Keeping in Touch with Collaborative UAVs: A Deep Reinforcement Learning Approach

Bo Yang, Min Liu
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
yangboac@gmail.com, liumin@ict.ac.cn

Abstract

Effective collaborations among autonomous unmanned aerial vehicles (UAVs) rely on timely information sharing. However, the time-varying flight environment and the intermittent link connectivity pose great challenges to message delivery. In this paper, we leverage the deep reinforcement learning (DRL) technique to address the UAVs' optimal links discovery and selection problem in uncertain environments. As the multi-agent learning efficiency is constrained by the high-dimensional and continuous action spaces, we slice the whole action spaces into a number of tractable fractions to achieve efficient convergences of optimal policies in continuous domains. Moreover, for the nonstationarity issue that particularly challenges the multi-agent DRL with local perceptions, we present a multi-agent mutual sampling method that jointly interacts the intra-agent and inter-agent state-action information to stabilize and expedite the training procedure. We evaluate the proposed algorithm on the UAVs' continuous network connection task. Results show that the associated UAVs can quickly select the optimal connected links, which facilitate the UAVs' teamwork significantly.

1 Introduction

Small unmanned aerial vehicles (UAVs) have an edge in ubiquitous-deployment, autonomous-mobility and flexible-maneuverability, which can serve our daily life in an intelligent and convenient manner. Moreover, the unwearied UAVs can also replace human labors to fulfill challenging tasks in remote, unsafe and stricken environments. The increasing popularization of UAVs has boosted numerous applications such as goods delivery, environment surveillance, disaster relief assistance and terrorist defence. However, the capabilities of coverage, sensing and execution as well as the budgets of payload and energy of one single UAV are very limited. Thus, it is advocated for multiple UAVs to fulfill a complex task collaboratively [Wang *et al.*, 2017].

The multi-UAV collaboration involves the necessary steps of connectivity maintenance and information sharing among distributed teammates in an uncertain or even unknown environment [Barrett *et al.*, 2017]. Due to the restriction of local perceptions and the lack of *a priori* knowledge about the potential teammates, it is difficult to learn about the existence of the counterparts in the free flight space. The varying space locations and the intermittent link connectivity also hinder UAVs to share the reactions to the environment changes. A feasible solution to get connected with teammates is to broadcast announcement messages over qualified channels and set up links by trial and error. However, dispersed UAVs share unpredictable common channels and the channel availability in the air fluctuates frequently over time, which further challenges the multi-UAV collaboration issue.

As reinforcement learning (RL) [Sutton and Barto, 1998] emerges as a viable and elegant approach to yield an optimized policy for sequential decision-making tasks, the uncertain and time-varying channel availability and the resultant connectivity decision is suitable to be tackled by RL. RL-based agents (i.e., UAVs[1]) gain experiences by exploration, and gradually converge to the optimal policy after repetitive interactions with the environment. However, the standard RL approach suffers from a high computational complexity due to the large-scale state-action spaces. To adapt to the environment variations quickly, the non-linear and parameterized Deep Neural Network (DNN) has become a dominant learning technique. The DNNs have several incomparable benefits such as compact and powerful encoding of the gained experiences, efficient training of the high-dimensional data samples, fine-grained approximation of the multi-scale objective functions, etc. [Sharma *et al.*, 2017]. Thus, we investigate the UAVs' optimal link discovery and selection problem under the deep reinforcement learning (DRL) framework.

Multi-agent DRL has the potential to execute the joint task remarkably, but also brings about particular challenges. It naturally incurs high-dimensional and large-scale state-action spaces to decide on connected links over multiple candidate channels in a continuous period of time, making the optimal policy convergence intractable and inefficient. Besides, each agent makes independent policies based on local perceptions and neglects the time-varying surroundings, making the learning process isolated and nonstationary [Ye *et al.*, 2017].

In this paper, we present an efficient multi-agent DRL al-

[1]Hereafter we use the terms *UAV* and *agent* interchangeably.

gorithm to decide the optimal communication link for UAVs' continuous connectivity in uncertain environments. To improve the computational efficiency, we slice the holistic high-dimensional action spaces into a number of tractable fractions. The "divide-and-conquer" strategy scales well to the massively increasing data over time. Moreover, to facilitate collaborations among autonomous UAVs, we investigate the multi-agent mutual sampling and joint learning method. The cross-network interactions are beneficial to stabilize and expedite the optimal policies learning. We evaluate the proposed Fractional Slicing & Mutual Sampling (FSMS) DRL algorithm on the UAVs' continuous network connectivity task. Results reveal that our algorithm enables UAVs to quickly discover potential links towards teammates.

2 Related Work

We review the literature on *single-agent* and *multi-agent* DRL algorithms.

2.1 Single-Agent Deep Reinforcement Learning

Deep Q-network (DQN) [Mnih *et al.*, 2015] is a representative work that iteratively approximates the state-action value (i.e., Q-value) function with a parameterized DNN. The core ingredients of *experience replay* and *target network* can improve the computational efficiency. However, the deficiency is that DQN overestimates the action values in practice. To this end, [Hasselt *et al.*, 2016] propose a Double Q-learning algorithm to achieve the objectives of large-scale function approximation and stable learning. [Zhang *et al.*, 2017] further propose a weighted double Q-learning algorithm to improve the underestimation of action values by heuristically selecting the weights from empirical results. [Anschel *et al.*, 2017] extend the DQN algorithm by averaging over the afore learned Q-values, which can reduce the target approximation errors and stabilize the training process. [Wang *et al.*, 2016] introduce a dueling network architecture that decouples the state value function and the action advantage function in DQN, which can figure out the appropriate action quickly during policy evaluation when similar actions are mixed. [Kulkarni *et al.*, 2016] propose a hierarchical DRL framework, in which the top-level picks a policy over intrinsic goals and the bottom-level learns a policy over atomic actions to fulfill the chosen goals. However, the afore mentioned works concern on the discrete learning process with a limited number of action spaces. Accordingly, [Gu *et al.*, 2016] present a normalized advantage function to simplify the sampling complexity and accelerate the model-free continuous Q-learning. [Mnih *et al.*, 2016] execute agents asynchronously on continuous tasks and the shared model among the parallel actor-learners leads to a stabilized learning process. [Xu *et al.*, 2018] propose a traffic-aware method to train an inexperienced DRL agent to gain exploration and make a rational policy in the continuous learning scenario with massive action spaces.

2.2 Multi-Agent Deep Reinforcement Learning

The general DRL algorithms deal with a certain number of agents in a coexisting environment, which should learn to maximize each individual's benefit while minimizing the influences on others. [Tampuu *et al.*, 2017] investigate how multiple agents learn and make different policies under both collaborative and competitive tasks. [Foerster *et al.*, 2016] tackle the multi-agent communication issue with partial observability in the decentralized execution but centralized learning setting. [He *et al.*, 2016] model the implicit representation of the opponent's strategy according to past observations and then derive an adaptive response that better optimizes available rewards. [Chen *et al.*, 2017] address the multi-agent cooperative collision avoidance problem by characterizing the stochastic behaviors as the compliance/violation of social norms. [Leibo *et al.*, 2017] propose a sequential social dilemma (SSD) model to capture the structure of cooperation and defection when independent agents learn dynamic policies by using Markov games. [Omidshafiei *et al.*, 2017] introduce a decentralized and stable experience replay method, which is combined with a generalized recurrent multi-task network to achieve multi-agent coordination.

Due to the powerful learning and decision capability, the DRL technique has been widely applied to the community of autonomous vehicle networks. For examples, [Liu *et al.*, 2017] avoid the traffic jam problem by utilizing cooperative Q-values to balance the traffic flows among adjacent intersections. [Chinchali *et al.*, 2018] tackle the traffic scheduling problem in IoTs in order to adapt to traffic variations dynamically and increase the network utilization significantly. [Wu *et al.*, 2017] implement a DRL-based UAV sensing system, which can increase the airborne broadcast dissemination rate and reduce the data loss rate adaptively. [Xiao *et al.*, 2018] accelerate the game between the UAV transmission and the smart attacks by removing the high dimension bottleneck of state spaces. [Zhang *et al.*, 2018] control energy-limited UAVs to sample the most required data in the target zone, and also to schedule the ground unmanned charging vehicles to the UAV refueling point with the shortest time.

Although the essential DRL algorithms have various extensions and widespread applications, there have been relatively few investigations on efficient DRL solutions for continuous and stationary multi-agent collaborations, particularly for the durative UAV communications in uncertain environments. Thus, we aim at designing an effective DRL algorithm to address the UAVs' continuous network connectivity problem.

3 Background

In this section, we introduce the relevant background on RL and particularly on the deep Q-learning algorithm.

3.1 Reinforcement Learning

RL focuses on how the agents take proper actions in a specific or unknown environment so as to maximize (*resp.* minimize) the cumulative objective function of the long-term reward (*resp.* cost). Agents need to interact with the environment through direct experience by a sequential learning and decision-making process. At each time step t ($t = 0, 1, \ldots$), one agent observes one possible environment state $s^{(t)} \in \mathcal{S}$ and takes a potential optimal action $a^{(t)} \in \mathcal{A}$, which leads to an immediate optimized reward $r^{(t)} \in \mathcal{R}$. The environment state is transited to the next step $s' = s^{(t+1)} \in \mathcal{S}$ by following a probability $p(s'|s^{(t)}, a^{(t)})$. We consider the infinite horizon with a discounted cumulative reward $R^{(t)} = \sum_{t'=t}^{+\infty} \gamma^{t'-t} r^{(t')}$, where $\gamma \in (0,1)$ denotes the discounted factor. Given the state set $\mathcal{S}$ and the action set $\mathcal{A}$, a policy corresponds to a function π that guides the agent towards reward maximization. Particularly, each agent aims to figure out the optimal policy $\pi^* : \mathcal{S} \to \mathcal{A}$ that harvests the maximized $\mathbb{E}[R^{(t)}]$, where $\mathbb{E}[\cdot]$ denotes the expectation operation.

We are interested in the value-based RL methods that encode policies by applying the action-value function (i.e., Q-function), which is given as

$$Q^{\pi}(s,a) = \mathbb{E}^{\pi}\left[\sum_{t=0}^{+\infty} \gamma^t r^{(t)} | s^{(t)} = s, a^{(t)} = a\right]. \quad (1)$$

Accordingly, the optimal Q-function is $Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a)$ and the optimal policy can be derived as $\pi^* = \arg\max_a Q^*(s,a)$.

The above Q-function associated with the optimal policy π^* can be solved by the following value-iteration-based Q-learning algorithm [Sutton and Barto, 1998]:

$$\begin{aligned} Q(s^{(t)}, a^{(t)}) \leftarrow\ & Q(s^{(t)}, a^{(t)}) + \\ & \eta(r^{(t+1)} + \gamma \max_a Q^*(s^{(t+1)}, a) - Q(s^{(t)}, a^{(t)})), \end{aligned} \quad (2)$$

where η denotes the learning rate.

However, the number of training samples required for convergence in the above Q-learning algorithm grows immeasurably with the number of states. Thus, the standard Q-learning suffers from the curse of dimensionality as well as the low computational efficiency. As a countermeasure, the function approximation technique makes sense, which is explained in the next subsection.

3.2 Deep Q-Networks

The approximate non-linear Q-function is denoted as a neural network parameterized with θ (i.e., $Q(s,a) \approx Q(s,a;\theta)$), which is also referred to as a Q-network. The transitions of states, actions and rewards yielded from each step are explored and stored in memory $\mathcal{D}$ so that agents can *replay* these *experiences* later in diverse sorts to break the temporal correlation caused by the traditional sequential training. At the i-th training step, the data samples are randomly selected from $\mathcal{D}$ and are packed into a batch. The training objective at the i-th step is to update $\theta^{(i)}$, which minimizes the expected mean-square error (i.e., the loss function) of the samples within the batch operating the Bellman equation (2). The loss function is given as

$$L^{(i)}(\theta^{(i)}) = \mathbb{E}[(r^{(i)} + \gamma \max_{a'} Q(s', a'; \bar{\theta}^{(i)}) - Q(s, a; \theta^{(i)}))^2], \quad (3)$$

where the parameter $\bar{\theta}^{(i)}$ depends on the previous update and is utilized to approximate the optimal target value.

The updates are made at each training step by using the following gradients:

$$\begin{aligned} \nabla_{\theta^{(i)}} L^{(i)}(\theta^{(i)}) =& \mathbb{E}[2(r^{(i)} + \gamma \max_{a'} Q(s', a'; \bar{\theta}^{(i)}) - \\ & Q(s, a; \theta^{(i)}))\nabla_{\theta^{(i)}} Q(s, a; \theta^{(i)})]. \end{aligned} \quad (4)$$

During the process of minimizing the loss function, the cost of back-propagating over all the training samples is undesirable. As a result, the stochastic gradient descent (SGD) method, which only samples a subset of the candidate batch, is utilized for an improved computational efficiency.

4 DRL-based Optimal Link Selection

The link selection problem concerns how to decide the best link to get connected to another UAV. In this section, we present the DRL-based algorithm for optimal links decision.

4.1 Model Framework

We consider a multi-UAV coexisting environment, in which a set $\mathcal{U}$ of autonomous UAVs fulfill a collaborative task. From the perspective of DRL, each UAV $u \in \mathcal{U}$ is modeled as an agent, which has the capability of perceiving the local available channels and attempts to select a connected link over a common channel shared by another agent $v \in \mathcal{U}$ continuously. Due to the variations in space locations, channel quality, perception capabilities, etc., different agents tend to observe different local available channels. For any $u \in \mathcal{U}$, let $\mathcal{C}_u$ denote the set of its local channels. To ensure the connectivity, we assume the time-varying local channels between any pair of agents overlap (i.e., $\forall u, v \in \mathcal{U}, \mathcal{C}_u \cap \mathcal{C}_v \neq \emptyset$). If adjacent agents propagate announcement messages on the same channel simultaneously, a connected link can be built. Armed with no knowledge of the flight environment and no awareness of potential teammates, each agent explores to discover links and accumulates experiences by iteratively interacting with the surroundings. We define the necessary elements in the DRL framework as follows.

- *States.* At each time instant, an agent u propagating an announcement message corresponds to a state $s_u \in \mathcal{S}$, which is defined as whether the message is delivered successfully or not. Let $\mathcal{S} = \{succ, fail\}$ denote the state set. If $s_u = \{fail\}$, agent u obviously needs to discover another available link at the next time instant. Whereas if $s_u = \{succ\}$, agent u still needs to check whether the previous link is continuously connected or not at the next time instant.

- *Actions.* Actions indicate the decisions on which channel to propagate the messages. Let $\mathcal{T}_u$ denote the set of teammates that agent u may reach. For any $u \in \mathcal{U}, s_u \in \mathcal{S}$, the set of potential actions is denoted as

$$\mathcal{A}_u^{\mathcal{C}_u}(s_u) = \{a_u = (v, ch_u) | v \in \mathcal{T}_u, ch_u \in \mathcal{C}_u\},$$

where $a_u = (v, ch_u)$ denotes the action of propagating a message to teammate v over channel ch_u, and we use $\mathcal{A}_u$ instead of $\mathcal{A}_u^{\mathcal{C}_u}(s_u)$ for brevity.

- *Rewards.* As the channel availability fluctuates frequently in the air, we let $p_u^{(t)}(ch_u)$ denote the probability that channel ch_u is available for agent u at time t. The decision-making depends on the channel quality, thus, the reward r_u received by agent u is calculated as the number of messages that are propagated successfully over the most qualified common channel.

To make optimal link decisions, every time a message m is ready to be propagated, agent $u \in \mathcal{U}$ execute the DRL-based algorithm to learn the Q-function. According to the calculated action-value, agent u decides the best propagating action $a_u = (v, ch_u)$ for m (i.e., over the best channel ch_u to reach teammate v).

4.2 Learning to Get Connected

The essentials of the proposed FSMS algorithm are two-fold: *fractional slicing* and *mutual sampling*.

(i) *Fractional slicing*. The UAVs' optimal links discovery and selection in uncertain and time-varying environments directly lead to high-dimensional and continuous action spaces. To balance the trade-off between representability and tractability, we slice the whole action spaces into a number of non-overlapped fractions while reserving a common coordinator, which deals with the functions of division and aggregation for all the fractions.

For agent u and its complex action set $\mathcal{A}_u$ with F dimensions, let $\mathcal{A}_{u,f}$ $(f \in \{1, \ldots, F\})$ denote the f-th dimension of $\mathcal{A}_u$. Each fraction $\mathcal{A}_{u,f}$ includes a finite number of subactions $a_{u,f}$. The agent u trains all the fractions independently and their respective Q-values (i.e., $Q_{u,f}(s_u, a_{u,f})$) are aggregated by the coordinator. As the initial division is estimated roughly and the output weights are usually biased, the training process is confronted with the instability barrier. We thus compute the F-fold average Q-value as the result, i.e., $Q_u(s_u, a_u; \theta_u) = \frac{1}{F}\sum_{f=1}^{F} Q_{u,f}(s_u, a_{u,f}; \theta_u)$. In accordance with the Q-value updates, the DQN target value is defined with an average operation: $y_u = r_u + \gamma \cdot \frac{1}{F}\sum_f \bar{Q}_{u,f}(s'_u, \operatorname*{arg\,max}_{a'_{u,f} \in \mathcal{A}_{u,f}} Q_{u,f}(s'_u, a'_{u,f}; \theta_u))$ ($\bar{Q}_{u,f}$ denotes the f-th fraction of the target DQN $\bar{Q}_u$). Analogously, the loss function is the expectation value of the averaged y_u across all fractions: $L_u(\theta_u) = \mathbb{E}_{(s_u, a_u, r_u, s'_u)\sim\mathcal{D}}[\frac{1}{F}\sum_f (y_{u,f} - Q_{u,f}(s_u, a_{u,f}; \theta_u))^2]$.

(ii) *Mutual sampling*. From the perspective of one agent, the environment is nonstationary in the sense that the agents only own local perceptions and learn the separate polices all by themselves. This phenomenon shields off the macro-perspective and slows down the convergence efficiency. To this end, we focus on the multi-agent mutual sampling and joint learning, i.e., each agent not only samples the correlative state-action information by itself but also from other peers.

In particular, we add a mutual learning parameter θ_u^{mutual}, which is distinguished from the original independent learning parameter θ_u^{own}, for each agent $u \in \mathcal{U}$ in the DQN. As a result, the training data is not only passed to all action fractions of one agent, but also passed to the associated cross-network fractions from other agents. Let $Q_u\left(s_u, a_u; \theta_u^{own}, \theta_u^{mutual}\right)$

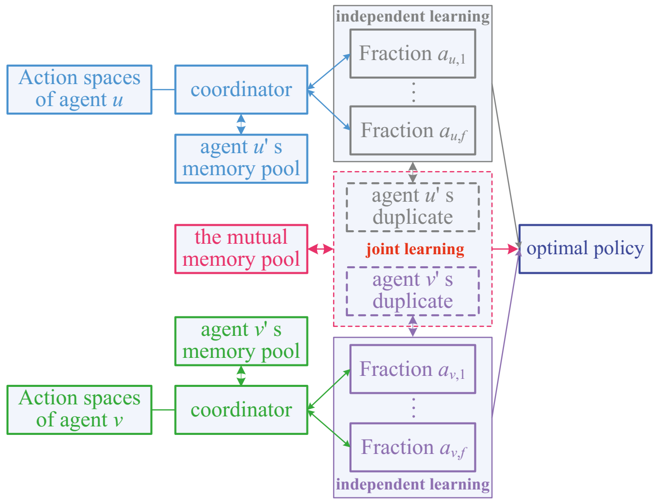


Figure 1: FSMS architecture

denote the deep Q-function of agent u. Correspondingly, the gradient training on θ_u^{own} and θ_u^{mutual} is conducted on the joint loss function $L_u(\theta_u^{own}, \theta_u^{mutual})$.

Then a problem arises, that one agent's own training data and the multi-agent mutual training data are mixed, making the passing of target information confusing. We thus utilize the duplication separation technique. In other words, each agent has a duplicate, which shares the same parameters as the original agent. These two parts can communicate with each other without any difficulty and are trained by performing the same algorithm. However, the underlying experience replay pools of these two parts are different, i.e., one part stores agent u's own experience $\mathcal{D}_u$ while the other part stores the interacting agent v's mutual experience $\mathcal{D}_{v\to u}(v \in \mathcal{T}(u))$. An improved convergence speed can be expected under the separation technique. For example, the cross-network training is only conducted on the reduced data set from $\mathcal{D}_{u\cap v}$ instead of the redundant data set from $\mathcal{D}_u$ or $\mathcal{D}_v$.

In Fig.1, we take the case of two agents to illustrate the FSMS algorithm architecture.

We formally present the FSMS algorithm in Algorithm 1, which takes the current state as well as the local available channels as inputs. When agent u needs to get connected to another teammate, agent u updates the state-action model and calculates the new Q-function (lines 3$\sim$5), where the Q-value is averaged across all fractions in its own action spaces and all associated cross-network fractions from the interacting agent v. To balance between exploration (gain of knowledge) and exploitation (usage of knowledge), we utilize the ϵ-greedy ($\epsilon \in (0, 1]$) strategy (lines 6$\sim$12). Specifically, agent u explores a random action with probability ϵ or exploits existing knowledge to select the optimal channel to propagate messages. According to the update results, agent u selects a proper action. After executing the current action $a_u^{(t)}$ and yielding the corresponding reward $r_u^{(t)}$, we keep track of agent u's state transition information in both its own memory $\mathcal{D}_u$ and the mutual memory $\mathcal{D}_{v\to u}$ with teammate v (lines 14$\sim$15). The parameters θ_u^{own} and θ_u^{mutual} are learnt jointly through stochastic gradient descent (i.e., only a subset of transitions is sampled for a reduced complexity as shown in line 16). Combining the techniques of *fractional slicing* and *mutual sampling*, the loss function is trained across all independent fractions within agent u's own DQN as well as the mutual DQN with the interacting teammate v (i.e., lines 17∼22). In consideration of the duplication separation technique that separates the pools of $\mathcal{D}_u$ and $\mathcal{D}_{v\to u}$, the duplicate needs to be trained in a similar way (i.e., lines 23∼26).

Algorithm 1: FSMS algorithm

```
1  Initialize the experience memory $\mathcal{D}_u$ and $\mathcal{D}_{v\to u}$;
2  Initialize the parameters of $\theta_u^{own}$ and $\theta_u^{mutual}$ randomly;
3  for $\forall s_u \in \mathcal{S}$ do
4      for $\forall a_u \in \mathcal{A}_u^{\mathcal{C}_u}(s_u)$ do
5          $Q_u(s_u, a_u) = \frac{1}{F}\sum_{f=1}^{F} Q_{u,f}\left(s_u, a_{u,f}; \theta_u^{own}, \theta_u^{mutual}\right)$;
6          $\rho = random\_number(0,1)$;
7          if $\rho < \epsilon$ then
8              $ch_u$ = a randomly selected channel in $\mathcal{C}_u$;
9              $(v, ch_u) = \underset{a_u \in \mathcal{A}_u^{\{ch_u\}}(s_u)}{\arg\max} Q_u(s_u, a_u; \theta_u^{own}, \theta_u^{mutual})$;
10         else
11             Calculate $Q_u(s_u, a_u)$ for all actions $a_u \in \mathcal{A}_u^{\mathcal{C}_u}(s_u)$;
12             $(v, ch_u) = \underset{a_u \in \mathcal{A}_u^{\mathcal{C}_u}(s_u)}{\arg\max} Q_u(s_u, a_u; \theta_u^{own}, \theta_u^{mutual})$;
13         $a_u^{(t)} = (v, ch_u)$;
14         Execute action $a_u^{(t)}$ while yielding reward $r_u^{(t)}$ and the next state $s_u^{(t+1)}$;
15         Keep the transition $(s_u^{(t)}, a_u^{(t)}, r_u^{(t)}, s_u^{(t+1)})$ in memory $\mathcal{D}_u$ and $\mathcal{D}_{v\to u}$ for all $v \in \mathcal{T}(u)$;
16         Sample a subset of transitions $(s_u, a_u, r_u, s'_u)$ from $\mathcal{D}_u$ randomly;
17         if $s'_u$ is the terminal state then
18             $y_u = r_u$;
19         else
20             $y_u = r_u + \gamma \cdot \frac{1}{F}\sum_f \bar{Q}_{u,f}(s'_u, \underset{a'_{u,f} \in \mathcal{A}_{u,f}}{\arg\max} Q_{u,f}(s'_u, a'_{u,f}; \theta_u^{own}, \theta_u^{mutual}))$;
21         Perform the gradient descent training on the loss function of $L_u(\theta_u) = \mathbb{E}[\frac{1}{F}\sum_f \left(y_{u,f} - Q_{u,f}\left(s_u, a_{u,f}; \theta_u^{own}, \theta_u^{mutual}\right)\right)^2]$;
22         Share $\theta_u^{mutual}$ to all agents $v \in \mathcal{T}(u)$;
23         for $v \neq u$ do
24             Sample a subset of transitions from $\mathcal{D}_{v\to u}$;
25             Update $\theta_u^{own}, \theta_u^{mutual}$ as the above iterations;
26             Assign $\theta_u^{mutual}$ to agent $v$;
```

Compared with the algorithm with the holistic action spaces, the FSMS algorithm's computational complexity is reduced from $O(|\mathcal{T}_u| \cdot |\mathcal{C}_u|^2)$ to $O(|\mathcal{T}_u| \cdot \sum_{f=1}^{F} |\mathcal{C}_{u,f}|^2)$, where $|\mathcal{T}_u|$ denotes the number of u's neighbors and $|\mathcal{C}_{u,f}|$ denotes the number of u's local channels in its f-th fraction.

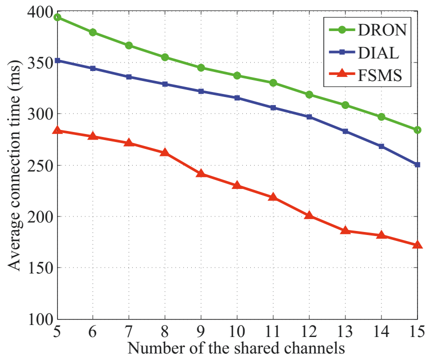
5 Experiments

In this section, we conduct experiments to evaluate the FSMS algorithm. We select the DRON [He *et al.*, 2016] and the DIAL [Foerster *et al.*, 2016] algorithms as the benchmarks and apply them to the UAVs' optimal link discovery and selection problem. According to the number of UAVs participated in the connection task, we divide the experiments into two sets: *five UAVs* and *ten UAVs*. We utilize the performance metrics of *average connection time* among all the collaborative UAVs and *average reward* (number of messages that are delivered successfully) yielded by each iteration to evaluate the compared algorithms. Each result is obtained by averaging over 1,000 independent runs of the algorithms.

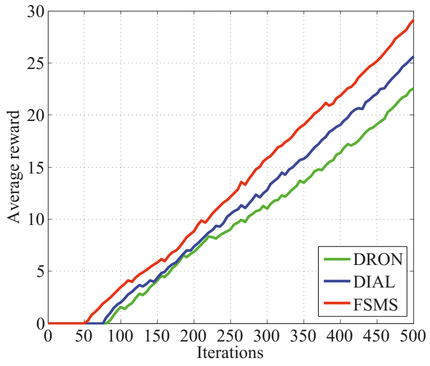
In terms of the metric of average connection time, as the number of common channels shared by the coexisting UAVs is a significant influence factor, we thus observe its variation trend under a fixed total number of channels and a varying number of shared channels. We assume there are totally 30 channels that are potentially available for all the UAVs. The local available channels perceived by the dispersed UAVs overlap but are different. We obtain the average connection time by varying the number of shared channels from 5 to 15. As for the metric of average reward, we compare the accumulative number of messages that are propagated among the connected UAVs at each iteration of the learning process.

Fig.2(a) shows the comparisons on average connection time with five UAVs. It can be seen that all algorithms need shorter time to get connected as the number of shared channels increases due to the fact that more shared channels indicate more connected opportunities. The DRON algorithm takes the longest average connection time, followed by the DIAL algorithm. Our FSMS algorithm takes the shortest connection time under all cases. The DRON model learns a reactive policy in a stochastic environment from the view point of one single agent and the policy space of other agents is unknown. Hence, DRON suffers from high complexity and nonstationary learning. The DIAL algorithm combines the centralized learning with the DQN, which indicates that the learning gradients can be passed from agent to agent. The end-to-end training across agents in DIAL is similar with our mutual sampling method. However, DIAL relies on the centralized learning, and the lack of duplication separation makes the interactive training confusing. As an improvement, FSMS utilizes fine-grained fractional slicing to achieve high-rate learning. Moreover, duplication separation is applied to the mutual sampling method, which can distinguish the mixed training information and expedite the convergence process. Fig.2(b) compares the average reward during 500 times of iterations. It can be seen that the FSMS algorithm yields the most rewards, followed by the DIAL and the DRON algorithms. The reason is that the better connected link is the foundation of successfully propagating more messages.

Fig.3(a) displays the comparisons on average connection time with ten collaborative UAVs. The downward trends of the three algorithms are similar with those of Fig.2(a). The d-

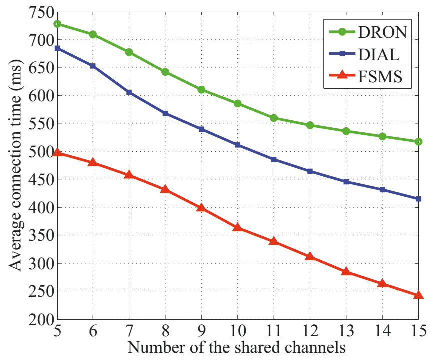


(a) Average connection time (ms)

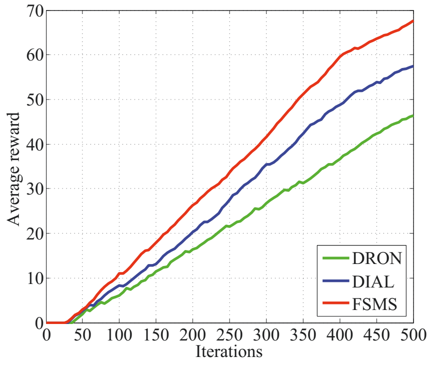


(b) Average reward

Figure 2: Evaluation results of five collaborative UAVs



(a) Average connection time (ms)



(b) Average reward

Figure 3: Evaluation results of ten collaborative UAVs

ifference is that all algorithms take longer time to get connected as the number of UAVs increases. Moreover, the superiority of the FSMS algorithm becomes more apparent. Fig.3(b) compares the average reward with ten UAVs. The reward results of all the three algorithms are relatively larger than those of Fig.2(b). The reason is that more interacting UAVs can facilitate the mutual learning about the connected links, leading to a more continuous propagation of messages.

From the evaluation results, we know that the key techniques of fractional slicing and mutual sampling in the FSMS algorithm are feasible and efficient in making optimal decisions on the UAVs' continuous link selection issue.

6 Conclusion

In this paper, we study the UAVs' optimal link discovery and selection problem, in which UAVs must select the most qualified link towards peers to support rapid information sharing and efficient teamwork. The usage of RL methods here is challenging since the action spaces are high-dimensional and continuous, and the UAVs should overcome the constraints of local perceptions and nonstationary learning. We propose a novel DRL algorithm to address these issues. The *fractional slicing* technique can train the sub-actions in a fine-grained manner and reduce the computational complexity. The *mutual sampling* technique can extend the perception range and expedite the policy-convergence process. Experiment results also verify the feasibility and efficiency of the proposed algorithm on the UAVs' continuous network connection task.

Acknowledgments

This work was supported by National Natural Science Foundation of China under Grants 61732017, 61472404, 61472402, 61501125 and 61502457, and China Postdoctoral Science Foundation under Grant 2018M631582. Min Liu is the contact author.

References

[Anschel *et al.*, 2017] Oron Anschel, Nir Baram and Nir Baram. Averaged-DQN: Variance reduction and stabilization for deep reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, Sydney, Australia, 2017.

[Barrett *et al.*, 2017] Samuel Barrett, Avi Rosenfeld, Sarit Kraus and Peter Stone. Making friends on the fly: Cooperating with new teammates. *Artificial Intelligence*, 242: 132–171, 2017.

[Chen *et al.*, 2017] Yu Fan Chen, Michael Everett, Miao Liu and Jonathan P. How. Socially aware motion planning with deep reinforcement learning. In *Proceedings of 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vancouver, Canada, 2017.

[Chinchali *et al.*, 2018] Sandeep Chinchali, Pan Hu, Tianshu Chu, Manu Sharma, Manu Bansal, Rakesh Misra, Marco Pavone and Sachin Katti. Cellular network traffic scheduling with deep reinforcement learning. In *Pro-*

ceedings of the 32th AAAI Conference on Artificial Intelligence, New Orleans, USA, 2018.

[Foerster *et al.*, 2016] Jakob N. Foerster, Yannis M. Assael, Nando de Freitas and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In *Proceedings of the 30th Conference on Neural Information Processing Systems*, Barcelona, Spain, 2016.

[Gu *et al.*, 2016] Shixiang Gu, Timothy Lillicrap, Ilya Sutskever and Sergey Levine. Continuous deep Q-learning with model-based acceleration. In *Proceedings of the 33th International Conference on Machine Learning*, New York, USA, 2016.

[Hasselt *et al.*, 2016] Hado van Hasselt, Arthur Guez and David Silver. Deep reinforcement learning with double Q-learning. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, Phoenix, USA, 2016.

[He *et al.*, 2016] He He, Jordan Boyd-Graber, Kevin Kwok and Hal Daume III. Opponent modeling in deep reinforcement learning. In *Proceedings of the 33th International Conference on Machine Learning*, New York, USA, 2016.

[Kulkarni *et al.*, 2016] Tejas D. Kulkarni, Karthik R. Narasimhan and Ardavan Saeedi. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Proceedings of the 30th Conference on Neural Information Processing Systems*, Barcelona, Spain, 2016.

[Leibo *et al.*, 2017] Joel Z. Leibo, Vinicius Zambaldi, Marc Lanctot, Janusz Marecki and Thore Graepel. Multi-agent reinforcement learning in sequential social dilemmas. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, São Paulo, Brazil, 2017.

[Liu *et al.*, 2017] Weirong Liu, Gaorong Qin, Yun He and Fei Jiang. Distributed cooperative reinforcement learning-based traffic signal control that integrates V2X networks' dynamic clustering. *IEEE Transactions on Vehicular Technology*, 66(10): 8667–8681, 2017.

[Mnih *et al.*, 2015] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, *et al.* Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533, 2015.

[Mnih *et al.*, 2016] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Tim Harley, Timothy P. Lillicrap, David Silver and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33th International Conference on Machine Learning*, New York, USA, 2016.

[Omidshafiei *et al.*, 2017] Shayegan Omidshafiei, Jason Pazis, Christopher Amato, Jonathan P. How and John Vian. Deep decentralized multi-task multi-agent reinforcement learning under partial observability. In *Proceedings of the 34th International Conference on Machine Learning*, Sydney, Australia, 2017.

[Sharma *et al.*, 2017] Sahil Sharma, Aravind S. Lakshminarayanan and Balaraman Ravindran. Learning to repeat: Fine grained action repetition. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, 2017.

[Sutton and Barto, 1998] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT Press, Cambridge, 1998.

[Tampuu *et al.*, 2017] Ardi Tampuu, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru and Raul Vicente. Multiagent cooperation and competition with deep reinforcement learning. *PLoS ONE*, 12(4): 1–15, 2017.

[Wang *et al.*, 2017] Jingjing Wang, Chunxiao Jiang, Zhu Han, Yong Ren, Robert G. Maunder and Lajos Hanzo. Taking drones to the next level: Cooperative distributed unmanned-aerial-vehicular networks for small and mini drones. *IEEE Vehicular Technology Magazine*, 12(3): 73–82, 2017.

[Wang *et al.*, 2016] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot and Nando de Freitas. Dueling network architectures for deep reinforcement learning. In *Proceedings of the 33th International Conference on Machine Learning*, New York, USA, 2016.

[Wu *et al.*, 2017] Di Wu, Dmitri I. Arkhipov, Minyoung Kim, Carolyn L. Talcott, Amelia C. Regan, Julie A. McCann and Nalini Venkatasubramanian. Addsen: Adaptive data processing and dissemination for drone swarms in urban sensing. *IEEE Transactions on Computers*, 66(2): 183–197, 2017.

[Xiao *et al.*, 2018] Liang Xiao, Caixia Xie, Minghui Min and Weihua Zhuang. User-centric view of unmanned aerial vehicle transmission against smart attacks. *IEEE Transactions on Vehicular Technology*, 67(4): 3420–3430, 2018.

[Xu *et al.*, 2018] Zhiyuan Xu, Jian Tang, Jingsong Meng, Weiyi Zhang, Yanzhi Wang, Chi Harold Liu and Dejun Yang. Experience-driven networking: A deep reinforcement learning based approach. In *Proceedings of the 37th IEEE International Conference on Computer Communications*, Honolulu, USA, 2018.

[Ye *et al.*, 2017] Dayong Ye, Minjie Zhang and Athanasios V. Vasilakos. A survey of self-organization mechanisms in multiagent systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(3): 441–461, 2017.

[Zhang *et al.*, 2018] Bo Zhang, Chi Harold Liu, Jian Tang, Zhiyuan Xu, Jian Ma and Wendong Wang. Learning-based energy-efficient data collection by unmanned vehicles in smart cities. *IEEE Transactions on Industrial Informatics*, 14(4): 1666–1676, 2018.

[Zhang *et al.*, 2017] Zongzhang Zhang, Zhiyuan Pan and Mykel J. Kochenderfer. Weighted double Q-learning. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, Melbourne, Australia, 2017.