2020 American Control Conference
Denver, CO, USA, July 1-3, 2020

Automata Guided Semi-Decentralized Multi-Agent Reinforcement Learning

Chuangchuang Sun, Xiao Li and Calin Belta

Abstract— **This paper investigates the problem of deploying a multi-robot team to satisfy a syntactically co-safe Truncated Linear Temporal Logic (scTLTL) task specification via multi-agent Reinforcement Learning (MARL). Due to the heterogeneous agents considered here, typical approaches cannot avoid the task assignment problem, which is inherently difficult and can sacrifice optimality (e.g., shortest path) through manual manipulation. scTLTL is exploited here to eliminate the task assignment as part of the learning process. MARL usually requires some direct or indirect coordination among agents to promote convergence and a tracker is needed to track the progress of satisfying the scTLTL specification given its temporal nature. We use the Finite State Automaton (FSA) to address these two issues. An FSA augmented Markov Decision Process (MDP) is constructed with each agent sharing the FSA state carrying the global information. Moreover, a metric called robustness degree is employed to replace the Boolean semantics and quantify the reward of gradually satisfying the scTLTL. Consequently, a language guided semi-decentralized Q-learning algorithm is proposed to maximize the return over the FSA augmented MDP. Simulation results demonstrate the effectiveness of the semi-decentralized multi-agent Q-learning while the complexity is significantly reduced.**

I. Introduction

Due to the insufficiency of a single robot, multi-robot team is pervasively deployed to accomplish tasks cooperatively, including but not limited to coverage, exploration and mapping [1], [2]. For tasks with complex structure and long horizons, formal specification languages, particularly temporal logics (TL) are commonly used as the task specification language due to its expressiveness and computable semantics. TL has found wide adoption in applications such as [3]. This paper addresses the problem of deploying a multi-robot team to satisfy a syntactically co-safe Truncated Linear Temporal Logic (scTLTL) task specification.

Related works: With heterogeneous complementary agents considered, task assignment is usually required to allocate agents to accomplish certain sub-tasks they are capable of. In a line of work in distributed sensing [4]–[6], either sub-teams are determined a prior or an online sub-team election is determined by controlling the agents to service the nearest regions of interest to minimize cost. However, manually tailored sub-team allocation will inevitably sacrifice optimality. Moreover, to satisfy the TL specification, in [7], abstracted workspace with path satisfying the TL specification is generated via sampling-based algorithms. While leveraged knowledge from the TL specification is exploited to enhance efficacy, they do not consider known environment abstractions and do not apply to multiple agents cases. To avoid the sub-team selection for a multi-robot team and achieve the optimal satisfaction of the scTLTL specification, multi-agent reinforcement learning technique is applied here.

One of the core issues in RL is the design of the reward function and accurately incorporating task specifications into the reward. If the reward is inappropriately chosen to capture the semantics of complex task, the task might not be satisfied even when the return is maximized [8]. Work in [9] transforms the Boolean semantics of scTLTL into a continuous quantitative semantics called robustness degree. It is well suited to transform a TL formula into a real-valued reward function.

In multi-agent reinforcement learning, centralized learning cannot avoid the curse of dimensionality and thus arises the necessity of decentralized multi-agent learning. The authors of [10] propose an algorithm for multi-agent learning in a "centralized training with decentralized execution" scheme. Coordination-free methods for decentralized multi-agent learning assume that the optimal joint action is unique [11], which, however, is barely the case. As a result, coordination, either direct or indirect, is usually required in multi-agent reinforcement learning to promote convergence and optimality. The former includes coordination graphs [12] and using communication to negotiate action choices [13], while the latter includes joint action learners [14] and frequency maximum q-value heuristic [15]. In a word, the decentralized MARL algorithms require such a *coordinator*. Moreover, due to the temporal nature of TL tasks, the satisfaction of a given scTLTL is not only related to the current joint agent states but also historical trajectory. Work in [16] introduced the Finite State Automaton (FSA) as a tracker to track the process of satisfying an scTLTL, for a single agent. The automaton state carries the global information which depends on the joint states of all the agents operating cooperatively to satisfy a certain scTLTL. As a result, the FSA can serve both to coordinate the agents and track the satisfaction of the scTLTL.

Contribution: In this paper, we demonstrate the applicability of FSA in guiding MARL problems under temporal logic constraints. Specifically, we use the FSA to orchestrate the reward distribution and track task satisfaction in a multi-agent system. We propose a semi-decentralized Q-learning method that maximizes the expected long-term return based

Chuangchuang Sun is with the Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA. Email: ccsun1@mit.edu; Work performed while at Boston University. Xiao Li and Calin Belta are with the Department of Mechanical Engineering, Boston University, Brookline, MA 02446, USA. Emails: {xli87, cbelta}@bu.edu.

978-1-5386-8266-1/$31.00 ©2020 AACC

on the FSA augmented MDP and show its advantage in improved sample complexity and scalability over the centralized approach.

Structure: The rest of the paper is organized as follows. Preliminary knowledge on (multi-agent) RL, TL and FSA are briefly introduced in Section II. Detailed problem formulation and the transformation via the quantitative robustness degree and augmented MDP are presented in Section III. Based on the augmented MDP, both centralized and semi-decentralized Q-learning methods are proposed in Section IV. Numerical simulation results are demonstrated in Section V to validate the effectiveness of the algorithm and we conclude the paper in Section VI with a few remarks.

II. Preliminary

A. Multi-Agent Reinforcement Learning

We first start with the definition of Markov Decision Process (MDP) with discrete state-action spaces.

Definition 1: The Markov decision process is a tuple $\langle S, A, p, r\rangle$ where S is the set of the agent state in the environment, A is the set of agent actions, $p : S \times A \times S \to [0, 1]$ is the state transition probability distribution and $r : S \times A \times S \to \mathbb{R}$ is the reward function.

At state s_k with action a_k taken, the agent will end up in state s_{k+1} with probability $p(s_k, a_k, s_{k+1})$. Correspondingly, as the feedback of the current action from the environment, the agent will receive a reward $r(s_k, a_k, s_{k+1})$. Then the agent's goal is to find a optimal policy $\pi^* : S \to A$ (or $\pi^* : S \times A \to [0, 1]$ for stochastic policies) to maximize the expected discounted return. Mathematically, it is expressed as

$$\pi^* = \arg\max_{\pi} \mathbb{E}^{\pi}\Big[\sum_{t=0}^{T-1} \gamma^t r(s_t, a_k, s_{t+1})\Big], \tag{2.1}$$

where $\gamma \in (0, 1)$ is the discount factor and $\mathbb{E}^{\pi}(\bullet)$ is the expectation operator over policy π. Moreover, finite horizon is adopted here with T as the maximum time-steps of the policy π. Moreover, the action-value function (Q-function) $Q^{\pi} : S \times A \to \mathbb{R}$ is the expected return of a state-action pair (s, a) under a given policy π, i.e., $Q^{\pi}(s, a) = \mathbb{E}^{\pi}\Big[\sum_{t=0}^{T-1} \gamma^t r(s_t, a_k, s_{t+1})|s_0 = s, a_0 = a\Big]$. To maximize the return, the agent can first calculate the optimal Q-function, i.e., $Q^*(s, a) = \max_{\pi} Q^{\pi}(s, a)$ and then choose action for a given state s by the greedy policy as $\pi(s) = \arg\max_a Q^*(s, a)$. While it is computationally intensive to compute the Q-function directly, the Q-learning method keeps interacting with the environment and get rewards to estimate Q^* iteratively as follows [17]

$$\begin{aligned} Q(s_k, a_k) := Q(s_k, a_k) \\ +\alpha[r_{k+1} + \gamma \max_{a'} Q(s_{k+1}, a') - Q(s_k, a_k)], \end{aligned} \tag{2.2}$$

where $\alpha \in (0, 1]$ is the learning rate. The sequence generated in (2.2) is provably convergent to the optimal Q^* under certain conditions [17].

Subsequently, the definitions of multi-agent RL is presented as follows:

Definition 2: A stochastic game (SG) (Markov game) is a tuple $\langle N, S, \{A_i\}_{i\in N}, p, \{r_i\}_{i\in N}\rangle$, where $N = [1, \ldots, n]$ is the set of n agents, $\{S_i\}_{i\in N}$ is the set of states, $\{A_i\}_{i\in N}$ are the possible actions for n agents and thus lead to the joint action set as the Cartesian product of the action sets for each agent, i.e., $A = \times_{i=1}^{n} A_i$. Similar to the single agent case, $p : S \times A \times S \to [0, 1]$ is the state transition probability distribution and $r_i : S \times A \times S \to \mathbb{R}, \forall i \in N$ are the reward function for the agents.

In the multi-agent case, for a certain agent, all the other agents are also part of the environment. As a result, the environment is no longer stationary. The state transition p, agent reward r_i and hence the return R_i all depend on the joint action $a_k = [a_{1,k}^{\mathrm{T}}, \ldots, a_{n,k}^{\mathrm{T}}]^{\mathrm{T}}$ with $a_{i,k} \in A_i$ and $a_k \in A$. The joint policy π is formed by combining the single agent policy $\pi_i : S \times A_i \to [0, 1]$. The Q-function of each agent $Q_i^{\pi} : S \times A \to \mathbb{R}$ also depends on the joint action and is conditioned on the joint policy. For a comprehensive survey on multi-agent RL, interested readers are referred to [18].

B. Temporal Logic Rewards and Finite State Automata

A Truncated Linear Temporal Logic (TLTL) [9] formula is defined over the predicate $f(s) \le c$, where $f : \mathbb{R}^n \to \mathbb{R}$ is a function and $c \in \mathbb{R}$ is a constant. Derived from TLTL, tasks considered here are specified via syntactically co-safe Truncated Linear Temporal Logic (scTLTL). The scTLTL specification has the following syntax

$$\begin{aligned} p := \top | f(s) \le c | \bigcirc p_1 | \neg p_1 | \Diamond p_1 | p_1 \wedge p_2 | p_1 \vee p_2 | \\ p_1 \Rightarrow p_2 | p_1 \mathcal{U} p_2 | p_1 \mathcal{T} p_2, \end{aligned} \tag{2.3}$$

where $\top$ is the true boolean constant, $\bigcirc$(next), $\Diamond$ (eventually), $\mathcal{U}$ (until), $\mathcal{T}$ (then) are temporal operators, and $\neg$(negation/not), $\wedge$ (conjunction/and), and $\vee$ (disjunction/or) are Boolean connectives. Consider the example below. It denotes that the trajectory of the state will eventually visit p_1 and p_2 in this order.

Example 3: Consider the following scTLTL formula $\phi := \Diamond p_2 \wedge \neg p_2 \mathcal{U} p_1$, where ϕ entails that eventually p_2 is true and p_2 should not be true before p_1 is true (sequencing).

Given a finite horizon state trajectory $s_{t:t+k} := s_t s_{t+1} \ldots s_{t+k}$, a quantitative metric $\rho(s_{t:t+k}, \phi)$ called robustness degree (or robustness in short) is employed here to measure how a scTLTL formula is satisfied by $s_{t:t+k}$. Such robustness degree can be recursively defined as

$$\begin{aligned} \rho(s_{t:t+k}, \top) &= \rho_{\max} \\ \rho(s_{t:t+k}, f(s) \le c) &= c - f(s_t) \\ \rho(s_{t:t+k}, \neg\phi) &= -\rho(s_{t:t+k}, \phi) \\ \rho(s_{t:t+k}, \phi \Rightarrow \psi) &= \max(-\rho(s_{t:t+k}, \phi), \rho(s_{t:t+k}, \psi)) \\ \rho(s_{t:t+k}, \phi_1 \wedge \phi_2) &= \min(\rho(s_{t:t+k}, \phi), \rho(s_{t:t+k}, \psi)) \\ \rho(s_{t:t+k}, \phi_1 \vee \phi_2) &= \max(\rho(s_{t:t+k}, \phi), \rho(s_{t:t+k}, \psi)) \\ \rho(s_{t:t+k}, \bigcirc\phi) &= \rho(s_{t+1:t+k}, \phi), k > 0 \end{aligned} \tag{2.4, 2.5}$$

where $\rho_{\max}$ is the maximum robustness degree. A positive robustness degree implies that a specification is satisfied

and negative otherwise. Mathematically, $\rho(s_{t:t+k}, \phi) > 0 \Rightarrow s_{t:t+k} \models \phi$ and $\rho(s_{t:t+k}, \phi) < 0 \Rightarrow s_{t:t+k} \not\models \phi$. Maximizing the robustness degree can serve as medium to enforce the satisfaction of the specification.

An FSA is commonly used to track the satisfaction of a scTLTL formula on an automaton graph.

Definition 4: A finite state automata corresponding to an scTLTL formula ϕ is defined as a tuple $\mathcal{A}_\phi = \langle \mathbb{Q}_\phi, \Psi_\phi, q_0, p_\phi, \mathcal{F}_\phi \rangle$, where $\mathbb{Q}_\phi$ is a set of automaton states, Ψ_ϕ is the set of input alphabet, $q_0 \in \mathbb{Q}_\phi$ and $\mathcal{F}_\phi \in \mathbb{Q}_\phi$ are the initial state and set of final states, respectively. Moreover, $p_\phi : \mathbb{Q}_\phi \times \mathbb{Q}_\phi \to [0, 1]$ is the conditional state transition probability defined as

$$p_\phi(q_j|q_i) = \begin{cases} 1 & \text{if } \psi_{q_i,q_j} \text{ is true ,} \\ 0 & \text{otherwise,} \end{cases} \tag{2.6}$$

where $\psi_{q_i,q_j} \in \Psi_\phi$ is the input that could facilitate transition from the current state q_i to the next state q_j. Alternatively, due to the definition of the robustness degree, p_ϕ can also be defined as

$$p_\phi(q_j|q_i, s) = \begin{cases} 1 & \text{if } \rho(s, \psi_{q_i,q_j}) > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{2.7}$$

Note that in (2.7), ψ_{q_i,q_j} is a predicate without any temporal operators, the corresponding robustness degree is only evaluate at current time step, i.e., $\rho(s_{t:t+k}, \psi_{q_i,q_j}) = \rho(s_t, \psi_{q_i,q_j})$.

The transition between a pair of FSA states is directed and deterministic. For the directedness, it is intuitively understood that once an scTLTL specification is satisfied, it cannot be undone. The transition probability in (2.6) and (2.7) can be later integrated as part of an MDP transition. Also, the temporal operator $\Box$ (always) is neglected in (2.3) in order to establish the connection between the scTLTL and FSA. The transformation from an scTLTL formula to an FSA can be carried out automatically with packages like Lomap [19]. For the scTLTL formula in Example 3, the FSA is illustrated in Fig. 1. The FSA has four automaton states $\mathbb{Q}_\phi = \{q_0, q_1, q_f, q_{\text{trap}}\}$, where q_0 is the initial state, q_{trap} is the trap state denoting violation of the scTLTL, and the rest are to track the progress of satisfying ϕ. Reaching the final state $q_f \in \mathcal{F}_\phi$ indicates that the specification is satisfied. The input alphabet $\Psi_\phi = \{p_1 \wedge p_2, \neg p_1 \wedge \neg p_2, \neg_1 \wedge p_2, \neg p_1 \wedge \neg p_2\}$. Here the shorthand is used $p_2 = (p_2 \wedge p_1) \vee (p_2 \wedge \neg p_1)$.

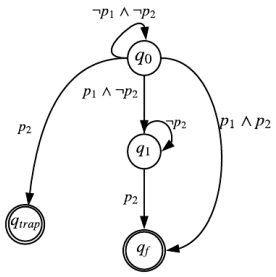


Fig. 1: Finite state automaton generated from formula $\phi := \Diamond p_2 \ \wedge \ \neg p_2 \mathcal{U} p_1$.

III. Problem Formulation

With a set of agents, the problem of interest here is to generate a joint state trajectory in the multi-agent system to satisfy an scTLTL specification ϕ. Mathematically, it is defined as follows.

Problem 5: Consider a Markov game $\mathcal{M} = \langle N, S, \{A_i\}_{i \in N}, p, \{r_i\}_{i \in N} \rangle$ as that in Definition 2 with unknown transition probability distribution. Find an overall joint policy $\pi^*_\phi = \times_{i=1}^n \pi^*_{i,\phi}$ with $\pi^*_{i,\phi}$ as the policy for agent i such that

$$\pi^*_\phi = \arg\max_{\pi_\phi} \mathbb{E}^\pi \left[\mathbf{1}\big(\rho(s_{0:T}, \phi) > 0\big) \right], \tag{3.8}$$

where $\mathbf{1}\big(\rho(s_{0:T}, \phi) > 0\big)$ is an indicator function with value 1 if $\rho(s_{0:T}, \phi) > 0$ and 0 otherwise. Moreover, the joint trajectory $s_{0:T}$ is defined as $s_{0:T} = \times_{i=1}^n s^{(i)}_{0:T}$ with $s^{(i)}_{0:T}$ as the trajectory of agent i.

Note that in N, the agent are heterogeneous, i.e., an agent is only capable of a certain subset of tasks defined by ϕ_1. Instead of assigning sub-team of agents to fulfill the task, we avoid this inherently difficult problem via appropriate reward function. For example, consider a sub-task ϕ_1 and a set of agents $N' \subseteq N$, which are capable of ϕ_1. Then for ϕ_1, $\rho(s_{0:T}, \phi_1) = \max_{i \in N'} \left[\rho(s^{(i)}_{0:T}, \phi_1) \right]$. Intuitively, it means that only those agents capable of ϕ_1 can get feedback (rewards) from visiting/accomplishing ϕ_1 while those incapable ones cannot. As a result, the learning process will automatically choose a capable agent to accomplish such a task in an optimal way.

Problem 5 is to find a policy, which generates a joint state trajectory satisfying the scTLTL specification ϕ in expectation. However, the satisfaction of an scTLTL depends on all the historical states such that it is necessary to have an extra state to track the process of satisfying ϕ. Consequently, given the finite state automaton introduced in Section II-B, we will integrate the automaton state into the MDP $\mathcal{M}$. On the other hand, as the FSA state depends on the states of all of the agents, so it has the global information and thus is a good candidate as the *coordinator* of the multi-agent learning. Otherwise, without communication with each other, each agent will simply learn their policy in a locally greedy way without pursuing global optimality. As a result, such disjoint local learning will either not converge due to the non-stationarity of the environment or only lead to a sub-optimum. Consequently, in the following, it is demonstrated that how an augmented MDP is built combining $\mathcal{M}$ and $\mathcal{A}_\phi$ in Definition 4.

A. FSA Augmented MDP

Definition 6: Consider a Markov game $\mathcal{M} = \langle N, S, \{A_i\}_{i \in N}, p, \{r_i\}_{i \in N} \rangle$ as that in Definition 2 and an FSA $\mathcal{A}_\phi = \langle \mathbb{Q}_\phi, \Psi_\phi, q_0, p_\phi, \mathcal{F}_\phi \rangle$ as that in Definition 4. Then the FSA augmented MDP is defined as $\mathcal{M}_\phi = \langle N, \hat{S}, \{A_i\}_{i \in N}, \hat{p}(\hat{s}'|\hat{s}, a), \{\hat{r}_i\}_{i \in N}, \mathcal{F}_\phi \rangle$, where $\hat{S} = S \times \mathbb{Q}_\phi$. Moreover, as the transition probability from $\hat{s}$ to $\hat{s}'$ with action a taken, $\hat{p}(\hat{s}'|\hat{s}, a)$ is defined,

$$\hat{p}(\hat{s}'|\hat{s}, a) \ = p((s', q')|(s, q), a) \tag{3.9}$$

$$= \begin{cases} p(s'|s,a) & \text{if } p_\phi(q'|q,s)=1 \\ 0 & \text{otherwise,} \end{cases}$$

where $p_\phi(q'|q,s)$ is defined in (2.7) and $\hat{s}=(s,q)$. Moreover, $\hat{r}_i : \hat{S}\times\hat{S}\to\mathbb{R}$ is the FSA augmented reward for agent i. For a fully cooperative multi-agent system, it is set that $\hat{r}_g=\hat{r}_1=\dots\hat{r}_n$ and $\hat{r}_g$ is defined as

$$\hat{r}_g(\times_{i=1}^n \hat{s}_i, \times_{i=1}^n \hat{s}_i') = \rho(\times_{i=1}^n \hat{s}_i', D_\phi^q) > 0, \tag{3.10}$$

where $D_\phi^q = \bigvee_{q'\in\Omega_q} \psi_{q,q'}$ represents the disjunction of all predicates guarding the transitions that originates from q and are the (non-trap) automata states connected with q through outgoing edges.

The reward function in (3.10) can effectively generate intrinsic rewards, which coincides well with the goal of the original Problem 5. It is worth noting that $\hat{r}_g(\times_{i=1}^n \hat{s}, \times_{i=1}^n \hat{s}')$ get all the states of the agents involved and thus serves as a global reward. However, in the system where the agents are not fully cooperative or the cases where not new states from all agents are available due to delay, the agent might calculate its own local reward. Consequently, such local reward $\hat{r}_{j,l}$, which only depends on its own state update, is defined as

$$\hat{r}_{j,l}(\hat{s}_j \times_{i=1,i\neq j}^n \hat{s}_i', \times_{i=1}^n \hat{s}_i') = \rho(\times_{i=1}^n \hat{s}_i', D_\phi^q) > 0, \tag{3.11}$$

with D_ϕ^q defined identically as that in (3.10). With the reward defined in (3.10) and (3.11), the learning process will drive the multi-agent system out of the current automaton state into the next one by maximizing the expected return. Eventually, the system will arrive at the final state $q_f \in \mathcal{F}_\phi$, which indicates the satisfaction of theϕ and thus solves Problem 5. *Note that at if state $s_{0:T}$ leads the automata to transition from q_0 to q_f, then $\rho(s_{0:T},\phi)>0$ which motivates* (3.8) *and* (3.11). Additionally, the FSA augmented MDP in Definition 6 can be built with any standard MDP and an scTLTL and thus can be applied to a wide range of applications expressible by scTLTL.

IV. FSA Guided Multi-Agent Reinforcement Learning

In this section, we will develop our method based on the tabular Q-learning [17], however the ideas presented here are extensible to the continuous case. We start with the centralized Q-learning as a benchmark.

A. Centralized Q-Learning

In the centralized setting, slight modifications are made to the original definition of the FSA augmented MDP. Particularly, $\mathcal{M}_\phi = \langle N, \hat{S}, \{A_i\}_{i\in N}, \hat{p}(\hat{s}'|\hat{s},a), \hat{r}, \mathcal{F}_\phi\rangle$, with $\hat{S}=\times_{i=1}^n S_i\times\mathbb{Q}_\phi$, $\hat{s}=\times_{i=1}^n s_i\times q_\phi$, $a=\times_{i=1}^n a_i, a_i\in A_i$ and $\hat{r}=\hat{r}_1=\dots\hat{r}_n$. Then given (2.1), the goal of the centralized Q-learning is to maximize the following expected sum of discounted return

$$\pi_\phi^* = \arg\min_{\pi_\phi} \mathbb{E}^{\pi_\phi}\Big[\sum_{k=0}^{T-1}\gamma^t \hat{r}(\hat{s}_k,\hat{s}_{k+1})\Big], \tag{4.12}$$

with the reward function $\hat{r}(\hat{s}_k,\hat{s}_{k+1})$ defined in (3.10). Standard Q-learning update is applied to the agent as

$$\begin{aligned} Q(\hat{s}_k,a_k) &= Q(\hat{s}_k,a_k) \\ &\quad +\alpha[\hat{r}_{k+1}+\gamma\max_{a'} Q(\hat{s}_{k+1},a') - Q(\hat{s}_k,a_k)]. \end{aligned} \tag{4.13}$$

In this case, there is only one *pseudo agent*, whose state/action is the joint state/action of all of the n agents. The environment is therefore stationary and will no longer encounter possible divergence [17]. However, this centralized Q-learning inevitably has the issue of "curse of dimensionality", as the dimension of the joint state/action space is increasing exponentially with the agent number n.

B. Semi-Decentralized Q-Learning

The curse of dimensionality encountered by the centralized Q-learning motivates our development of a semi-decentralized version of Algorithm 1. Instead of learning jointly as an integrated *pseudo agent* with one Q table, in the semi-decentralized setting, every agent has its own learning process. For agent i, the FSA augmented MDP is $\mathcal{M}_{i,\phi} = \langle \hat{S}_i, A_i, \hat{p}(\hat{s}_i'|\hat{s}_i,a_i), \hat{r}_i, \mathcal{F}_\phi\rangle$, with $\hat{S}_i = S_i\times\mathbb{Q}_\phi$, $\hat{s}_i = s_i\times q_\phi$, $a_i\in A_i$. For each agent, we add the FSA state as an extra dimension in the state space. As discussed before, such FSA state possesses the global information and serves to track the process of the multi-agent system in satisfying the scTLTL task specification. In other words, it acts as a global coordinator and thus the semi-decentralized Q-learning here belongs to the category of direct coordination methods [18]. The interaction between the coordinator and the agents are illustrated in Fig. 2. Agent i provides its own current state $\hat{s}_i=(s_i,q)$, then the coordinator will evaluate the next state s_i' as well as the FSA state q' based on (2.6) or (2.7). The communication graph here is a time-invariant star graph as shown in Fig. 2. Note that synchronous learning is implemented here such that all of the agents in the system will share the identical FSA state. The global and local reward $\hat{r}_g$ and $\hat{r}_{i,l}$ are evaluated according to (3.10) and (3.11), respectively. The reward for the ith agent $\hat{r}_i$ is set as the weighted sum of the global and local rewards as follows

$$\hat{r}_i = w_i\hat{r}_g + (1-w_i)\hat{r}_{i,l}, \tag{4.14}$$

where $w_i\in(0,1]$ as the weight coefficient for agent i. It is worth noting that with $w_i=1$, the system is fully cooperative, otherwise the agents are somewhat selfish as they take their own reward into consideration. In the extreme case where $w_i=0,\forall i\in N$, the agents are greedily maximizing only its own return and ignoring that of the whole system. The coordination in that case is invalid and thus divergence can be expected.

With the FSA augmented MDP for each agent defined, the agent will try to maximize its own expected sum of return as

$$\pi_{i,\phi}^* = \arg\min_{\pi_{i,\phi}} \mathbb{E}^{\pi_{i,\phi}}\Big[\sum_{k=0}^{T-1}\gamma\hat{r}_i(\hat{s}_k,\hat{s}_{k+1})\Big]. \tag{4.15}$$

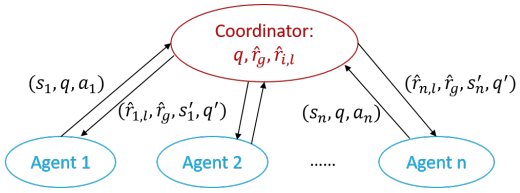


Fig. 2: Illustration of the interaction between the coordinator and the agents.

Consequently, the Q-function update for each agent is

$$Q_i(\hat{s}_{i,k}, a_{i,k}) = Q_i(\hat{s}_{i,k}, a_{i,k}) + \alpha[\hat{r}_{i,k+1} + \gamma \max_{a'} Q_i(\hat{s}_{i,k+1}, a') - Q_i(\hat{s}_{i,k}, a_{i,k})]. \tag{4.16}$$

The semi-decentralized Q-learning on multi-agent FSA augmented MDP is summarized in Algorithm 1. Due to the semi-decentralized nature of Algorithm 1, the size of the Q table is linearly increasing with the agent number n, and thus significantly reducing the memory and computing burden.

Algorithm 1 Semi-Decentralized Q-Learning on Multi-Agent FSA Augmented MDP

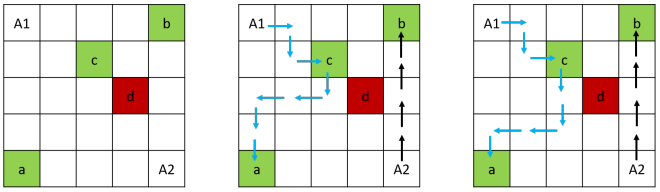
Inputs: scTLTL task specification ϕ, $\hat{S}_i, A_i, \hat{p}(\hat{s}_i'|\hat{s}_i, a_i), \hat{r}_i, \mathcal{F}_\phi$, learning rate $\alpha \in (0, 1]$, discount factor $\gamma \in (0, 1)$, training episode n_E and maximum training iterations in each episode n_s.
Initialize: Initial states $\hat{s}_i$, randomly initialized Q-function Q_i
Outputs: Q-function and optimal policy π_ϕ^*
Construct the FSA augmented MDP $\mathcal{M}_{i,\phi}$ for each agent.
for $k_E = 1 \dots n_e$ **do**
  for $k = 1 \dots n_s$ **do**
    i) Take action for each agent as $a_{i,k} = \arg\min_{a'} Q_i(\hat{s}_{i,k}, a')$. ii) Get new state $\hat{s}_{i,k+1}$ through $\hat{p}(\hat{s}_i'|\hat{s}_i, a_i)$. iii) Evaluate Global and local reward $\hat{r}_g$ and $\hat{r}_{i,l}$ are evaluate according to (3.10) and (3.11), and then get $\hat{r}_i$ based on (4.14).
    for each agent $i \in N$ **do**
      Do the Q-function update

$$Q_i(\hat{s}_{i,k}, a_{i,k}) = Q_i(\hat{s}_{i,k}, a_{i,k}) + \alpha[\hat{r}_{i,k+1} + \gamma \max_{a'} Q_i(\hat{s}_{i,k+1}, a') - Q_i(\hat{s}_{i,k}, a_{i,k})]. \tag{4.17}$$

    end for
    if $q_{k+1} \in \mathcal{F}_\phi$ **then**
      break
    end if
  end for
end for
With trained optimal Q-function Q_i^*, get optimal greedy policy based on $\pi_{i,\phi}^*(\hat{s}) = \arg\min_{a'} Q_i^*(\hat{s}, a')$.

V. Experiments

In this section, We apply our semi-decentralized Q-learning algorithm to satisfy a scTLTL task specification.



(a) (b) (c)

Fig. 3: (a): Finite environment as the agent workspace. a, b and c are sites of interest while d is the obstacle to avoid. A1 and A2 represent the initial locations of agent one and agent two, respectively. (b) (c): State trajectory of the two agents from centralized Q-learning and semi-decentralized Q-learning (Algorithm 1) to satisfy scTLTL (5.18), respectively.

The environment of the agent work space is modelled as finite grid, illustrated in Fig. 3a. Such abstraction, while conservative, works for some robot dynamics [20]. In this environment, states a, b, c are labelled as goals while d is labelled as the obstacle. Consider the scTLTL formulae

$$p := \Diamond a \wedge \Diamond b \wedge \Diamond c, \tag{5.18}$$

which states "eventually visit a, b and c". We have a team of agents $N = \{1, 2\}$, which shall work cooperatively to satisfy ϕ . The agents in N are heterogeneous, which means that they have different capacities over the sub-tasks specified in (5.18). T_i is defined to denote the set of sub-tasks that agent i can do. In this case, it is set that $T_1 = \{a, c\}$ and $T_2 = \{b, c\}$. Instead of manually assigning sub-teams of agents to accomplish the sub-tasks, we will let the learning process itself to manage that in an optimal way. Fig. 3b and 3c demonstrate the state trajectory of the two agents in satisfying the given scTLTL task specification. The automaton trajectory is: $\left\{q_0 \xrightarrow[c]{\text{A1}} q_3 \xrightarrow[b]{\text{A2}} q_6 \xrightarrow[a]{\text{A1}} q_f\right\}$ (transition labels show each sub-task accomplished by the corresponding agent). As presented there, both the centralized and semi-decentralized settings are able to learn the optimal solution. However, the scale of the multi-agent learning problem is significantly reduced in the semi-decentralized algorithm. In centralized learning, the dimension of the state-action space is $(|\mathbb{Q}_\phi|\Pi_{i=1}^n|S_i||A_i|)$, while that for the semi-decentralized learning is $|\mathbb{Q}_\phi|\sum_{i=1}^n|S_i||A_i|$, with $|\bullet|$ as the cardinality operator of a set. More specifically in this case study, the centralized learning has a state-action space of dimension $80,000$, while that of the semi-decentralized learning is $1,600$. Such scale reduction not only saves the computation and memory burden but also speeds up the convergence. In addition, the episodic reward for centralized learning and Algorithm 1 are presented in Fig. 5 and 6. Results show that both algorithms converge and the semi-decentralized version uses about a quarter of the experience necessary for that of the centralized version.

VI. Conclusions

In this paper, the multi-agent Reinforcement Learning (RL) technique is applied to the problem of deploying a multi-robot team to satisfy a syntactically co-safe Truncated

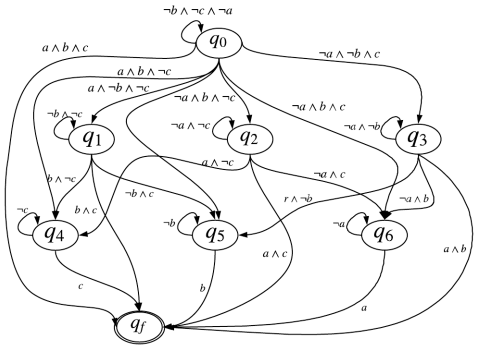


Fig. 4: The FSA of scTLTL formula $p := \Diamond a \wedge \Diamond b \wedge \Diamond c$.

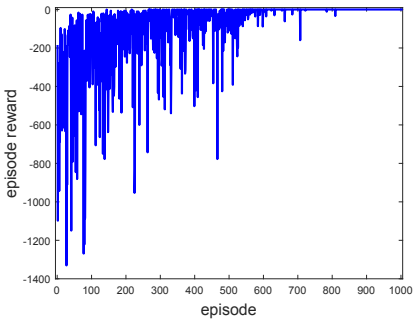


Fig. 5:

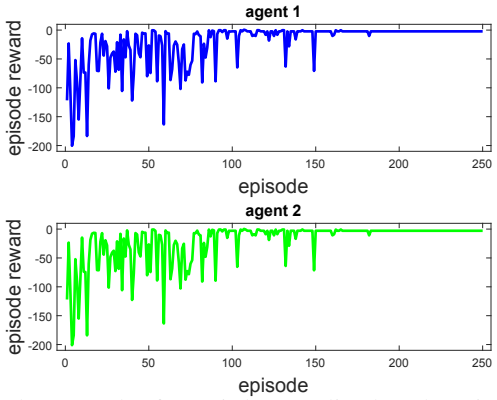


Fig. 6: Episode reward of semi-decentralized Q-learning (Algorithm 1). In order to make the agents work highly cooperatively, the weight of global reward w_i is chosen as a large number in (4.14), so the episode rewards of the two agents look close but still distinct.

Linear Temporal Logic (scTLTL) task specification. One of the advantages of multi-agent RL is that the difficult task assignment problem on heterogeneous agents is integrated as part of the learning problem. Without manual manipulation on task assignment, it is more promising to enhancing optimality. A continuous metric called robustness degree is used to reward progress towards satisfying the scTLTL formulae. Moreover, FSA is introduced to coordinate multi-agent RL to promote convergence and act as a tracker to track the process of satisfying the scTLTL. Subsequently, an FSA augmented MDP is constructed for each agent, which share the FSA state carrying the global information. Then over the FSA augmented MDP, a semi-decentralized Q-learning algorithm is proposed to maximize the return. We apply our algorithms on certain cases and the results demonstrate the effectiveness of the semi-decentralized multi-agent Q-learning with reduced complexity.

REFERENCES

[1] W. Burgard, M. Moors, C. Stachniss, and F. E. Schneider, "Coordinated multi-robot exploration," *IEEE Transactions on robotics*, vol. 21, no. 3, pp. 376–386, 2005.

[2] I. Rekleitis, V. Lee-Shue, A. P. New, and H. Choset, "Limited communication, multi-robot team based coverage," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, vol. 4. IEEE, 2004, pp. 3462–3468.

[3] S. Andersson, A. Nikou, and D. V. Dimarogonas, "Control synthesis for multi-agent systems under metric interval temporal logic specifications," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 2397–2402, 2017.

[4] Z. Serlin, K. Leahy, R. Tronl, and C. Beita, "Distributed sensing subject to temporal logic constraints," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4862–4868.

[5] K. Leahy, A. Jones, M. Schwager, and C. Belta, "Distributed information gathering policies under temporal logic constraints," in *2015 54th IEEE Conference on Decision and Control (CDC)*. IEEE, 2015, pp. 6803–6808.

[6] A. Jones, M. Schwager, and C. Belta, "Information-guided persistent monitoring under temporal logic constraints," in *2015 American Control Conference (ACC)*. IEEE, 2015, pp. 1911–1916.

[7] J. McMahon and E. Plaku, "Robot motion planning with task specifications via regular languages," *Robotica*, vol. 35, no. 1, pp. 26–49, 2017.

[8] E. Plaku, L. E. Kavraki, and M. Y. Vardi, "Hybrid systems: from verification to falsification by combining motion planning and discrete search," *formal methods*, vol. 34, no. 2, pp. 157–182, 2009.

[9] X. Li, C.-I. Vasile, and C. Belta, "Reinforcement learning with temporal logic rewards," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 3834–3839.

[10] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *neural information processing systems*, pp. 6379–6390, 2017.

[11] M. L. Littman, "Value-function reinforcement learning in markov games," *Cognitive Systems Research*, vol. 2, no. 1, pp. 55–66, 2001.

[12] C. Guestrin, M. G. Lagoudakis, and R. Parr, "Coordinated reinforcement learning," in *ICML '02 Proceedings of the Nineteenth International Conference on Machine Learning*, 2002, pp. 227–234.

[13] F. Fischer, M. Rovatsos, and G. Weiss, "Hierarchical reinforcement learning in communication-mediated multiagent coordination," in *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, 2004. AAMAS 2004.*, 2004, pp. 1334–1335.

[14] C. Claus, C.and Boutilier, "The dynamics of reinforcement learning in cooperative multiagent systems," in *Conference on Artificial Intelligence and 10th Conference on Innovative Applications of Artificial Intelligence (AAAI/IAAI-98)*, 1998, p. 746752.

[15] S. Kapetanakis and D. Kudenko, "Reinforcement learning of coordination in cooperative multi-agent systems," in *Eighteenth national conference on Artificial intelligence*, 2002, pp. 326–331.

[16] X. Li, Y. Ma, and C. Belta, "Automata guided reinforcement learning with demonstrations," *arXiv preprint arXiv:1809.06305*, 2018.

[17] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[18] L. Busoniu, R. Babuska, and B. De Schutter, "Multi-agent reinforcement learning: A survey," in *2006 9th International Conference on Control, Automation, Robotics and Vision*. IEEE, 2006, pp. 1–6.

[19] C.-I. Vasile, "Github repository," 2017.

[20] K. Leahy, D. Zhou, C.-I. Vasile, K. Oikonomopoulos, M. Schwager, and C. Belta, "Provably correct persistent surveillance for unmanned aerial vehicles subject to charging constraints," in *Experimental Robotics*. Springer, 2016, pp. 605–619.