma game. In Figure 7 we plotted the direction field of the Replicator equations for the prisoner's dilemma game and we also plotted the Cross learning process for this same game.

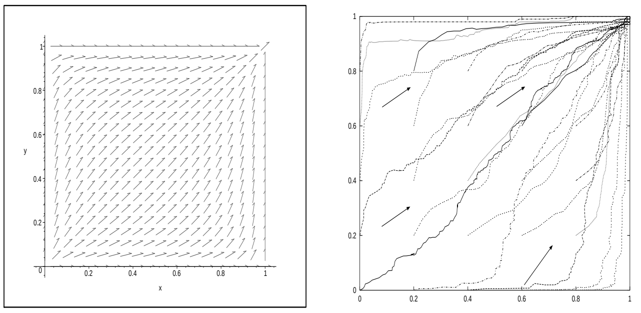

Figure 7 Left: The direction field plot of the RD of the prisoner's dilemma game. The x-axis represents the probability with which the first player (or row player) plays *defect*, and the y-axis represents the probability with which the second player (or column player) plays *defect*. The strong attractor and Nash equilibrium of the game lies at coordinates $(1, 1)$ as one can see in the plot. Right: The paths induced by the Cross learning process of the prisoner's dilemma game. The arrows point out the direction of the learning process. These probabilities are now learned by the Cross learning algorithm.

For both players we plotted the probability of choosing their first strategy (in this case defect). The x-axis represents the probability with which the row player plays *defect*, and the y-axis represents the probability with which the column player plays this same strategy. As you can see the sample paths of the Cross learning process approximates the paths of the RD.

In previous work the authors have extended the results of Börgers and Sarin to popular Reinforcement Learning (RL) models as Learning Automata (LA) and Q-learning. In [Tuy02] the authors have shown that the Cross learning model is a Learning Automaton with a linear-reward-inaction updating scheme. All details and experiments are available in [Tuy02].

Next, we continue with the mathematical relation between Q-learning and the Replicator Dynamics. In [Tuy03b] we derived mathematically the dynamics of Boltzmann Q-learning. We investigated here whether there is a possible relation with the evolutionary dynamics of Evolutionary Game Theory. More precisely we constructed a continuous time limit of the Q-learning model, where Q-values are interpreted as Boltzmann probabilities for the action selection, in an analogous manner of Börgers and Sarin for Cross learning. We briefly summarize the findings here. All details can be consulted in [Tuy03b]. The derivation has been restricted to a 2 player situation for reasons of simplicity. Each agent(or player) has a probability vector over his action set , more precisely $x_1, ..., x_n$ over action set $a_1, ..., a_n$ for the first player and $y_1, ..., y_m$ over $b_1, ..., b_m$ for the second player. Formally the Boltzmann distribution is described by,

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

where $x_i(k)$ is the probability of playing strategy i at time step k and τ is the temperature. The temperature determines the degree of exploring different strategies. As the trade-off between exploration-exploitation is very important in RL, it is important to set this parameter correctly. Now suppose that we have payoff matrices A and B for the 2 players. Calculating the time limit results in,

$$\frac{dx_i}{dt} = x_i\alpha\tau((A\mathbf{y})_i - \mathbf{x}\cdot A\mathbf{y}) + x_i\alpha\sum_j x_j ln(\frac{x_j}{x_i}) \tag{25}$$

for the first player and analogously for the second player in,

$$\frac{dy_i}{dt} = y_i\alpha\tau((B\mathbf{x})_i - \mathbf{y}\cdot B\mathbf{x}) + y_i\alpha\sum_j y_j ln(\frac{y_j}{y_i}) \tag{26}$$

Comparing (25) or (26) with the RD in (22), we see that the first term of (25) or (26) is exactly the RD and thus takes care of the selection mechanism, see [Wei96]. The mutation mechanism for Q-learning is therefore left in the second term, and can be rewritten as:

$$x_i\alpha\sum_j x_j ln(x_j) - ln(x_i) \tag{27}$$

In equation (27) we recognize 2 entropy terms, one over the entire probability distribution x, and one over strategy x_i. Relating entropy and mutation is not new. It is a well known fact [Schneid00, Sta99] that mutation increases entropy. In [Sta99], 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. Exploration can be considered as the mutation concept, as both concepts take care of providing variety.

Equations 25 and 26 now express the dynamics of both Q-learners in terms of Boltzmann probabilities, from which the RD emerge.

In [Tuy03c] we answered the question whether it is possible to first define the dynamic behavior in terms of Evolutionary Game Theory (EGT) and then develop the appropriate RL-algorithm.

For these reasons we extended the RD of EGT. We call it the Extended Replicator Dynamics. All details on this work can be found in [Tuy03c]. The main result is that the extended dynamics guarantee an evolutionary stable outcome in all types of 1-stage games.

Finally, in [Hoe04] the authors have shown how the EGT approach can be used for Dispersion Games [Gre02]. In this cooperative game, n agents must learn to choose from k tasks using local rewards and full utility is only achieved if each agent chooses a distinct task. We visualized the learning process of the MAS and showed typical phenomena of distributed learning in a MAS. Moreover, we showed how the derived fine tuning of parameter settings from the RD can support application of the COllective INtelligence (COIN) framework of Wolpert et al. [Wol98, Wol99] using dispersion games. COIN is a proved engineering approach for learning of cooperative tasks in MASs. 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. Thus, the challenge is to define suitable private utility functions for the individual agents, given the performance of the collective. We showed that the derived link between RD and RL predicts performance of the COIN framework and visualizes the incentives provided in COIN toward cooperative behavior.

5 Final Remarks

In this survey paper we investigated Reinforcement Learning and Evolutionary Game Theory in a Multi-Agent setting. We provided most of the fundamentals of RL and EGT and moreover showed their remarkable similarities. We also discussed some of the excellent existing Multi-agent RL algorithms today and gave a more detailed description of the Evolutionary Game Theoretic approach of the authors. However, still a lot of work needs to be done and some problems are still unresolved. Especially, overcoming problems of incomplete information and large state spaces, in Multi-Agent Systems as for instance Sensor Webs, are still hard. More precisely, under these conditions it becomes impossible to learn models over other agents, storing information on them and using a lot of communication.

6 Acknowledgments

After writing this survey paper an acknowledgment is in place. We wish to thank our colleagues of the Computational Modeling Lab from the Vrije Universiteit Brussel, Belgium. Most of all we want to express our appreciation and gratitude to Katja Verbeeck and Maarten Peeters for their important input and support in writing this article.

We also wish to express our gratitude to Dr. ir Pieter-Jan 't Hoen of the CWI (center for mathematics and computer science) in the Netherlands, especially for his input on the COIN framework.

References

[And87]Anderson C.W., Strategy Learning with multilayer connectionist representations, Proceedings of the 4th international conference on Machine Learning, pp. 103-114.

[Bar83]Barto A., Sutton R., and Anderson C., Neuronlike adaptive elements that can solve difficult learning control problems, IEEE Transactions on Systems, Man and Cybernetics, Vol. 13, No. 5,pp. 834-846.

[Baz03]Bazzan A. L. C., Klugl Franziska, Learning to Behave Socially and Avoid the Braess Paradox In a Commuting Scenario. Proceedings of the first international workshop on Evolutionary Game Theory for Learning in MAS,july 14 2003, Melbourne Australia.

[Baz97]Bazzan A. L. C., A game-theoretic approach to coordination of traffic signal agents. PhD thesis, Univ. of Karlsruhe, 1997.

[Bel62]Bellman R.E., and Dreyfuss S.E., Applied Dynamical Programming, Princeton University press.

[Ber76]Bertsekas, D.P., Dynamic Programming and Stochastic Control. Mathematics in Science and Engineering, Vol. 125, Academic Press, 1976.

[Bjo95]Bjornerstedt J., and Weibull, J. Nash Equilibrium and Evolution by Imitation. The Rational Foundations of Economic Behavior, (K. Arrow et al, ed.), Macmillan, 1995.

[Bör97]Börgers, T., Sarin, R., Learning through Reinforcement and Replicator Dynamics. Journal of Economic Theory, Volume 77, Number 1, November 1997.

[Bus55]Bush, R. R., Mosteller, F., Stochastic Models for Learning, Wiley, New York, 1955.

[Cas94]Cassandra, A. R., Kaelbling, L. P., and Littman , M. L., Acting optimally in partially observable stochastic domains. In Proceedings of the Twelfth National Conference on Artificial Intelligence, Seattle, WA, 1994.

[Cla98]Claus, C., Boutilier, C., The Dynamics of Reinforcement Learning in Cooperative Multi-Agent Systems, Proceedings of the 15th international conference on artificial intelligence, p.746-752, 1998.

[Gin00]Gintis, C.M., Game Theory Evolving. University Press, Princeton, June 2000.

[Gre02]T. Grenager, R. Powers, and Y. Shoham. Dispersion games: general definitions and some specific learning results. In AAAI 2002, 2002.

[Hir74]Hirsch, M.W., and Smale, S., Differential Equations, Dynamical Systems and Linear Algebra. Academic Press, Inc, 1974.

[Hoe04]'t Hoen, P.J., Tuyls, K., Engineering Multi-Agent Reinforcement Learning using Evolutionary Dynamics. Proceedings of the 15th European Conference on Machine Learning (ECML'04), LNAI Volume 3201, 20-24 september 2004, Pisa, Italy.

[Hof98]Hofbauer, J., Sigmund, K., Evolutionary Games and Population Dynamics. Cambridge University Press, November 1998.

[Hu99]Hu, J., Wellman, M.P., Multiagent reinforcement learning in stochastic games. Cambridge University Press,November 1999.

[Jaf01]Jafari, C., Greenwald, A., Gondek, D. and Ercal, G., On no-regret learning, fictitious play, and nash equilibrium. Proceedings of the Eighteenth International Conference on Machine Learning,p 223 - 226, 2001.

[Kae96]Kaelbling, L.P., Littman, M.L., Moore, A.W., Reinforcement Learning: A Survey. Journal of Artificial Intelligence Research, 1996.

[Kap02]Kapetanakis S. and Kudenko D., Reinforcement Learning of Coordination in Cooperative Multi-agent Systems, AAAI 2002.

[Kap04]Kapetanakis S., Independent Learning of Coordination in Cooperative Single-stage Games, PhD dissertation, University of York, 2004.

[Kon04]Ville Kononen, Multiagent Reinforcement Learning in Markov Games: Asymmetric and Symmetric approaches, PhD dissertation, Helsinki University of Technology, 2004.

[Lau00]Lauer M. and Riedmiller M., An algorithm for distributed reinforcement learning in cooperative multi-agent systems, Proceedings of the seventeenth International Conference on Machine Learning, 2000.

[Lit94]Littman, M.L., Markov games as a framework for multi-agent reinforcement learning. Proceedings of the Eleventh International Conference on Machine Learning, p 157 - 163, 1994.

[Loc98]Loch, J., Singh, S., Using eligibility traces to find the best memoryless policy in a partially observable markov process. Proceedings of the fifteenth International Conference on Machine Learning, San Francisco, 1998.

[May82]Maynard-Smith, J., Evolution and the Theory of Games. Cambridge University Press, December 1982.

[May73]Maynard Smith, J., Price, G.R., The logic of animal conflict. Nature, 146: 15-18, 1973.

[Muk01]R.Mukherjee and S.Sen, Towards a Pareto Optimal Solution in general-sum games, Working Notes of Fifth Conference on Autonomous Agents, 2001, pages 21 - 28.

[Nar74]Narendra, K., Thathachar, M., Learning Automata: A Survey. IEEE Trans. Syst., Man, Cybern., Vol. SMC-14, pages 323-334, 1974.

[Nar89]Narendra, K., Thathachar, M., Learning Automata: An Introduction. Prentice-Hall, 1989.

[Now99a]Nowé, A., Parent, J., Verbeeck, K., Social agents playing a periodical policy. Proceedings of the 12th European Conference on Machine Learning, p 382 - 393, 2001.

[Now99b]Nowé A. and Verbeeck K., Distributed Reinforcement learning, Loadbased Routing a case study, Notes of the Neural, Symbolic and Reinforcement Methods for sequence Learning Workshop at ijcai99, 1999, Stockholm, Sweden.

[Neu44]von Neumann, J., Morgenstern, O., Theory of Games and Economic Behaviour, Princeton University Press, 1944.

[Osb94]Osborne J.O., Rubinstein A., A course in game theory. Cambridge, MA: MIT Press,1994.

[Pee03]Maarten Peeters, A Study of Reinforcement Learning Techniques for Cooperative Multi-Agent Systems, Computational Modeling Lab, Vrije Universiteit Brussel, 2003.

[Pen98]Pendrith M.D., McGarity M.J., An analysis of direct reinforcement learning in non-Markovian domains. Proceedings of the fifteenth International Conference on Machine Learning, San Francisco,1998.

[Per02]Perkins T.J., Pendrith M.D., On the Existence of Fixed Points for Q-learning and Sarsa in Partially Observable Domains. Proceedings of the International Conference on Machine Learning (ICML02),2002.

[Red01]Redondo, F.V., Game Theory and Economics, Cambridge University Press, 2001.

[Sam97]Samuelson, L. Evolutionary Games and Equilibrium Selection, MIT Press, Cambridge, MA, 1997.

[Schneid00]Schneider, T.D., Evolution of biological information. Journal of NAR, volume 28, pages 2794 - 2799, 2000.

[Sta99]Stauffer, D., Life, Love and Death: Models of Biological Reproduction and Aging. Institute for Theoretical physics, Köln, Euroland, 1999.

[Ste97]Steenhaut K., Now A., Fakir M. and Dirkx E., Towards a Hardware Implementation of Reinforcement Learning for Call Admission Control in Networks for Integrated Services. In the proceedings of the International Workshop on Applications of Neural Networks and other Intelligent Techniques to Telecommunications 3, Melbourne, 1997.

[Sut88]Sutton, R.S., Learning to Predict by the Methods of Temporal Differences, Machine Learning 3, Kluwer Academic Publishers, Boston, pp. 9-44.

[Sut00]Sutton, R.S., Barto, A.G., Reinforcement Learning: An introduction. Cambridge, MA: MIT Press, 1998.

[Sto00]Stone P., Layered Learning in Multi-Agent Systems. Cambridge, MA: MIT Press, 2000.

[Tha02]Thathacher M.A.L., Sastry P.S., Varieties of Learning Automata: An Overview. IEEE Transactions on Systems, Man, And Cybernetics-Part B: Cybernetics, Vol. 32, NO.6, 2002.

[Tse62]Tsetlin M.L., On the behavior of finite automata in random media. Autom. Remote Control, vol. 22, pages 1210-1219, 1962.

[Tse73]Tsetlin M.L., Theory and Modeling of Biological Systems. New York: Academic, 1973.

[Tsi93]Tsitsiklis, J.N., Asynchronous stochastic approximation and Q-learning. Internal Report from the laboratory for Information and Decision Systems and the Operation Research Center, MIT 1993.

[Tuy02]Tuyls, K., Lenaerts, T., Verbeeck, K., Maes, S. and Manderick, B, Towards a Relation Between Learning Agents and Evolutionary Dynamics. Proceedings of the Belgium-Netherlands Artificial Intelligence Conference 2002 (BNAIC). KU Leuven, Belgium.

[Tuy03a]Tuyls, K., Verbeeck, K., and Maes, S. On a Dynamical Analysis of Reinforcement Learning in Games: Emergence of Occam's Razor. Lecture Notes in Artificial Intelligence, Multi-Agent Systems and Applications III, Lecture Notes in AI 2691, (Central and Eastern European conference on Multi-Agent Systems 2003). Prague, 16-18 june 2003, Czech Republic.

[Tuy03b]Tuyls, K., Verbeeck, K., and Lenaerts, T. A Selection-Mutation model for Q-learning in Multi-Agent Systems. The ACM International Conference Proceedings Series, Autonomous Agents and Multi-Agent Systems 2003. Melbourne, 14-18 juli 2003, Australia.

[Tuy03c]Tuyls, K., Heytens, D., Nowé, A., and Manderick, B., Extended Replicator Dynamics as a Key to Reinforcement Learning in Multi-Agent Systems. Proceedings of the European Conference on Machine Learning'03, Lecture Notes in Artificial Intelligence. Cavtat-Dubrovnik, 22-26 september 2003, Croatia.

[Ver02]K.Verbeeck and A. Nowé and T.Lenaerts and J. Parent, Learning to reach the Pareto Optimal Nash Equilibrium as a Team, Proceedings of the 15th Australian Joint Conference on Artificial Intelligence, 2002, pp. 407 - 418, publisher="Springer-Verlag LNAI2557

[Wat92]Watkins, C. and Dayan, P., Q-learning. Machine Learning, 8(3):279-292, 1992.

[Wei96]Weibull, J.W., Evolutionary Game Theory, MIT Press 1996.

[Wei98]Weibull, J.W., What we have learned from Evolutionary Game Theory so far? Stockholm School of Economics and I.U.I. may 7, 1998.

[Wei99]Weiss, G., Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence. Edited by Gerard Weiss Cambridge, MA: MIT Press. 1999.

[Wol98]Wolpert, D.H., Tumer, K., and Frank, J., Using Collective Intelligence to Route Internet Traffic. Advances in Neural Information Processing Systems-11, pages 952–958. Denver, 1998.

[Wol99]Wolpert, D.H., Wheller, K.R., and Tumer, K., General principles of learning-based multi-agent systems. Proceedings of the Third International Conference on Autonomous Agents (Agents'99), ACM Press. Seattle, WA, USA, 1999.

[Woo02]Wooldridge, M., An Introduction to MultiAgent Systems. Published in February 2002 by John Wiley, Sons, Chichester, England, 2002.

View publication stats