# Distributed Multiagent Reinforcement Learning on Coordination Graphs

Chao Yu[1], Yinzhao Dong[2], and Xin Wang[2]

[1] School of Data and Computer Science, Sun Yat-Sen University, 510006, Guangzhou, China
[2] School of Computer Science and Technology, Dalian University of Technology, 116024, Dalian, China
yuchao3@mail.sysu.edu.cn, {1447866357,1109525927}@qq.com

**Abstract.** Tremendous progress has been made in the investigation of coordinated multiagent reinforcement learning (RL) under the framework of coordination graphs (CGs) by simplifying a global decision making problem into sub-ones by exploiting structural dependencies among agents. Most existing work only focuses on coordinating agents' behaviors over one particular component of RL algorithms, i.e., the global TD errors. In this paper, a general methodology is proposed to demonstrate that learning on CGs can be greatly enriched by coordinating over various components of RL. Using this methodology, several distinct types of coordinated learning methods can be clarified, and their advantages as well as disadvantages can be comparatively studied. Our work provides a unified and systematic understanding of various coordinated learning methods on CGs, and therefore, sheds some light on designing and choosing more efficient learning methods for a specific coordination problem. Simulation results in the distributed sensor network problem verify that the proposed coordinated learning methods can achieve various benefits in performance improvement.

**Keywords:** Multiagent Systems · Coordinated Learning · Reinforcement Learning · Coordination Graphs · Variable Elimination.

## 1 Introduction

Coordination is crucial for enabling full functionalities of multiagent systems (MASs) [17]. The need for coordination arises because the effect of an agent's action also depends on the actions of other agents in the same environment. Hence, agents' actions must be mutually consistent in order to achieve their intended effects. This is specially the case in fully cooperative MASs where all agents share a common goal, and it should be guaranteed that the individual decisions of the agents result in jointly optimal decisions for the whole group.

Multiagent reinforcement learning (MARL) is a promising technique to achieve coordination among agents in MASs, and has been widely studied in the past decades in various theoretical and practical perspectives [7, 11, 17]. The core issue is that the computational complexity in MARL grows exponentially with the number of agents, which renders it impractical to search in the joint state/action space of all the agents. One way to alleviate this problem is to exploit certain level of independence among agents. Coordination Graph (CG) [6] is one of such effective techniques to enable decomposition of the global decision problem into a combination of local sub-problems. A CG can be expressed as an undirected graph, in which each node represents an agent and an edge indicates the corresponding agents having to coordinate their actions. By using various message passing algorithms, e.g., Variable Elimination (VE) [6], a joint optimal action of agents can be computed to maximize the global decision function.

Several studies have investigated how to realize coordinated MARL on CGs. In [6], three explicit MARL algorithms have been proposed, including two methods with value function approximation (a variant of Q-learning [15] and Least Squares Policy Iteration (LSPI) [13]) as well as a direct policy search method [5]. The basic idea is to maintain a local value function for each agent and update it along the gradient of the global mean square Temporal Difference (TD) error with the optimal joint actions being computed by the VE algorithm. In [11], the Sparse Cooperative Q-Learning algorithm was proposed to enable agent-based or edge-based update of local Q values on CGs. In all these algorithms, structured communication and coordination between agents only resides in the TD error of the learning process.

However, as we look back at the learning process and update rule of general RL algorithms, we can easily find that, apart from the TD error, there are other basic components in RL algorithms where coordination can be executed. To this end, in this paper, we provide a general methodology of realizing coordinated learning on CGs. By focusing on different components of an RL process, where coordination can be conducted, several types of learning algorithms can be clarified, and their advantages as well as disadvantages can be comparatively studied. We evaluate the proposed coordinated MARL algorithms on the distributed sensor network (DSN) problem, which is a typical cooperative MAS and a general testing domain for studying MARL algorithms. Our work provides a unified and systematic understanding of various coordinated learning methods on CGs, and therefore, sheds some light on designing and choosing efficient learning methods for a specific coordination problem.

This paper is organized as follows. Section 2 introduces MARL and CG, which are the preliminaries of this paper. Section 3 formalizes the DSN problem and presents the general MARL algorithms based on CGs. Section 4 verifies the benefits of the proposed MARL algorithms. Section 5 discusses related work. Finally, Section 6 concludes the paper with directions for future research.

## 2 Preliminaries

RL is a general class of algorithms in the field of machine learning that aims at allowing an agent to learn how to make sequential decisions in an environment, where the only feedback consists of a scalar reward signal [16]. A Markov Decision Process (MDP) is a commonly adopted framework for RL problems, which can be defined by a 4-tuple $M = (S, A, P, R)$, where $S$ is a finite state space, $A$ is a set of actions available to the agent, $P(s, a, s') : S \times A \times S \to [0, 1]$ is a Markovian transition function when the agent transits from state $s$ to $s'$ after taking action $a$, and $R : S \times A \to R$ is a reward function that returns immediate reward $R(s, a)$ to the agent after taking action $a$ in state $s$. An agent's *policy* $\pi : S \times A \to [0, 1]$ is a probability distribution that maps a state $s \in S$ to an action $a \in A$. The goal in an MDP is to learn a policy $\pi$ so as to maximize the expected discounted reward $V^{\pi}(s)$ for each state $s \in S$, as given by Eq. (1).

$$V^{\pi}(s) = E_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t)) | s_0 = s\right], \tag{1}$$

where $E_{\pi}$ is the expectation of policy $\pi$, $s_t$ denotes the state at time $t$, and $\gamma \in [0, 1)$ is a discount factor.

For any finite MDP, there is at least one *optimal policy* $\pi^*$, such that $V^{\pi^*}(s) \geq V^{\pi}(s)$ for every policy $\pi$ and every state $s \in S$. $\pi^*$ can be computed by using linear programming or dynamic programming techniques if an agent fully knows the reward and transition functions of the environment. When these functions are unknown, finding an optimal policy in MDP can be solved using RL methods. One of the most important and widely used RL approach is Q-learning [15], which is an off-policy model-free temporal difference control algorithm. The one step updating rule is given by Eq. (2), where $\alpha \in (0, 1]$ is a learning rate.

$$Q_{t+1}(s, a) = Q_t(s, a) + \alpha[R(s, a) + \gamma \max_{a'} Q_t(s', a') - Q_t(s, a)] \tag{2}$$

Every Q value can be stored in a table for discrete state-action space. It is proved that this tabular Q-learning method converges to the optimal $Q^*(s, a)$ w.p.1 when all state-action pairs are visited infinitely and an appropriate exploration strategy and learning rate are chosen [15].

When multiple agents coexist in the same environment and conduct learning simultaneously, it becomes the MARL problem. MARL is a promising paradigm to study how agents can learn to achieve satisfactory behaviors in complex environments, providing additional advantages that cannot be seen in single-agent learning. The major difficulty in MARL is that the computation complexity grows exponentially with the number of agents. One way to alleviate this problem is to exploit certain level of independence among agents. Coordination Graph (CG) [6] is one of such effective approaches, which enables decomposing global payoff function $Q(\mathbf{s}, \mathbf{a})$ into a linear combination of local payoff functions. This decomposition can be depicted using an undirected graph $G = (V, E)$ in which each node $i \in V$ represents an agent and an edge $(i, j) \in E$ indicates that the corresponding agents have to coordinate their actions. Allowing payoff functions defined over at most two agents, the global payoff function $Q(\mathbf{s}, \mathbf{a})$ can be given as follows:

$$Q(\mathbf{s}, \mathbf{a}) = \sum_{(i,j) \in E} Q_{ij}(\bar{s_{ij}}, a_i, a_j) \tag{3}$$

where $\mathbf{s} = \langle s_1, ..., s_n \rangle \in S$ and $\mathbf{a} = \langle a_1, ..., a_n \rangle \in A$ are the joint states and joint actions of all agents, respectively, and $\bar{s_{ij}}$ represents the relevant state variables for agent $i$ and $j$.

The main goal of CG is to find a coordination strategy of actions for the agents to maximize $Q(\mathbf{s}, \mathbf{a})$ at state $\mathbf{s}$. The VE algorithm [6] can be applied to solve the coordination problem and find the optimal $\mathbf{a}$. In VE, an agent first collects all payoff functions related to its edges before it is eliminated. It then computes a conditional payoff function which returns the maximal value for the agent to contribute to the system regarding every action combination of its neighbors, and a best-response function (or conditional strategy) which returns the action corresponding to the maximizing value. The conditional payoff function is communicated to its neighbors and the agent is eliminated from the graph. The agents are iteratively eliminated until one agent remains, and then a reverse order is performed in which every agent computes its optimal action based on its conditional strategy and fixed actions of its neighbors.

Having discussed the problem of selecting an optimal joint action for a given CG structure, it is then possible to employ coordinated RL approaches [6] for solving sequential decision making problems. The basic idea is to maintain a local value function for each agent, which can be represented, e.g., using a neural network, and update it along the gradient of the mean square TD error, resulting the following update rule $\Delta w_i = \alpha[r + \gamma \max_{\mathbf{a}} Q(\mathbf{s}', \mathbf{a}, w) - Q(\mathbf{s}, \mathbf{a}, w)] \nabla_{w_i} Q_i(s_i, a_i, w_i)$, in which the optimal joint action $\mathbf{a}$ in the new state and the previous joint Q-value can be computed by applying VE on the CG.

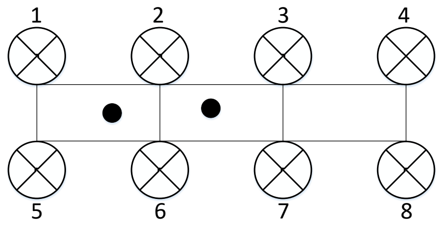
## 3 General MARL Algorithms on CGs

This section first describes the DSN problem and its MDP formulation, and then proposes the general MARL algorithms on CGs.

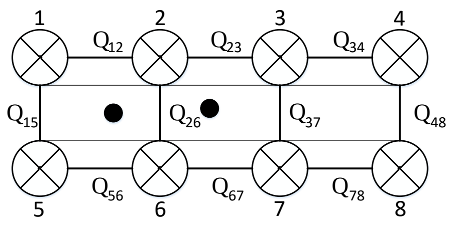
### 3.1 The DSN Problem

As a general testing bed for MARL algorithms, the distributed sensor network (DSN) problem [1, 11] is a sequential decision-making variant of the distributed constraint optimization problem that consists of a set of sensors monitoring certain targets. In a DSN, the monitoring space is divided into several cells and each cell, surrounded by exactly four sensors, can be occupied by a target. A target moves to the cell on its left, to the cell on its right, or remains on its current position with equal probability. Actions that move a target to a cell occupied by another target or outside the grid cannot be executed. Figure 1(a) gives a basic DSN setting of eight sensors and two targets, in which sensors and targets are depicted as $\otimes$ and $\bullet$, respectively.

Each sensor is able to perform three actions: focuses on a target in the cell to its immediate left, to its immediate right, or does not focus at all. Every focusing action has a small cost. Each target starts with a default energy level of 3. When at least three of the four surrounding sensors focus on a target at



(a) A basic DSN setting



(b) The corresponding CG

**Fig. 1.** A basic DSN setting of eight sensors(⊗) and two targets(●), with its corresponding CG.

the same time, the target is 'hit' and its energy is decreased by one. When the energy level reaches zero, the target is captured and removed, and the sensors involved in the capture each time receive a positive reward. In the case when four sensors are involved in a capture, only the three sensors with the highest index receive the reward. An episode finishes when all targets are captured.

A DSN is a typical cooperative MAS which can be modeled by a Multiagent Markov Decision Process (MMDP) [2], as a tuple $\langle \mathcal{A}, \{S_i\}, \{A_i\}, P, R, \gamma \rangle$, where $\mathcal{A} = \{\alpha_1, \alpha_2, ..., \alpha_n\}$ is the set of agents that represent the sensors, while $S_i$ and $A_i$ are sets of states and actions available to agent $\alpha_i$, respectively. The $S = \prod_{i \in S_i}$ and $A = \prod_{i \in A_i}$ are the sets of joint states and actions, respectively. Thus, the transition probability function $P$ can be defined as $P : S \times A \times S \to [0, 1]$ and the reward function $R = \prod_{i \in R_i}$ can be defined as $R : S \times A \times S \to R$. The parameter $\gamma$ is a discount factor in $[0, 1]$.

***State*** $S_i$**.** Due to partial observability, the local state of each agent is composed of the energy level in the cell on its left $e_l$ and right $e_r$. Thus, the local state $S_i$ is a two tuple of state $\langle e_l, e_r \rangle$ for each agent.

***Action*** $A_i$**.** As the goal is to capture all the targets at the lowest cost as soon as possible, three main actions are: focusing on a target in the cell to its left, focusing on a target in the cell to its right, or not focusing at all.

***Reward*** $R_i$**.** The reward function is crucial in RL, which indicates the learning goal for an agent. As we aim at decreasing the energy level of all the targets to 0, the reward function of an agent is defined as follow:

$$R_i = \begin{cases} 10 & \text{if the energy level of a target decreases to 0 and the agent is capturing it} \\ 0 & \text{if the agent does not focus at all} \\ -1 & \text{else} \end{cases} \tag{4}$$

Since the DSN problem is a typical cooperative MAS, the choice of actions for each agent always depends on the actions of other agents in the same environment. Agents must coordinate their actions in order to capture the targets as quickly as possible. Therefore, a link is built between two neighboring agents to indicate direct dependency between them. Figure 1(b) gives the corresponding CG of the basic DSN problem in Figure 1(a).

## 3.2 General MARL Algorithms on CGs

The previous coordinated MARL algorithms [6] can be abstracted as the following extremely simple form of $\Delta Q_i \leftarrow \alpha_i \delta_i$, combined with a proper action exploration strategy, e.g., the $\epsilon_i$-exploration strategy. Here, $Q_i$ is the Q value of agent $i$, $\alpha_i$ is the learning rate, $\epsilon_i$ is the exploration rate, and $\delta_i$ is the TD error. In an MARL problem, $\delta_i$ can indicate the global TD error of the whole system, i.e., $\delta_i^g = [r + \gamma \max_{\mathbf{a}} Q(\mathbf{s}', \mathbf{a}) - Q(\mathbf{s}, \mathbf{a})]$, where $\mathbf{a}$ and $\mathbf{s}$ are the joint actions and states for all the agents, and $r$ is the team reward. On the other hand, $\delta_i$ can also indicate the local TD error just for agent $i$, i.e., $\delta_i^l = [r_i + \gamma \max_{a_i} Q_i(s_i', a_i) - Q_i(s_i, a_i)]$. Therefore, an MARL algorithm can be clearly defined by a three-tuple parameters or components as $\zeta = \langle \alpha_i, \epsilon_i, \delta_i \rangle$.

Targeting on coordinating over different components, we now come up with a general methodology for conducting coordinated learning on CGs, which can be briefly summarized as the following steps:

**Step 1: Computing the coordinated joint actions.** Based on current structure of CG and the value functions of all the agents at this step, the VE algorithm is applied to compute the optimal joint actions for the agents. It is noted that the joint actions are optimal regarding the agents' estimated value functions at each learning step.

**Step 2: Determining the coordinated components.** We can simply coordinate over any one component in the tuple of $\zeta = \langle \alpha_i, \epsilon_i, \delta_i \rangle$, or any combination of these components.

**Step 3: Updating the chosen components.** After choosing the coordinated component(s), its(their) values can be updated using some predefined heuristics as introduced below.

The action performed by an agent at time step $t$, $a_i^t$, may be different from the action $a_i'$ as indicated by the optimal joint actions. Depending on this consistency, the component can be updated based on the following mechanisms:

**Component-$\alpha$:** In RL, the learning performance heavily depends on the learning rate parameter, which is difficult to tune. When agent $i$ has chosen the same action with the one indicated by the optimal joint actions, it decreases its learning rate to maintain its current state, otherwise, it increases its learning rate to learn faster from its interaction experience. Formally, learning rate $\alpha_i$ can be adjusted as follows:

$$\alpha_i = \begin{cases} \alpha_{min} & if\ a_i^t = a_i', \\ \alpha_{max} & otherwise. \end{cases} \tag{5}$$

where $\alpha_{min}$ is a "winning" learning rate and $\alpha_{max}$ is a "losing" learning rate.

**Component-$\epsilon$:** Exploration-exploitation trade-off has a crucial impact on the learning process. Therefore, this mechanism adapts the exploration rate $\epsilon$. The motivation of this mechanism is that an agent needs to explore more of the environment when it is performing poorly and explore less otherwise. Similarly, the exploration rate $\epsilon$ can be adjusted according to Equation 6:

$$\epsilon_i = \begin{cases} \epsilon_{min} & if\ a_i^t = a_i', \\ \epsilon_{max} & otherwise. \end{cases} \tag{6}$$

in which $\epsilon_{min}$ and $\epsilon_{max}$ are the "winning" and "losing" exploration rates, respectively.

**Component-$\alpha$-$\epsilon$:** This mechanism adapts the learning rate and the exploration rate at the same time based on the above two mechanims.

The above mechanisms are based on the concept of "winning" and "losing" in the well-known MARL algorithm WoLF (Win-or-Learn-Fast) [3]. Although the original meaning of "winning" or "losing" in WoLF and its variants is to indicate whether an agent is doing better or worse than its Nash-Equilibrium policy, this heuristic is gracefully introduced into the proposed framework to evaluate an agent's performance against the action suggested by the joint optimal actions computed over the CG. Specifically, an agent is considered to be winning (i.e., performing well) if its action is the same with the optimal action and losing (i.e., performing poorly) otherwise. The different situations of "winning" or "losing" thus indicate whether the agent is complying with the behavior of optimal social welfare in the society. If an agent is in a losing state (i.e., it is against the behavior of optimal social welfare), it needs to learn faster or explore more of the environment in order to escape from this adverse situation. On the contrary, it should decrease its learning/exploration rate to stay in the winning state.

## 4 Experiments

To validate the proposed coordinated MARL algorithms that coordinate over various components in RL, we solve the DSN problem in a specific setting of eight agents and two targets as given by Fig. 1(a). Parameters of different MARL algorithms are listed in Table 1, where $IL$ denotes *individual learning* where each agent learns simply based on its own local state, action and reward, while $G\_TD$ and $L\_TD$ denote coordinating over global and local TD errors, respectively.

Fig. 2 gives the performance in terms of average reward and time steps to complete the sensing task using different MARL algorithms coordinating over the learning rate $\alpha$. It is clear that the coordinated MARL algorithms can achieve better performance compared with the corresponding basic MARL algorithms. Since no coordination among agents has been conducted using $IL$, it achieves the worse performance among all the algorithms. A significant improvement can be achieved when coordinating over the learning rate $\alpha$ using $IL$. Since global information of TD errors is used, $G\_TD$ outperforms $L\_TD$ both in efficiency and final convergence level. However, when introducing the coordination component

**Table 1.** Parameters of different learning algorithms.

| Component | Learning Algorithm | $\alpha$ | $\epsilon$ | $\gamma$ | $\alpha_{max}$ | $\alpha_{min}$ | $\epsilon_{max}$ | $\epsilon_{min}$ |
|---|---|---|---|---|---|---|---|---|
| | IL | 0.1 | 0.1 | 0.95 | - | - | - | - |
| | G_TD | 0.1 | 0.1 | 0.95 | - | - | - | - |
| | L_TD | 0.1 | 0.1 | 0.95 | - | - | - | - |
| | $\alpha$_IL | - | 0.1 | 0.95 | 0.5 | 0.05 | - | - |
| $\alpha$ | $\alpha$_G_TD | - | 0.1 | 0.95 | 0.11 | 0.09 | - | - |
| | $\alpha$_L_TD | - | 0.1 | 0.95 | 0.5 | 0.05 | - | - |
| | $\epsilon$_IL | 0.1 | - | 0.95 | - | - | 0.5 | 0.1 |
| $\epsilon$ | $\epsilon$_G_TD | 0.1 | - | 0.95 | - | - | 0.5 | 0.1 |
| | $\epsilon$_L_TD | 0.1 | - | 0.95 | - | - | 0.5 | 0.1 |
| | $\alpha$_$\epsilon$_IL | - | - | 0.95 | 0.5 | 0.05 | 0.5 | 0.1 |
| $\alpha$ and $\epsilon$ | $\alpha$_$\epsilon$_G_TD | - | - | 0.95 | 0.11 | 0.09 | 0.5 | 0.1 |
| | $\alpha$_$\epsilon$_L_TD | - | - | 0.95 | 0.5 | 0.05 | 0.5 | 0.1 |

of learning rate, $\alpha\_L\_TD$ achieves a slower learning speed than $\alpha\_G\_TD$ at the beginning, but converges to a much higher level of rewards as the learning proceeds. This result indicates that the integration of learning rate can significantly bias the learning efficiency when coordinating over TD errors.

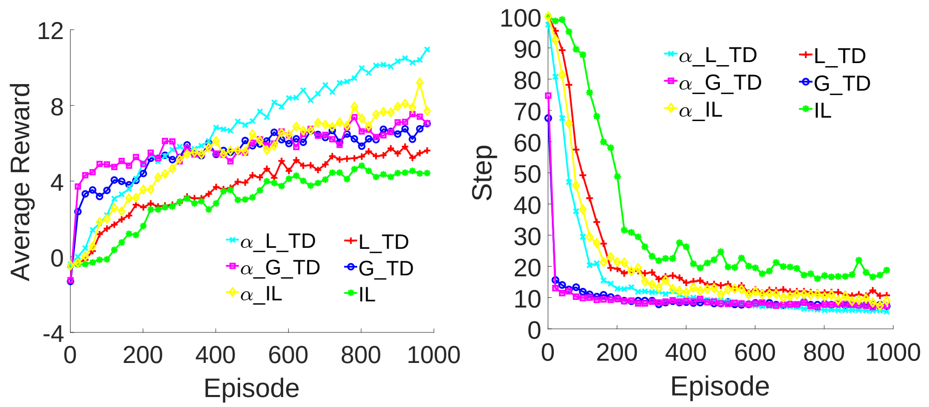


**Fig. 2.** Learning performance in terms of average reward and time steps to complete the sensing task using different MARL algorithms coordinating over the component $\alpha$.

The learning patterns are slightly different when coordinating over the exploration rate $\epsilon$, as given in Fig. 3. While all the coordinated learning algorithms still perform better than the non-coordinated $IL$ algorithm, $\epsilon\_L\_TD$ now performs worse than $\epsilon\_G\_TD$, which is contradictory to the result when coordinating over learning rate. Fig. 4 further gives the performance of different MARL algorithms when coordinating over both the learning rate $\alpha$ and the exploration rate $\epsilon$. The result patterns are similar to those when coordinating only over learning rate.

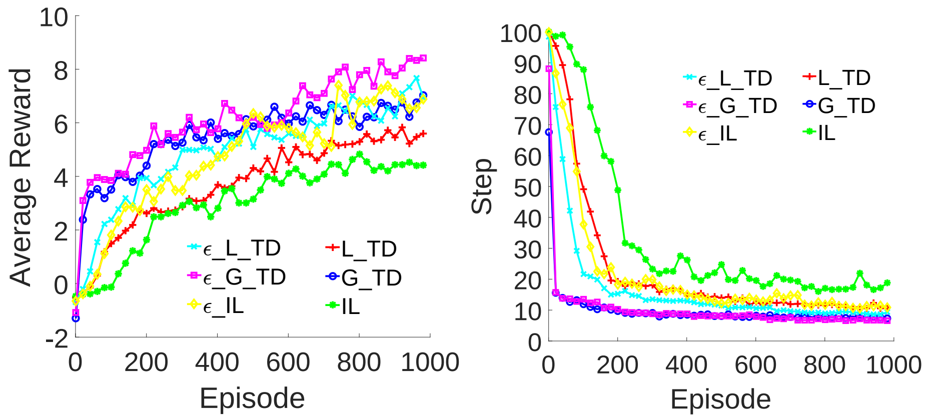


**Fig. 3.** Learning performance in terms of average reward and time steps to complete the sensing task using different MARL algorithms coordinating over the component $\epsilon$.

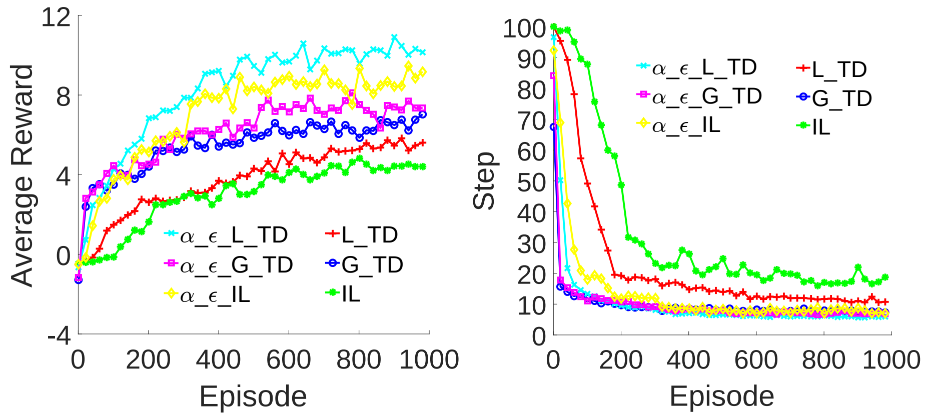


**Fig. 4.** Learning performance in terms of average reward and time steps to complete the sensing task using different MARL algorithms coordinating over $\alpha$ and $\epsilon$.

The final learning performance of different MARL algorithms is shown in Table 2. Among all the algorithms, $\alpha\_L\_TD$ receives the highest average reward of 10.96 and the lowest steps of 5.60, followed by $\alpha\_\epsilon\_L\_TD$. This indicates that the combination of learning rate and local TD is most beneficial in facilitating coordination among agents. Integration of exploration rate further, however, impairs the learning performance slightly. The component of exploration rate seems to be the most unhelpful factor in improving coordination, unless combined with global TD errors (i.e., $\epsilon\_G\_TD$).

## 5 Related Work

MARL [4] has gained a great deal of interest in the MAS and RL research in the past years. Plenty of studies have adopted CGs to decompose a global decision

**Table 2.** Performance of MARL algorithms coordinating over various components.

| Component | Learning Algorithms | Average Reward | Steps |
|---|---|---|---|
| | IL | 4.42 | 18.79 |
| | G_TD | 7.03 | 7.32 |
| | L_TD | 5.60 | 10.80 |
| | $\alpha$_IL | 7.71 | 8.97 |
| $\alpha$ | $\alpha$_G_TD | 7.06 | 7.40 |
| | $\alpha$_L_TD | 10.96 | 5.60 |
| | $\epsilon$_IL | 6.88 | 10.80 |
| $\epsilon$ | $\epsilon$_G_TD | 8.42 | 6.57 |
| | $\epsilon$_L_TD | 6.94 | 9.31 |
| | $\alpha$_$\epsilon$_IL | 9.18 | 7.08 |
| $\alpha$ and $\epsilon$ | $\alpha$_$\epsilon$_G_TD | 7.35 | 6.90 |
| | $\alpha$_$\epsilon$_L_TD | 10.14 | 5.92 |

making problem into a combination of local sub-problems for alleviating the computational compelxity [8,9]. Particularly, several studies have focused on coordinated learning based on CGs by exploiting local interactions or structural dependencies among agents to improve learning performance. Guestrin et al. [6] proposed a novel coordinated approach based on Q-learning, LSPI, and policy search RL algorithms. Kok et al. [10] proposed the *Utile Coordination* algorithm, which learns a policy comparable to the method that learns in the complete joint-action space both in a small coordination problem and a much larger predator-prey problem. Later, the authors [11] proposed the *Sparse Cooperative Q-learning* that enables agent-based or edge-based update of local Q values on CGs. Kuyer et al. [12] applied CG-based MARL in urban traffic control. More recently, Yu et al. applied coordinated MARL based on CGs in decision making of overtaking and following maneuvers in cooperative autonomous driving [22], as well as structural decomposition and decentralized learning in robotic control [21].

The WoLF heuristic is a general mechanism that has been widely used in different forms by previous studies, e.g., in MARL research [3], or collective dynamics [14]. Recently, this mechanism has also been extended to study the problem of norm learning in MASs, where the "winning" and "losing" indicate whether an agent's behavior is consistent with a supervision policy that is generated through a social learning process based on the agent's historical learning experience [18], or indicate whether an agent's behavior complies with the suggested social norm in organizational MASs [19,20].

## 6 Conclusions

In this paper, we proposed a set of general MARL algorithms that are able of coordinating over various leaning components on CGs. Simulation results in the DSN problems verified that coordinated learning over CGs could achieve better performance than individual learning and distinct learning patterns could be observed by coordinating over different learning components in RL. By providing a unified and systematic understanding of various coordinated learning methods on CGs, our work can thus shed some light on designing and choosing efficient coordinated learning methods for a specific coordination problem.

In the future work, we plan to extend the general algorithms to policy search and batch learning settings. Moreover, more detailed investigation of the underlying interplay and dynamics among different kinds of coordinated learning algorithms should be carried out, in order to unveil the reasons that lead to the various learning performance.

## Acknowledgments

This work is supported in part by the Ministry of Military Equipment Development of China under Grant 61403120203, in part by the Dalian High Level Talent Innovation Support Program under Grant 2017RQ008, and in part by the Dalian Science and Technology Innovation Fund under Grant 2018J12GX046. .

## References

1. Ali, S., Koenig, S., Tambe, M.: Preprocessing techniques for accelerating the dcop algorithm adopt. In: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems. pp. 1041–1048 (2005)
2. Boutilier, C.: Planning, learning and coordination in multiagent decision processes. In: Proceedings of the 6th Conference on Theoretical Aspects of Rationality and Knowledge. pp. 195–210 (1996)
3. Bowling, M., Veloso, M.: Multiagent learning using a variable learning rate. Artificial Intelligence **136**, 215–250 (2002)
4. Busoniu, L., Babuska, R., De Schutter, B.: A comprehensive survey of multiagent reinforcement learning. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews **38**(2), 156–172 (2008)
5. Deisenroth, M.P., Neumann, G., Peters, J.: A Survey on Policy Search for Robotics. Now Publishers Inc. (2013)
6. Guestrin, C., Lagoudakis, M., Parr, R.: Coordinated reinforcement learning. In: Proceedings of the 19th International conference on Machine Learning. pp. 227–234. Citeseer (2002)
7. Hu, Y., Gao, Y., An, B.: Multiagent reinforcement learning with unshared value functions. IEEE transactions on cybernetics **45**(4), 647–662 (2015)
8. Kok, J.R., Spaan, M.T., Vlassis, N., et al.: Multi-robot decision making using coordination graphs. In: Proceedings of the 11th International Conference on Advanced Robotics, ICAR. vol. 3, pp. 1124–1129 (2003)
9. Kok, J.R., Vlassis, N.: Using the max-plus algorithm for multiagent decision making in coordination graphs. In: Robot Soccer World Cup. pp. 1–12. Springer (2005)
10. Kok, J., Hoen, P., Bakker, B., Vlassis, N.: Utile coordination: Learning interdependencies among cooperative agents. In: Proceedings of the Symposium on Computational Intelligence and Games. pp. 29–36 (2005)
11. Kok, J., Vlassis, N.: Collaborative multiagent reinforcement learning by payoff propagation. The Journal of Machine Learning Research **7**, 1789–1828 (2006)

12. Kuyer, L., Whiteson, S., Bakker, B., Vlassis, N.: Multiagent reinforcement learning for urban traffic control using coordination graphs. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 656–671. Springer (2008)
13. Lagoudakis, M.G., Parr, R.: Least-squares policy iteration. Journal of Machine Learning Research **4**(6), 1107–1149 (2003)
14. Szolnoki, A., Wang, Z., Perc, M.: Wisdom of groups promotes cooperation in evolutionary social dilemmas. Scientific Reports **2**, 576 (2012)
15. Watkins, C., Dayan, P.: Q-learning. Machine Learning **8**(3), 279–292 (May 1992)
16. Wiering, M., Van Otterlo, M.: Reinforcement learning. Adaptation, Learning, and Optimization **12** (2012)
17. Yu, C., Zhang, M., Ren, F., Tan, G.: Multiagent learning of coordination in loosely coupled multiagent systems. IEEE Transactions on Cybernetics **45**(12), 2853–2867 (2015)
18. Yu, C., Lv, H., Ren, F., Bao, H., Hao, J.: Hierarchical learning for emergence of social norms in networked multiagent systems. In: Australasian Joint Conference on Artificial Intelligence. pp. 630–643. Springer (2015)
19. Yu, C., Lv, H., Sen, S., Hao, J., Ren, F., Liu, R.: An adaptive learning framework for efficient emergence of social norms. In: Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems. pp. 1307–1308 (2016)
20. Yu, C., Tan, G., Lv, H., Wang, Z., Meng, J., Hao, J., Ren, F.: Modelling adaptive learning behaviours for consensus formation in human societies. Scientific reports **6**, 27626 (2016)
21. Yu, C., Wang, D., Ren, J., Ge, H., Sun, L.: Decentralized multiagent reinforcement learning for efficient robotic control by coordination graphs. In: Pacific Rim International Conference on Artificial Intelligence. pp. 191–203. Springer (2018)
22. Yu, C., Wang, X., Xu, X., Zhang, M., Ge, H., Ren, J., Sun, L., Chen, B., Tan, G.: Distributed multiagent coordinated learning for autonomous driving in highways based on dynamic coordination graphs. IEEE Transactions on Intelligent Transportation Systems, doi: 10.1109/TITS.2019.2893683 (2019)