# Multi-agent Reinforcement Learning for Control Systems: Challenges and Proposals

Manuel Graña[1,2](✉) and Borja Fernandez-Gauna[1]

[1] Grupo de Inteligencia Computacional (GIC), Universidad del País Vasco (UPV/EHU), San Sebastián, Spain
manuel.granaromay@pwr.edu.pl

[2] ENGINE Centre, Wrocław University of Technology,
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland

**Abstract.** Multi-agent Reinforcement Learning (MARL) methods offer a promising alternative to traditional analytical approaches for the design of control systems. We review the most important MARL algorithms from a control perspective focusing on on-line and model-free methods. We review some of sophisticated developments in the state-of-the-art of single-agent Reinforcement Learning which may be transferred to MARL, listing the most important remaining challenges. We also propose some ideas for future research aiming to overcome some of these challenges.

## 1 Introduction

Reinforcement Learning (RL) [37] methods gaining popularity in the area of control because they allow to build control systems without detailed modeling of the underlying dynamics, because they learn how to maximize the control objective by means of interacting with the environment. This is quite an advantage over compared with traditional analytical control techniques requiring a deatiled formal model, which may be difficult to construct for complex non-linear systems. The quality of these approaches rely on the quality of the model itself and thus, require a good understanding of the problem at hand. In the RL approach, parameter tuning is substituted by iterative adaptation to an stochastic environment. Some systems (i.e., Multi-component Robotic Systems [13]) are best approached from a multi-agent perspective in order to better exploit the computation capabilities and robustness of distributed control systems. Multi-Agent Reinforcement Learning (MARL) is the extension of single-agent RL to multi-agent scenarios. MARL methods have already been successfully applied to several multi-agent control scenarios [17,36,41,43].

## 2 Reinforcement Learning

### 2.1 Single-agent Reinforcement Learning

*Markov Decision Process.* Single-agent RL methods use Markov Decision Processes (MDPs) to model the interaction between the agent and the environment. An

K. Jackowski et al. (Eds.): IDEAL 2015, LNCS 9375, pp. 18–25, 2015.
DOI: 10.1007/978-3-319-24834-9_3

MDP $< S, A, P, R >$ is defined by the set of *states* $(S)$, the set of *actions* from which the agent can choose $(A)$, a *transition function* $(P)$ that determines state transitions produced by actions, and a *reward function* $(R)$ that gives a numerical value assessing how good a state transition was. $S$ can be a finite set of states (i.e., a cell number in a grid-world) or a vector of real values (i.e., the $x$ and $y$ coordinates read from a GPS receiver). The goal of the agent is to learn a policy $\pi$ that maximizes the expected *return* $R$ by means of interacting with the environment. The state-action value function $Q^{\pi}(s, a)$ is the value of taking action $a$ in state $s$.

*Policy learning methods.* There are basically three classes of RL methods [7]: value iteration, policy iteration and policy search methods. Value iteration methods (such as Q-Learning) generally learn the optimal state-action value-function $Q^*$ and then derive the optimal policy. Policy iteration methods usually follow some policy, evaluate its value by learning $V^{\pi}$, and then, aim to improve $\pi$. Actor-critic methods belong to this class: an actor implements a parametrized policy, a critic learns its value function (i.e., *Least-Squares Temporal Difference* [6,42]). Value updates are then fed back to the actor, which can us it to improve its policy (i.e., *Natural Policy Gradient* [5]). Finally, policy search methods directly search on the policy space for the optimal policy that maximizes the expected return for any possible initial state.

## 2.2 Multi-agent Reinforcement Learning

*Stochastic Games.* The interaction model in Multi-agent Reinforcement Learning (MARL) is the Stochastic Game (SG), that is defined by the number of agents $(n)$ and the tuple $\langle S, A_1, \ldots A_n, P, R_1, \ldots, R_n \rangle$. Each $i$-th agent chooses actions from its own local action space $A_i$ and receives its own reward $R_i$. Multi-agent systems can be competitive, cooperative or mixed. In fully cooperative systems, all the agents share the same goals and so, the same reward signals $R_1 = R_2 = \ldots = R_n$. MARL algorithms can also be classified depending on whether they use models of the other agents or not. In this paper, we will focus on model-free methods because we expect them to scale better to multi-dimensional control problems.

*Distributed Q-Learning* (D-QL) [26] is an example of independent learning. Each agent assumes that the remaining agents will behave optimally thus projecting the virtual centralized state-action values $Q(s, \mathbf{a})$ $(\mathbf{a} \in \mathbf{A})$ to its own local action space $Q_i(s, a)$, $a \in A_i$. An instance of the MARL algorithms in which agents are aware of other agents' choices, is the *Team Q-Learning* [28], where (Team-QL) agents learn the joint state-action $Q(s, \mathbf{a})$. The algorithm uses the Q-Learning update rule, but using the joint-actions $\mathbf{a}$ and $\mathbf{a}'$ instead of $a$ and $a'$. This algorithm converges to optimal values under an additional assumption: a unique optimal action exists in each state. This implicit coordination mechanism ensures that agents will exploit the Q-function in a coordinated manner. Some other implicit coordination mechanisms based on heuristics [8,23] or models of the other agents [30,40] can be found in the literature. MARL methods aware of the other agents' actions eliminates the non-stationarity due to other agents' policy changes, but then it becomes a more complex problem.

In order to reduce this complexity, it can be assumed that agents need not coordinate in every state with every other agent, but only with some of them. Under this assumption, the agents first learn when and which agents to coordinate with, and then use an explicit coordination mechanism [20,24] to select a joint-action that maximizes the expected return. *Coordinated Reinforcement Learning* (Coordinated-RL) [20] builds a *Coordination Graph* (CG) for each state that defines with whom agents do need to coordinate. An agent's local state-action values thus only depend on its own local action and those taken by the agents connected to it through the CG. Agents can maximize the global value using a message-based coordination mechanism. An improved version of this algorithm based on an edge-based decomposition of the CG instead of an agent-based decomposition was proposed in [24]. This method scales linearly on the number of agents. The downside of these methods is having to learn and store the CG and the additional processing time introduced by the coordination mechanism. *Hierarchical Reinforcement Learning* (HRL) is another interesting approach to reduce the complexity of a task by decomposing it as a hierarchical structure of subtasks. Single-agent MAXQ [11] allows agents to learn concurrently low-level subtasks and higher-level tasks based on these subtasks. This idea is extended to multi-agent problems in [19]: *Cooperative HRL* assumes costless communication, and *COM-Cooperative HRL* considers communication as part of the problem. A communication level is added to the subtask hierarchy, so that the agent can learn when to communicate. Decomposing the task into a hierarchy of subtasks is not always trivial, and the decomposition of the task itself determines how good the system will approach the optimal solution. This has led research towards automated decomposition of task, both in single-agent [22,29] and multi-agent environments [10].

## 3 Control Applications of MARL

Last years have seen a number of novel MARL applications: traffic light control [1,4,25,35], robotic hose maneuvering [17], micro-grid control [27], structure prediction of proteins [9], route-choosing [2], supply chains [43], or management of the cognitive radio spectrum [41].

*Advantages.* MARL-based approaches to control systems offer some inherent advantages over traditional control methods. For example, MARL algorithms can adapt to changes in the environment thanks to their learning nature. Another big advantage over analytical approaches is that model-free algorithms do not require the dynamics of the environment to be fully understood, thus enabling the control of more complex systems. There is still quite a gap between single-agent RL and MARL techniques, but we expect more works currently restricted to the single-agent case to be extended to the multi-agent case in the near future. An example of this is Multi-objective Reinforcement Learning [12], which aims to maximize different possibly conflicting objectives at the same time.

*Challenges.* We have not found in the literature MARL algorithms able to deal with continuous state-action spaces [21]. Continuous controllers have been shown to outperform algorithms with discretized state and actions in general feedback control tasks [15]. Several methods have been developed in single-agent RL paradigm to deal control tasks involving continuous action spaces: actor-critic learning architectures [5,18,21,32], policy search methods [3] or parametrized controllers [34]. A lot of effort has been devoted in recent years towards obtaining efficient policy gradients [5] and data-efficient value estimation methods [6,31]. For a complete review of the state-of-the-art on single-agent RL using VFA, we refer the reader to [42]. On the other hand, MARL algorithms are mostly based on Q-Learning, hence they estimate the state-action value. General Value Function Approximation (VFA) methods can been used to approximate the state-action value function. This allows continuous states [7], but greedy selection of the action with the highest value will correspond to the center value of one feature. This limits the ability of Q-Learning to output continuous action spaces.

Most of the MARL applications to realistic control problems so far found in the literature are either uncoupled systems of agents operating with no influence on each other [2,41], or loosely coupled tasks, such as traffic light control [1,4]. Some loosely coupled problems may be better approached using systems of unaware agents. Regarding fully-coupled systems in which agents' actions have an effect on other agents' decisions, only a few instances can be found [16,17,25]. This kind of systems require either full observation and learning on the joint state-action space, which does not scale well to real-world environments. Between unaware multi-agent systems and learning on the full joint state-action space, there are alternative approaches, such as exploiting the coordination requirements of the task using CG(i.e., *Coordinated-RL*), or decomposing tasks into a structure of hierarchical subtasks. Both CGs and task hierarchies can be designed by hand in small-scale or clearly structured tasks [25], but manual design is not feasible in more complex or unstructured problems. Some advances have been done towards automatic learning of Coordination Graphs [10] and hierarchies of tasks [27], but none is applicable to continuous state or action spaces. It is not clear either how good these methods will scale to more complex MARL problems. CG-based algorithms require communication each time step. A variable-elimination procedure was proposed in [20] to give an exact solution to the joint-action value maximization process. The number of messages exchanged at each decision step depends on the topology of the specific CG. In order to alleviate this problem, two anytime algorithms were proposed in [39] to approximate the optimal joint-action in a predetermined time: *Coordinated Ascent* and *Max-Plus*. Whether these methods provide an acceptable solution in complex real-time control scenarios within an acceptable time-frame remains an open question.

Another important uncontested challenge is learning an initial policy from scratch in large real-world applications, where it is unaffordable to allow agents thousands of trials before they can start completing the task (i.e., robotic maneuvering tasks [16]). There are several approaches to this problem, all based on some form of Learning Transfer [38]: agents can be first trained in a simulated environment and then allowed to face the real task [14], they can be initialized resembling some initial policy (i.e., a PID controller) available to the system designer [18], or agents may be trained to imitate some expert performing the task [32].

*Proposals for Future Research.* MARL methods are mostly based on general heterogeneous Stochastic Games and, thus, they work under very broad assumptions. From a control perspective though, one can further assume fully cooperative tasks and homogeneous learning agents. This kind of systems might be better approached from a distributed point of view. Consider a multiple output continuous actor-critic architecture: an approximated value function estimated by the critic and an actor with several VFAs, each representing a different output of the actor. When an improvement in the value function is detected, the actors update its policies towards the last action explored. This same idea can be translated to a multi-agent system in which each agent keeps an instance of the critic learning the value function and an actor with a subset of the system's output. Agents would only require to coordinate exploration and exploitation, which could be achieved by using consensus [33] to share and update the exploration parameters using some preset schedules. This learning structure would allow to use the state-of-the-art in single-agent model-free environments. Full observation of the state could also be alleviated by deferred updates of the critic/actor: agents can follow their policies tracking their local actions and yet incomplete states, and defer the update of the actor and policy until all the state variables have been received.

## 4 Conclusions

In this paper, we have reviewed the basics of MARL and some recent works in the literature of this field applied to control systems. MARL offers some advantages over traditional analytical control techniques. The most important is that the system designer needs not fully understand or have an accurate model of the system. MARL-based methods also pose some interesting challenges when applied to real-world control problems. Most of the algorithms have been developed with small environments in mind. In this respect, we point out that the main gap between single-agent and MARL algorithms to date is the ability to deal with continuous state and action spaces.

**Acknowledgments.** This research has been partially funded by grant TIN2011-23823 of the Ministerio de Ciencia e Innovación of the Spanish Government (MINECO), and the Basque Government grant IT874-13 for the research group. Manuel Graña was supported by EC under FP7, Coordination and Support Action, Grant Agreement Number 316097, ENGINE European Research Centre of Network Intelligence for Innovation Enhancement.

## References

1. Arel, I., Liu, C., Urbanik, T., Kohls, A.: Reinforcement learning-based multi-agent system for network traffic signal control. Intell. Transport Syst. IET **4**(2), 128–135 (2010)
2. Arokhlo, M., Selamat, A., Hashim, S., Selamat, M.: Route guidance system using multi-agent reinforcement learning. In: 2011 7th International Conference on Information Technology in Asia (CITA 2011), pp. 1–5, July 2011
3. Bagnell, J.A.D., Schneider, J.: Autonomous helicopter control using reinforcement learning policy search methods. In: 2001 Proceedings of the International Conference on Robotics and Automation. IEEE, May 2001
4. Bazzan, A.: Opportunities for multiagent systems and multiagent reinforcement learning in traffic control. Auton. Agents Multi-Agent Syst. **18**(3), 342–375 (2009)
5. Bhatnagar, S., Sutton, R., Ghavamzadeh, M., Lee, M.: Natural actor-critic algorithms. Automatica Int. Fed. Autom. Control **45**(11), 2471–2482 (2009)
6. Boyan, J.A.: Technical update: least-squares temporal difference learning. Mach. Learn. **49**, 233–246 (2002)
7. Bussoniu, L., Babuska, R., Schutter, B.D., Ernst, D.: Reinforcement Learning and Dynamic Programming Using Function Approximators. CRC Press, Boca Raton (2010)
8. Claus, C., Boutilier, C.: The dynamics of reinforcement learning in cooperative multiagent systems. In: Proceedings of the Fifteenth National Conference on Artificial Intelligence, pp. 746–752. AAAI Press (1997)
9. Czibula, G., Bocicor, M.I., Czibula, I.G.: A distributed reinforcement learning approach for solving optimization problems. In: Proceedings of the 5th WSEAS International Conference on Communications and Information Technology, CIT 2011, pp. 25–30. World Scientific and Engineering Academy and Society (WSEAS), Stevens Point (2011)
10. De Hauwere, Y.M., Vrancx, P., Nowé, A.: Learning multi-agent state space representations. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2010, vol. 1, pp. 715–722. International Foundation for Autonomous Agents and Multiagent Systems, Richland (2010)
11. Dietterich, T.G.: An overview of MAXQ hierarchical reinforcement learning. In: Choueiry, B.Y., Walsh, T. (eds.) SARA 2000. LNCS (LNAI), vol. 1864, p. 26. Springer, Heidelberg (2000)
12. Drugan, M., Nowe, A.: Designing multi-objective multi-armed bandits algorithms: a study. In: The 2013 International Joint Conference on Neural Networks (IJCNN), pp. 1–8, August 2013
13. Duro, R., Graña, M., de Lope, J.: On the potential contributions of hybrid intelligent approaches to multicomponen robotic system development. Inf. Sci. **180**(14), 2635–2648 (2010)
14. Fernandez-Gauna, B., Lopez-Guede, J., Graña, M.: Transfer learning with partially constrained models: application to reinforcement learning of linked multicomponent robot system control. Robot. Auton. Syst. **61**(7), 694–703 (2013)
15. Fernandez-Gauna, B., Ansoategui, I., Etxeberria-Agiriano, I., Graña, M.: Reinforcement learning of ball screw feed drive controllers. Eng. Appl. Artif. Intell. **30**, 107–117 (2014)
16. Fernandez-Gauna, B., Graña, M., Etxeberria-Agiriano, I.: Distributed round-robin q-learning. PLoS ONE **10**(7), e0127129 (2015)

17. Fernandez-Gauna, B., Marques, I., Graña, M.: Undesired state-action prediction in multi-agent reinforcement learning. application to multicomponent robotic system control. Inf. Sci. **232**, 309–324 (2013)
18. Fernandez-Gauna, B., Osa, J.L., Graña, M.: Effect of initial conditioning of reinforcement learning agents on feedback control tasks over continuous state and action spaces. In: de la Puerta, J.G., Ferreira, I.G., Bringas, P.G., Klett, F., Abraham, A., de Carvalho, A.C.P.L.F., Herrero, Á., Baruque, B., Quintián, H., Corchado, E. (eds.) International Joint Conference SOCO'14-CISIS'14-ICEUTE'14. AISC, vol. 299, pp. 125–133. Springer, Heidelberg (2014)
19. Ghavamzadeh, M., Mahadevan, S., Makar, R.: Hierarchical multi-agent reinforcement learning. Auton. Agents Multi-Agent Syst. **13**, 197–229 (2006)
20. Guestrin, C., Lagoudakis, M., Parr, R.: Coordinated reinforcement learning. In: Proceedings of the IXth ICML, pp. 227–234 (2002)
21. van Hasselt, H.: Reinforcement Learning in Continuous State and Action Spaces. In: Wiering, M., van Otterlo, M. (eds.) Reinforcement Learning: State of the Art, pp. 207–246. Springer, Heidelberg (2011)
22. Hengst, B.: Discovering hierarchy in reinforcement learning with HEXQ. In: Maching Learning: Proceedings of the Nineteenth International Conference on Machine Learning, pp. 243–250. Morgan Kaufmann (2002)
23. Kapetanakis, S., Kudenko, D.: Reinforcement learning of coordination in cooperative multi-agent systems. In: AAAI/IAAI 2002, pp. 326–331 (2002)
24. Kok, J.R., Vlassis, N.: Collaborative multiagent reinforcement learning by payoff propagation. J. Mach. Learn. Res. **7**, 1789–1828 (2006)
25. Kuyer, L., Whiteson, S., Bakker, B., Vlassis, N.: Multiagent reinforcement learning for urban traffic control using coordination graphs. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part I. LNCS (LNAI), vol. 5211, pp. 656–671. Springer, Heidelberg (2008)
26. Lauer, M., Riedmiller, M.A.: An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In: Proceedings of the Seventeenth International Conference on Machine Learning, ICML 2000, pp. 535–542. Morgan Kaufmann Publishers Inc., San Francisco (2000)
27. Li, F.D., Wu, M., He, Y., Chen, X.: Optimal control in microgrid using multi-agent reinforcement learning. ISA Trans. **51**(6), 743–751 (2012)
28. Littman, M.L.: Value-function reinforcement learning in Markov games. Cogn. Syst. Res. **2**(1), 55–66 (2001)
29. Mehta, N., Ray, S., Tadepalli, P., Dietterich, T.: Automatic discovery and transfer of MAXQ hierarchies. In: Proceedings of the 25th International Conference on Machine Learning, ICML 2008, pp. 648–655. ACM, New York (2008). http://doi.acm.org/10.1145/1390156.1390238
30. Melo, F., Ribeiro, M.: Coordinated learning in multiagent MDPS with infinite state-space. Auton. Agents Multi-Agent Syst. **21**, 321–367 (2010)
31. Nedic, A., Bertsekas, D.: Least squares policy evaluation algorithms with linear function approximation. Discrete Event Dyn. Syst. **13**(1–2), 79–110 (2003)
32. Peters, J., Schaal, S.: Policy gradient methods for robotics. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS (2006)
33. Ren, W., Beard, R.W.: Distributed Consensus in Multi-vehicle Cooperative Control: Theory and Applications. Springer, London (2007)
34. Roberts, J.W., Manchester, I.R., Tedrake, R.: Feedback controller parameterizations for reinforcement learning. In: IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (2011)

35. Salkham, A., Cunningham, R., Garg, A., Cahill, V.: A collaborative reinforcement learning approach to urban traffic control optimization. In: Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, WI-IAT 2008, vol. 2, pp. 560–566. IEEE Computer Society, Washington, DC (2008)
36. Servin, A., Kudenko, D.: Multi-agent reinforcement learning for intrusion detection. In: Tuyls, K., Nowe, A., Guessoum, Z., Kudenko, D. (eds.) ALAMAS 2005, ALAMAS 2006, and ALAMAS 2007. LNCS (LNAI), vol. 4865, pp. 211–223. Springer, Heidelberg (2008)
37. Sutton, R.S., Barto, A.G.: Reinforcement Learning I: Introduction. MIT Press, Cambridge (1998)
38. Taylor, M.E., Stone, P.: Transfer learning for reinforcement learning domains: a survey. J. Mach. Learn. Res. **10**(1), 1633–1685 (2009)
39. Vlassis, N., Elhorst, R., Kok, J.R.: Anytime algorithms for multiagent decision making using coordination graphs. In: Proceedings of the International Conference on Systems, Man, and Cybernetics (2004)
40. Wang, X., Sandholm, T.: Reinforcement learning to play an optimal nash equilibrium in team Markov games. In: Advances in Neural Information Processing Systems, pp. 1571–1578. MIT Press (2002)
41. Wu, C., Chowdhury, K., Di Felice, M., Meleis, W.: Spectrum management of cognitive radio using multi-agent reinforcement learning. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Industry Track, AAMAS 2010, pp. 1705–1712. International Foundation for Autonomous Agents and Multiagent Systems, Richland (2010)
42. Xu, X., Zuo, L., Huang, Z.: Reinforcement learning algorithms with function approximation: recent advances and applications. Inf. Sci. **261**, 1–31 (2014)
43. Zhao, G., Sun, R.: Application of multi-agent reinforcement learning to supply chain ordering management. In: 2010 Sixth International Conference on Natural Computation (ICNC), vol. 7, pp. 3830–3834, August 2010