Delft University of Technology

Delft Center for Systems and Control

Technical report 06-025

Multi-agent reinforcement learning: A survey*

L. Buşoniu, R. Babuška, and B. De Schutter

If you want to cite this report, please use the following reference instead:
L. Buşoniu, R. Babuška, and B. De Schutter, "Multi-agent reinforcement learning: A survey," *Proceedings of the 9th International Conference on Control, Automation, Robotics and Vision (ICARCV 2006)*, Singapore, pp. 527–532, Dec. 2006.

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.51.19 (secretary)
fax: +31-15-278.66.79
URL: `http://www.dcsc.tudelft.nl`

*This report can also be downloaded via `http://pub.deschutter.info/abs/06_025.html`

Multi-Agent Reinforcement Learning: A Survey

Lucian Buşoniu Robert Babuška Bart De Schutter
Delft Center for Systems and Control
Delft University of Technology
2628 CD Delft, The Netherlands
Email: {i.l.busoniu,b.deschutter,r.babuska}@tudelft.nl

***Abstract*—Multi-agent systems are rapidly finding applications in a variety of domains, including robotics, distributed control, telecommunications, economics. Many tasks arising in these domains require that the agents learn behaviors online. A significant part of the research on multi-agent learning concerns reinforcement learning techniques. However, due to different viewpoints on central issues, such as the formal statement of the learning goal, a large number of different methods and approaches have been introduced. In this paper we aim to present an integrated survey of the field. First, the issue of the multi-agent learning goal is discussed, after which a representative selection of algorithms is reviewed. Finally, open issues are identified and future research directions are outlined.**

***Keywords*—multi-agent systems, reinforcement learning, game theory, distributed control**

I. Introduction

Multi-agent systems are rapidly finding applications in a wide variety of domains such as robotic teams, distributed control, collaborative decision support systems, data mining, etc. Although the individual agents can be programmed to exhibit some basic behaviors, many tasks require that agents learn new behaviors online, such that the performance of the agent or of the whole multi-agent system gradually improves.

A reinforcement learning (RL) agent learns by interacting with its environment, using a scalar reward signal as performance feedback [1]. The simplicity and generality of this setting make it attractive also for multi-agent learning. However, the main challenge in multi-agent RL (MARL) is that each learning agent must explicitly consider other learning (and therefore nonstationary) agents, and coordinate its behavior with theirs, such that a coherent joint behavior results.

Over the last years, many algorithms addressing this problem were proposed. These algorithms can be classified along several dimensions: the type of task they address, the homogeneity of the agent team, assumptions on the agents' knowledge and inputs, etc. Figure 1 organizes the algorithms by their field of origin, regarding MARL as a fusion of temporal-difference RL, game theory, and more general direct policy search techniques.

MARL surveys typically review the field from a game-theoretic perspective and focus on the central part of Fig. 1 [2]–[5]. The work in [6] is more general, but looking mainly at *cooperative* multi-agent learning.

The aim of our survey is to take a broader approach and give an overall view of the field. We address the different viewpoints on the learning goal in MARL, which leads to certain diversity in the set of MARL algorithms and techniques. Finally, we identify open issues in the field, and outline control-theoretic ways to address some of these issues.

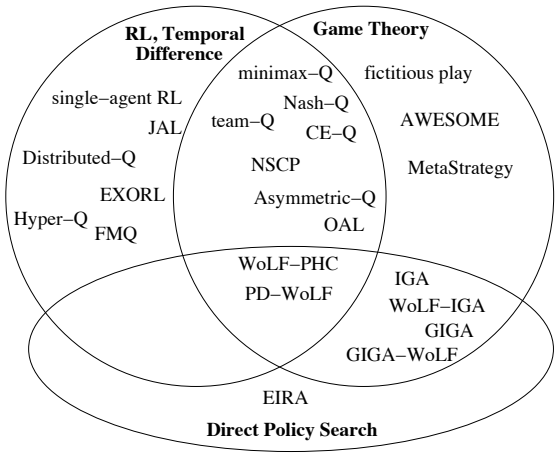

Figure 1. MARL encompasses temporal-difference reinforcement learning, game theory and direct policy search techniques.

This paper is organized as follows. Section II introduces the necessary background. Section III addresses the problem of a suitable multi-agent learning goal, and Section IV reviews a representative selection of the MARL algorithms, classifying them by the type of task they solve. Section V concludes the paper.

II. Background

Single-agent RL concepts are given first, followed by their extension to the multi-agent case.

A. The single-agent case

Definition 1: A *Markov decision process* is a tuple $\langle X, U, f, \rho \rangle$ where: X is the discrete set of environment states, U is the discrete set of agent actions, $f : X \times U \times X \to [0, 1]$ is the state transition probability distribution, and $\rho : X \times U \times X \to \mathbb{R}$ is the reward function.

As a result of action u_k, the environment changes state from x_k, ending up in x_{k+1} with probability $f(x_k, u_k, x_{k+1})$. The agent receives (possibly delayed) feedback on its performance via the scalar reward signal $r_{k+1} \in \mathbb{R}$, according to ρ:

$r_{k+1} = \rho(x_k, u_k, x_{k+1})$. For deterministic models, the transition distributions is replaced by a function, $\overline{f} : X \times U \to X$. The reward is then completely determined by the current state and action, $\overline{\rho} : X \times U \to \mathbb{R}$. The agent chooses actions according to its *policy* that may be either stochastic, $h : X \times U \to [0, 1]$, or deterministic, $\overline{h} : X \to U$. A policy is called stationary if it does not change over time.

The agent's goal is to maximize, at each time step k, the discounted return:

$$R_k = \sum\nolimits_{j=0}^{\infty} \gamma^j r_{k+j+1}, \tag{1}$$

where $\gamma \in (0, 1)$ is the discount factor. The *action-value function* (Q-function), $Q^h : X \times U \to \mathbb{R}$, is the expected return of a state-action pair under a given policy: $Q^h(x, u) = E\{R_k \,|\, x_k = x, u_k = u, h\}$. The agent can maximize its return by first computing the *optimal* Q-function, defined as $Q^*(x, u) = \max_h Q^h(x, u)$, and then choosing actions by the greedy policy $h^*(x) = \arg\max_u Q^*(x, u)$, which is optimal.

The *Q-learning* algorithm iteratively estimates Q^* by interaction with the environment, using observed rewards r_{k+1} and pairs of subsequent states x_k, x_{k+1} [7]:

$$\begin{aligned} Q_{k+1}(x_k, u_k) = Q_k(x_k, u_k) + \\ \alpha\big[r_{k+1} + \gamma \max_{u'} Q(x_{k+1}, u') - Q_k(x_k, u_k)\big], \end{aligned} \tag{2}$$

where $\alpha \in (0, 1]$ is the learning rate. The sequence Q_k provably converges to Q^* under certain conditions, including that the agent keeps trying all actions in all states with nonzero probability [7]. This means that the agent must sometimes *explore*, i.e., perform other actions than dictated by the current greedy policy.

B. The multi-agent case

Definition 2: A *stochastic game (SG)* (Markov game) is a tuple $\langle A, X, \{U_i\}_{i \in A}, f, \{\rho_i\}_{i \in A} \rangle$ where: $A = \{1, \ldots, n\}$ is the set of n agents, X is the discrete set of environment states, $\{U_i\}_{i \in A}$ are the discrete sets of actions available to the agents, yielding the joint action set $\boldsymbol{U} = \times_{i \in A} U_i$, $f : X \times \boldsymbol{U} \times X \to [0, 1]$ is the state transition probability distribution, and $\rho_i : X \times \boldsymbol{U} \times X \to \mathbb{R}, i \in A$ are the reward functions of the agents.

Note that the state transitions, agent rewards $r_{i,k+1}$, and thus also the agent returns $R_{i,k}$, depend on the *joint action* $\boldsymbol{u}_k = [u_{1,k}, \ldots, u_{n,k}]^{\mathrm{T}}, \boldsymbol{u}_k \in \boldsymbol{U}$, $u_{i,k} \in U_i$. The policies $h_i : X \times U_i \to [0, 1]$ together form the joint policy $\boldsymbol{h}$. The Q-function of each agent depends on the joint action and is conditioned on the joint policy, $Q_i^{\boldsymbol{h}} : X \times \boldsymbol{U} \to \mathbb{R}$.

If $X = \emptyset$, the SG reduces to a static game. A static game, when played repeatedly by the same agents, is called a repeated game. If $\rho_1 = \cdots = \rho_n$, the SG is fully cooperative. If $n = 2$ and $\rho_1 = -\rho_2$, the SG is fully competitive.

In a static game, the policy loses the state argument and transforms into a strategy $h_i : U_i \to [0, 1]$. Similarly, a policy conditioned on a given state x yields a strategy. The best response of agent i to a set of opponent strategies is a strategy that achieves the maximum expected reward given the opponents' strategies. A *Nash equilibrium* is a set of strategies such that each is a best-response to the others.

The purpose of *coordination* is to make sure that all agents coherently choose their part of a desirable joint policy. In a game with multiple equilibria, coordination boils down to equilibrium selection, where the agents need to consistently pick their part of the same equilibrium.

Table I
STABILITY AND ADAPTATION IN MULTI-AGENT LEARNING.

Stability property	Adaptation property	Relevant work
convergence	rationality	[8], [9]
convergence	no-regret	[10]
opponent-independent	opponent-aware	[5], [11]
prediction	rationality	[4]
—	{ targeted optimality, compatibility, safety	[2], [12]

III. MULTI-AGENT LEARNING GOAL

In fully cooperative SGs, the common return can be jointly maximized. In other cases, however, specifying a good MARL goal is difficult, because the agents' returns are correlated and cannot be maximized independently.

In this section, we review the learning goals put forward in the literature. These goals incorporate *stability* of the learning process on the one hand, and *adaptation* to the dynamic behavior of the other agents on the other hand. Stability essentially means the convergence to stationary policies, whereas adaptation ensures that performance is maintained or improved. The goals typically formulate conditions for static games, but some can be extended to dynamic games by requiring that conditions are satisfied stage-wise for all the states of the dynamic game.

Convergence to equilibria is a basic stability requirement, postulated already in the early MARL literature [13], [14]. Nash equilibria are most frequently used. However, concerns have been voiced regarding their usefulness [2], due to the unclear link between stage-wise convergence to Nash equilibria and performance in the dynamic game.

In [8], *rationality* is added as an adaptation criterion. It requires the agent to converge to a best response when other agents remain stationary. An alternative to rationality is the concept *no-regret*, which prevents the learner from 'being exploited' by the other agents [10].

Targeted optimality / compatibility / safety [12] replace convergence with adaptation requirements, in the form of average reward bounds for three classes of opponents: those deemed interesting (targeted), those using the learner's algorithm, and remaining opponents.

Table I summarizes the desirable properties of MARL algorithms, as discussed above and in the literature. Algorithms focused on stability only are typically *independent* of other agents; those that consider adaptation clearly need to be *aware* of their behavior. If only adaptation is considered and stability is disregarded, algorithms are *tracking* the behavior of other agents.

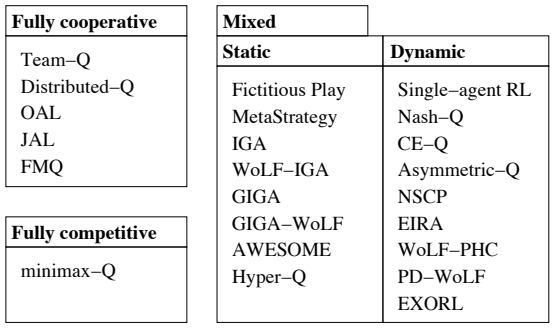

Figure 2. Taxonomy of MARL algorithms by the type of task they address.

IV. Multi-Agent Reinforcement Learning Algorithms

The MARL algorithms are organized here by the type of task they address: fully cooperative, fully competitive, and mixed tasks. Some algorithms can only solve static games; for mixed tasks, these algorithms are treated separately, as depicted in Fig. 2.

A. Fully cooperative tasks

In a fully cooperative SG, $\rho_1 = \ldots = \rho_n$ and the learning goal is to maximize the common discounted return. If a centralized controller is available, the task reduces to a Markov decision process whose action space is the joint action space of the SG. The goal can be achieved by learning the optimal joint-action values with Q-learning:

$$Q_{k+1}(x_k, \boldsymbol{u}_k) = Q_k(x_k, \boldsymbol{u}_k) + \alpha\big[r_{k+1} + \gamma \max_{\boldsymbol{u}'} Q(x_{k+1}, \boldsymbol{u}') - Q_k(x_k, \boldsymbol{u}_k)\big], \quad (3)$$

and using the greedy policy. If the agents are independent decision makers, a coordination problem arises even if all the agents use the same algorithm to learn in parallel the common optimal Q-function. In principle, they could then use the greedy policy to maximize the common return. However, the greedy action selection mechanism breaks ties randomly, which means that in the absence of additional mechanisms, different agents may break a tie in different ways, and the resulting joint action may be suboptimal.

Coordination-free methods. The *Team Q-learning* algorithm [11] solves the problem by assuming that the optimal joint actions are unique (which will rarely be the case). Then, (3) can directly be used.

The *Distributed Q-learning* algorithm [15] solves the cooperative task without assuming coordination, however it is only valid in the deterministic setting. Each agent i maintains an explicit policy $h_i(x)$, and a local Q-function $Q_i(x, u_i)$, depending only on its own action. Both are updated only in the direction that increases Q_i:

$$Q_{i,k+1}(x_k, u_{i,k}) = \max\big\{Q_{i,k}(x_k, u_{i,k}), r_{k+1} + \gamma \max_{u_i} Q_{i,k}(x_{k+1}, u_i)\big\} \quad (4)$$

$$h_{i,k+1}(x_k) = \begin{cases} u_{i,k} & \text{if } \max_{u_i} Q_{i,k+1}(x_k, u_i) \neq \max_{u_i} Q_{i,k}(x_k, u_i) \\ h_{i,k}(x_k) & \text{otherwise} \end{cases} \quad (5)$$

Under the conditions that $Q_{i,0} = 0$ and the common reward function is positive, the policies of the agents provably converge to the optimal joint policy $\boldsymbol{h}^*$.

Direct coordination methods. A more general approach to solving the coordination problem is to make sure that ties are broken by all agents in the same way. This clearly requires that random action choices are somehow coordinated or negotiated:

- *Social conventions* [16] and *roles* [17] restrict the action choices of the agents.
- *Coordination graphs* simplify coordination when the global Q-function can be additively decomposed in local Q-functions which only depend on the actions of a subset of agents [18], [19].
- *Communication* is used to negotiate action choices, either alone or in combination with the above techniques, as in e.g., [20], [21].

Indirect coordination methods bias action selection toward actions that promise to yield better values, and thus steer the agents toward coordination. *Joint Action Learners* (JAL) [22] employ empirically learned models of the other agents' behavior. The *Frequency Maximum Q-value* (FMQ) heuristic [23] is based on the frequency with which actions yielded good values in the past. In *Optimal Adaptive Learning* (OAL), the bias is towards recently chosen Nash equilibria [24]. Using an additional mechanism to guarantee that optimal Nash equilibria are eventually selected, OAL provably converges to optimal joint policies (at the cost of increased complexity). JAL and FMQ only work in static games.

Remarks and open issues. Coordination-free methods are teammate-independent, whereas indirect coordination methods are teammate-aware. Direct coordination methods are teammate-independent if they rely on common-knowledge assumptions, and teammate-aware if they use negotiation.

To improve the applicability of algorithms in practice, effort must be invested in their scalability and their robustness to uncertain or incomplete observations. Coordination-free methods are especially vulnerable to uncertain observations.

Communication has the potential to provide straightforward, efficient solutions to the coordination problem in MARL. This potential has not yet been fully exploited.

B. Fully competitive tasks

In a fully competitive SG (for two agents, $\rho_1 = -\rho_2$), the minimax principle can be applied: maximize one's benefit under the assumption that the opponent will always act so as to minimize it. The resulting algorithm is *minimax-Q* [11], given

here for agent 1:

$$h_{1,k}(x_k, \cdot) = \arg \mathbf{m}_1(Q_k, x_k) \tag{6}$$

$$Q_{k+1}(x_k, u_{1,k}, u_{2,k}) = Q_k(x_k, u_{1,k}, u_{2,k}) + \alpha\big[r_{k+1} + \gamma \mathbf{m}_1(Q_k, x_{k+1}) - Q_k(x_k, u_{1,k}, u_{2,k})\big] \tag{7}$$

where $\mathbf{m}_1$ is the minimax return of agent 1:

$$\mathbf{m}_1(Q, x) = \max_{h_1(x,\cdot)} \min_{u_2} \sum_{u_1} h_1(x, u_1) Q(x, u_1, u_2) \tag{8}$$

The Q-table is not subscripted by the agent index, because the equations use the implicit assumption that $Q_1 = Q = -Q_2$. Minimax-Q is truly opponent-independent, because even if the minimax optimization has multiple solutions, any of them will achieve at least the minimax return regardless of what the opponent is doing.

If the learner has a model of the opponent's policy (i.e., is opponent-aware), it might actually do better than the minimax return (8). An opponent model can be learned using e.g., the *M** algorithm [25].

C. Mixed tasks

In the general case, no constraints are imposed on the reward functions of the agents. This is of course appropriate for self-interested agents, but even cooperating agents may encounter situations where their immediate interests are in conflict, e.g., when they need to compete for some resource. The game-theoretic elements, especially the concept of equilibrium, are most influential in this category. When multiple equilibria exist in a particular state of an SG, the equilibrium selection problem arises: the agents need to consistently pick their part of the same equilibrium.

MARL algorithms for static, repeated games are first presented, followed by those addressing dynamic SGs.

1) Repeated games: In repeated games, one of the essential properties of RL, delayed reward, is lost. However, the learning problem is still nonstationary due to the dynamic behavior of the agents that play the repeated game. This is why methods in this category always consider adaptation to the other agents.

Algorithms in this category typically require that the agents know the task model (i.e., the reward function), and assume observable actions (some of them, even observable strategies).

Agent-tracking methods adapt to learned models of the opponents' behavior. *Fictitious play* uses a best response to these empirical models, whereas *Hyper-Q* incorporates the models in the state vector, and learns on their basis [26]. The *MetaStrategy* combines modified versions of fictitious play, minimax and a game-theoretic strategy called Bully to achieve the targeted optimality / compatibility / safety triple goal [12]. These methods do not necessarily converge to stationary strategies.

Agent-aware methods target convergence as well. The *AWESOME* algorithm uses fictitious play, but monitors the other agents and switches to a precomputed Nash equilibrium when it concludes that they are adapting (hence the name: Adapt When Everyone is Stationary, Otherwise Move to Equilibrium) [9].

Some methods in the area of direct policy search use gradient update rules that guarantee convergence in specific classes of games: *Infinitesimal Gradient Ascent* (IGA) [27], *Win-or-Learn-Fast IGA* (WoLF-IGA) [8], *Generalized IGA* (GIGA) [28], and *GIGA-WoLF* [10].

Remarks and open issues. A perfect task model is rarely available in practice. Thus, versions of the methods above that work with imperfect and / or learned models would be interesting (e.g., GIGA-WoLF provides a heuristic extension that does that).

Static, repeated games represent a limited set of applications, among which are included negotiation, auctions, and bartering. The algorithms above provide valuable theoretical results; these results should be however extended to the dynamical case in order to become interesting for more general classes of applications (e.g., WoLF-PHC, discussed in Sec. IV-C2).

2) Dynamic stochastic games: Mixed, dynamic tasks correspond to the unrestricted SG, which exhibits all the MARL challenges: delayed reward, nonstationary agents, and conflicting goals.

Single-agent RL can be directly applied to the multi-agent case [29]. However, the nonstationarity of the MARL problem invalidates most of the single-agent RL theoretical results. Single-agent RL might not work when agents severely interfere with one another. As they do not take into account the behavior of the other agents, single-agent methods are agent-independent.

Despite its limitations, this approach found applications, mainly because of its simplicity [30], [31]. In applications, information about other agents is typically encoded in the learner's input, thus indirectly enabling it to make decisions on the basis of their behavior.

Agent-independent methods share a common structure based on Q-learning, where policies and state values are computed with game-theoretic solvers for the stage games arising in the states of the SG [5], [14]. Denoting by $\{Q_{\cdot,k}(x, \cdot)\}$ the stage game arising in state x and given by all the agents' Q-functions at time k:

$$h_{i,k}(x, \cdot) = \mathbf{solve}_i \{Q_{\cdot,k}(x_k, \cdot)\} \tag{9}$$

$$Q_{i,k+1}(x_k, \boldsymbol{u}_k) = Q_{i,k}(x_k, \boldsymbol{u}_k) + \alpha\big[r_{i,k+1} + \gamma \cdot \mathbf{eval}_i \{Q_{\cdot,k}(x_{k+1}, \cdot)\} - Q_{i,k}(x_k, \boldsymbol{u}_k)\big] \tag{10}$$

$\mathbf{solve}_i$ returns the i'th agent's part of some type of equilibrium (a strategy), and $\mathbf{eval}_i$ gives the agent's expected return at this equilibrium. When the solution of $\mathbf{solve}$ is not unique, the equilibrium selection problem arises. The goal is the convergence to an equilibrium in every state.

The updates use the Q-tables of all the agents. So, each agent needs to model the Q-tables of the other agents. It can do that by applying (10). This requires two assumptions: that all agents use the same algorithm, and that all actions and rewards are observable.

A particular instance of $\mathbf{solve}$ and $\mathbf{eval}$ for e.g., *Nash Q-*

learning [13] is:

$$\begin{cases} \mathbf{eval}_i \{Q_{\cdot,k}(x,\cdot)\} & = V_i(x, \mathbf{NE}\{Q_{\cdot,k}(x,\cdot)\}) \\ \mathbf{solve}_i \{Q_{\cdot,k}(x,\cdot)\} & = \mathbf{NE}_i \{Q_{\cdot,k}(x,\cdot)\} \end{cases} \tag{11}$$

where $\mathbf{NE}$ computes a Nash equilibrium, and $\mathbf{NE}_i$ is agent i's strategy component of this equilibrium, and $V_i(x, \mathbf{NE}\{Q_{\cdot,k}(x,\cdot)\})$ is the expected return of agent i from x under this equilibrium. Instantiations of *correlated Q-learning* (CE-Q) [14] or *asymmetric Q-learning* [32] can be performed in a similar fashion, by using correlated or Stackelberg (leader-follower) equilibria, respectively. For asymmetric-Q, the follower does not need to model the leader's Q-table; however, the leader must know how the follower chooses its actions.

Agent-tracking methods adapt to learned models of the other agents' nonstationary policies without considering convergence. Actions have to be observable. The *Non-Stationary Converging Policies* (NSCP) algorithm computes a best-response to the models and uses it in estimating value functions [33].

Agent-aware methods typically do consider convergence. *Win-or-Learn-Fast Policy Hill-Climbing* (WoLF-PHC) combines the basic Q-learning update rule (2) with a gradient-based policy update originating in WoLF-IGA [8]:

$$h_{i,k+1}(x_k,u_i) = h_{i,k}(x_k,u_i) + \begin{cases} \delta_{i,k} & \text{if } u_i = \arg\max_{\tilde{u}_i} Q_{i,k+1}(x_k,\tilde{u}_i) \\ -\frac{\delta_{i,k}}{|U_i|-1} & \text{otherwise} \end{cases} \tag{12}$$

The gradient step $\delta_{i,k}$ is δ_l when the agent is losing and δ_w when it is winning, with $\delta_\mathrm{l} > \delta_\mathrm{w}$. The win criterion is based either on a comparison of an average policy with the current one, in the original version of WoLF-PHC, or on the second-order difference of policy elements, in PD-WoLF [34]. The rationale is that the agent should escape fast from losing situations, while adapting cautiously when it is winning, in order to encourage convergence.

The *Extended Optimal Response* (EXORL) heuristic applies a similar idea in two-agent tasks: the policy update is biased in a way that minimizes the other agent's incentive to deviate from its current policy [35].

Environment-Independent Reinforcement Acceleration (EIRA) pushes policies onto, and pops policies from, a policy stack in such a way that long-term reinforcement improvements are guaranteed [36]. EIRA does not make any assumptions on the environment and on the other agents. In this sense, it is very general. However, it may not be able to take advantage of the task's structure.

Remarks and open issues. Game theory induces a bias toward static (stage-wise) solutions in the dynamic case – see e.g., equations (9)–(10) and the state-wise win/lose criteria in WoLF. However, the suitability of such state-wise solutions in the context of the dynamic task is currently unclear [2], [6].

Agents in mixed SGs are generally regarded as self-interested. Consequently, cooperative coordination techniques, such as communication, social conventions, or roles, are not investigated. However, in many mixed tasks the agents are cooperative, with competition arising in certain situations such as when they compete for a resource. In these tasks, cooperative coordination methods are a viable alternative.

Many algorithms for mixed SGs suffer from scalability issues and are sensitive to imperfect observations; the latter holds especially for agent-independent methods.

V. Conclusion and Future Perspectives

We have reviewed the challenges of multi-agent reinforcement learning, the methods to address them, and we have provided specific conclusions and open issues for each class of methods. More general open problems are given next.

First, the stage-wise application of game-theoretic techniques may not be the most suitable approach, given that the environment and the behavior of learning agents are generally dynamic processes. So far, game-theory-based analysis has only been applied to the learning dynamics [3], [37]. We expect that tools developed in the area of robust control will play an important role in the analysis and synthesis of the learning process as a whole (i.e., the environment and the learning dynamics). In addition, this framework can incorporate prior knowledge on bounds for imperfect observations, such as noise-corrupted variables.

Second, the issue of a suitable learning goal requires additional work. MARL goals are typically formulated in terms of static games. Their extension to dynamic tasks is not always straightforward or even possible. If an extension via stage games is possible, the relationship between the extended goals and performance in the dynamic task is not clear, and is not made explicit in the literature. This holds for stability requirements, like convergence to equilibria, as well as for adaptation requirements, like rationality.

Stability of the learning process is desirable, because the behavior of stable agents is more amenable to analysis and meaningful performance guarantees. Adaptation to the other agents is desirable because their dynamics are generally unpredictable. Therefore, a good multi-agent learning goal must include both components. This means that MARL algorithms should not be purely agent-independent nor purely agent-tracking. The control-theoretic concept of robustness can help integrate stability and adaptation into a unified goal. If a learning algorithm is robustly stable with respect to nonstationarity in the other agents, it will converge while allowing for bounded changes in the behavior of these agents.

Moreover, from a practical viewpoint, a realistic learning goal should include bounds on the transient performance, in addition to the usual asymptotic requirements. Examples of such bounds include maximum time constraints for reaching a desired performance level, or a lower bound on instantaneous performance levels. First steps in this direction have been taken in [10], [12].

In our view, significant progress in the field of multi-agent learning can be achieved by a more intensive cross-fertilization between the fields of machine learning, game theory, and control theory.

ACKNOWLEDGEMENT

This research is financially supported by Senter, Ministry of Economic Affairs of the Netherlands within the BSIK-ICIS project "Interactive Collaborative Information Systems" (grant no. BSIK03024).

REFERENCES

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

[2] Y. Shoham, R. Powers, and T. Grenager, "Multi-agent reinforcement learning: A critical survey," Computer Science Dept., Stanford University, California, US, Tech. Rep., 16 May 2003.

[3] K. Tuyls and A. Nowé, "Evolutionary game theory and multi-agent reinforcement learning," *The Knowledge Engineering Review*, vol. 20, no. 1, pp. 63–90, 2005.

[4] G. Chalkiadakis, "Multiagent reinforcement learning: Stochastic games with multiple learning players," Dept. of Computer Science, University of Toronto, Canada, Tech. Rep., 25 March 2003, URL: http://www.cs.toronto.edu/~gehalk/DepthReport/DepthReport.ps.

[5] M. Bowling, "Multiagent learning in the presence of agents with limitations," Ph.D. dissertation, Computer Science Dept., Carnegie Mellon University, Pittsburgh, US, May 2003.

[6] L. Panait and S. Luke, "Cooperative multi-agent learning: The state of the art," *Autonomous Agents and Multi-Agent Systems*, vol. 11, no. 3, pp. 387–434, November 2005.

[7] C. J. C. H. Watkins and P. Dayan, "Technical note: Q-learning," *Machine Learning*, vol. 8, pp. 279–292, 1992.

[8] M. Bowling and M. Veloso, "Multiagent learning using a variable learning rate," *Artificial Intelligence*, vol. 136, no. 2, pp. 215–250, 2002.

[9] V. Conitzer and T. Sandholm, "AWESOME: A general multiagent learning algorithm that converges in self-play and learns a best response against stationary opponents," in *Proceedings Twentieth International Conference on Machine Learning (ICML-03)*, Washington, US, 21–24 August 2003, pp. 83–90.

[10] M. Bowling, "Convergence and no-regret in multiagent learning," in *Advances in Neural Information Processing Systems 17 (NIPS-04)*, Vancouver, Canada, 13–18 December 2004, pp. 209–216.

[11] M. L. Littman, "Value-function reinforcement learning in Markov games," *Journal of Cognitive Systems Research*, vol. 2, pp. 55–66, 2001.

[12] R. Powers and Y. Shoham, "New criteria and a new algorithm for learning in multi-agent systems," in *Advances in Neural Information Processing Systems 17 (NIPS-04)*, Vancouver, Canada, 2004, pp. 1089–1096.

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

[14] A. Greenwald and K. Hall, "Correlated-Q learning," in *Proceedings Twentieth International Conference on Machine Learning (ICML-03)*, Washington, US, 21–24 August 2003, pp. 242–249.

[15] M. Lauer and M. Riedmiller, "An algorithm for distributed reinforcement learning in cooperative multi-agent systems," in *Proceedings Seventeenth International Conference on Machine Learning (ICML-00)*, Stanford University, US, 29 June – 2 July 2000, pp. 535–542.

[16] C. Boutilier, "Planning, learning and coordination in multiagent decision processes," in *Proceedings Sixth Conference on Theoretical Aspects of Rationality and Knowledge (TARK-96)*, De Zeeuwse Stromen, The Netherlands, 17–20 March 1996, pp. 195–210.

[17] M. T. J. Spaan, N. Vlassis, and F. C. A. Groen, "High level coordination of agents based on multiagent Markov decision processes with roles," in *Workshop on Cooperative Robotics, 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-02)*, Lausanne, Switzerland, 1 October 2002, pp. 66–73.

[18] C. Guestrin, M. G. Lagoudakis, and R. Parr, "Coordinated reinforcement learning," in *Proceedings Nineteenth International Conference on Machine Learning (ICML-02)*, Sydney, Australia, 8–12 July 2002, pp. 227–234.

[19] J. R. Kok, M. T. J. Spaan, and N. Vlassis, "Non-communicative multi-robot coordination in dynamic environment," *Robotics and Autonomous Systems*, vol. 50, no. 2–3, pp. 99–114, 2005.

[20] N. Vlassis, "A concise introduction to multiagent systems and distributed AI," University of Amsterdam, The Netherlands, Tech. Rep., September 2003, URL: http://www.science.uva.nl/~vlassis/cimasdai/cimasdai.pdf.

[21] F. Fischer, M. Rovatsos, and G. Weiss, "Hierarchical reinforcement learning in communication-mediated multiagent coordination," in *Proceedings 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-04)*, New York, US, 19–23 August 2004, pp. 1334–1335.

[22] C. Claus and C. Boutilier, "The dynamics of reinforcement learning in cooperative multiagent systems," in *Proceedings 15th National Conference on Artificial Intelligence and 10th Conference on Innovative Applications of Artificial Intelligence (AAAI/IAAI-98)*, Madison, US, 26–30 July 1998, pp. 746–752.

[23] S. Kapetanakis and D. Kudenko, "Reinforcement learning of coordination in cooperative multi-agent systems," in *Proceedings 18th National Conference on Artificial Intelligence and 14th Conference on Innovative Applications of Artificial Intelligence (AAAI/IAAI-02)*, Menlo Park, US, 28 July – 1 August 2002, pp. 326–331.

[24] X. Wang and T. Sandholm, "Reinforcement learning to play an optimal Nash equilibrium in team Markov games," in *Advances in Neural Information Processing Systems 15 (NIPS-02)*, Vancouver, Canada, 9–14 December 2002, pp. 1571–1578.

[25] D. Carmel and S. Markovitch, "Opponent modeling in multi-agent systems," in *Adaptation and Learning in Multi-Agent Systems*, G. Weiß and S. Sen, Eds. Springer Verlag, 1996, pp. 40–52.

[26] G. Tesauro, "Extending Q-learning to general adaptive multi-agent systems," in *Advances in Neural Information Processing Systems 16 (NIPS-03)*, Vancouver and Whistler, Canada, 8–13 December 2003.

[27] S. Singh, M. Kearns, and Y. Mansour, "Nash convergence of gradient dynamics in general-sum games," in *Proceedings 16th Conference on Uncertainty in Artificial Intelligence (UAI-00)*, San Francisco, US, 30 June – 3 July 2000, pp. 541–548.

[28] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *Proceedings Twentieth International Conference on Machine Learning (ICML-03)*, Washington, US, 21–24 August 2003, pp. 928–936.

[29] S. Sen, M. Sekaran, and J. Hale, "Learning to coordinate without sharing information," in *Proceedings 12th National Conference on Artificial Intelligence (AAAI-94)*, Seattle, US, 31 July – 4 August 1994, pp. 426–431.

[30] M. J. Matarić, "Learning in multi-robot systems," in *Adaptation and Learning in Multi–Agent Systems*, G. Weiß and S. Sen, Eds. Springer Verlag, 1996, pp. 152–163.

[31] R. H. Crites and A. G. Barto, "Improving elevator performance using reinforcement learning," in *Advances in Neural Information Processing Systems*, vol. 8, 1996, pp. 1017–1023.

[32] V. Könönen, "Asymmetric multiagent reinforcement learning," in *Proceedings IEEE/WIC International Conference on Intelligent Agent Technology (IAT-03)*, Halifax, Canada, 13–17 October 2003, pp. 336–342.

[33] M. Weinberg and J. S. Rosenschein, "Best-response multiagent learning in non-stationary environments," in *Proceedings 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-04)*, New York, US, 19-23 August 2004, pp. 506–513.

[34] B. Banerjee and J. Peng, "Adaptive policy gradient in multiagent learning," in *Proceedings 2nd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-03)*, Melbourne, Australia, 14–18 July 2003, pp. 686–692.

[35] N. Suematsu and A. Hayashi, "A multiagent reinforcement learning algorithm using extended optimal response," in *Proceedings 1st International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-02)*, Bologna, Italy, 15–19 July 2002, pp. 370–377.

[36] J. Schmidhuber, "A general method for multi-agent reinforcement learning in unrestricted environments," in *Working Notes AAAI Symposium on Adaptation, Co-evolution and Learning in Multiagent Systems*, Stanford University, US, March 25–27 1996, pp. 84–87.

[37] J. M. Vidal, "Learning in multiagent systems: An introduction from a game-theoretic perspective," in *Adaptive Agents: Lecture Notes in Artificial Intelligence*. Springer Verlag, August 2003, vol. 2636, pp. 202–215.