# Coordinated Multi-Agent Learning for Decentralized POMDPs

Chongjie Zhang
Computer Science Dept.
University of Massachusetts
Amherst, MA 01002, US
chongjie@cs.umass.edu

Victor Lesser
Computer Science Dept.
University of Massachusetts
Amherst, MA 01002, US
lesser@cs.umass.edu

## ABSTRACT

In many multi-agent applications such as distributed sensor nets, a network of agents act collaboratively under uncertainty and local interactions. Networked Distributed POMDP (ND-POMDP) provides a framework to model such cooperative multi-agent decision making. Existing work on ND-POMDPs has focused on offline techniques that require accurate models, which are usually costly to obtain in practice. This paper presents a model-free, scalable learning approach that synthesizes multi-agent reinforcement learning (MARL) and distributed constraint optimization (DCOP). By exploiting structured interaction in ND-POMDPs, our approach distributes the learning of the joint policy and employs DCOP techniques to coordinate distributed learning to ensure the global learning performance. Our approach can learn a globally optimal policy for ND-POMDPs with a property called *groupwise observability*. Experimental results show that, with communication during learning and execution, our approach significantly outperforms the nearly-optimal non-communication policies computed offline.

## INTRODUCTION

Decentralized partially observable MDP (DEC-POMDP) provides a powerful framework for modeling cooperative multi-agent decision making problems under uncertainty. Due to the intractability of optimally solving general DEC-POMDPs, research has focused on restricted versions of DEC-POMDP that are easier to solve yet rich enough to represent many practical applications. Networked Distributed POMDP (ND-POMDP) [1] is one such model that is inspired by a real-world sensor network coordination problem [2]. ND-POMDP assumes transition and observation independence and locality of interaction.

A rich portfolio of algorithms have been developed for solving ND-POMDPs [1, 3, 4]. One good feature of these techniques is that, although computing policies is centralized or requires extensive communication, executing computed policies does not require explicit communication. However, this feature may prevent agents from better coordination during execution when communication is allowed. In fact, in many practical applications, communications (at least between neighboring agents) are necessary for agents to perform tasks. For example, for target tracking in sensor networks, agents need to fuse their observations and actions to determine sensing results. The work [5] introduced communication in ND-POMDPs to periodically synchronize the belief state and extended existing algorithms to obtain policies with longer horizons. However, extensive communication is required for global synchronization, which is not scalable. In addition, all these algorithms for ND-POMDPs are offline techniques and require accurate models of the environment, which are usually costly to obtain in practice.

In this paper, we present a model-free, scalable learning approach to developing policies for ND-POMDPs. Our approach synthesizes multi-agent reinforcement learning (MARL) and distributed constraint optimization (DCOP). By exploiting locality of interactions in ND-POMDPs, our approach factors a global joint action-value function and distributes the learning of the joint policy, which potentially scales up the learning to large-scale ND-POMDPs. Using communication between neighboring agents, our approach employs DCOP techniques to coordinate distributed learning to ensure the global performance. Coordinated reinforcement learning based on coordination graphs [6] has been explored in [7, 8] for factored MDPs. In contrast to these previous work, in this paper, we explore coordinated multi-agent reinforcement learning in a principled way in ND-POMDPs and prove that our coordinated learning approach can learn the globally optimal policy for ND-POMDPs with a property, called *groupwise observability*. In addition, we also demonstrate that a max-sum algorithm [9] can be used for an approximate solution to our distributed coordination problem in learning, which requires limited communication overhead (typically scaling linearly with the number of agents) and computation. This DCOP algorithm can be readily implemented as an anytime algorithm to trade off solution quality and cost of computation and communication. Unlike the message-passing algorithm in [8], this algorithm can be directly used for coordinating interactions involving more than two agents. Our previous work [10, 11] presented a general supervisory framework for coordinating MARL, but did not provide a general coordination algorithm. In this paper, we demonstrate that DCOP algorithms can be used as general techniques for coordinating MARL in ND-POMDPs. Experimental results show that, even in ND-POMDPs without groupwise observability, our approach scales to larger domains and performs significantly better and with orders of magnitude time savings (in the offline mode) over the previous best offline algorithm. Note that, as our approach needs communication during execution, a direct comparison among approaches is not appropriate. However, the offline results do provide a way to evaluate our approach by providing a baseline (i.e., nearly-optimal performance without communication).

## BACKGROUND

This section briefly introduces an illustrative problem in the sensor network domain, the ND-POMDP model, and basic learning

*The Seventh Annual Workshop on Multiagent Sequential Decision-Making Under Uncertainty* (*MSDM-2012*), held in conjunction with *AAMAS*, June 2012, in Valencia, Spain.


**Figure 1: A 4-chain sensor configuration**

approaches.

## Illustrative Domain

This illustrative problem is motivated by a real-world challenge, where a network of agents (sensors) are used to track targets. Figure 1 shows a specific problem instance consisting of four sensors. Here, each sensor node can scan in one of four directions: North, South, East or West. To track a target and obtain the associated reward, two sensors with overlapping scanning areas must coordinate by scanning the same area simultaneously. For example, sensor1 needs to scan East and sensor2 needs to scan West simultaneously to track a target in location1. Thus, sensors have to act in a coordinated fashion. The movement of targets is unaffected by sensor agents. Sensors have imperfect observability of the target, so there can be false positive and negative observations. Sensors receive a reward on successfully tracking a target, and they incur a cost, when they either scan an area in an uncoordinated fashion or when the target is absent.

## Networked Distributed POMDPs

Observe that sensors in this domain are mostly independent. Their state transitions, given the target location and the observations, are independent of the actions of the other agents. The only dependence arises from the fact that two agents must coordinate by scanning the same region to track a target. This dependence can be translated into a joint reward function. Such dependence is usually localized among a few agents (only two agents in this sensor network problem). The ND-POMDP model [1] was introduced to express such a type of interactions.

DEFINITION 1. *An* **ND-POMDP** *is defined by the tuple* $\langle I, S, A, \Omega, P, O, R, b \rangle$*, where*

- $I$ *is a set of agent indices.*
- $S$ $= \times_{i \in I} S_i \times S_u$. $S_i$ *refers to the local state of agent* $i$. $S_u$ *refers to a set of uncontrollable states that are independent of the actions of the agents. In the sensor network example,* $S_i$ *is empty, while* $S_u$ *corresponds to the set of locations where targets can be present.*
- $A$ $= \times_{i \in I} A_i$*, where* $A_i$ *is the set of actions for agent* $i$*. For the sensor network example,* $A_1 = \{N, W, E, S, Off\}$.
- $\Omega$ $= \times_{i \in I} \Omega_i$ *is the joint observation set.*
- $P$ $P(s'|s,a) = P_u(s'_u|s_u) \cdot \prod_{i \in I} P_i(s'_i|s_i, s_u, a_i)$*, where* $a = \langle a_1, \ldots, a_n \rangle$ *is the joint action performed in joint state* $s = \langle s_u, s_1, \ldots, s_n \rangle$ *resulting in joint state* $s' = \langle s'_u, s'_1, \ldots, s'_n \rangle$*. (This models the transition independence.)*
- $O$ $O(\omega|s,a) = \prod_{i \in I} O_i(\omega_i|s_i, s_u, a_i)$*, where* $s$ *is the joint state resulting after taking joint action* $a$ *and receiving joint observation* $\omega$*. (This models the observation independence.)*
- $R$ $R(s,a) = \sum_l R_l(s_l, s_u, a_l)$*. The reward function is decomposable among sub groups of agents referred by* $l$*. If* $k = |l|$ *agents* $i_1, \ldots, i_k$ *are involved in a particular sub group* $l$*, then* $s_l$ *denotes the state of group* $l$*, i.e.,* $\langle s_{l1}, \ldots, s_{lk} \rangle$*. Similarly,* $a_l = \langle a_{l1}, \ldots, a_{lk} \rangle$*. In the sensor domain, the reward function is expressed as the sum of rewards between sensor agents that have overlapping areas* $(k = 2)$ *and the reward functions for an individual agent's cost for sensing*$(k = 1)$*. Based on the reward function, an interaction hypergraph* $G = (I, E)$ *can be constructed, where* $I$ *is a vertex (i.e., agent) set and* $E$ *is a set of hyperlinks. A hyperlink* $l \in E$ *connects the subset of agents which form the reward component* $R_l$*. Note that this interaction hypergraph will be used to develop our learning approach in later sections.*
- $b$ $b = (b_u, b_1, \ldots, b_n)$ *is the initial belief (or distribution) for joint state* $s = \langle s_u, s_1, \ldots, s_n \rangle \in S$ *and* $b(s) = b(s_u) \cdot \prod_{i \in I} b_i(s_i)$*, where* $b_u$ *and* $b_i$ *are the initial distribution over* $S_u$ *and* $S_i$*.*

The goal for ND-POMDPs is to compute a joint policy $\pi$ that maximizes the total expected reward of all agents over a finite horizon $T$ starting from $b$. Without communication, agents can only act based on its local observations. In this case, a joint policy $\pi$ is defined by $\langle \pi_1, \ldots, \pi_n \rangle$, where $\pi_i$ refers to the individual policy of agent $i$ that maps its history of observations to an action $a_i \in A_i$. If communication is allowed, a joint policy $\pi$ can also be defined by one policy, called *global policy*, that maps from a history of joint observations to a joint action $a \in A$. This is because agents can exchange their observations and select actions based on joint observations. Obviously, the optimal global policy inherently performs better than the optimal set of individual policies. In this paper, we assume agents can communicate (at least with their neighbors) during the execution time and focus on representing and learning the optimal *global policy* in a scalable way.

## Basic Learning Approaches

To learn the joint policy, we need to define Q-function (or Q-value function). Let Q-function $Q(\vec{h}, a)$ represent the expected reward of doing joint action $a$ with history $\vec{h}$ of joint observations and actions and behaving optimally from then on. The globally joint policy $\pi$ can be derived from $Q(\vec{h}, a)$ by setting $\pi(\vec{h}) = \text{argmax}_{a \in A} Q(\vec{h}, a)$.

In principle, we can directly estimate $Q(\vec{h}, a)$ by using standard single-agent Q-learning:

$$Q(\vec{h}^t, a^t) = (1 - \alpha) Q(\vec{h}^t, a^t) + \alpha [r^t + \gamma \max_a Q(\vec{h}^{t+1}, a)] \quad (1)$$

where $\alpha \in (0, 1)$ is the learning rate, $r^t$ is the immediate reward of doing $a^t$ for observation history $\vec{h}^t$, $\gamma \in [0, 1]$ is the discount factor, which is usually set to 1 for a finite horizon. We call this approach *globally joint learning*. Although this approach leads to an optimal policy, it is practically intractable, because the policy space is exponential in the number of agents and the agents might not have access to the needed information (i.e., observations, actions, and rewards of all other agents) for learning and selecting actions.

At the other extreme, we can have the *independent learning* approach [12] in which agents ignore the actions and rewards of the other agents, and concurrently learn their own action-value functions solely based on their local observations and rewards. To provide local rewards in ND-POMDPs, we can split the reward component $R_l$ evenly among agents in group $l$. This approach is distributed, results in big storage and computational savings in the policy space, and does not require communication during learning and execution. However, this approach lacks coordination and

might lead to oscillations or converge to local optimal policies. For example, in Figure 1, if location1, location2, and location3 always have targets with sensing reward 50, 60, and 50, respectively, then, by using independent learning approach, sensor2 and sensor3 will learn to always sense location2, which is locally optimal with average expected reward 60. However, the optimal policy is that sensor1 and sensor2 always sense location1 and sensor3 and sensor4 always sense location3, whose global expected reward is 100. Therefore, some form of coordination is needed in order to learn the globally optimal policy.

# COORDINATED MULTI-AGENT REINFORCEMENT LEARNING

As discussed in the previous section, directly learning the globally joint policy in a centralized way is infeasible from a practical perspective, while independent learning is a distributed, scalable approach, but may yield poor global performance. In this section, we present a coordinated multi-agent learning approach for ND-POMDPs that attempts to achieve both scalability and optimality (or near-optimality). This approach distributes the learning by exploiting structured interactions in ND-POMDPs and coordinates distributed learning to ensure the global performance.

Our approach optimizes a decomposable Q-function $\hat{Q}(\vec{h}, a)$ that is used to approximate the global Q-function $Q(\vec{h}, a)$. This Q-function $\hat{Q}(\vec{h}, a)$ is defined as a sum of smaller local Q-functions based on hyperlinks in the interaction hypergraph of ND-POMDPs, that is,

$$\hat{Q}(\vec{h}, a) = \sum_{l \in E} Q_l(\vec{h}_l, a_l), \tag{2}$$

where $Q_l(\vec{h}_l, a_l)$ is the expected reward for agents on hyperlink $l$ by doing joint action $a_l^t$ at joint history $\vec{h}_l^t$ and behaving *globally* optimally from then on in respect to maximizing $\hat{Q}(\vec{h}, a)$. We will show in the next subsection that this approximation becomes exact for ND-POMDPs with a property called *groupwise observability*, which will lead to the theoretical result of optimality for our approach. In fact, this approximation is reasonable for general ND-POMDPs. This is because the global reward in ND-POMDPs is the sum of local rewards of groups defined on hyperlinks in the interaction hypergraph, and, as a result, $Q(\vec{h}, a)$ and $\hat{Q}(\vec{h}, a)$ are strongly positively correlated. Therefore, maximizing $\hat{Q}(\vec{h}, a)$ can potentially optimize $Q(\vec{h}, a)$. Our experimental results will verify this hypothesis on ND-POMDPs without the groupwise observability property.

Q-learning is used to learn the optimal $\hat{Q}(\vec{h}, a)$. With the decomposition in (2), the global Q-learning update rule in (1) can be rewritten as

$$\begin{aligned} \textstyle\sum_{l \in E} Q_l(\vec{h}_l^t, a_l^t) = \quad & (1-\alpha) \textstyle\sum_{l \in E} Q_l(\vec{h}_l^t, a_l^t) \\ & + \alpha[\textstyle\sum_{l \in E} r_l^t + \gamma \max_a \hat{Q}(\vec{h}^{t+1}, a)] \end{aligned} \tag{3}$$

Note that the discounted future reward, $\max_a \hat{Q}(\vec{h}^{t+1}, a)$, can not be directly written as the sum of local discounted future rewards, because it depends on the joint action that maximizes the global value. Fortunately, we can accomplish this by defining the joint action $a^* = \text{argmax}_a \hat{Q}(\vec{h}^{t+1}, a)$ and $\max_a \hat{Q}(\vec{h}^{t+1}, a) = \hat{Q}(\vec{h}^{t+1}, a^*) = \sum_{l \in E} Q_l(\vec{h}_l^{t+1}, a_l^*)$. We are now able to decompose all terms in (3) and write the update rule for each group $l$:

$$Q_l(\vec{h}_l^t, a_l^t) = (1-\alpha) Q_l(\vec{h}_l^t, a_l^t) + \alpha[r_l^t + \gamma Q_l(\vec{h}_l^{t+1}, a_l^*)] \tag{4}$$

Similar to Sparse Cooperative Q-Learning [8], update rule in (4) is based on local reward and Q-function, except for $a_l^*$. Note that the local contribution $Q_l(\vec{h}_l^{t+1}, a_l^*)$ of group $l$ to the global action value might be lower than $\max_{a_l} Q_l(\vec{h}_l^{t+1}, a_l)$, the maximizing value of its local Q-function, because it is unaware of the dependencies among groups. We will use distributed constraint optimization (DCOP) techniques to compute $a_l^*$, which will be discussed later. Update rule in (4) is different from coordinated reinforcement learning approach in [7], where local Q-function update depends on the global reward signal and the global Q-value, which are not usually specifically tailored to local behaviors, thus resulting in slower learning convergence.

Using update rule in (4), our approach distributes the learning of the global function $\hat{Q}$ among groups. Our approach assumes that each group has a delegate agent (which can be chosen arbitrarily from a group) that learns $Q_l$ on behalf of the group. The basic learning process is as follows. During each learning cycle $t$, after executing actions $a_l^t$, agents in group $l$ receive and transmit their observations to the delegate agent of their group and the delegate agent receives its group reward signal $r_l^t$. Using its updated observation history $\vec{h}_l^{t+1}$, the delegate agent then computes the next best action $a_l*$ for $\vec{h}_l^{t+1}$ by using a DCOP technique and updates its Q-function $Q_l$ using rule (4). Finally, it distributes the next actions to its group members, which will be $a_l*$ or some exploration actions.

The learned global Q-function is distributedly represented by local Q-functions of delegate agents. As a result, during execution, agents' action selections are computed online in a distributed manner by a DCOP algorithm from local Q-fuctions. Note that local Q-function $Q_l(\vec{h}_l^t, a_l^t)$ is defined on the observation history of group $l$, which scales exponentially with the horizon. To deal with a large horizon, one approach is to use a fixed-size window of observations, as we did in our experiments. Other more sophisticated approaches (i.e., utile suffix memory [13]) for dealing with partial observability can also be used with our approach.

In next two subsections, we will formally analyze the optimality of our approach and discuss how to compute joint action selections for learning or execution.

## Optimality Analysis

In this section, we first define a property for ND-POMDPs, called *groupwise observability*, and then prove that our approach can learn an optimal policy for ND-POMDPs with this property.

DEFINITION 2. *An ND-POMDP is said to have* **groupwise observability** *if, for all $l \in E$, the set of observations $\omega_l = \langle \omega_{l1}, \ldots, \omega_{lk} \rangle$ made by agents on hyperlink $l$ together fully determine the current uncontrolled state, that is, if $\forall l \forall \omega_l \exists s_u : \Pr(s_u | \omega_l) = 1$.*

Note that this property does not imply that agents can observe their local states or states of other agents. It does imply that, for each agent $i \in l$, $P_i(s_i' | s_i, s_u, a_i, \omega_l) = P_i(s_i' | s_i, a_i, \omega_l)$ and $O_i(\omega_i | s_i, s_u, a_i, \omega_l) = O_i(\omega_i | s_i, a_i, \omega_l)$, which means, given joint observation $\omega_l$, observation and transition of agent $i$ on $l$ are completely independent of observations and actions of other agents, and, as a result, its local belief update only depends on its local action and observation. This further implies that, in ND-POMDPs with groupwise observability, the local belief of agent $i \in l$ can be fully determined by its initial local state and the history of joint observations and actions of agents on $l$.

The theoretical result of optimality of our approach is as follows.

THEOREM 1. *For ND-POMDPs with groupwise observability, under basic assumption of Q-learning and by using update rule (4), $Q_l(\vec{h}_l, a_l)$ will converge to the optimal $Q_l^*(\vec{h}_l, a_l)$, for all $l \in E$,*

*and the policy* $\pi^*(\vec{h}) = \text{argmax}_a \sum_{l \in E} Q_l^*(\vec{h}_l, a_l)$ *is globally optimal.*

The proof for this theorem can be conducted by showing that Q-function $\hat{Q}$ defined in Equation (2) is exactly the same as the objective function $Q$ of ND-POMDPs. This is because, if the approximation of $\hat{Q}$ is exact, then our coordinated learning approach described above is essentially a distributed version of update rule (1) that uses Q-learning, which leads to the global optimal $Q^*(\vec{h}, a)$. The exactness of this approximation for ND-POMDPs with groupwise observability will be shown by Proposition 2.

Our proof first defines a Q-function with state variables, then shows it is decomposable, and finally uses this result to prove the approximation of $\hat{Q}$ to $Q$ is exact for ND-POMDPs with groupwise observability. To simplify the equations, we introduce some abbreviations:

$$
\begin{aligned}
p_i^t &\equiv P_i(s_i^{t+1}|s_i^t, s_u^t, a_i^t) \cdot O_i(\omega_i^{t+1}|s_i^{t+1}, s_u^{t+1}, a_i^t) \\
p_u^t &\equiv P_u(s_u^{t+1}|s_u^t) \\
r_l^t &\equiv R_l(s_l, s_u, a_l) \\
Q^t &\equiv Q^t(s^t, \vec{h}^t, a^t) \\
Q^{t*} &\equiv \max_a Q^t(s^t, \vec{h}^t, a) \\
Q_l^t &\equiv Q_l^t(s_l^t, s_u^t, \vec{h_l}^t, a_l^t)
\end{aligned}
$$

The global Q-function $Q(s^t, \vec{h}^t, a^t)$ with state will satisfy the Bellman equation:

$$
Q(s^t, \vec{h}^t, a^t) = R(s^t, a^t) + \gamma \sum_{s^{t+1}, w^{t+1}} p_u^t p_1^t \dots p_n^t Q^{(t+1)*},
$$

where $\vec{h}^{t+1}$ is $\vec{h}^t$ appended by $\langle a^t, \omega^{t+1} \rangle$.

Let $b^t$ be the belief state at time $t$. As $b^t$ is fully determined by the initial belief $b$ and history $\vec{h}^t$ of joint observations and actions, we have

$$
Q(\vec{h}^t, a^t) = \sum_{s \in S} b^t(s) Q(s^t, \vec{h}^t, a^t). \tag{5}
$$

Similarly, we define a Q-function for each hyperlink $l$:

$$
Q_l(s_l^t, s_u^t, \vec{h_l}^t, a^t) = r_l^t + \gamma \sum_{s_l^{t+1}, \omega_l^{t+1}} p_u^t p_{l1}^t \dots p_{lk}^t Q_l^{t+1*},
$$

where $\vec{h_l}^{t+1}$ is $\vec{h_l}^t$ appended by $\langle a_l^t, \omega_l^{t+1} \rangle$ and $Q_l^{t+1*}$ denotes $Q_l(s_l^{t+1}, \vec{h_l}^{t+1}, a_l^*)$, where $a_l^*$ is the *globally* optimal joint action taken by agents on $l$ in the next global state and history of joint observations and actions of all agents.

For ND-POMDPs with groupwise observability, as $b_u^t(s_u)$ is fully determined by history $\vec{h_l}^t$ of joint observations and actions, and, for $i \in l$, $b_i^t(s_i)$ is fully determined by the initial belief $b_i(s_i)$ and history $\vec{h_l}^t$, we then have

$$
Q(\vec{h_l}^t, a_l^t) = \sum_{s_l, s_u} b_l^t(s_u, s_l) Q_l(s_l, s_u, \vec{h_l}^t, a_l^t). \tag{6}
$$

PROPOSITION 1. *In ND-POMDPs, the global function* $Q^t(s^t, \vec{h}^t, a^t)$ *for any finite horizon* $T$ *is decomposable, that is,*

$$
Q^t(s^t, \vec{h}^t, a^t) = \sum_{l \in E} Q_l^t(s_l^t, s_u^t, \vec{h_l}^t, a_l^t). \tag{7}
$$

PROOF. Proof is by mathematical induction. Proposition holds for $t = T - 1$ because $r^t = \sum_{l \in E} r_l^t$ and there is no future reward. Assume it holds for $t$ where $1 \le t \le T - 1$, that is, $Q^t = \sum_{l \in E} Q_l^t$.

Now let us show that proposition holds for $t - 1$.

$$
\begin{aligned}
Q^{t-1} &= R(s^{t-1}, a^{t-1}) + \gamma \sum_{s^t, w^t} p_u^{t-1} p_1^{t-1} \dots p_n^{t-1} Q^{t*} \\
&= \sum_{l \in E} r_l^{t-1} + \gamma \sum_{s^t, w^t} p_u^{t-1} p_1^{t-1} \dots p_n^{t-1} \sum_{l \in E} Q_l^{t*} \\
&= \sum_{l \in E} [r_l^{t-1} + \gamma \sum_{s_l^t, s_u^t, w_l^t} p_u^{t-1} p_{11}^{t-1} \dots p_{lk}^{t-1} Q_l^{t*}] \\
&= \sum_{l \in E} Q_l^{t-1}
\end{aligned}
$$

□

Based on Proposition 1, Equation 5 and 6, we can show an exact decomposition of the Q-function without state.

PROPOSITION 2. *In ND-POMDPs with groupwise obserbability, the global Q-value function* $Q^t(\vec{h}^t, a^t)$ *for any finite horizon* $T$ *is decomposable, that is,*

$$
Q^t(\vec{h}^t, a^t) = \sum_{l \in E} Q_l^t(\vec{h_l}^t, a_l^t). \tag{8}
$$

PROOF.

$$
\begin{aligned}
Q(\vec{h}^t, a^t) &= \sum_{s_u, s_1, \dots, s_n} b_u^t(s_u) b_1^t(s_1) \dots b_n^t(s_n) \cdot \\
&\quad \sum_{l \in E} Q_l^t(s_l, s_u, \vec{h_l}^t, a_l^t) \\
&= \sum_{l \in E} \sum_{s_l, s_u} b_l^t(s_u, s_l) Q_l^t(s_l, s_u, \vec{h_l}^t, a_l^t) \\
&= \sum_{l \in E} Q_l^t(\vec{h_l}^t, a_l^t).
\end{aligned}
$$

□

This proposition completes the proof of Theorem 1. Note that the work [1] also showed that the value function of a given joint policy can be decomposed and the work [14] generalized its result to factored Dec-POMDPs. However, unlike our Q-value function decomposition, decomposition components of their value function are defined on a joint policy, so their results are not directly applicable to the proof of Theorem 1.

## Optimal Joint Action Selection

Our learning approach requires computing the joint action that maximizes the global Q-value function for updating local Q-functions or for acting during execution. We can formulate this problem as a DCOP, which is defined by a set of discrete variables $a = \{a_1, \dots, a_n\}$, where $a_i \in A_i$ is controlled by agent $i$ and represents its action choice, and a set of functions $Q = \{Q_l | l \in E\}$, where $Q_l$ is the Q-value function for hyperlink $l$. Note that history $\vec{h}$ is fixed for every computation, so we will ignore it in the following discussion and denote $Q_l(\vec{h}, a_l)$ by $Q_l(a_l)$. The goal is to find the joint action $a^*$, such that the global Q-value function, the sum of all Q-functions, is maximized, that is, $a^* = \text{argmax}_a \sum_{l \in E} Q_l(a_l)$. We can represent this DCOP as a factor graph by creating a node for each variable and for each function and connecting a function node to a variable node if the corresponding function is dependent upon that variable. The resulting graph is bipartite.

A variable elimination algorithm [6] can be used to compute an optimal solution for this DCOP, but it requires extensive communication and computation (scaling exponentially with the induced width of the agent interaction graph). In this paper, we investigate the max-sum algorithm [9] for an approximate solution, which requires much less communication and computation and can be readily implemented as an anytime algorithm to trade off the quality and efficiency of computing joint actions. Unlike the max-plus algorithm in [8], this algorithm can be directly used for coordinating interactions involving more than two agents.

The max-sum algorithm operates directly on the factor graph, and does so by specifying the messages that should be passed from variable to function nodes, and from function nodes to variable nodes, which are defined as follows:

- **Message from variable node $i$ to function node $l$:**

$$q_{i\to l}(a_i) = \sum_{g\in\mathcal{F}_i\setminus l} r_{g\to i}(a_i) + c_{il}$$

where $\mathcal{F}_i$ is a vector of function indexes, indicating which function nodes are connected to variable node $i$, and $c_{il}$ is a normalizing constant to prevent the messages from increasing endlessly in cyclic graphs.

- **Message from function node $l$ to variable node $i$:**

$$r_{l\to i}(a_i) = \max_{\mathbf{a}_l\setminus a_i}[Q_l(\mathbf{a}_l) + \sum_{g\in\mathcal{V}_l\setminus i} q_{g\to l}(a_g)]$$

where $\mathcal{V}_l$ is a vector of variable indexes, indicating which variable nodes are connected to function node $l$ and $\mathbf{a}_l\setminus a_i = \{a_g : g \in \mathcal{V}_l\setminus i\}$.

Here variable node $i$ is agent $i$ who needs to select its action and function node $l$ is the delegate agent of hyperlink $l$ that hosts the Q-value function $Q_l$. If the factor graph is cycle-free, the algorithm is guaranteed to converge to the optimal global solution such that each agent $i$ can find its optimal action $a_i^*$ by locally calculating $a_i^* = \text{argmax}_{a_i} z_i(a_i)$, where $z_i(a_i) = \sum_{g\in\mathcal{F}_i} r_{g\to i}(a_i)$. Otherwise, there is no guarantee of convergence. However, extensive empirical results show that, even in this case, the algorithm frequently provides good solutions. Before convergence, the value $z_i(a_i)$ of agent $i$ calculated from incoming messages is actually an approximation of the exact value of action $a_i$ given other agents act optimally. Therefore, the max-sum algorithm can be implemented as an anytime algorithm by controlling the number of rounds of passing messages, which will trade off the quality and efficiency (or communication cost) of the action selection. In addition, the max-sum algorithm is essentially distributed. Its messages are small (linearly scaling with the maximum number of actions of agents), the number of messages typically varies linearly with the number of agents and hyperlinks, and its computational complexity scales exponentially with the maximum size of hyperlinks (which typically is much less than the total number of agents).

# EXPERIMENTS

To evaluate our coordinated learning (CL) approach in general ND-POMDPs, we experimented it in the illustrative sensor network domain, which does not have the groupwise observability property. We compared CL with the independent learning (IL) approach (described in the Background Section) and CBDP [4], one of the most efficient algorithms for ND-POMDPs. We conducted experiments with configurations shown in Figure 2. The first three configurations are introduced in [3], but we changed their initial beliefs to an uniform distribution over ten states to increase problem difficulty. The 25-grid sensor network has two targets with the same sensing rewards as 15-3D, but has a larger state space and longer target paths.

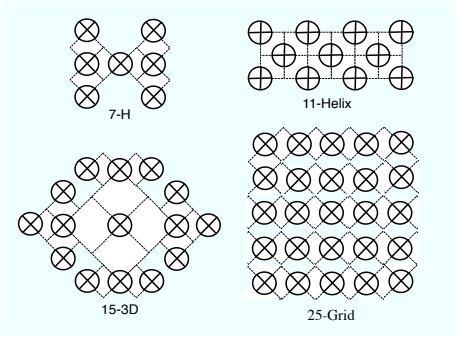


**Figure 2: Sensor network configurations**

Since both CL and IL are model-free, we develop a simulator for ND-POMDPs to learn and evaluate policies. The evaluation process is as follows: for each ND-POMDP, we use CBDP to solve it and get its joint policy, then run both learning approaches in a simulator for that ND-POMDP, whose learning time is set to some ratio of CBDP's computation time, and, finally evaluate learned policies and CBDP's policy in the simulator. The solution quality for each horizon is indicated by the expected global reward for that horizon. Solution quality is computed over 10000 simulation runs. Results are then averaged over 10 experiments and the deviation is computed, which is very small (under 5) and not shown properly in the following figures. The learning rate $\alpha$ is set to 0.001 and discount factor $\gamma = 1$. Both learning approaches learned policies that map fixed-windows of observations (with size $\leq 4$) to an action even for scanarios with horizon greater than 5. To trade off the speed and solution quality, we restricted the max-sum algorithm passing messages at most 4 rounds for each joint action computation (except for experiments of controlling communication).

Figure 3 (a) shows the solution quality of CL and IL with different learning time on the configuration 15-3D with horizon $T = 10$. The configuration 15-3D is the most complex problem instance for CBDP. The $x$ axis represents the ratio of learning time to CBDP's computation time, which is plotted with a logarithmic scale. The performance of both CL and IL generally increases with more training time. We observe that CL can learn policies, whose performance surpasses that of CBDP's policy, with learning time two orders of magnitude less than CBDP's computation time. However, IL performs much worse than CL and CBDP. One reason is that, as we have discussed, IL can only converge to local optima, which is far away from the global optimal solution on the configuration 15-3D. This result actually illustrates the importance of the coordination during learning and execution. Another reason is that IL (and CL) uses fixed-window policy that maps up to 4 observations to an action, while CBDP's policies with horizon $T = 10$ maps up to 9 observations to an action. We did observe that IL could perform comparably or better than CBDP on smaller problems with small horizon (e.g., one the domain 11-Helix with 5 horizon).

Figure 3 (b) shows the solution quality over a range of horizons on the configuration 15-3D. We can see that the solution quality of CL linearly increases with the horizon size, whose increase rate is greater than CBDP. This indicates that CL can potentially scale better than CBDP with the horizon size. Figure 3 (c) shows the solution quality on other configurations, where 15-Mod is the modified

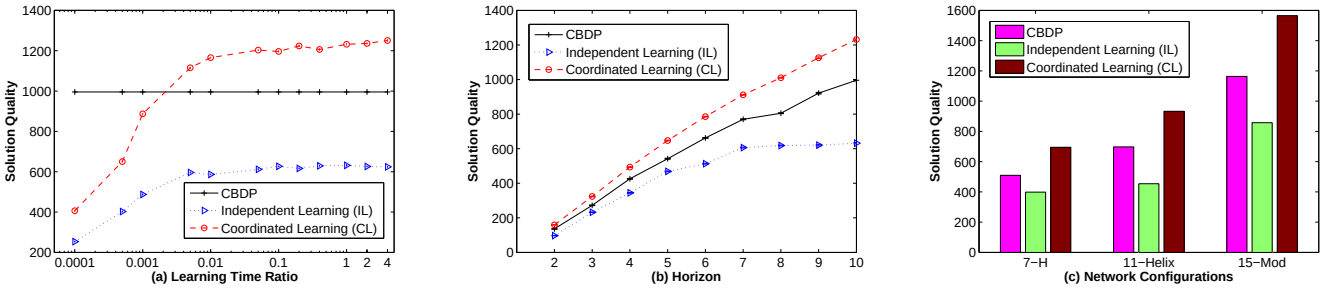


**Figure 3: Solution quality over (a) different ratios of learning time of IL and CL to CBDP's policy computation time on 15-3D with horizon $T = 10$, (b) over different horizons on 15-3D, and (c) different network configurations with $T = 10$. Note that IL and CL in (b) and (c) use the same learning time as CBDP's policy computation time.**

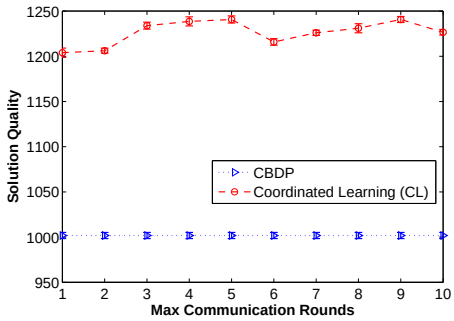


**Figure 4: Trade-off of solution quality and communication**

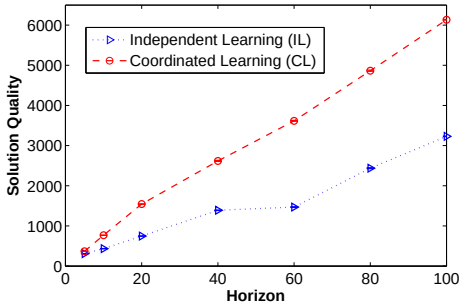


**Figure 5: Solution quality for a range of horizons on 25-grid**

version of 15-3D with different target paths. Consistent with results on 15-3D, CL performs best, then CBDP, and finally IL.

By controlling the maximum round of message passing between agents and their group delegates for computing joint actions, we can trade off solution quality and cost of communication and computation. Figure 4 show the solution quality of CL over different maximum rounds of message passing on the domain 15-3D with horizon 10 and the same learning time as CBDP's computation time. We can observe that CL still performs significantly better than CBDP, even when using only one-round message passing. Note that when using fixed learning time, more rounds of message passing do not necessarily yield better learning performance. This is because, although using more rounds of message passing computes better joint actions, it results in more communication and computation at each learning cycle and learning with less total cycles.

We also evaluated CL and IL on the 25-grid problem, where CBDP could not scale even to horizon 2. The learning time is set to 200 seconds for horizon 5 and linearly increases with the horizon. Figure 5 shows solution quality over horizons up to 100. The solution quality of CL almost doubled that of IL and increases linearly with the horizon.

## SUMMARY

We have introduced a model-free, multi-agent learning approach for ND-POMDPs. This approach decomposes and distributes the learning of the optimal global joint policy by exploiting its structured interactions through a decomposable reward function and independence among agents. Distributed learning is coordinated through joint action selection computed by distributed constraint optimization (DCOP) techniques, which ensure the optimality of the learning for ND-POMDPs with *groupwise observability*. By exploiting the property of locality of interactions in ND-POMDPs, the learning complexity potentially scales linearly with the number of agents. To trade off solution quality and communication and computation efficiency, a max-sum algorithm is used to compute an approximate solution for our DCOP. Experimental results show that, by exploiting extra communication during learning and execution, this approach significantly outperforms off-line construction of nearly-optimal no-communication policies.

## ACKNOWLEDGEMENTS

This work is supported by the National Science Foundation under Award No. IIS-1116078. Any opinions, ?ndings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily re?ect those of the National Science Foundation.

## REFERENCES

[1] Pradeep Varakantham, Milind Tambe, and Makoto Yokoo. Networked distributed pomdps: A synthesis of distributed constraint optimization and pomdps. In *AAAI*, pages 133–139, 2005.

[2] V. Lesser, C. Ortiz, and M. Tambe, editors. *Distributed Sensor Networks: A Multiagent Perspective (Edited book)*, volume 9. Kluwer Academic Publishers, 2003.

[3] Janusz Marecki, Tapana Gupta, Pradeep Varakantham, Milind Tambe, and Makoto Yokoo. Not all agents are equal: Scaling up distributed pomdps for agent networks. In *AAMAS*, pages 485–492, 2008.

[4] Akshat Kumar and Shlomo Zilberstein. Constraint-based dynamic programming for decentralized pomdps with structured interactions. In *AAMAS*, 2009.

[5] M. Tasaki, Y. Yabu, Y. Iwanari, M. Yokoo, M. Tambe, J. Marecki, and P. Varakantham. Introducing communication in dis-pomdps with locality of interaction. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, volume 2, pages 169 –175, 2008.

[6] Carlos Guestrin, Daphne Koller, and Ronald Parr. Multiagent planning with factored mdps. In *NIPS-14*, pages 1523–1530, 2001.

[7] Carlos Guestrin, Michail G. Lagoudakis, and Ronald Parr. Coordinated reinforcement learning. In *ICML '02: Proceedings of the Nineteenth International Conference on Machine Learning*, pages 227–234, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

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

[9] Ruben Stranders, Alessandro Farinelli, Alex Rogers, and Nicholas R. Jennings. Decentralised coordination of mobile sensors using the max-sum algorithm. In *IJCAI*, pages 299–304, 2009.

[10] Chongjie Zhang, Sherief Abdallah, and Victor Lesser. Integrating organizational control into multi-agent learning. In *AAMAS'09*, 2009.

[11] Chongjie Zhang, Victor Lesser, and Sherief Abdallah. Self-organization for coordinating decentralized reinforcement learning. In *AAMAS'10*, 2010.

[12] Caroline Claus and Craig Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. In *AAAI'98*, pages 746–752. AAAI Press, 1998.

[13] R. Andrew Mccallum. Instance-based utile distinctions for reinforcement learning with hidden state. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 387–395. Morgan Kaufmann, 1995.

[14] Frans A. Oliehoek, Matthijs T. J. Spaan, Shimon Whiteson, and Nikos Vlassis. Exploiting locality of interaction in factored dec-pomdps. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems - Volume 1*, AAMAS '08, pages 517–524, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems.