A COMPREHENSIVE SURVEY OF MULTIAGENT REINFORCEMENT LEARNING

BY: BUSONIU, L., R. BABUSKA, AND B. DE SCHUTTER
Introduction

- A reinforcement learning (RL) agent learns by trial-and-error interaction with its dynamic environment.
- Well-understood algorithms with good convergence and consistency properties are available for solving the single-agent RL task.
  - Both when the agent knows the dynamics of the environment and the reward function (the task model), and when it does not.
- Together with the simplicity and generality of the setting, this makes RL attractive also for RL in multiagent systems.
Introduction: Challenges

- Difficult to define a good learning goal for the multiple RL agents

- Most of the times each learning agent must keep track of the other learning (and therefore nonstationary) agents
  - Only then will it be able to coordinate its behavior with heirs, such that a **coherent** joint behavior results
  - Nonstationarity also invalidates the convergence properties of most single-agent RL algorithms

- Scalability of algorithms to realistic problem sizes is an even greater cause for concern in multiagent reinforcement learning (MARL)
Background: Reinforcement Learning

- Recall: states ($X$), actions ($U$), reward functions ($\rho$)
The joint action set: $U = U_1 \times \ldots \times U_n$

The state transition probability function: $f: X \times U \times X \rightarrow [0,1]$

The reward function of agent $i$: $\rho_i: X \times U \times X \rightarrow \text{Real}$

Together, they form the collection of reward functions.

In MARL, the state transitions are the result of the joint action of ALL the agents.

Consequently, the rewards and the returns also depend on the joint action.

The policies are: $h_i: X \times U_i \rightarrow [0,1]$ (all $\rightarrow$ joint policy $h$).

The Q-function of each agent depends on the joint action and is conditioned on the joint policy, $Q_{h,i}: X \times U \rightarrow \text{Real}$.
If \( \rho_1 = \ldots = \rho_n \), then all the agents have the same goal (to maximize the same expected return), and the system is fully cooperative.

If \( n = 2 \) and \( \rho_1 = -\rho_2 \), then all the two agents have opposite goals, and the system is fully competitive.

Mixed-game systems are stochastic systems that are neither fully cooperative nor fully competitive.
Benefits of MARL

- A speedup of MARL can be realized (thanks to parallel computation) when the agents exploit the decentralized structure of the task.
- Experience sharing can help agents with similar tasks to learn faster and better.
- When one or more agents fail in a MAS, the remaining agents can take over some of their tasks; robustness.
Challenges in MARL

- Curse of dimensionality
  - Complexity of MARL is exponential in the number of agents, because each agent adds its own variables to the joint state-action space

- Specifying a good MARL goal in the general stochastic setup is a difficult challenge, as the agents’ returns are correlated and cannot be maximized independently

- Non-stationarity of the multiagent learning problem arises because all the agents in the system are learning simultaneously

- Need for coordination as actions by agents depend on others’ actions
The exploration-exploitation tradeoff requires online RL algorithms to strike a balance between the exploitation of the agent’s current knowledge, and exploratory, information-gathering actions taken to improve that knowledge.

- In MARL, further complications arise due to presence of multiple agents.
- Exploring agents do not just obtain info about the environment, but also about the other agents.
- Too much exploration can destabilize the learning dynamics of the other agents **(WHY?)**
Specifying a good MARL goal is, in general, a difficult problem especially in situations where agents are not fully cooperative.

Goals incorporate two key factors:

1. Stability of the learning dynamics of the agent
   - Convergence to a stationary policy
2. Adaptation to the dynamic behavior of the other agents
   - Performance is maintained or improved as the other agents are changing their policies
Convergence to equilibria is a basic stability requirement

- Agents’ strategies should eventually converge to a coordinated equilibrium
- Nash equilibria are most frequently used

Rationality, an adaptation criterion, to add to stability

- The requirement that the agent converges to a best response when the other agents remain stationary
An alternative to rationality is the concept of no-regret:
- The requirement that the agent achieves a return that is at least as good as the return of any stationary strategy.
- Prevents the learner from “being exploited” by the other agents.

Targeted optimality/compatibility/safety are adaptation requirements expressed in the form of average reward bounds:
- E.g., targeted optimality demands an average reward, against a targeted set of algorithms, which is at least the average reward of a best response.
## MARL Goal, 4

<table>
<thead>
<tr>
<th>Stability Property</th>
<th>Adaptation Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convergence</td>
<td>Rationality</td>
</tr>
<tr>
<td>Convergence</td>
<td>No-Regret</td>
</tr>
<tr>
<td>--</td>
<td>Targeted optimality, compatibility, safety</td>
</tr>
<tr>
<td>Opponent-independent</td>
<td>Opponent-aware</td>
</tr>
<tr>
<td>Equilibrium learning</td>
<td>Best-response learning</td>
</tr>
<tr>
<td>Prediction</td>
<td>Rationality</td>
</tr>
</tbody>
</table>
## Taxonomy of MARL Algorithms

### Breakdown of MARL Algorithms by Task Type and Degree of Agent Awareness

<table>
<thead>
<tr>
<th>Agent Awareness</th>
<th>Cooperative</th>
<th>Competitive</th>
<th>Mixed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Independent</td>
<td>Coordination-free</td>
<td>Opponent-independent</td>
<td>Agent-independent</td>
</tr>
<tr>
<td>Tracking</td>
<td>Coordination-based</td>
<td>---</td>
<td>Agent-tracking</td>
</tr>
<tr>
<td>Aware</td>
<td>Indirect coordination</td>
<td>Opponent-aware</td>
<td>Agent-aware</td>
</tr>
</tbody>
</table>
## Taxonomy of MARL Algorithms, 2

<table>
<thead>
<tr>
<th>Task Type</th>
<th>Static or Dynamic?</th>
<th>Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully Cooperative</td>
<td>Static</td>
<td>Joint Action Learners (JAL), Frequency Maximum Q-value (FMQ)</td>
</tr>
<tr>
<td></td>
<td>Dynamic</td>
<td>Team-Q, Distributed-Q, Optimal Adaptive Learning (OAL)</td>
</tr>
<tr>
<td>Fully Competitive</td>
<td>NA</td>
<td>Minimax-Q</td>
</tr>
<tr>
<td>Mixed</td>
<td>Static</td>
<td>Fictitious Play, MetaStrategy, Infinitesimal Gradient Ascent (IGA), Win-or-Learn-Fast-IGA (WoLF-IGA), Generalized IGA (GIGA), GIGA-WoLF, AWESOME, Hyper-Q</td>
</tr>
<tr>
<td></td>
<td>Dynamic</td>
<td>Single-agent RL, Nash-Q, Correlated Equilibrium Q-learning (CE-Q), Asymmetric-Q, Non-Stationary Converging Policies (NSCP), WoLF-Policy Hill Climbing (WoLF-PHC), PD-WoLF, EXORL</td>
</tr>
</tbody>
</table>
## Taxonomy of MARL Algorithms, 3

<table>
<thead>
<tr>
<th>Task Type</th>
<th>Open Issues</th>
</tr>
</thead>
</table>
| Fully Cooperative       | • Rely on exact measurements of the state  
• Many also require exact measurements of the other agents’ actions  
• Communication might help relax these strict requirements  
• Most suffer from the curse of dimensionality |
| Mixed                   | • Static, repeated games represented a limited set of applications  
• Most static game algorithms assume the availability of an exact task model, which is rarely the case in practice  
• Many suffer from the curse of dimensionality  
• Many are sensitive to imperfect observations |
Application Domains

- Mostly in simulation but also to some real-life tasks
- Simulation domains dominate because:
  - Results in simpler domains are easier to understand and to use for gaining insight
  - In real life, *scalability* and *robustness* to imperfect observations are necessary, and few MARL algorithms exhibit these properties
  - In real-life applications, more direct derivations of single-agent RL are preferred
Distributed Control

- A set of autonomous, interacting controllers act in parallel on the same process
- Cooperative in nature
- E.g., process control, control of traffic signals, control of electrical power networks
Robotic Teams

- Most popular application domain
- Many MARL researchers are active in the robotics field
- Real and simulation
- E.g., navigation, area sweeping (object recovery), search-and-rescue, exploration and target tracking, predator-and-prey games, object transportation, Robocup (soccer, disaster response, …)
- Cooperative, competitive
Automated Trading

- Software trading agents exchange goods on e-markets on behalf of a company or a person, using mechanisms such as auctions and negotiations
- Trading Agent Competition (TAC): plane tickets, goods, and hotel bookings
- Cooperative, self-interested
Resource Management

Agents form a cooperative team, and they can be one of:

- Managers of resources
- Clients of resources

Network routing, elevator scheduling, load balancing

Performance measures include average job processing times, minimum waiting time for resources, resource usage, and fairness in serving clients
Outlook

- Scalability is the central concern for MARL as it stands today
  - Approximate solutions are sought
- Providing domain knowledge to the agents can greatly help them in learning solutions to realistic tasks
  - Approximations, informative reward functions, human teaching agents, pre-programmed reflex behaviors, hierarchical RL, task-model-based initialization of Q-functions
MARL goals are typically formulated in terms of static games; their extension to dynamic tasks is not always clear or even possible.

- Stability and adaptation are needed.
- MARL algorithms should neither be totally independent of the other agents, nor just track their behavior without concerns for convergence.
The stagewise application of game-theoretic techniques to solve dynamic multiagent tasks is a popular approach.

- May not be the most suitable, given that both 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 of the agents, while the dynamics of the environment have not been explicitly considered.