Multiagent Learning Paradigms

K. Tuyls[1,2] and P. Stone[3]

[1] DeepMind, London, UK
karltuyls@google.com
[2] University of Liverpool, UK
[3] University of Texas, Austin, USA
pstone@cs.utexas.edu

Abstract. *"Perhaps a thing is simple if you can describe it fully in several different ways, without immediately knowing that you are describing the same thing"* – Richard Feynman

This articles examines multiagent learning from several paradigmatic perspectives, aiming to bring them together within one framework. We aim to provide a general definition of multiagent learning and lay out the essential characteristics of the various paradigms in a systematic manner by dissecting multiagent learning into its main components. We show how these various paradigms are related and describe similar learning processes but from varying perspectives, e.g. an individual (cognitive) learner vs. a population of (simple) learning agents.

1 Introduction

Multiagent systems (MAS) are distributed systems of independent actors, called *agents*, that are each independently controlled, but that interact with one another in the same environment [47]. In their recent book entitled *Multiagent Systems*, Shoham and Leyton-Brown define multiagent systems as "those systems that include multiple autonomous entities with either diverging information or diverging interests, or both." [36] Examples of multiagent systems applications include automated driving, disaster rescue aided by teams of robots, and autonomous bidding agents for electricity power markets. Because of the complexity of most MAS it is often impossible, or at least impractical, to engineer effective agent behaviors by hand. Rather, it is preferable for agents to be able to *learn* to behave effectively from experience in the environment, and from interactions with other agents. Tom Mitchell, in his book *Machine Learning* defines machine learning (ML) as "the study of computer algorithms that improve automatically through experience." [27] Using these definitions of MAS and ML as bases, we consider "multiagent learning" to be:

> *The study of multiagent systems in which one or more of the autonomous entities improves automatically through experience.*

As stated, this definition is quite broad, leaving open the possibility for many types of autonomous entities, systems of these entities, and foci of learning. For

example, there could be many simple agents (like an ant colony), or a small number of sophisticated agents (like a soccer team). The agents could interact over long periods of time with exactly the same other agents, or with a series of different randomly chosen other agents, each for a short interaction. And the agents could learn about the environment itself, about the behaviors of the other agents, or directly about what actions are most effective. The main commonality in all of these above scenarios, and indeed the prerequisite for learning in the first place (as pointed out by Shoham and Leyton-Brown), is that there is a temporal nature to the scenario that exhibits regularity across time. Thus past experience is somehow predictive of future expectations.

Multiagent learning has received most attention from the reinforcement learning community [23, 7, 17, 39]. For an overview see [18, 44]. In [37] Shoham et al. explore what research questions multiagent learning is trying to answer by defining five research agenda's that MAL research is pursuing and classifying the state of the art therein. As not all work falls into one of these agenda's, this implies that either we need more agenda's, or some work needs to be revisited. The purpose of the paper was to inititiate a discussion within the community leading to several response articles, e.g. [38, 34, 41]. The current paper is different, in that it considers several multiagent learning paradigms, and not only RL, and furthermore aims to understand what the different MAL components are, bringing several of the paradigms together within one framework.

1.1 Multiagent Learning Components

In this paper, we consider the full spectrum of such scenarios, in which multiagent learning is possible. As illustrated in Figure 1, we think of a multiagent learning scenario as consisting of four distinct components: the environment, the agents, the interaction mechanism, and the learning mechanism itself.

First, the environment, or domain, specifies the state space, action space, and transition function. The state space specifies the set of states that an *individual* agent can be in at any given time. The action space is the set of actions available to an individual agent at any given time, and the transition function, or the environment dynamics, specifies the (possibly stochastic) way in which the environment changes as a result of each agent (or a subset of agents) executing an action in a given state. For the purposes of exposition, we assume that the environment proceeds in discrete, evenly-spaced time steps and that all actions are available at all times. But these assumptions are easily relaxed, and indeed must be in many practical settings.

Second, the agents are defined by their communication channels with the environment for sensing the (possibly partial) state and for specifying actions; their communication channels between one another; their utility functions indicating their preferences over environmental states; and their policies for selecting actions.

Third, the interaction mechanism defines how long agents interact with one another, with which other agents, and what they observe about other agents. For

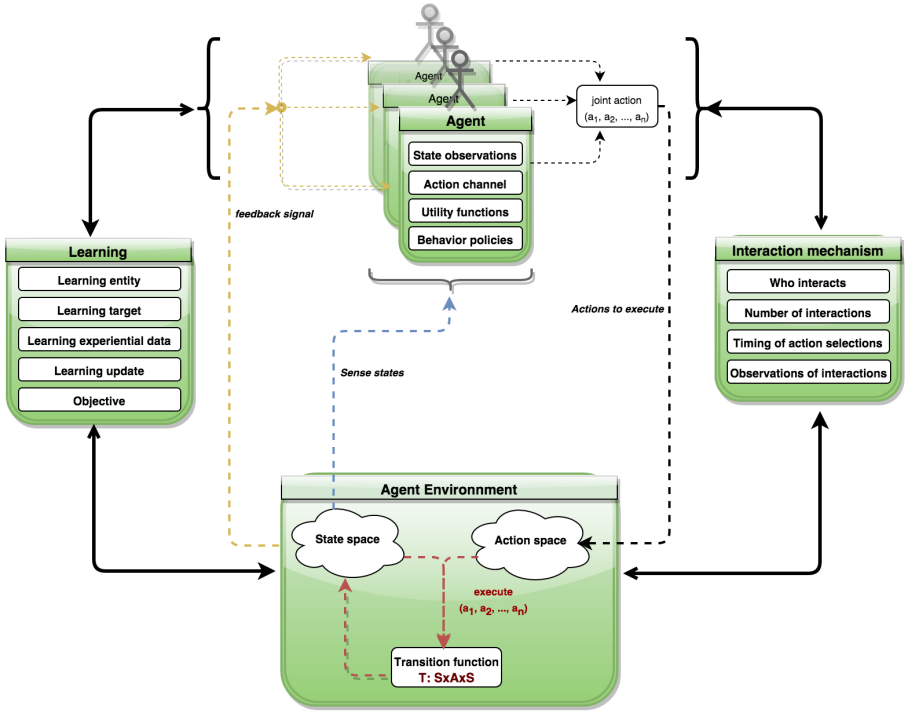

Fig. 1. A depiction of the general multiagent learning scenario.

example, at one extreme, agents may be fully aware of each other's behavior policies; or, at the other extreme, they may only observe the effects of their actions on the environment. As intermediate possibilities, they may observe each other's selected actions, their utilities (payoffs), or both. The interaction mechanism also dictates the frequency (or number) of interactions among any given agents, as well as whether their actions are selected simultaneously or sequentially (the timing of action selections).

Fourth, the learning mechanism is defined by the learning entity, learning target, the learning experiential data, the learning update, and the objective of learning. The learning entity specifies whether the learning happens at the individual agent level, e.g. by an intelligent cognitive agent, or at the group level, e.g. by a population of cognitively limited agents. The learning target describes what is being learnt. For example, it could be the interaction mechanism that is being learnt, or the policies of the individual agents. The learning experiential data describes what information is available to the learning entity as the basis for learning. The learning update defines how the learning entity is updated during the learning process; and the objective is a representation of the goal, or evaluation function, of the learning process.

1.2 Classes of multiagent learning

Multiagent learning scenarios are challenging both to design and to analyze for a number of reasons. To begin with, even defining an objective function is far from trivial. For example, is the goal to achieve some desired configuration for the entire set of agents with no regard for individual utility (e.g. disaster rescue with multiple robots); to achieve a game theoretic equilibrium (e.g. autonomous cars selecting travel routes); or to achieve maximum utility for some subset of designated agents (e.g. agents bidding in a marketplace on one person's behalf)? In addition, from the algorithmic perspective, as long as multiple agents are learning, multiagent learning scenarios are inherently non-stationary, meaning that they violate the Markov assumption that is typically leveraged by sequential decision making algorithms.

For the purpose of description and analysis, in this paper we divide multiagent learning scenarios into three distinct classes based on how many of the agents, and to what extent the system of interactions is "in our control" as designers of algorithms. Since each class has historically been addressed by different types of algorithms, we divide the paper into sections accordingly. However, we find that ultimately there are many commonalities among these algorithms, which we emphasize and unify throughout.

First, we consider the case in which just one of the agents is in our control as it interacts repeatedly with the same small set of relatively sophisticated agents. This class of scenarios, which we refer to as *individual learning* scenarios, is traditionally the realm of multiagent reinforcement learning (RL) algorithms. Second, we consider the case in which we have control over all of the agents and their interactions as they interact repeatedly with randomly chosen members of a large population of relatively simple agents. In such *population learning*

scenarios, the next "state" may be defined by the distribution of other agents in the population (in particular their behaviors) that defines who the agents will interact with. This class of scenarios is traditionally the realm of co-evolutionary approaches and swarm algorithms. Third, we consider the case in which none of the agents are in our control, but we can define the system of interactions. We refer to this case as *protocol learning*. While much less common than the prior two cases, protocol learning covers multiagent systems research such as adaptive mechanism design for autonomous bidding agents.

While the distinctions among the three types of scenarios may appear sharp as stated, in practice they are quite fuzzy. For example, multiagent reinforcement learning algorithms can be analyzed from the perspective of all agents being in our control, and swarm algorithms can include relatively sophisticated agents.

In the next section we further refine the three classes of problems into five paradigmatic settings in which to consider the multiagent learning problem.

2 Paradigms

As stated above multiagent learning is a process by which agents learn to behave in order to achieve their goal(s), while interacting with other agents (possibly co-operative or adversarial) that are potentially learning as well. These learned behaviours can be generated by a variety of techniques coming from different paradigms. We distinguish five such paradigms from which such learning can be studied.

We distinguish between three higher level types of agents or learning scenarios, i.e., individual learning in which a relatively sophisticated agent learns at the individual level; population learning in which a population of cognitively-limited agents learn at the group level by using simple local interactions; and protocol learning in which the interaction mechanism among the agents is itself learned. The five paradigmatic settings we consider are:

1. *Online RL towards individual utility*
2. *Online RL towards social welfare*
3. *Co-evolutionary learning*
4. *Swarm Intelligence*
5. *Adaptive mechanism design*

Paradigms 1 and 2 concern *individual learners*, paradigms 3 and 4 concern *population learners*, and paradigm 5 concerns *protocol learning*. In addition to the 5 paradigms we also consider MAL tools for analyzing and predicting learning behaviour, and for building opponent models. Specifically, we consider:

- Analysis and prediction tools, for example to analyze the resulting equilibrium behavior of coevolutionary approaches; and
- Teammate and opponent modeling tools that can be useful for predicting agent behaviors in any of the five paradigms.

In the next section we describe the five paradigms systematically in prototypical multiagent learning scenarios that fully specify the environment, the agents, the interaction, and the objective, following the taxonomy laid out in Figure 1.

3 Paradigm Descriptions

This section describes the five paradigms introduced above in more detail, and categorizes them according to the taxonomy introduced above.

3.1 Paradigm 1: Online RL towards individual utility

One of the most-studied scenarios in multiagent learning is that in which multiple independent agents take actions in the same environment, and learn online to maximize their own individual utility functions.

This paradigm, in turn, is most often reduced to the abstract game-theoretic, artificial scenario of a repeated normal form game. In a normal form game, each agent (or "player") has a set of possible actions, players select an action simultaneously, and each player gets a payoff that is a function of the full set of actions. Perhaps the most famous normal form game is the Prisoner's Dilemma, a 2-player game with actions and utilities shown in Figure 2. The motivation is that two prisoners committed a crime together and are being interrogated separately. If neither of them confesses to the crime (they both "cooperate"), then they will both get a small punishment (corresponding to a payoff of 3 in the figure). If one of them confesses (or "defects") but the other does not, then the one that confesses gets off for free (payoff of 5), but the other gets the worst punishment possible (payoff of 0). If they both defect, they both get a fairly bad punishment (payoff of 1). Normal form games can also have more than 2 players, and more than 2 actions per player.

$$\begin{array}{cc} & \begin{array}{cc} D & C \end{array} \\ \begin{array}{c} D \\ C \end{array} & \begin{pmatrix} 1,1 & 5,0 \\ 0,5 & 3,3 \end{pmatrix} \end{array}$$

Fig. 2. Payoff tables of the PD game. Strategies D and C correspond with Defect and Cooperate respectively.

Normal form games were initially introduced and studied as one-shot interactions. The players knew each other's full utility functions, and played the game only once. It was in this setting that the famous Nash equilibrium was introduced as a set of actions such that no player would be better off deviating given that the other players' actions are fixed. Games can have one, or multiple Nash Equilibria. In the prisoner's dilemma, the only Nash Equilibrium is for both agents to defect.

In these traditional one-shot settings, there is no opportunity for learning because there is no temporal nature to the scenario. However, it is also possible to consider *repeated* normal form games such that the same players interact with one another multiple times in the same game, with the objective of maximizing

their (possibly discounted)[4] sum of utilities over time. In repeated normal form games, the repetition provides the temporal nature. The regularity across time comes from the assumption that players' past actions are somehow predictive of their future actions.

While normal form games are the most common way to formulate this paradigm, it has also been studied extensively in the context of the pursuit domain [4], and applies also to a wide variety of more complex domains. The essential defining characteristics for multiagent learning settings that fall under this paradigm (i.e. the characteristic necessary to fall under this paradigm), are as laid out in Table 1. When many options are possible, they are enumerated, or the field is left blank.

Online RL towards individual utility	
Component	**Description**
Agent Environment	
state space:	
action space:	
transition function:	
Agents	
state observations:	
action channel:	
utility functions:	(discounted) sum of rewards
behavior policies:	the other agents' policies are unknown; our agent's policy is the target of the learning (this is what is in our control)
Interaction mechanism	
who interacts:	same set of agents
frequency of interactions:	repeat multiple times or in one continual process
timing of action selections:	actions taken simultaneously or sequentially
observations of interactions:	agents may or may not observe the other agents' actions, payoffs, or policy
Learning	
learning entity:	individual learner
learning target:	agent's policy
learning experiential data:	agents' joint action, reward, next state observation
learning update:	behavior update from last experience
objective:	maximize our own agent's sum of utilities over time

Table 1. The essential characteristics of paradigm 1. When many options are possible, they are enumerated, or the field is left blank.

[4] Discounted utilities are used to represent that near-term payoffs are more important to the agent than longer term payoffs.

3.2 Paradigm 2: Online RL towards social welfare

A slight variation on the above scenario is that we may assume that all agents are using exactly the same learning-based behavior policy, and adopt the objective that, perhaps after some transient initial phase, they arrive at a steady state of always both selecting the cooperate action (which maximizes their sum of utilities).

In this case, the main things that change from Paradigm 1 are the behavior policies (in particular, what is in our control), and the objective.

The essential characteristics for this paradigm are laid out in Table 2.

Online RL towards social welfare	
Component	**Description**
Agent Environment	
state space:	
action space:	
transition function:	
Agents	
state observations:	
action channel:	
utility functions:	(discounted) sum of rewards
behavior policies:	**all agents' policy is the target of the learning (this is what is in our control)**
Interaction mechanism	
who interacts:	same set of agents
frequency of interactions:	repeat multiple times or in one continual process
timing of action selections:	actions taken simultaneously or sequentially
observations of interactions:	**each agent observes the other agents' actions and payoffs, but not policy**
Learning	
learning entity:	each agent learns individually
learning target:	each agent's policy
learning experiential data:	agents' joint action, reward next state observation
learning update:	behavior updates from last experience
objective:	**maximize the sum of all agents' utilities over time**

Table 2. The essential characteristics of paradigm 2. When many options are possible, they are enumerated, or the field is left blank. The differences from Paradigm 1 are highlighted in bold.

3.3 Paradigm 3: Co-evolutionary approaches

Evolution can be used to learn agent behaviors as well. In this paradigm, abstract Darwinian models of evolution are applied to refine populations of agents (known

as individuals) representing candidate solutions to a given problem [26, 11, 32]. This process consists of five steps: representation, selection, generation of new individuals (crossover and mutation), evaluation, and replacement. An evolutionary algorithm (EA) begins with an initial population of randomly-generated agents. Each member of this population is then evaluated and assigned a fitness value. The EA then uses a fitness-oriented procedure to select agents, breeds and mutates them to produce child agents, which are then added to the population, replacing older agents. One evaluation, selection, and breeding cycle is known as a generation. Successive generations continue to refine the population until time is exhausted or a sufficiently fit agent is discovered. Coevolution is an intuitive extension of evolutionary algorithms for domains with multiple learning agents. In co-evolution, the fitness of an individual is based on its interaction with other individuals in the population.

In essence EAs are training a "policy" to perform a state to action mapping. In this approach, rather than update the parameters of a single agent interacting with the environment as is done in reinforcement learning, one searches through a population of policies to find one that is appropriate for the task. This type of policy search approach is well suited to domains with continuous states and actions where traditional reinforcement learning approaches generally encounter difficulties. One can use a probability vector or distribution as representation of the policy, but an often-used policy in conjunction with evolutionary algorithms is a feed-forward neural network with non-linear activation functions (referred to as neuro-evolution [35, 10, 45, 20]). The aim of the neural network is to perform a mapping between its inputs (state) and its outputs (actions), that satisfies the agent's task. For example, a mobile robot using a neural network to navigate can map the sensory inputs it receives to direction and velocity. The key then is to find the correct parameters for the neural network that will provide the desired behavior.

The essential characteristics of this paradigm are laid out in Table 3.

3.4 Paradigm 4: Swarm Intelligence

Swarm Intelligence is a bio-inspired machine learning technique, largely based on the behavior of social insects (e.g. ants and honeybees), that is concerned with developing self-organized and decentralized adaptive algorithms [9, 24]. The type and form of learning in a swarm intelligence is characterized by a large population of cognition-limited agents that locally interact. Rather than developing complex behaviors for single individuals, as is done in reinforcement learning, swarm intelligence investigates the emerging (intelligent) behavior of a group of simple individuals that achieve complex behavior through their interactions with one another. Consequently, swarm intelligence can be considered as a cooperative multiagent learning approach in that the behavior of the full set of agents is determined by the actions of and interactions among the individuals.

In swarm intelligence, each individual in the group follows simple rules without central control structures. By interacting locally, a global behavior emerges, yet the individual has no knowledge of this 'big picture' behavior. Examples of

Co-evolutionary approaches	
Component	**Description**
Agent Environment	
state space:	
action space:	
transition function:	
Agents	
state observations:	
action channel:	
utility functions:	payoff accumulated from utility matrix
behavior policies:	agents' policies are fixed parameterized functions
Interaction mechanism	
who interacts:	many sets of 2 agents, one from each population, randomly grouped
frequency of interactions:	repeat multiple times
timing of action selections:	actions taken simultaneously or sequentially
observations of interactions:	each agent may or may not observe the other agents' actions, payoffs, or policies
Learning	
learning entity:	population
learning target:	proportion of populations with each set of possible parameters
learning experiential data:	groupings of agents to generate utilities
learning update:	change of populations based on utilities of the individuals
objective:	maximize the sum of utilities over a group-wise interaction for the best agents in the populations

Table 3. The essential characteristics of paradigm 3. When many options are possible, they are enumerated, or the field is left blank.

such systems are ant foraging, bird flocking, fish schooling, and animal herding [12, 3, 21, 14]. Currently the most well-known swarm intelligence algorithms are pheromone-based (stigmergic), such as Ant Colony Optimization [8].

Ant Colony Optimization is a class name for ant-inspired algorithms solving combinatorial optimization problems. Algorithms belonging to it are stochastic search procedures in which the central component is the pheromone model. Pheromone-based algorithms are inspired by the behavior of ants and are the most well-known swarm intelligence algorithm. The algorithms are based on the fact that ants deposit a pheromone trail on the path they take during travel. Using this trail, they are able to navigate toward their nest or food. Ants employ an indirect recruitment strategy by accumulating pheromone trails in the environment. The ants communicate indirectly via the environment, a phenomenon called *stigmergy*. When a trail is strong enough, other ants are attracted to it and, with high probability, will follow this trail toward a destination. In other words, the more ants follow a trail, the more that trail becomes attractive for being followed. However, pheromones evaporate over time, meaning that unless they are reinforced by other ants, they will disappear. Since long paths take more time to traverse, and pheromones evaporate, it will require more ants to sustain a long path. As a consequence, short paths will eventually prevail. The dissipation of pheromones ensures that "old" solutions can be forgotten, and that the ants will not get stuck in a local optimum.

Optimization problems best suited to be solved by ant colony optimization are those that can be cast as computational problems on a graph, implying that optimal solutions will correspond to specific paths in such a graph. Successful examples of such problems include the traveling salesman problem, various routing problems, job shop scheduling and even "coverage problems" with robots [9, 2, 6].

The essential characteristics of this paradigm are laid out in Table 4.

3.5 Paradigm 5: Adaptive mechanism design

Thus far, the thing under our control has been the algorithms of the agents, while their method of interaction has been taken as given. For example, in repeated games, it was given that the agents play the same game over and over again, taking actions simultaneously. The MAL algorithm defined the behavior(s) of the agent(s).

However, it is also possible to think of a multiagent learning setting as being one in which the agents are fixed (or at least beyond our control — so to the extent that they learn, they do so in a way that we cannot affect), but the interaction mechanism is to be learned [33].

Consider, for example, an auction house that interacts with a population of bidders. When auctioning several artworks, there are several parameters that can be adjusted, such as the reserve price, whether the auctions are simultaneous or sequential, and the mechanism by which the winner is determined and the price is set (e.g. English auction, Vickrey auction, Dutch auction, etc.). The auction

Swarm Intelligence	
Component	**Description**
Agent Environment	
state space:	pheromone levels and, or agent locations in the environment
action space:	
transition function:	changes in pheromone levels and, or agent locations after all take actions
Agents	
state observations:	pheromone levels and/or agent locations either globally or locally
action channel:	
utility functions:	
behavior policies:	agent's policy is a fixed function (simple control rules based on pheromone levels and/or agent locations
Interaction mechanism	
who interacts:	agents operate in the same environment
frequency of interactions:	repeated task executions by each agent
timing of action selections:	actions taken simultaneously
observations of interactions:	each agent alters the environment, affects other agents' decisions via stigmergy
Learning	
learning entity:	population
learning target:	proportion of pheromones in the environment dropped by the entire population
learning experiential data:	amounts of pheromones dropped in the environment that will determine optimal path (or utilities)
learning update:	change of pheromones levels in the environment based on ant utility
objective:	maximize the level of pheromones on the optimal path in the environment

Table 4. The essential characteristics of paradigm 4. When many options are possible, they are enumerated, or the field is left blank.

house presumably wants to maximize the selling prices of the items, which will in turn maximize its commission.[5]

In this case, the auction house is not able to control the bidders (the interaction agents) themselves. As people tend to be, they may be irrational to varying degrees. Instead, it can only control the rules of interaction, in this case the bidding rules.

Note that the ideal auction mechanism may depend on the characteristics of the goods being auctioned. Compared to artwork, people bidding on electronic equipment may bid differently (or the auction may simply attract a different population). Furthermore, the population's bidding strategies as a whole may change over time (for example due to changes in the overall economy). Thus the auction house will need to continually adapt the parameters of its auctions if it is to maximize its profits [31, 30].

The essential characteristics of this paradigm are laid out in Table 5.

4 Multiagent Learning Tools

In addition to the five MAL paradigms presented in Section 3, in this Section we summarize two useful tools for the study and development of MAL algorithms. The first, using evolutionary game theory, is an analysis tool designed to enable researchers to predict the eventual stable state (fixed point) of an MAL system assuming self-interested agents continually adapt to each other's behaviors using known learning rules.

The second, opponent modeling, is a tool used by agents within a multiagent system themselves to predict the future actions of other agents in the environment. An opponent model could itself be learnt, in which case it falls under the "learning target" within our taxonomy shown in Figure 1. However it may also be provided to the agent a priori. For this reason we treat it here as a tool to be used by an agent in a MAL system.

4.1 Analysis and prediction tool

The first MAL tool we discuss leverages Evolutionary Game Theory (EGT) as an analysis and prediction tool for the dynamics of MAL. It is well known that by using concepts from EGT, such as replicator equations and evolutionary stability, we can say something useful about the properties of learning trajectories and equilibria that are learnt by a variety of multi agent learning algorithms [43, 40, 48, 22, 13]. We now first briefly outline the differences between EGT and traditional Game Theory, and present some intuitions of the replicator equations and how they can be used as an analysis tool in MAL. For a good overview see [5].

[5] In some public auctions, the objective may instead be to maximize social welfare — striving to sell each item to the bidder who values it most.

Adaptive mechanism design	
Component	**Description**
Agent Environment	
state space:	
action space:	
transition function:	determined by the auction mechanism - how the prices change over time as a function of bids. This is one of the things we control
Agents	
state observations:	
action channel:	
utility functions:	
behavior policies:	various - not in our control
Interaction mechanism	
who interacts:	random subsets from populations of agents interact in a series of auctions
frequency of interactions:	repeat continually
timing of action selections:	Under our control, the subject of the adaptive algorithm
observations of interactions:	Under our control, the subject of the adaptive algorithm
Learning	
learning entity:	the mechanism; the entity that sets the rules of interaction
learning target:	the current mechanism: – do agents observe each other's bids? just prices? – is price set by highest bid or 2nd highest? – are auctions run sequentially or simultaneously? – are bids sequential or simultaenous?
learning experiential data:	alteration of the mechanism for next round
learning update:	objective value with new mechanism
objective:	maximize profit or social welfare

Table 5. The essential characteristics of paradigm 5. When many options are possible, they are enumerated, or the field is left blank.

Classical game theory assumes that full knowledge of the normal form game is available to all players, which together with the assumption of individual rationality, or perfectly logical players, does not necessarily reflect the dynamic nature of real world interactions. EGT relaxes the rationality assumption and replaces it by biological operators such as natural selection, crossover and mutation [46, 25, 16, 15]. Central to evolutionary game theory are the replicator dynamics that describe how a population of individuals or agents evolves over time under evolutionary pressure. Each individual has a certain phenotype, using the same pure strategy during its lifetime, and individuals are randomly paired in interaction. The population mix evolves over time according to the reproduction rates of strategies under exponential growth or decay. Their reproductive success is determined by their fitness, which results from these interactions.

The replicator dynamics dictate that the population share of a certain phenotype will increase if the individuals of this type have a higher fitness than the population average when interacting with the current distribution of agents; otherwise their population share will decrease. The population can be described by the state vector $x = (x_1, x_2, ..., x_n)$, with $0 <= x_i <= 1$ for all i and $\sum_i x_i = 1$, representing the fractions of the population belonging to each of the phenotypes or strategies. Now suppose the fitness of type i is given by the fitness function $f_i(x)$, and the average fitness of the population is given by $f^(x) = \sum_j [x_j f_j(x)]$ The population change over time can then be written as: $\frac{dx_i}{dt} = x_i[f_i(x) - f(x)]$ or,

$$\dot{x}_i = x_i \left[f_i(\mathbf{x}) - \bar{f}(\mathbf{x}) \right] \tag{1}$$

which is known as the single population replicator equation.

Let us now consider an example of a population playing the prisoner's dilemma with payoff tables shown in Figure 2. An individual playing the strategy $i = 1$, i.e. cooperate, on average encounters x_1 individuals also cooperating and x_2 individuals defecting. This means that the average fitness of an individual playing cooperate is $(Ax)_1 = 3x_1 + 0x_2$ Similarly, the average payoff of an individual playing defect is $(Ax)_2 = 5x_1 + 1x_2$. The payoff matrix A determines the payoff an individual receives when interacting with others. The state vector x describes the frequencies of all pure strategies within the population. Success of a strategy i is measured by the difference between its current payoff $(Ax)_i$ and the average payoff of the entire population xAx. Hence, strategies that perform better than average grow in population share and those performing worse than average diminish. We can now also plot the phase plot of the trajectories of the dynamical system, which will predict learning traces of various learning algorithms, for an extensive overview see [5]. An RL researcher or experimentalist can now easily investigate the directional field plot of the learning behavior described by various replicator dynamics models, providing insight into the equilibrium structure of games and their basins of attraction when various learning strategies are examined. Figure 3 shows the directional field plot for the prisoner's dilemma using equation 1.

On the one hand a population is a collection of individuals, each representing a certain phenotype, i.e., a pure strategy. An individual never changes its

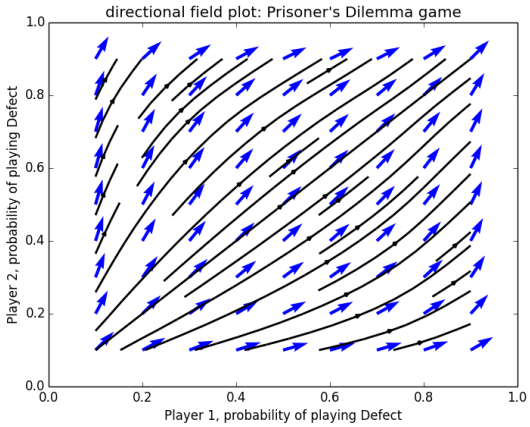

Fig. 3. Directional field plot of the replicator dynamics in the Prisoner's Dilemma Game. The Nash equilibrium is situated at the top right corner.

phenotype during the course of its lifetime. Individuals are randomly matched and play the game according to their predetermined phenotypes; subsequently, phenotypes replicate according to the realized payoffs. Thus phenotypes compete with each other, fitter strategies prevail while inferior strategies eventually die out. On the other hand, a **population** might also represent the behavior of a **particular agent**. The population shares reflect the current preferences over different strategies and thus defines the agent's policy. The asymmetric replicator dynamics provide a model for two learning agents pitted against each other and thus, two populations co-evolving.

The single population replicator dynamics are only applicable to symmetric games. An asymmetric two-player game comprises different payoff tables for the two players and possibly different action sets (e.g. matching pennies). Likewise we need two separate populations to describe the dynamics. At each time step a random individual from one population interacts with a randomly matched individual from the other population. Instead of one payoff matrix we will now have two payoff matrixes A and B, which are of size $m \times n$. m is the number of actions the row player can choose from and n the number of actions the column player can choose from. The state vectors of the two populations will now be denoted as x and y, and the dynamics are now specified by a coupled dynamical system consisting of $m + n$ equations. m for the replicators of x and n for the replicators of y. The fitness of an individual of population x playing strategy i against population y is $f_i(x) = (Ay)_i$, and the expected fitness of a random individual of x against y is $f(x) = x^T Ay$. Similarly we can compute the fitness for individuals of population y. For the two populations the replicator equations now look as follows:

$$\begin{aligned} \dot{x}_i &= x_i \left[(\mathbf{Ay})_i - \mathbf{x}^\top \mathbf{Ay} \right] \\ \dot{y}_i &= y_i \left[(\mathbf{x}^\top \mathbf{B})_i - \mathbf{x}^\top \mathbf{By} \right]. \end{aligned} \tag{2}$$

Note that a recent result shows how to decompose an asymmetric game into its symmetric counterparts (using replicator dynamics), allowing to discover the Nash structure of an asymmetric game using its symmetric counterparts, for details see [42].

There exist RD models of various reinforcement learning algorithms such as: Q-learning [43], lenient Q-learning [29, 28], regret minimization [19], FAQ-learning [18] etc. These are derived by constructing a continuous time limit of the difference equation of two consecutive updates of the respective learning update rule. Taking the limit for δt approaching zero of this difference equation, i.e. the time between the two updates becomes infinitesimally small, yields the RD model of the respective learning algorithm. These models can now be used by researchers to gain insight in the learning behavior by examining the respective phase-plots of the various dynamical systems in a specific game.

4.2 Opponent modeling tool

In contrast to replicator dynamics, which is a tool used by an MAL experimentalist, the second MAL tool we consider is one used by an agenet within an MAL system. An opponent model predicts the future actions of other agents in the systems, and may be given a priori or itself learned. In the latter case, it is one example of a "learning target within our taxonomy show in Figure 1.

A recently-published survey of methods for agents modeling other agents provides a comprehensive review of types of opponent modeling methods that can be used for constructing such a tool [1]. These methods include policy reconstruction, type-based reasoning, classification, plan recognition, recursive reasoning, graphical models, and group modeling.

An extended version of the current paper will describe opponent modelling in greater detail. For further details, we refer the reader to the survey [1], which compares and contrasts these methods in Table 1, and summarizes numerous examples from the literature in Tables 2-9.

5 Conclusion

The purpose of this paper has been to identify, compare, and contrast the main prevalent research paradigms within the multiagent learning literature. To this end, we begin with an overarching taxonomy of multiagent learning, as illustrated in Figure 1. We then identify three high-level types of agent learning scenarios — individual learning in which a relatively sophisticated agent learns at the individual level; population learning in which a population of cognitively-limited agents learn at the group level by using simple local interactions; and protocol learning in which the interaction mechanism among the agents is itself learned —

and then further subdivide them into the five paradigms specified in Section **??**. We then conclude with coverage of two classes of MAL tools in Section **??**: one for use by researchers or experimentalists to predict the dynamics of an MAL system, and one for use by the agents themselves.

While this paper provides a high-level classification of MAL paradigms, it does not survey the literature in any particular detail. We hope that the provided perspective and terminology will prove to be useful to the community for description of existing and future multiagent learning approaches.

References

1. Albrecht, S.V., Stone, P.: Autonomous agents modelling other agents: A comprehensive survey and open problems. Artif. Intell. 258, 66–95 (2018)
2. Altshuler, Y., Bruckstein, A.M.: Static and expanding grid coverage with ant robots: Complexity results. Theor. Comput. Sci. 412(35), 4661–4674 (2011)
3. Banerjee, A.: A simple model of herd behavior. Quarterly Journal of Economics 107, 797–817 (1992)
4. Barrett, S., Stone, P., Kraus, S.: Empirical evaluation of ad hoc teamwork in the pursuit domain. In: 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011), Taipei, Taiwan, May 2-6, 2011, Volume 1-3. pp. 567–574 (2011)
5. Bloembergen, D., Tuyls, K., Hennes, D., Kaisers, M.: Evolutionary dynamics of multi-agent learning: A survey. J. Artif. Intell. Res. 53, 659–697 (2015)
6. Broecker, B., Caliskanelli, I., Tuyls, K., Sklar, E.I., Hennes, D.: Hybrid insect-inspired multi-robot coverage in complex environments. In: Towards Autonomous Robotic Systems - 16th Annual Conference, TAROS 2015, Liverpool, UK, September 8-10, 2015, Proceedings. pp. 56–68 (2015)
7. Claus, C., Boutilier, C.: The dynamics of reinforcement learning in cooperative multiagent systems. In: Proceedings of the Fifteenth National Conference on Artificial Intelligence and Tenth Innovative Applications of Artificial Intelligence Conference, AAAI 98, IAAI 98, July 26-30, 1998, Madison, Wisconsin, USA. pp. 746–752 (1998)
8. Colorni, A., Dorigo, M., Maniezzo, V.: Distributed optimization by ant colonies. In: Varela, F.J., Bourgine, P. (eds.) Towards a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life, pp. 134–142. MIT Press, Cambridge, MA (1992)
9. Dorigo, M., Stützle, T.: Ant colony optimization. MIT Press (2004)
10. Fogel, D.B.: Evolving behaviors in the iterated prisoner's dilemma. Evolutionary Computation 1(1), 77–97 (1993)
11. Fogel, D.B.: Evolutionary computation - toward a new philosophy of machine intelligence. IEEE (1995)
12. Galef, B.: Imitation in animals: History, denition, and interpretation of data from the psychological laboratory. In: Zentall, T., Galef, B. (eds.) Social Learning: Psychologicand Biological Perspectives. Lawrence Erlbaum Associates, Hillsdale, NJ (1988)
13. Gatti, N., Restelli, M.: Sequence-form and evolutionary dynamics: Realization equivalence to agent form and logit dynamics. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA. pp. 509–515 (2016)

14. Genter, K.L., Stone, P.: Influencing a flock via ad hoc teamwork. In: Swarm Intelligence - 9th International Conference, ANTS 2014, Brussels, Belgium, September 10-12, 2014. Proceedings. pp. 110–121 (2014)
15. Gintis, H.: Game Theory Evolving. University Press, Princeton NJ, 2nd edn. (2009)
16. Hofbauer, J., Sigmund, K.: Evolutionary games and population dynamics. Cambridge University Press (1998)
17. Hu, J., Wellman, M.P.: Nash q-learning for general-sum stochastic games. Journal of Machine Learning Research 4, 1039–1069 (2003)
18. Kaisers, M., Tuyls, K.: Frequency adjusted multi-agent q-learning. In: 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), Toronto, Canada, May 10-14, 2010, Volume 1-3. pp. 309–316 (2010)
19. Klos, T., van Ahee, G.J., Tuyls, K.: Evolutionary dynamics of regret minimization. In: Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2010, Barcelona, Spain, September 20-24, 2010, Proceedings, Part II. pp. 82–96 (2010)
20. Knudson, M., Tumer, K.: Policy transfer in mobile robots using neuro-evolutionary navigation. In: Genetic and Evolutionary Computation Conference, GECCO '12, Philadelphia, PA, USA, July 7-11, 2012, Companion Material Proceedings. pp. 1411–1412 (2012)
21. Laland, K., Richerson, P., Boyd, R.: Animal social learning: Toward a new theoretical approach. In: Klopfer, P., Bateson, P., Thomson, N. (eds.) Perspectives in Ethology. Plenum press, New York (1993)
22. Lanctot, M.: Further developments of extensive-form replicator dynamics using the sequence-form representation. In: International conference on Autonomous Agents and Multi-Agent Systems, AAMAS '14, Paris, France, May 5-9, 2014. pp. 1257–1264 (2014)
23. Littman, M.: Markov games as a framework for multi-agent reinforcement learning. In: Proceedings of the Eleventh International Conference on Machine Learning, p 157 - 163 (1994)
24. Manderick, B., Spiessens, P.: Fine-grained parallel genetic algorithms. In: Proceedings of the 3rd International Conference on Genetic Algorithms, George Mason University, Fairfax, Virginia, USA, June 1989. pp. 428–433 (1989)
25. Maynard Smith, J., Price, G.R.: The logic of animal conflict. Nature 246(2), 15–18 (1973)
26. Mitchell, M.: An introduction to genetic algorithms. MIT Press (1998)
27. Mitchell, T.M.: Machine learning. McGraw Hill series in computer science, McGraw-Hill (1997)
28. Palmer, G., Tuyls, K., Bloembergen, D., Savani, R.: Lenient multi-agent deep reinforcement learning. Accepted for AAMAS 2018 (2018)
29. Panait, L., Tuyls, K., Luke, S.: Theoretical advantages of lenient learners: An evolutionary game theoretic perspective. Journal of Machine Learning Research 9, 423–457 (2008)
30. Pardoe, D., Stone, P., Saar-Tsechansky, M., Keskin, T., Tomak, K.: Adaptive auction mechanism design and the incorporation of prior knowledge. INFORMS Journal on Computing 22(3), 353–370 (2010)
31. Pardoe, D., Stone, P., Saar-Tsechansky, M., Tomak, K.: Adaptive mechanism design: a metalearning approach. In: Proceedings of the 8th International Conference on Electronic Commerce: The new e-commerce - Innovations for Conquering Current Barriers, Obstacles and Limitations to Conducting Successful Business on the Internet, 2006, Fredericton, New Brunswick, Canada, August 13-16, 2006. pp. 92–102 (2006)

32. Paredis, J.: Coevolutionary computation. Artificial Life 2(4), 355–375 (1995)
33. Parkes, D.C.: On Learnable Mechanism Design, p. 107–131. Springer-Verlag (2004)
34. Sandholm, T.: Perspectives on multiagent learning. Artif. Intell. 171(7), 382–391 (2007)
35. Saravanan, N., Fogel, D.B.: Evolving neurocontrollers using evolutionary programming. In: Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence, Orlando, Florida, USA, June 27-29, 1994. pp. 217–222 (1994)
36. Shoham, Y., Leyton-Brown, K.: Multiagent Systems - Algorithmic, Game-Theoretic, and Logical Foundations. Cambridge University Press (2009)
37. Shoham, Y., Powers, R., Grenager, T.: If multi-agent learning is the answer, what is the question? Artif. Intell. 171(7), 365–377 (2007)
38. Stone, P.: Multiagent learning is not the answer. it is the question. Artif. Intell. 171(7), 402–405 (2007)
39. Stone, P., Veloso, M.M.: Multiagent systems: A survey from a machine learning perspective. Auton. Robots 8(3), 345–383 (2000)
40. Tuyls, K., Hoen, P.J., Vanschoenwinkel, B.: An evolutionary dynamical analysis of multi-agent learning in iterated games. Autonomous Agents and Multi-Agent Systems 12(1), 115–153 (2006)
41. Tuyls, K., Parsons, S.: What evolutionary game theory tells us about multiagent learning. Artif. Intell. 171(7), 406–416 (2007)
42. Tuyls, K., Pérolat, J., Lanctot, M., Ostrovski, G., Savani, R., Leibo, J.Z., Ord, T., Graepel, T., Legg, S.: Symmetric decomposition of asymmetric games. Scientific Reports 8(1), 1015 (2018)
43. Tuyls, K., Verbeeck, K., Lenaerts, T.: A selection-mutation model for q-learning in multi-agent systems. In: The Second International Joint Conference on Autonomous Agents & Multiagent Systems, AAMAS 2003, July 14-18, 2003, Melbourne, Victoria, Australia, Proceedings. pp. 693–700 (2003)
44. Tuyls, K., Weiss, G.: Multiagent learning: Basics, challenges, and prospects. AI Magazine 33(3), 41–52 (2012)
45. Urzelai, J., Floreano, D.: Evolutionary robotics: Coping with environment change. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '00), Las Vegas, Nevada, USA, July 8-12, 2000. pp. 941–948 (2000)
46. Weibull, J.W.: Evolutionary game theory. MIT press (1997)
47. Wooldridge, M.J.: Introduction to multiagent systems. Wiley (2002)
48. Wunder, M., Littman, M.L., Babes, M.: Classes of multiagent q-learning dynamics with epsilon-greedy exploration. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel. pp. 1167–1174 (2010)