Learning-based Physical Layer Communications for Multiagent Collaboration

Arsham Mostaani[1], Osvaldo Simeone[2], Symeon Chatzinotas[1], Bjorn Ottersten[1]
Emails: arsham.mostaani, symeon.chatzinotas, bjorn.ottersten@uni.lu, osvaldo.simeone@kcl.ac.uk
[1] SnT, University of Luxembourg, Luxembourg
[2] Department of Informatics, King's College London, UK

***Abstract*—Consider a collaborative task carried out by two autonomous agents that can communicate over a noisy channel. Each agent is only aware of its own state, while the accomplishment of the task depends on the value of the joint state of both agents. As an example, both agents must simultaneously reach a certain location of the environment, while only being aware of their own positions. Assuming the presence of feedback in the form of a common reward to the agents, a conventional approach would apply separately: (*i*) an off-the-shelf coding and decoding scheme in order to enhance the reliability of the communication of the state of one agent to the other; and (*ii*) a standard multiagent reinforcement learning strategy to learn how to act in the resulting environment. In this work, it is argued that the performance of the collaborative task can be improved if the agents learn how to jointly communicate and act. In particular, numerical results for a baseline grid world example demonstrate that the jointly learned policy carries out compression and unequal error protection by leveraging information about the action policy.**

I. Introduction

Consider the rendezvous problem illustrated in Fig. 1 and Fig. 2. Two agents, e.g., members of a SWAT team, need to arrive at the goal point in a grid world at precisely the same time, while starting from arbitrary positions. Each agent only knows its own position but is allowed to communicate with the other agent over a noisy channel. This set-up is an example of cooperative multiple agent problems in which each agent has partial information about the environment [1], [2]. In this scenario, communication and coordination are essential in order to achieve the common goal [3]–[5], and it is not optimal to design the communication and control strategies separately [5], [6].

Assuming the presence of a delayed and sparse common feedback signal that encodes the team reward, cooperative multiagent problems can be formulated in the framework of multiagent reinforcement learning. As attested by the references [1], [2], [7] mentioned above, as well by [8], [9], this is a well-studied and active field of research. To overview some more recent contributions, paper [10] presents simulation results for a distributed tabular Q-learning scheme with instantaneous communication. Deep learning approximation methods

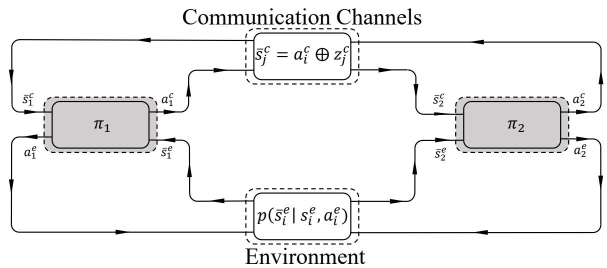

Figure 1. An illustration of the cooperative multiagent system with noisy communicaitons under study.

are applied in [11] for Q-learning and in [12] for actor-critic methods. In [13], a method is proposed that keeps a centralized critic in the form of a Q-function during the learning phase and uses a counter-factual approach to carry out credit assignment for the policy gradients.

The works mentioned above assume a noiseless communication channel between agents or use noise as a form of regularization [9]. In contrast, in this paper, we consider the problems of simultaneously learning how to communicate on a noisy channel and how to act, creating a bridge between the emerging literature on machine learning for communications [14] and multiagent reinforcement learning. A closely related work is [15], in which, however, the focus is on the joint optimization of scheduling and actions in a multiagent system.

For a sequential two-agent reinforcement learning problem, we formulate distributed Q-learning algorithms that learn simultaneously what to communicate over the inter-agent noisy channel and which actions to take in the environment. As a numerical example, we consider models with sequential or simultaneous communications and actions. For the rendezvous problem illustrated in Fig. 2, we provide numerical performance comparisons between the proposed multiagent reinforcement learning scheme and a conventional method based on the separate optimization of action and communication policies. While the proposed scheme jointly learns how to act and communicate, the baseline conventional method applies separately an off-the-shelf channel coding scheme for communication of an agent's state and multiagent

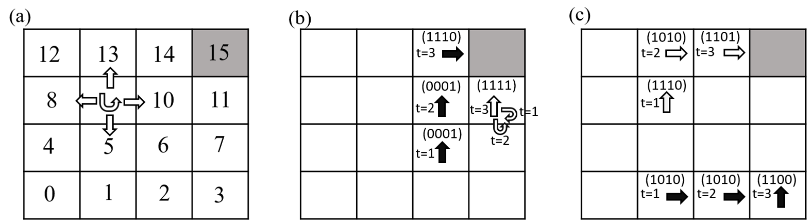

Figure 2. The rendezvous problem: (a) illustration of the environment state-space, $\mathcal{S}^e$, i.e., the location on the grid, of the environment action space $\mathcal{A}^e$, denoted by arrows, and of the goal state, marked with gray background; (b) demonstration of a sampled episode, where arrows show the environment actions taken by the agents (empty arrows: actions of agent 1, solid arrows: actions of agent 2) and the $B = 4$ bits represent the message sent by each agent. A larger reward $R_2 > R_1$ is given to both agents when they enter the goal point at the same time, as in the example; (c) in contrast, R_1 is the reward accrued by agents when only one agent enters the goal position.

reinforcement learning to adapt the action policies. The advantages of the jointly optimized policy are seen to result from data compression and unequal error protection mechanisms that are tailored by each agent to the action policy.

II. Problem Set-up

As illustrated in Fig. 1 and Fig. 2, we consider a cooperative multiagent system comprising of two agents that communicate over noisy channels. The system operates in discrete time, with agents taking actions and communicating in each time step $t = 1, 2, ...$ While the approach can be applied to any multiagent system, in order to fix the ideas, we focus here on the rendezvous problem illustrated in Fig. 2. The two agents operate on an $n \times n$ grid world and aim at arriving at the same time at the goal point on the grid. The position of each agent $i \in \{1, 2\}$ on the grid determines its environment state $s_i^e \in \mathcal{S}^e = [n] \times [n]$, where $[n] = \{1, 2, ..., n\}$. Each agent ith environment state $s_i^e \in \mathcal{S}^e$ can also be written as the pair $s_i^e = \langle s_{i,x}^e, s_{i,y}^e \rangle$, with $s_{i,x}^e, s_{i,y}^e \in [n]$ being respectively the horizontal and vertical coordinates. Each episode terminates as soon as an agent or both visit the goal point which is denoted as $\mathcal{S}_T^e = \{s_T^e\}$. At time $t = 1$, the initial position $\mathrm{s}_{i,t=1}^e$, is randomly and uniformly selected amongst the non-goal states. Note that, throughout, we use Roman font to indicate random variables and the corresponding standard font for their realizations.

At any time step $t = 1, 2, ...$ each agent i has information about its position, or environment state, $s_{i,t}^e$ and about the signal $s_{i,t}^c$ received from the other agent $j \neq i$ at the previous time step $t - 1$. Based on this information, agent i selects its environment action $a_i^e = \langle a_{i,x}^e, a_{i,y}^e \rangle$ from the set $\mathcal{A}^e = \{\langle 1, 0\rangle, \langle -1, 0\rangle, \langle 0, 0\rangle, \langle 0, 1\rangle, \langle 0, -1\rangle\}$, where $a_{i,x}^e$ and $a_{i,y}^e$ represent the horizontal and vertical move of agent i on the grid. Furthermore, it chooses the communication message to send to the other agent by selecting a communication action $a_i^c \in \mathcal{A}^c = \{0, 1\}^B$ of B bits.

The environment state transition probability for agent i can be described by the equation $\mathrm{s}_{i,t+1}^e = \mathrm{s}_{i,t}^e + \mathrm{a}_{i,t}^e$, with the caveat that, if an agent on an edge of the grid world selects an action that transfers it out, the environment keeps the agent at its current location.

Agents communicate over interference-free channels using binary signaling, and the channels between the two agents are independent Binary Symmetric Channels (BSCs), such that the received signal is given as

$$\mathrm{s}_{j,t+1}^c = a_{i,t}^c \oplus \mathrm{z}_{j,t}^c, \tag{1}$$

where the XOR operation $\oplus$ is applied element-wise, and $\mathrm{z}_{j,t}^c$ has independent identically distributed (i.i.d.) Bernoulli entries with bit flipping probability $q \leq 0.5$. Extensions to other channels are conceptually straight forward.

We first consider a scenario with simultaneous communication and actions. Accordingly, agent i follows a policy π_i that maps the observations $s_i = \langle s_i^e, s_i^c \rangle$ of the agent into its actions $a_i = \langle a_i^e, a_i^c \rangle$. The policy is generally stochastic, and we write it as the conditional probability $\pi_i(a_i|s_i)$ of taking action a_i while in state s_i. We assume the joint policy π_i to select both the environment action a_i^e and transmitted signal a_i^c based on the overall state s_i. The overall joint policy π is given by the product $\pi = \pi_1 \times \pi_2$. It is noted that the assumed memoryless stationary policies are sub-optimal under partial individual observability of environment state [1]. A model that assumes sequential communications and actions will be covered in Sec. III. B.

At each time t, given states $\langle s_1, s_2 \rangle$ and actions $\langle a_1, a_2 \rangle$, both agents receive a single team reward

$$\mathrm{r}_t = \begin{cases} R_1, & \text{if } s_i^e \neq s_j^e \in \mathcal{S}_T^e \\ R_2, & \text{if } s_i^e = s_j^e \in \mathcal{S}_T^e, \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

where $R_1 < R_2$. Accordingly, when only one agent arrives at the target point s_T^e, a smaller reward R_1 is obtained at the end of the episode, while the larger reward R_2 is attained when both agents visit the goal point at the same time. Note that this reward signal encourages coordination between agents which in turn can benefit from inter-agent communications.

The goal of the multiagent system is to find a joint policy π that maximizes the expected return. For given initial states, $(s_{1,t=1}, s_{2,t=1})$, this amounts to solving the problem

$$\underset{\pi}{\text{maximize}} \qquad \mathbb{E}_\pi[\mathrm{G}_t|\mathrm{s}_{1,t=1} = s_{1,t=1}, \mathrm{s}_{2,t=1} = s_{2,t=1}], \tag{3}$$

where

$$\mathrm{G}_t = \sum_{t=1}^{\infty} \gamma^t \mathrm{r}_t \tag{4}$$

is the long-term discounted return, with $\gamma \in (0, 1]$ being the reward discount factor. The expected return in (3) is calculated with respect to the probability of the trace of states, actions, and rewards induced by the policy π [16].

III. Learned Communication

In this section we consider a strategy that jointly learns the communication and the environment action policies of both agents, with the aim of addressing problem (3). To this end we adapt the distributed Q-learning algorithm to the setup at hand [5]. Accordingly, given the received communication signal s_i^c and the local environment state s_i^e, each agent i selects its environment actions a_i^e and communication actions a_i^c simultaneously by following its policy π_i. We will then also consider an alternative scenario with sequential communications and actions.

A. Simultaneous Communications and Actions

In order to define agent ith policy π_i, we introduce the state-action value function $Q_i^e(s_i^e, s_i^c, a_i^e, a_i^c)$. We recall that a state-action function $Q(s, a)$ provides an estimate of the expected return (4) when starting from the state s and taking action a. As detailed, the action-value function is trained based on its interaction with environment. For a given action-value function, to control the trade-off between exploitation and exploration, we adopt the Upper Confidence Bound (UCB) method [16]. UCB selects the actions $\mathrm{a}_{i,t}^c$ and $\mathrm{a}_{i,t}^e$ as

$$\langle a_{i,t}^e, a_{i,t}^c \rangle = \underset{a_i^e, a_i^c}{\operatorname{argmax}}\ Q_i(s_{i,t}^e, s_{i,t}^c, a_i^e, a_i^c) + c\sqrt{\frac{\ln(T_t)}{N_i(s_{i,t}^e, s_{i,t}^c, a_i^e, a_i^c)}}, \tag{5}$$

where $c > 0$ is a constant; T_t is the total number of time steps in the episodes considered up to the current time t in a given training epoch; and table $N_i(s_{i,t}^e, s_{i,t}^c, a_i^e, a_i^c)$ counts the total number of times that the states $\langle s_{i,t}^e, s_{i,t}^c \rangle$ has been visited and the actions $\langle a_i^e, a_i^c \rangle$ selected among the previous T_t steps. When c is large enough, UCB encourages the exploration of the state-action tuples that have been experienced fewer times.

The update of the action value function based on the available observations at time t follows the off-policy Q-learning algorithm, i.e., [16]

$$Q_i(s_{i,t}^e, s_{i,t}^c, a_{i,t}^e, a_{i,t}^c) \leftarrow (1-\alpha) Q_i(s_{i,t}^e, s_{i,t}^c, a_{i,t}^e, a_{i,t}^c) + \alpha\gamma\Big(r_t + \max_{a_i^e, a_i^c} Q_i(s_{i,t+1}^e, s_{i,t+1}^c, a_i^e, a_i^c)\Big), \tag{6}$$

where $\alpha > 0$ is a learning rate parameter. The full algorithm is detailed in Algorithm 1. At the end of the training process, policy $\pi_i(a_i^e, a_i^c|s_i^e, s_i^c)$ can be obtained by

$$\pi_i(a_i^e, a_i^c|s_i^e, s_i^c) = \delta\Big(\langle a_i^e, a_i^c \rangle - \underset{\langle a_i^e, a_i^c \rangle}{\operatorname{argmax}}\, Q_i(s_i^e, s_i^c, a_i^e, a_i^c)\Big), \tag{7}$$

where $\delta(\cdot)$ is the Dirac delta function.

As a baseline, we also consider a conventional scheme that optimizes communications and actions separately. For communication, each agent i sends its environment state s_i^e to the other agent by using a channel code for the given noisy channel. Note that compression is not possible, since all states are a priori equally likely. Agent j obtains an estimate $\hat{s}_i^e$ of the environment state of i by using a channel decoder based on the received signal. This estimate is used as if it were the correct position of the other agent to define the environment state-action value function $Q_j^e(s_j^e, \hat{s}_i^e, a_j^e)$. This function is updated using Q-learning and the UCB policy in a manner similar to Algorithm 1.

Algorithm 1 Learned Simultaneous Communications and Actions

Input: γ, α, and c
Initialize all-zero Q-table $Q_i(s_i^e, s_i^c, a_i^e, a_i^c)$ and table $N_i(s_i^e, s_i^c, a_i^e, a_i^c)$, for $i = 1, 2$
for each episode $m = 1 : M$ **do**
  Randomly initialize $\langle s_{1,t=1}^e, s_{2,t=1}^e \rangle \notin \mathcal{S}_T^e$
  Randomly initialize $\langle s_{1,t=1}^c, s_{2,t=1}^c \rangle$
  set $t_m = 1$
  while $\langle s_{1,t}^e, s_{2,t}^e \rangle \notin \mathcal{S}_T^e$ **do**
    Jointly select $a_{i,t}^e = a_i^e \in \mathcal{A}_i^e$ and $a_{i,t}^c = a_i^c \in \mathcal{A}_i^c$ by solving (5), for $i = 1, 2$
    Increment $N_i(s_{i,t}^e, s_{i,t}^c, a_{i,t}^e, a_{i,t}^c)$, for $i = 1, 2$
    Obtain message $s_{i,t+1}^c$, for $i = 1, 2$
    Obtain reward r_t and move to $s_{i,t+1}^e$, for $i = 1, 2$
    for $i = 1, 2$ **do**
      Update $Q_i(s_{i,t}^e, s_{i,t}^c, a_{i,t}^e, a_{i,t}^c)$ by following (6)
    end
    $t_m = t_m + 1$
  end
  Compute $\sum_{t=1}^{t_m-1} \gamma^t r_t$ for the mth episode
end
Output: $\pi_i(a_i^e, a_i^c|s_i^e, s_i^c)$ by following (7) for $i = 1, 2$

B. Sequential Communications and Actions

In the problem formulation considered so far, agents select both environment and communication actions simultaneously, which inherently leads to a delay in the inter-agent communication. In fact, information embeded by agent i in its communication action $a^c_{i,t}$ cannot be used earlier than in time step $t+1$ by the other agent $j \neq i$. As we have seen in Sec. II, this model can be formalized using the standard Markov Decision Process formulation. We now study an alternative set-up, in which communication is followed by the selection of an environment action at each time instant t. A similar model was considered in [17].

To elaborate, at each time t, each agent i first observes its environment state $s^e_{i,t}$ and then selects a communication action $a^c_{i,t}$ by following a policy $\pi^c_i(a^c_i|s^e_i)$. In the second phase of time step t, agent i receives the communication message $s^c_{i,t}$ over the channel. Finally, agent i selects its environment action $a^e_{i,t}$ by following its policy $\pi^e_i(a^e_i|s^e_i, s^c_i)$. Both conventional communication and jointly learned communication schemes can be easily adapted to this communication model. We provide details for learned communication in Appendix.

IV. Results and Discussions

In this section, we provide numerical results for the rendezvous problem described in Sec. II. As in Fig. 2, the grid world is of size 4×4, i.e. $n=4$, and it contains one goal point at the right-top position. Environment states are numbered row-wise starting from the left-bottom as shown in Fig. 2(a). All the algorithms are run for 50 independent epochs. For each agent i the initial state $s^e_{i,t=1} \notin \mathcal{S}^e_T$ in each episode is drawn uniformly from all non-terminal states.

We compare the conventional communication and the learned communication schemes reviewed in the previous section. Conventional communication transmits the position of an agent on the grid as the 4-bit binary version of the indices in Fig. 2(a) after encoding via a binary cyclic (B,4) code, with the generator polynomial $1+X^2+X^3$. The received message is then decoded by syndrome decoding.

In order to reduce the dimensions of the state-action space for learned simultaneous communications and actions, in our experiments, we use disjoint policies π^e_i and π^c_i to separately select environment and communication actions. Accordingly, given the received communication signal s^c_i and the local environment state s^e_i, each agent i selects its environment actions a^e_i by following a policy π^e_i based on a state-action value function $Q^e_i(s^e_i, s^c_i, a^e_i)$, while it chooses its communication action a^c_i by following a second policy π^c_i, based on a state-action function $Q^c_i(s^e_i, a^c_i)$.

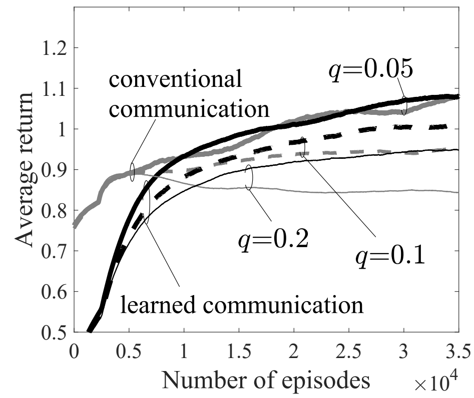

Figure 3. Average return for conventional communication and learned communication when $B=7$ with simultaneous actions and communications.

The performance of each scheme is evaluated in terms of the discounted return in (4), averaged over all epochs and smoothed using a moving average filter of memory equal to 4,000 episodes. The rewards in (2) are selected as $R_1 = 1$ and $R_2 = 3$, while the discount factor is $\gamma = 0.9$. A constant learning rate $\alpha = 0.15$ is applied, and the exploration rate c of the UCB policy is selected from the set $\{0.3, 0.4, 0.5\}$ such that it maximizes the average return at the end of the episodes in an epoch.

We first investigate the impact of the channel noise by considering different values of the bit flip probability q for simultaneous communications and actions. In Fig. 3 it is observed that conventional communication performs well at the low bit flipping rate of $q = 0.05$, but at higher rates of q learned communication outperforms conventional communication after a sufficiently large number of episodes. Importantly, for $q = 0.2$, the performance of conventional communication degrades through episodes due to the accumulation of noise in the observations, while learned communication is seen to be robust against channel noise.

In Fig. 4, we consider the case of sequential communications and actions. In this case, for $q=0$ conventional communication is optimal [1]. In contrast, for noisy channels, learned communication provides significant gain, e.g., 20% for $q=0.15$ and $q=0.20$.

We now discuss the reasons that underlie the performance advantages of learned communication. We start by analyzing the capability of learned communication to compress the environment state information before transmission. To obtain quantitative insights, we measure the mutual information $I(\mathrm{s}^e_i; \mathrm{a}^c_i)$ between the environment state s^e_i and the communication action a^c_i of an agent i as obtained under the policy learned after 20,000 episodes for $q = 0, 0.05, 0.1, 0.15, 0.2$ [18]. Fig. 5 plots the mutual information as a function of the bit flipping probability q for learned communication. For conventional communication scheme the communication

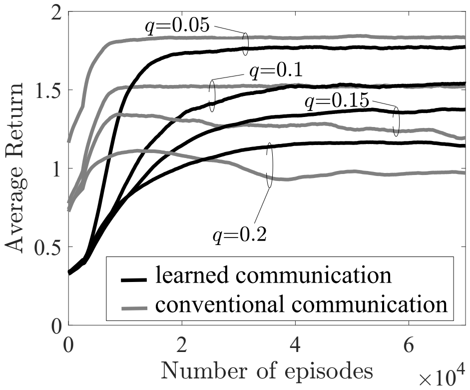

Figure 4. Average return for the conventional and for the learned communication, when $B = 7$ with sequential communications and actions.

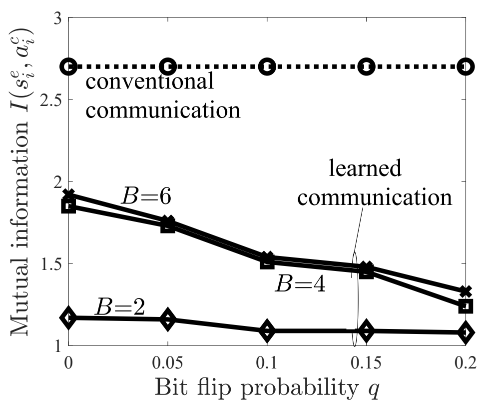

Figure 5. Mutual information between an agent's environment state s_i^e and the communication action a_i^c versus the bit flip probability q for simultaneous conventional and learned communication after 20,000 episodes ($B = 2, 4, 6,\ q = 0.05, 0.10, 0.15, 0.20$).

message a_i^c is a deterministic function of the state s_i^e and hence we have $I(\mathrm{s}_i^e; \mathrm{a}_i^e) = H(\mathrm{s}_\mathrm{i}^\mathrm{e})$, which is independent of q and B. In the absence of channel noise, i.e., $q = 0$, learned communication compresses by almost 30% the information about the environment state distribution s_i^e when $B = 6$. This reduction becomes even more pronounced as the channel noise increases or when agents have a tighter bit-budget, i.e., for smaller values of B.

We proceed by investigating how compression is carried out by jointly optimizing the agent's environment action and communication action policies. We will also see that learned communication carries out a form of unequal error protection. To this end, Fig. 6 illustrates a sample of the learned action and communication policies π_i^e and π_i^c for agent $i = 1$ when $q = 0.05$ and $B = 4$ after 30,000 episodes of training in the presence of communication delays. In this figure, arrows show the dominant environment action(s) a_i^e selected at each location; the bit sequences represent the communication action a_i^c selected at each location; and the colour of each square shows how likely it is for the position to be visited by agent i.

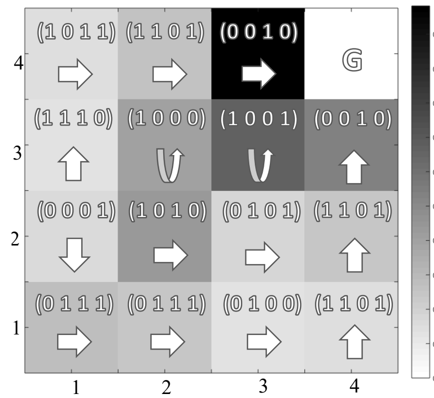

Figure 6. Illustration of a learned communication action policy when there is no communication delay ($B = 4$, $q = 0.05$). Locations with brighter colors are more likely to be visited. Arrows show the dominant action selected at any location. Bit strings show the message sent at a certain location.

We can observe that compression is achieved by assigning same message to different locations. In this regard, it is interesting to note the interplay with the learned action policy: groups of states are clustered together if states have similar distance from the goal point, such as $\{\langle 4, 3\rangle, \langle 3, 4\rangle\}$ and $\{\langle 4, 2\rangle, \langle 2, 4\rangle\}$; or if they are very far from the goal point such as $\{\langle 1, 2\rangle, \langle 1, 3\rangle\}$. Furthermore, it is seen that the Hamming distance of the selected messages depends on how critical it is to distinguish between the corresponding states. This is because it is important for an agent to realize whether the other agent is close to the terminal point.

V. Conclusions

In this paper, we have studied the problem of decentralized control of agents that communicate over a noisy channel. The results demonstrate that jointly learning communication and action policies can significantly outperform methods based on standard channel coding schemes and on the separation between the communication and control policies. We have illustrated that the underlying reason for the improvement in performance is the learned ability of the agents to carry out data compression and unequal error protection as a function of the action policies.

VI. Acknowledgements

The work of Arsham Mostaani, Symeon Chatzinotas and Bjorn Ottersten is supported by European Research Council (ERC) advanced grant 2022 (Grant agreement ID: 742648). Arsham Mostaani and Osvaldo Simeone have received funding from the ERC under the European Union's Horizon 2020 Research and Innovation Program (Grant Agreement No. 725731).

References

[1] D. V. Pynadath and M. Tambe, "The communicative multiagent team decision problem: Analyzing teamwork theories and models," *Journal of Artificial Intelligence Research*, vol. 16, pp. 389–423, Jun. 2002.

[2] G. Weiss, *Multiagent systems: a modern approach to distributed artificial intelligence*, MIT press, 1999.

[3] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *Proc. International Conference on Machine Learning*, San Francisco, CA, USA, 1998, pp. 487–494, Morgan Kaufmann Publishers Inc.

[4] L. Busoniu, R. Babuska, and B. D. Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Trans. Systems, Man, and Cybernetics, Part C*, vol. 38, no. 2, pp. 156–172, Mar. 2008.

[5] M. Lauer and M. A. Riedmiller, "An algorithm for distributed reinforcement learning in cooperative multi-agent systems," in *Proc. Conference on Machine Learning*. 2000, pp. 535–542, Morgan Kaufmann Publishers Inc.

[6] A. Sahai and P. Grover, "Demystifying the witsenhausen counterexample [ask the experts]," *IEEE Control Systems Magazine*, vol. 30, no. 6, pp. 20–24, Dec. 2010.

[7] F. Fischer, M. Rovatsos, and G. Weiss, "Hierarchical reinforcement learning in communication-mediated multiagent coordination," in *Proc. IEEE Joint Conference on Autonomous Agents and Multiagent Systems, 2004. AAMAS 2004.*, New York, Jul. 2004, pp. 1334–1335.

[8] S. Sukhbaatar, R. Fergus, et al., "Learning multiagent communication with backpropagation," in *Proc. Advances in Neural Information Processing Systems*, Barcelona, 2016, pp. 2244–2252.

[9] J. Foerster, Y. Assael, N. D. Freitas, and Sh. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Proc. Advances in Neural Information Processing Systems*, Barcelona, 2016, pp. 2137–2145.

[10] Q. Huang, E. Uchibe, and K. Doya, "Emergence of communication among reinforcement learning agents under coordination environment," in *Proc. IEEE Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*, Sept. 2016, pp. 57–58.

[11] J. Foerster, N. Nardelli, G. Farquhar, Ph. Torr, P. Kohli, and Sh. Whiteson, "Stabilising experience replay for deep multi-agent reinforcement learning," *arXiv preprint arXiv:1702.08887*, 2017.

[12] R. Lowe, L. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. Advances in Neural Information Processing Systems*, Long Beach, 2017, pp. 6382–6393.

[13] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and Sh. Whiteson, "Counterfactual multi-agent policy gradients," *arXiv preprint arXiv:1705.08926*, 2017.

[14] Osvaldo Simeone, "A very brief introduction to machine learning with applications to communication systems," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 4, pp. 648–664, 2018.

[15] Daewoo Kim, Sangwoo Moon, David Hostallero, Wan Ju Kang, Taeyoung Lee, Kyunghwan Son, and Yung Yi, "Learning to schedule communication in multi-agent reinforcement learning," *arXiv preprint arXiv:1902.01554*, 2019.

[16] R. Sutton and A. G. Barto, *Introduction to reinforcement learning*, vol. 135, MIT Press, 2 edition, Nov. 2017.

[17] P. Xuan, V. Lesser, and Sh. Zilberstein, "Communication decisions in multi-agent cooperation: Model and experiments," in *Proc. ACM Conference on Autonomous Agents*, 2001, pp. 616–623.

[18] Ryan Lowe, Jakob Foerster, Y-Lan Boureau, Joelle Pineau, and Yann Dauphin, "On the pitfalls of measuring emergent communication," *arXiv preprint arXiv:1903.05168*, 2019.

Appendix

Details of the sequential communications-actions policy can be found in Algorithm 2 below. At the end of the training process, policy $\pi_i^e(a_i^e|s_i^e, s_i^c)$ can be obtained by

$$\pi_i^e(a_i^e|s_i^e, s_i^c) = \delta\Big(a_i^e - \operatorname*{argmax}_{a_i^e} Q_i(s_i^e, s_i^c, a_i^e)\Big), \qquad (8)$$

and policy $\pi_i^c(a_i^c|s_i^e)$ by

$$\pi_i^c(a_i^c|s_i^e) = \delta\Big(a_i^c - \operatorname*{argmax}_{a_i^c} Q_i(s_i^e, a_i^c)\Big). \qquad (9)$$

Algorithm 2 Learned Sequential Communications and Actions

Input parameters: γ, α, and c
Initialize all-zero Q-tables $Q_i^e(s_i^e, s_i^c, a_i^e)$, $Q_i^c(s_i^e, s_i^c, a_i^c)$ and tables $N_i^e(s_i^e, s_i^c, a_i^e)$, $N_i^c(s_i^e, a_i^c)$, for $i = 1, 2$
for each episode $m = 1 : M$ **do**
  Randomly initialize $\langle s_{1,t=1}^e, s_{2,t=1}^e\rangle \notin \mathcal{S}_T^e$
  Randomly initialize $\langle s_{1,t=0}^c, s_{2,t=0}^c\rangle$
  $t = 1$
  while $\langle s_{1,t}^e, s_{2,t}^e\rangle \notin \mathcal{S}_T^e$ **do**
    for $i = 1, 2$ **do**
      Select $a_{i,t}^c \in \mathcal{A}_i^c$, by following UCB policy
      $a_{i,t}^c = \operatorname*{argmax}_{a_i^c} Q_i^c(s_{i,t}^c, a_i^c) + c\sqrt{\frac{\ln(\sum_{k=1}^m t_k)}{N_i^c(s_{i,t}^c, a_i^c)}}$
    end
    Obtain message $s_{i,t}^c$, for $i = 1, 2$, from channel
    if $t \geq 2$ **then**
      for $i = 1, 2$ **do**
        $Q_i^e(s_{i,t-1}^e, s_{i,t-1}^c, a_{i,t-1}^e) \leftarrow$
        $(1-\alpha)Q_i^e(s_{i,t-1}^e, s_{i,t-1}^c, a_{i,t-1}^e) +$
        $\alpha\gamma\big(r_{t-1} + \max_{a_i^e} Q_i^e(s_{i,t}^e, s_{i,t}^c, a_i^e)\big)$
      end
      if $s_{1,t}^e \in \mathcal{S}_T^e$ or $s_{2,t}^e \in \mathcal{S}_T^e$ **then break**
    end
    for $i = 1, 2$ **do**
      Select $a_{i,t}^e \in \mathcal{A}_i^e$ by following UCB policy
      $a_{i,t}^e = \operatorname*{argmax}_{a_i^e} Q_i^e(s_{i,t}^e, s_{i,t}^c, a_i^e) +$
      $c\sqrt{\frac{\ln(\sum_{k=1}^m t_k)}{N_i^e(s_{i,t}^e, s_{i,t}^c, a_i^c)}}$
    end
    Obtain reward r_t and move to the next environment state $s_{i,t+1}^e$, for $i = 1, 2$
    $Q_i^c(s_{i,t}^e, s_{i,t-1}^c, a_{i,t}^c) \leftarrow$
    $(1-\alpha)Q_i^c(s_{i,t}^e, s_{i,t-1}^c, a_{i,t}^c) + \alpha\gamma\big(r_t +$
    $\max_{a_i^c} Q_i^c(s_{i,t+1}^e, s_{i,t}^c, a_i^c)\big)$, for $i = 1, 2$
    $t = t + 1$
  end
  Compute $\sum_{t=1}^{t_m - 1} \gamma^t r_t$ for the mth episode
end
Output: $\pi_i^e(a_i^e|s_i^e, s_i^c)$ by following (8) for $i = 1, 2$
$\pi_i^c(a_i^c|s_i^e)$ by following (9) for $i = 1, 2$