Coordinated Multiagent Reinforcement Learning for Teams of Mobile Sensing Robots

Extended Abstract

Chao Yu
School of Computer Science & Technology, Dalian University of Technology, Dalian, China
cy496@dlut.edu.cn

Xin Wang
School of Computer Science & Technology, Dalian University of Technology, Dalian, China
1109525927@qq.com

Zhanbo Feng
School of Computer Science & Technology, Dalian University of Technology, Dalian, China
571102482@qq.com

ABSTRACT

A mobile sensing robot team (MSRT) is a typical application of multi-agent systems. This paper investigates multiagent reinforcement learning in the MSRT problem. A naive coordinated learning approach is first proposed that uses a coordination graph to model interaction relationships among robots. To further reduce the computation complexity in the context of continuously changing topology caused by robots' movement, we then propose an on-line transfer learning method that is capable of transferring the past interaction experience and learned knowledge to a new context in a dynamic environment. Simulations verify that the method can achieve reasonable team performance by properly balancing robots' local selfish interests and global team performance.

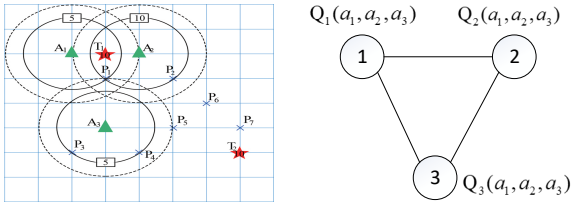
KEYWORDS

Mobile Sensing Robot Team; Coordination; Reinforcement Learning; Transfer Learning; Coordination Graph

ACM Reference Format:
Chao Yu, Xin Wang, and Zhanbo Feng. 2019. Coordinated Multiagent Reinforcement Learning for Teams of Mobile Sensing Robots. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 3 pages.

1 INTRODUCTION

A *mobile sensing robot team* (MSRT) is one type of *multi-agent systems* (MASs), in which a group of mobile robots perform coverage or monitoring tasks by sensing collaboratively in a common environment [7]. An MSRT problem [11] can be given by a tuple $< \mathcal{A}, \mathcal{P}, \mathcal{T}, \mathcal{G} >$, in which $\mathcal{A} = \{\alpha_1, \alpha_2, ..., \alpha_n\}$ is a finite set of robots (agents), $\mathcal{P} = \{P_1, P_2, ..., P_x\}$ is a set of possible locations of the agents, $\mathcal{T} = \{T_1, T_2, ..., T_m\}$ is a set of targets that the agents are aiming to cover, and $\mathcal{G}$ is a goal function. In MSRTs, each agent α_i is physically situated in the environment and its *current position* is denoted by $CP_i \in \mathcal{P}$. The maximum distance that α_i can travel in a single time step is defined by its *mobility range* MR_i. Agents have limited sensing ranges such that agent α_i can only provide information on targets within its *sensing range* SR_i. Agents may also differ in the *credibility* CR_i (i.e., quality) of their sensing abilities, with higher values indicating better sensing abilities. When a set of S agents are sensing the same target at the same time, the *joint credibility* JC of these agents can be calculated as $JC(S) = \sum_{\alpha_i \in S} CR_i$. Each target T_i is represented implicitly by an *environmental requirement* value ER_i representing the credibility required for that target to be adequately sensed. Thus, the *remaining coverage requirement* of target T_i can be given as $RR_i = \max\{0, ER_i - JC(S_{T_i})\}$, where S_{T_i} is the set of agents that are covering target T_i. A major goal of the MSRT problem is for the agents to explore the environment sufficiently to be aware of the presence of targets and position themselves to minimize RR_i for all targets, $\mathcal{G} : min \sum_{T_i \in \mathcal{T}} RR_i$. Figure 1(a) gives an illustration of an MSRT problem with three agents and two targets.



(a) An MSRT problem (b) The CG structure

Figure 1: An MSRT with 3 agents (green triangles) and 2 targets (red stars), and its CG structure.

2 COORDINATED MARL FOR MSRTS

In this paper, we model the MSRT problem as a *multiagent reinforcement learning* (MARL) problem [1], in which agents learn to coordinate their behaviors for a maximized global team performance. The local state S_i involves the set of targets that agent i can sense at present as well as after a single time step. The local action set A_i can be simply defined as the set of all the positions within the mobility range of agent i. The individual reward for agent i can be given by:

$$r_i = \sum_{T_j \in \mathcal{T}} \{\min\{ER_j, CR_{i\{i \in \mathcal{A}'_{T_j}\}}\} - \min\{ER_j, CR_{i\{i \in \mathcal{A}_{T_j}\}}\}\},$$

To model the influence of an agent's action on the whole team, a reward function for group evaluation is given as:

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada. © 2019 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

$$r_g = \sum_{T_j \in \mathcal{T}} \{|ER_j - \sum_{i \in \mathcal{A}_{T_j}} CR_i| - |ER_j - \sum_{i \in \mathcal{A}'_{T_j}} CR_i|\},$$

The overall reward function then can be defined as a weighted sum of these two components as $R_i = w \times r_i + (1 - w) \times r_g$, where w is a weight to model a trade-off between agents' local selfish interests and global team performance.

In the proposed *Distributed Coordinated Learning* (DCL) approach, the global value function can be decomposed into a linear combination of local value functions as $Q(\mathbf{js}, \mathbf{ja}) = \sum_i Q_i(s_i, \mathbf{ja}_i)$, where $Q(\mathbf{js}, \mathbf{ja})$ stands for the global value function for the joint state $\mathbf{js}$ and joint action $\mathbf{ja}$ of all the agents, while $Q_i(s_i, \mathbf{ja}_i)$ is the local Q value function of agent i, and can be updated by,

$$Q_i(s_i, \mathbf{ja}_i) = Q_i(s_i, \mathbf{ja}_i) + \alpha[R_i + \gamma \max_{\mathbf{ja}'_i} Q'_i(s'_i, \mathbf{ja}'_i) - Q_i(s_i, \mathbf{ja}_i)], \quad (1)$$

where R_i is the reward value that agent i receives, and $\max_{\mathbf{ja}'_i} Q'_i(s'_i, \mathbf{ja}'_i)$ is the maximum value function for agent i that is computed by VE on the new CG at next time step.

3 KNOWLEDGE TRANSFER LEARNING IN MSRTS

Simulation results verifies the effectiveness of the *DCL* approach in solving a simple MSRT in Figure 1. However, as the domain size gets larger, the storage and computation complexity grows exponentially with the increase of number of agents, causing significant scalability issues. To solve this problem, we propose a transfer learning method in the coordinated learning process that is able to adaptively transfer the learning information at previous step to that at current step as the topology of agents is changing dynamically. This can be achieved by two knowledge transfer processes: the *knowledge distilling process*, and the *knowledge synthesis process*.

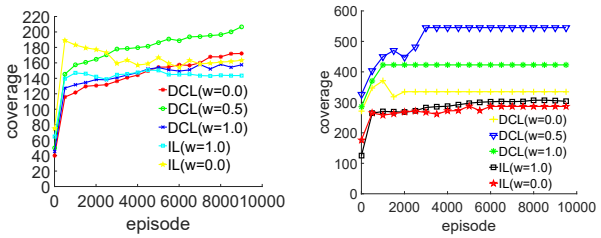
The knowledge distilling process: In order to adapt to the continuously changing environment, an agent must transfer its knowledge about previous neighbors to new neighbors. To enable this knowledge transfer, agent i must first extract its own knowledge in terms of $Q_i(s_i, a_i)$ out of the higher-dimensional knowledge $Q_i(s_i, \mathbf{ja}_i)$ that is conditioned on the joint actions over all its neighbors. This process can be realized by simply discarding the redundant information of the neighbors as given by Equation 2.

$$Q_i(s_i, a_i) = Q_i(s_i, \mathbf{ja}_i) \times (D(i) + 1) - \sum_{(i,j) \in E} \frac{\sum_{s_j \in S_j} Q_j(s_j, a_j)}{|S_j|}, \quad (2)$$

where $D(i)$ is the number of neighbors of agent i, E is the set of neighboring edges on the current topology structure, and S_j is the set of states that involves neighbor j.

The knowledge synthesis process: While the role of knowledge distilling is to solve the computation complexity problem by reducing the dimension of information, the process of knowledge synthesis is to restore the coordinated Q value function $Q(s_i, \mathbf{a}_i)$ on the agent i for computing its coordinated joint action value with its new neighbors and maximizing the global payoff function. This process is a reverse process of knowledge distilling, as given by:

$$Q_i(s_i, \mathbf{ja}_i) = \frac{Q_i(s_i, a_i) + \sum_{(i,j) \in E} \frac{\sum_{s_j \in S_j} Q_j(s_j, a_j)}{|S_j|}}{(D(i) + 1)}, \quad (3)$$



(a) 20×20 grid (b) 50×50 grid

Figure 2: Results in two sizes of MRST domains.

Figure 2(a) gives the results in a 20×20 grid environment, involving 8 agents to cover 4 randomly located targets. The *DCL* approach with an equal weight can achieve far better performance than the other approaches. When coordinating more agents in an even larger domain, the CG may be too condense for the VE algorithm to compute a global optimal joint action efficiently at each time step. To address this issue, a heuristic is proposed to reduce the complexity of CG based on the influence of neighbors on a local agent. If some neighboring edges are considered to play minor influence, these edges can be deleted from the CG. Figure 2(b) shows the final results when coordinating 20 agents in a 50×50 grid environment. It is clear that the *DCL* approaches can achieve far better performance than the *IL* approaches, which fully demonstrates the benefits of coordinated learning in larger complex domains.

4 CONCLUSIONS

In this paper, we solve the MSRT problem from a learning perspective, which deviates from the existing mainstream of research that focuses on using search or inference algorithms for solving a DCOP problem [2, 6–8, 10, 11]. Unlike other existing studies in MARL that still focus on static and close environments [3–5, 9], learning efficient coordinated behaviors in the MSRT problems is challenging due to the mobility, limited communication and observability range of agents. Thus, this paper makes an initial progress in addressing MARL problems in a dynamic learning environment where the agents' sensing tasks, actions available and their mutual relationships are changing continuously.

5 ACKNOWLEDGMENTS

This work was supported by the Joint Key Program of National Natural Science Foundation of China and Liaoning Province under Grant U1808206.

REFERENCES

[1] Lucian Busoniu, Robert Babuska, and Bart De Schutter. 2008. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, And Cybernetics-Part C: Applications and Reviews, 38 (2), 2008* (2008).

[2] Alessandro Farinelli, Alex Rogers, Adrian Petcu, and Nicholas R Jennings. 2008. Decentralised coordination of low-power embedded devices using the max-sum algorithm. In *AAMAS'08*. ACM, New York, NY, USA, 639–646.

[3] Carlos Guestrin, Michail Lagoudakis, and Ronald Parr. 2002. Coordinated reinforcement learning. In *ICML'02*, Vol. 2. 227–234.

[4] Jelle R Kok and Nikos Vlassis. 2006. Collaborative multiagent reinforcement learning by payoff propagation. *Journal of Machine Learning Research* 7, Sep (2006), 1789–1828.

[5] Lior Kuyer, Shimon Whiteson, Bram Bakker, and Nikos Vlassis. 2008. Multiagent reinforcement learning for urban traffic control using coordination graphs. *Machine learning and knowledge discovery in databases* (2008), 656–671.

[6] Harel Yedidsion and Roie Zivan. 2016. Applying DCOP_MST to a Team of Mobile Robots with Directional Sensing Abilities. In *AAMAS'16*. ACM, New York, NY, USA, 1357–1358.

[7] Harel Yedidsion, Roie Zivan, and Alessandro Farinelli. 2014. Explorative max-sum for teams of mobile sensing agents. In *AAMAS'14*. ACM, New York, NY, USA, 549–556.

[8] Harel Yedidsion, Roie Zivan, and Alessandro Farinelli. 2018. Applying max-sum to teams of mobile sensing agents. *Engineering Applications of Artificial Intelligence* 71 (2018), 87–99.

[9] Chao Yu, Minjie Zhang, Fenghui Ren, and Guozhen Tan. 2015. Multiagent learning of coordination in loosely coupled multiagent systems. *IEEE transactions on cybernetics* 45, 12 (2015), 2853–2867.

[10] Roie Zivan, Robin Glinton, and Katia Sycara. 2009. Distributed constraint optimization for large teams of mobile sensing agents. In *IEEE/WIC/ACM WI-IAT'09*. IEEE, Los Alamitos, CA, 347–354.

[11] Roie Zivan, Harel Yedidsion, Steven Okamoto, Robin Glinton, and Katia Sycara. 2015. Distributed constraint optimization for teams of mobile sensing agents. *Autonomous Agents and Multi-Agent Systems* 29, 3 (2015), 495–536.