Towards Distributed Control Under Deficient Communication with Multi-agent Reinforcement Learning

Arne Kummerow(✉) and Torben Weis

University of Duisburg-Essen, Bismarckstraße 90, 47057 Duisburg, Germany
{arne.kummerow,torben.weis}@uni-due.de

Abstract. Multi-agent reinforcement learning solves optimization problems in sequential decision-making and enables controlling spatially distributed actuators. We consider a cooperative setting where agents can exchange information via a central controller, e.g. a cloud-based service. In real world applications however, communication channels are often error-prone and agents may become disconnected and can neither send its observation nor receive observations from other agents. We formalize this problem as a subclass of decentralized Markov decision processes and discuss the complexity of the problem. We then propose several solution concepts that involve breaking down the complexity by considering only a subset of failure scenarios, learning independent policies for each failure scenario, reconstructing missing information and learning policies that incorporate the state uncertainty in the training process.

Keywords: MARL · Dec-MDP · Dec-POMDP · Communication Failure

1 Introduction

Many real world control problems involve operating spatially distributed actuators by use of a potentially faulty communication network, e.g. a power grid, a water drainage network or a traffic light control system [1,2]. Reinforcement learning has been used extensively in recent years to solve optimization problems in sequential decision-making [3] by learning a policy that maps the system state to control actions, but using these algorithms in a distributed scenario requires perfect communication channels. This is caused by the fact that they learn policies that do not account for being incapable of action per se, as it becomes the case when a central controller running the policy loses connection to one or more of the actuators. Modeling our task as a multi-agent system overcomes this issue by separating available state information and accessible control actions to enable

Funded by the Federal Ministry of Economic Affairs and Climate Protection of Germany (BMWK) within the project RIWWER.

A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 503–510, 2024.
https://doi.org/10.1007/978-3-031-63992-0_33

generation of multiple policies that can be deployed locally where actuators are located, thus mitigating the need of a continuous communication stream.

In this paper we will focus on the cooperative multi-agent reinforcement learning (MARL) setting [5], where we want to optimize all agents against some common goal, since we are specifically interested in the problem that is posed by coordinating these agents across distant locations to solve a global objective. A lot of work has been done that incorporate the exchange of information as part of the learned policy, i.e. agents can decide what information they want to send to other agents and also to which of these. This concept is known as learning to communicate and is typically motivated by a limited communication bandwidth, latency or a restricted network topology (e.g. agents can only talk to adjacent agents). This might be the case in sensor networks or autonomous cars, for example. In the following we want to focus on scenarios where a central controller is present and no limits in communication bandwidth or latency exist. Agents may become disconnected to the central controller, however. We think that this type of information structure applies to a lot of applications, since we often encounter settings where agents have access to the internet via a cellular network and can communicate to a cloud-based service. The central controller in this case is very resilient against failure, but the communication channels to the agents that drive the actuators are not.

2 Preliminaries

A Markov decision process (MDP) is a discrete time decision framework for the single agent setting and is defined as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, R, \gamma)$, where $\mathcal{S}$ is the set of all states, $\mathcal{A}$ is the set of all action that the agent can take. Being in state $s \in \mathcal{S}$, the agent selects an action $a \in \mathcal{A}$ and transitions to next state s' with probability $\mathcal{T}(s'|s,a)$. $R\colon \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the reward function that acts as feedback for the agent, providing the optimization quantity for a given transition (s,a,s'). $\gamma \in [0,1)$ is the discount factor. The goal of the agent is to maximize the expected discounted reward $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1})]$ over encountered trajectories by learning a policy $\pi\colon \mathcal{S} \to \mathcal{A}$ that maps from states to actions. Depending on the environment, the optimal policy might however be a stochastic one, i.e. π is more generally defined as $\pi\colon \mathcal{S} \to \Delta(\mathcal{A})$, and the agent may select its action by $a \sim \pi(.|s)$.

Stochastic games are an extension to MDPs that capture the interaction of multiple agents with the environment in that they extend MDPs with a set of n agents denoted by $\mathcal{N}$. Every agent $i \in \mathcal{N}$ has its own action space $\mathcal{A}_i$ and gets reward R_i. We will from now on denote the joint action space as $\mathcal{A} = \times_{i \in \mathcal{N}} \mathcal{A}_i$. In the cooperative setting considered in this work, $R_i = R_j$ holds for all $i, j \in \mathcal{N}$.

To model the uncertainty of an agent's limited view of the environment, we define the set of joint observations as $\Omega = \times_{i \in \mathcal{N}} \Omega_i$, as well as an observation function $O\colon \mathcal{A} \times \mathcal{S} \to \Delta(\Omega)$ that indicates the probability $O(o|a,s')$ of the agents observing a joint observation o after taking joint action a and transition to state s'. Each agent can only see its corresponding part of the joint

observation. This leads us to the well known framework of decentralized partially observable Markov decision processes (Dec-POMDPs) [6], which capture the aforementioned structures as a tuple $(\mathcal{N}, \mathcal{S}, \mathcal{A}, \mathcal{T}, R, \Omega, O, \gamma)$. Regarding the observation function, different information structures have been identified [7] that describe the amount of information that an agent gets from its observation. If there exists a mapping from each observation that uniquely identifies the full state, then we can reduce the problem to a single agent decision problem by learning a joint policy that maps to the joint action space. During execution, each agent will select its action according to that policy. Another important class of observation functions are those where the composition of all observations can uniquely identify the system state. This class is referred to as jointly fully observable and the corresponding framework is denoted as a decentralized Markov decision process (Dec-MDP), which are a subclass of Dec-POMDPs. For $n > 3$ they are NEXP-complete in the worst case however, as it is the case for Dec-POMDPs for $n > 2$ [8]. A slightly stronger assumption than the joint full observability is a decomposable state $\mathcal{S} = (\mathcal{S}_1, ..., \mathcal{S}_n)$ whereat each agent observes its local state $\mathcal{S}_i$ [10,11]. Even though an agent may not observe the full state information during execution time, we will assume that the full state, observations and actions of all agents are known during training, known as the centralized-training-decentralized-execution scheme [5].

Note that during execution, an observation itself does not provide enough information to decide on an action without losing optimality [6], since the Markov property is no longer valid for partial observations. Therefore, policies have to be build on action-observation histories, which in practice is often realized by use of a recurrent architecture [5,9].

3 Problem Formulation

To address the issue of deficient communication channels, we formalize the problem as a subclass of Dec-MDPs, represented by a tuple $(\mathcal{N}, \mathcal{S}_E, \mathcal{M}, \mathcal{A}, \mathcal{T}_E, \mathcal{T}_M, R, \Omega, \mathcal{O}, \gamma)$. $\mathcal{N}$ is the set of n agents, $\mathcal{S} = \mathcal{S}_E \times \mathcal{M}$ is the set of states with $\mathcal{S}_E = \times_{i \in \mathcal{N}} \mathcal{S}_i$ being the set of environment states consisting of local states for each agent $i \in \mathcal{N}$. $\mathcal{M} = \{0, 1\}^n$ is the set of mask states that specify for each agent if it is connected to the central controller, with *1* representing that the agent is connected and can send its local state $\mathcal{S}_i$ and receive states of other agents that are connected and *0* representing that the agent is disconnected and can only observe its local state. We denote $0 = (0, \ldots, 0)$ for the case that every agent is disconnected and $1 = (1, \ldots, 1)$ if all agents are connected. Moreover, we define the relation $\trianglelefteq$ on $\mathcal{M}$ with $(m_1, \ldots, m_i, \ldots, m_n) \trianglelefteq (m'_1, \ldots, m'_i, \ldots, m'_n)$ if $m_i = 1 \implies m'_i = 1$ for all $i \in \mathcal{N}$ meaning that a mask state m is less or equally available than m' if all agents that are connected in m are also connected in m' (Fig. 1).

$\mathcal{A} = \times_{i \in \mathcal{N}} \mathcal{A}_i$ is the set of joint actions and $R \colon \mathcal{S}_E \times \mathcal{A} \times \mathcal{S}_E \to \mathbb{R}$ is the reward function for a common goal for all agents. $\mathcal{T} \colon \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition probability function, which can be factorized into two components $\mathcal{T}_E \colon \mathcal{S}_E \times \mathcal{A} \to \Delta(\mathcal{S}_E)$ and $\mathcal{T}_M \colon \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{M})$ so that

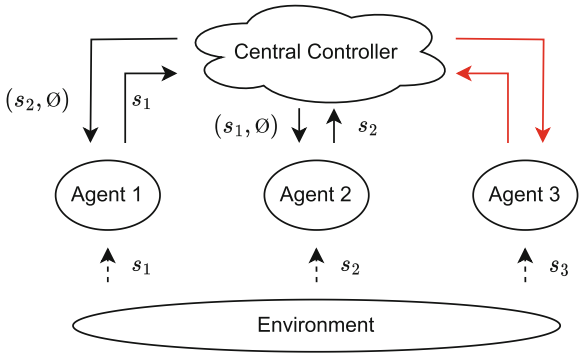

Fig. 1. Example of the proposed information structure. Three agents can exchange local states via a central controller. In the displayed situation, agent 1 and 2 are connected and see the states of each other, agent 3 is disconnected and observes local information only. Mask state is $(1, 1, 0)$.

$$\mathcal{T}((s'_E, m')|(s_E, m), a) = \mathcal{T}_E(s'_E|s_E, a)\mathcal{T}_M(m'|(s_E, m), a). \tag{1}$$

In other words, we assume that the environment state is not affected by the availability mask, but the mask state may be affected by the environment state. The process that determines whether an agent is connected to the central controller therefore does not change the actual state of system, but the communication channels may be influenced by the environment state.

$\Omega = \times_{i\in\mathcal{N}}\Omega_i$ is the set of joint observations. Each observation consists of the local state component, as well as the local state components of all other agents that are connected to the central controller, i.e.

$$\Omega_i = \mathcal{S}_i \cup (\mathcal{S}_i \times (\times_{j\in\mathcal{N}\setminus\{i\}}(\mathcal{S}_j \cup \{\emptyset\}))). \tag{2}$$

$\emptyset$ acts as a placeholder for the local state of an agent that is disconnected from the central controller. Note that we need the left union in Eq. 2 to distinguish between the case of an agent being disconnected and the case in which all other but that one agent is disconnected. Other than in the general case of a Dec-MDP, we assume the observation function to be deterministic and only dependent on the state. We thus write $\mathcal{O}\colon \mathcal{S} \to \Omega$ and define $\mathcal{O}(s) = (\mathcal{O}_1(s), \ldots, \mathcal{O}_n(s))$ with $\mathcal{O}_i\colon \mathcal{S} \to \Omega_i$ being defined as

$$\mathcal{O}_i(((s_1, \ldots, s_i, \ldots, s_n), (m_1, \ldots, m_i, \ldots, m_n))) = \begin{cases} s_i & \text{if } m_i = 0 \\ (s_i, (s_1 \otimes m_1, \ldots, s_{i-1} \otimes m_{i-1}, s_{i+1} \otimes m_{i+1}, \ldots, s_n \otimes m_n)) & \text{else.} \end{cases} \tag{3}$$

$\otimes\colon M \times \{0, 1\} \to M \cup \{\emptyset\}$ is the masking operator defined as $m \otimes 0 = \emptyset$ and $m \otimes 1 = m$ for any set M and $m \in M$.

3.1 Generalizations

In the previous section we assumed the probability function for the availability mask $\mathcal{T}_M$ to be dependent on the system state which is jointly fully observable. It might however be the case that the transition probabilities for the mask is not solely dependent on the system state, but also on some additional information that is not available to the agents. This is the case when probabilities change e.g. due to network congestion which is independent of the dynamics we want to control. Additionally, the definition of the mask transition as in Eq. 1 might violate the Markov assumption, since given the history of masks the probabilities might be different than given only the current mask (e.g. when an agent stays unavailable for at least two time steps). Alternatively, we can define the state as $\mathcal{S} = \mathcal{S}_E \times \mathcal{S}_M$, with $\mathcal{S}_M$ representing the communication state that is not observable by the agents. In that case the problem becomes a subclass of Dec-POMDPs. The same is true if the central controller has some additional information, since agents can't observe such in case of the mask state being 0 or when the observation function does not just reflect the local state, but is truly partially observable and has the general stochastic form known from Dec-POMDPs.

3.2 On the Complexity

When augmenting the state vector with additional $|\mathcal{M}| = 2^n$ failure scenarios, it can easily be seen that this drastically increases the complexity of the problem compared to the standard single agent decision problem that acts on complete information but also compared to the general case of a Dec-MDP in which agents only observe their local states. The size of the state space is now exponential in the number of agents, which is also true for their observation spaces. This implies that excessive amounts of collected data is needed during training because the number of samples required to get a good estimate of the expected reward also grows exponentially. This becomes even worse when the probability of disconnection is low, since the agent will only experience a failure state very rarely. Sparse exploration leads to a high variance in the reward signal and therefore to poor decisions. The action space is however not affected, due to the fact that the agents can always take the same actions, regardless of their connection status.

4 Solution Concepts

In this section we will discuss approaches to overcome issues with the proposed information structure, namely exponential state and observation space sizes, high variance for unlikely failure scenarios and varying input sizes for policies.

4.1 Subset of Failure Scenarios

In order to tackle the problem of exponential state and observation space sizes, a simple solution is to only account for a subset of failure scenarios $G_{\mathcal{M}} \subset \mathcal{M}$.

In case of a failure m that is not in that set, we choose some $m' \in G_{\mathcal{M}}$ for which $m' \trianglelefteq m$ holds and ignore some available information. Even though this leads to a suboptimal solution, we can reduce the number of training data needed, since we assumed the environment transition probabilities $\mathcal{T}_E$ not be affected by the communication state, i.e. we can control $\mathcal{S}_E$ regardless of $\mathcal{M}$. If more than one element of $G_{\mathcal{M}}$ is possible for m', we want to choose the element that is "closest" to m. For $m'_1 \trianglelefteq m'_2$ we choose m'_2, since we can make potentially better decisions the more information we include in the process. For $m'_1 \not\trianglelefteq m'_2$ however, we might want to select the mask that has more agents available, or randomly select one if the number of connected agents is equal. The minimal set $G_{\mathcal{M}}$ is $\{0\}$, since we can reduce every failure to that. When failures are very unlikely, it might for example make sense to train on $\{0, 1\}$, so we can act optimal most of the time and have some emergency strategy that applies to all the cases in which at least one agent has no connection to the central controller. When the probability for different agents to be unavailable is known a priori and does not depend on the environment state, optimization regarding $G_{\mathcal{M}}$ are possible, in that we include failure scenarios that are encountered most often. Otherwise, we can adapt $G_{\mathcal{M}}$ during training based on the frequency of experienced mask states. This might however prevent convergence of policies and further theoretical investigation is needed.

4.2 Independent Policies

When learning on an environment that exhibits low probabilities for certain failure scenarios or even for all other settings than 1, decision under such mask states might be poor due to the lack of experience. Instead, we can explicitly train these scenarios by restricting access to state information despite the actual connection state. While this reduces variance, it will lead to a pessimistic policy that overvalues communication failures. Reinforcement learning algorithms estimate the expected reward by drawing samples of state trajectories by interacting with the environment. Unfortunately, this process becomes biased when occurrences of mask states are altered. Alternatively, we can train multiple independent policies for each of the considered mask states and apply the action of the corresponding policy during execution. This however presumes a low rate in change of the mask state since the policies assume that the information structure they were trained on does not change at all. Frequent changes in the connection state will therefore degrade the overall performance. Nonetheless, it allows us to sample as much training data for a given mask state as needed to converge without affecting other policies. This idea has been implemented in [2], though restricted to the case of $G_{\mathcal{M}} = \{0, 1\}$, and showed being a robust backup for communication failures.

Note that an agent that is disconnected does not know the actual mask state and therefore can't decide which policy to apply. It hence must use the same policy whenever it is disconnected, so every agent needs to have only one policy for that case which will lead to a bias for these policies anyway, though it is assumed to be much smaller than using the same policy for all mask states

given that the probability for mask states during training does not reflect the occurrences of mask states during execution. Disconnected agents may also use some fixed rule based policy if they are disconnected. Another advantage of this approach is that policies have fixed input sizes which simplifies the use of artificial neural networks, which are commonly used in reinforcement learning to realize the policy [3].

4.3 Reconstruction

Another approach is to learn policies that act on complete information only. After convergence, fixing the policies enables the reduction of the underlying decision process to a Markov chain, since action probabilities do not change and can be seen as a part of transition function. This allows us to learn one or more prediction models by feeding back the reconstruction loss of the whole state given the partial state and the history of previous state information, e.g. by use of a recurrent neural network. During execution, we impute missing values by use of the prediction model and feed the complete state to the learned policy. Combinations with learning only on a subset of mask states are also possible in that we can fall back to one of the closest elements $m' \in G_{\mathcal{M}}$ with $m \trianglelefteq m'$ by imputing missing parts of m', given actual mask state m. Fallback in both directions are also possible, e.g. we can ignore information if the encountered mask state has only little more information than some mask state we trained on and impute values if only little information is missing.

A disadvantage of this approach is the fact that the longer a connection is broken, the more the performance is likely to degrade based on the assumption that the divergence between the predicted state and the actual state becomes greater the longer a communication failure persists. Policies cannot distinguish between missing information that has been imputed and a state component that was actually received by the central controller, since the prediction model has been trained after convergence of the policy. This state aliasing then may lead bad decisions. To overcome this issue, we can instead include the uncertainty induced by missing observations in the training process. Agents may for example maintain a belief over other agents' states, e.g. they learn the probability distribution of the state of other agents given the available information. This can then be used as an input to the policy, similar to work that has been done in [1].

5 Conclusion

In this paper we gave a formal definition of a communication structure that is posed by a central controller exchanging information with agents using error-prone communication channels in a spatially distributed control problem. We identified the problem to be a subclass of Dec-MDPs or Dec-POMDPs, dependent on whether the central controller has additional information that agents cannot observe locally and whether communication failures depend on some non-observable process. We discussed the complexity of the problem and argued

that the problem is even harder to solve than a multi-agent system with fixed observation spaces due to the fact that observation space sizes in the proposed formalism are exponential in the number of agents. In addition to that we identified further problems that may arise when using reinforcement learning algorithms to solve the problem, namely the high variance in the reward signal when the probabilities for communication failures are low and the varying input sizes for policies. Several solution concepts have been proposed that involve breaking down the complexity by considering only a subset of failure scenarios, learning independent policies for each failure scenario, reconstructing missing information and learning policies that incorporate the state uncertainty in the training process. Nevertheless, investigation of other information structures than those incorporating a central controller is needed. There is a lack of theoretical foundation for multi-agent systems that include the possibility of communication failures. We hope that this paper will motivate further research in this direction.

References

1. Zhou, H., et al.: Multiagent Bayesian deep reinforcement learning for microgrid energy management under communication failures. IEEE Internet Things J. **9**(14), 11685–11698 (2021)
2. Zhang, Z., Tian, W., Liao, Z.: Towards coordinated and robust real-time control: a decentralized approach for combined sewer overflow and urban flooding reduction based on multi-agent reinforcement learning. Water Res. **229**, 119498 (2023)
3. Li, Y.: Deep reinforcement learning: an overview. arXiv preprint arXiv:1701.07274 (2017)
4. Zhang, K., Yang, Z., Başar, T.: Multi-agent reinforcement learning: a selective overview of theories and algorithms. In: Vamvoudakis, K.G., Wan, Y., Lewis, F.L., Cansever, D. (eds.) Handbook of Reinforcement Learning and Control. SSDC, vol. 325, pp. 321–384. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-60990-0_12
5. Oroojlooy, A., Hajinezhad, D.: A review of cooperative multi-agent deep reinforcement learning. Appl. Intell. **53**(11), 13677–13722 (2023)
6. Oliehoek, F.A., Amato, C.: A Concise Introduction to Decentralized POMDPs, vol. 1. Springer International Publishing, Cham, Switzerland (2016). https://doi.org/10.1007/978-3-319-28929-8
7. Goldman, C.V., Zilberstein, S.: Decentralized control of cooperative systems: categorization and complexity analysis. J. Artif. Intell. Res. **22**, 143–174 (2004)
8. Bernstein, D.S., Givan, R., Immerman, N., Zilberstein, S.: The complexity of decentralized control of Markov decision processes. Math. Oper. Res. **27**(4), 819–840 (2002)
9. Zhong, Y., et al.: Heterogeneous-agent reinforcement learning. arXiv preprint arXiv:2304.09870 (2023)
10. Becker, R., Zilberstein, S., Lesser, V., Goldman, C.V.: Transition-independent decentralized Markov decision processes. In: Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 41–48 (2003)
11. Nair, R., Varakantham, P., Tambe, M., Yokoo, M.: Networked distributed POMDPs: a synthesis of distributed constraint optimization and POMDPs. In: AAAI, vol. 5, pp. 138–139 (2005)