Integrating Decision Sharing with Prediction in Decentralized Planning for Multi-Agent Coordination under Uncertainty

Minglong Li, **Wenjing Yang**, **Zhongxuan Cai**, **Shaowu Yang** and **Ji Wang**
State Key Laboratory of High Performance Computing, College of Computer,
National University of Defense Technology, China
{minglong_l, wjyang1088}@163.com, caizhongxuan1993@gmail.com, {shaowu.yang, wj}@nudt.edu.cn

Abstract

The performance of decentralized multi-agent systems tends to benefit from information sharing and its effective utilization. However, too much or unnecessary sharing may hinder the performance due to the delay, instability and additional overhead of communications. Aiming to a satisfiable coordination performance, one would prefer the cost of communications as less as possible. In this paper, we propose an approach to improve the sharing utilization by integrating information sharing with prediction in decentralized planning. We present a novel planning algorithm by combining decision sharing and prediction based on decentralized Monte Carlo Tree Search called Dec-MCTS-SP. Each agent grows a search tree guided by the rewards calculated by the joint actions, which can not only be sampled from the shared probability distributions over action sequences, but also be predicted by a sufficiently-accurate and computationally-cheap heuristics-based method. Besides, several policies including sparse and discounted UCT and DIY-bonus are leveraged for performance improvement. We have implemented Dec-MCTS-SP in the case study on multi-agent information gathering under threat and uncertainty, which is formulated as Decentralized Partially Observable Markov Decision Process (Dec-POMDP). The factored belief vectors are integrated into Dec-MCTS-SP to handle the uncertainty. Comparing with the random, auction-based algorithm and Dec-MCTS, the evaluation shows that Dec-MCTS-SP can reduce communication cost significantly while still achieving a surprisingly higher coordination performance.

1 Introduction

Communication is fundamental to coordinated behaviour, as robots need to develop decision strategies that take into account the actions of others [Best *et al.*, 2018]. Through the cooperative planning process, multi-robot systems become effective for complex real-world tasks, including cooperative localisation, target tracking, object recognition, exploration, surveillance, and environmental monitoring [Best *et al.*, 2019].

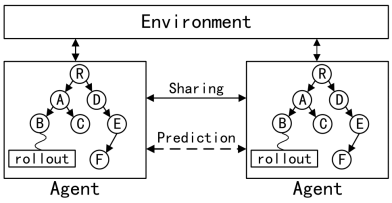

Figure 1: The interaction among the robots and the environment.

Full appreciation of the interactive nature of sequential decision making will enable us to better understand multi-agent coordination, and thus come up with more usable and effective solutions [Oliehoek, 2018], which means that the performance of the decentralized multi-agent systems tends to benefit from information sharing and its effective utilization. However, over communication may lead to large overheads, which hinders the suitability for realistic scenarios where communication resources are limited, the network is unreliable, or the signals are susceptible to interference from the other robots [Best *et al.*, 2018]. We hold the viewpoint that decision sharing can be integrated with prediction in decentralized planning to reduce communication cost significantly with equally-good or even higher coordination performance.

In this paper, we propose a novel decentralized planning approach by combining decision sharing and prediction in a more principled method. Our approach is based on the context of decentralized Monte Carlo Tree Search (Dec-MCTS) [Best *et al.*, 2019] , which is called Dec-MCTS-SP. Its intuition is in analogy with human, when several players cooperate to play soccer, they need not talk frequently to let others know what they are going to do the next moment. Instead, given an overall objective (to score a goal), the cooperation is always happening by prediction on others.

As a result, decision sharing through communication is critical but not everything for cooperation. In Dec-MCTS-SP, each agent grows a search tree guided by the rewards calculated by the joint actions as shown in Figure 1. The cooperation of Dec-MCTS-SP is composed of two features. One is through sampling from the shared probability distributions over action sequences by communication, and the other is predicted by a sufficiently-accurate and computationally-

cheap heuristics-based method. Besides, several policies are leveraged for further performance improvement including sparse and discounted UCT and DIY bonus. To validate Dec-MCTS-SP, we formulate a representative decentralized information gathering problem under uncertainty. It is based on decentralized partially observable Markov decision process (Dec-POMDP) [Bernstein *et al.*, 2002]. We use Dec-MCTS-SP to solve it and compare the performance with that of the state of the art.

In summary, our main contributions are as follows.

- We propose the framework that decision sharing can be integrated with the prediction for decentralized planning in a more principled way, and design Dec-MCTS-SP algorithm which combines these sharing and predicting cooperating methods in the context of Dec-MCTS.
- We incorporate the method of factored belief into Dec-MCTS-SP for empowering it to deal with uncertainty.
- The experimental results show the significant communication reductions with a surprisingly higher coordination performance, which outperforms those of random, auction-based algorithm and Dec-MCTS.

2 Problem Formulation

In a multi-agent information gathering problem, agents move and observe to gather information in the dynamic environment which may threaten their health. We represent the scenario as a graph. The targets with information and threat are modeled as graph nodes. The information and threat changing are modeled as probabilistic Markov state transition matrixes, which model the uncertainty. We assume that the agent can only observe the information and threat at the node where it locates, which captures the partially observable feature, and all the agents can share their observations. In this paper, we formulate the planning problem as a constrained Dec-POMDP.

Environment

We model the environment as an undirected graph $G = (V, E)$ and $N = |V|$. Each vertex has two states, one for information and the other for threat. The value of information is determined by the function $f^n : I^n \to \mathbb{R}$, where I^n is the information level of vertex v_n. In the same way, the threat value is determined by $c^n : R^n \to \mathbb{R}$. Figure 2 gives an instance of 3 agents ("blue triangles") patrolling an area of 36 locations ("black dots"). The information (positive) or threat (negative) states of the locations are shown by the yellow and red circles respectively. The size of the circles denotes the information or threat levels.

Multi-Agent Information Gathering

The problem is formulated as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP), which is $< \mathcal{M}, \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{O}, \Omega, r, h, b_0 >$, where

- $\mathcal{M}$ is the set of agents, $|\mathcal{M}| = m$.
- $\mathcal{S}$ is the set of states, and each $\boldsymbol{s} \in \mathcal{S}$ is defined as $[\boldsymbol{v}, (s_R^1, ..., s_R^N), (s_I^1, ..., s_I^N)]$, where $\boldsymbol{v}$ is the current positions of agents, $s_R^N \in R^n$ and $s_R^N \in I^n$ are the threat and information states at vertex $v_n \in \mathcal{V}$. We denote $\boldsymbol{s_e} = [(s_R^1, ..., s_R^N), (s_I^1, ..., s_I^N)] \in \mathcal{S}_e$ as the threat and information states at each position that captures the uncertainty of the environment.

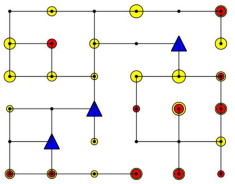

Figure 2: A scenario of 3 agents patrolling an area of 36 nodes generated randomly.

- $\mathcal{A}$ is the set of all joint actions, and the agents select adjacent vertices to visit as a joint action.
- $\mathcal{T}$ is the set of conditional transition probabilities. We assume that $\boldsymbol{v}$ is deterministic and only determined by the destinations of the joint movement of agents. $\boldsymbol{s_e}$ follows a discrete-time Markov process. We denote the Markov state transition processes as $\boldsymbol{P}_R^n$ and $\boldsymbol{P}_I^n$ for the threat and information state transition of vertex v_n.
- $\mathcal{O}$ is the set of joint observations. In this paper, we assume the observation of the environment is deterministic without uncertainty. The uncertainty comes from the changing environment.
- Ω is the set of observation probabilities. As an observation $\boldsymbol{o}$ is directly a part of some states, the observation probability $\Omega(\boldsymbol{o}|\boldsymbol{s}', \boldsymbol{a}) = 1$ if $\boldsymbol{o}$ is consistent with the corresponding part of $\boldsymbol{s}'$ and $\Omega(\boldsymbol{o}|\boldsymbol{s}', \boldsymbol{a}) = 0$ otherwise.
- r is the immediate reward function $\mathcal{S} \times \mathcal{A} \to \mathbb{R}$ calculated by Eqn. 1, where n_{v_i} is the number of agents visiting v_i. We assume that multiple agents at the same vertex of the same time suffer from multiple threats but only gather one information. The immediate reward for state $\boldsymbol{s}$ is shown as Eqn. 1, where α is the weighting parameter for balancing information and threat.

$$r(\boldsymbol{s}) = \sum_{v_i \in \boldsymbol{v}} (\alpha \frac{1}{n_{v_i}} f^i(s_I^i) - (1-\alpha) c^i(s_R^i)) \quad (1)$$

- h is the horizon of the problem. For example, each patrolling robot has a battery limit and can only move for 20 time steps during the whole information gathering task. Besides, there is no place for them to recharge.
- $\boldsymbol{b}_0$ is the initial state distribution at time $t = 0$. For the scenario, the initial information and threat states for all the vertex are zero vectors, and they start from the same starting places at the left bottom corner of the map.

Planning Objective

The objective of the agents is to choose the joint movement actions to maximize the total expected reward accumulated over h time steps shown as Eqn. 2, where $r(t)$ is the immediate reward of time step t.

$$R(h) = \sum_{t=0}^{t=h} r(t) \quad (2)$$

3 Integrating Decision Sharing with Prediction in Decentralized Planning

3.1 Decentralized Monte Carlo Tree Search

Monte Carlo Tree Search (MCTS) is a promising approach to online planning because it is not myopic and anytime [Kocsis and Szepesvári, 2006]. Recently, MCTS has been successfully applied to robotics such as adversarial patrol [Kartal *et al.*, 2015] and computer Go (MCTS is the basis for AlphaGo [Silver *et al.*, 2016] and AlphaGo Zero [Silver *et al.*, 2017]). By using Monte Carlo simulations to sample thousands of possible trajectories quickly, we can achieve good approximations of the values of possible actions.

Decentralized Monte Carlo Tree Search (Dec-MCTS) leverages the power of MCTS to select an effective and compact sample space of action sequences for decentralized online planning [Best *et al.*, 2019]. By sharing the decisions (intentions/plannings) with each other, i.e. the probability distributions of the action sequences, the multi-agent system gradually forms an emerging type of cooperation. The intuition behind Dec-MCTS is that one needs both think independently and negotiate with teammates when cooperating.

3.2 Decentralized MCTS Integrating Sharing with Prediction

We integrate sharing with prediction in the context of Dec-MCTS and propose Dec-MCTS-SP. Its performance relies on several important factors mainly including the decision sharing, the decision prediction and the joint-reward-guided decentralized tree search.

The result of one rollout is either "win" or "lose" in some scenarios such as "Chess" or "Go", so the reward is either "0" or "1" regardless of how many the look-ahead steps are. Unlike them, in Dec-MCTS-SP, we limit the maximal look-ahead steps to a constant variable D, which means the combined rewards guiding the tree to grow is D-step rewards, and the depth of the tree can not be deeper than D.

As mentioned above, the growth of the tree is guided by the combined reward calculated by the joint actions of the agents. The actions of the planning agent itself can be acquired easily from the tree (a branch from the root node to the leaf node is an action sequence for the planning agent itself). However, it is hard to get the actions of the other agents. To solve this problem, we divide D into two parts including D_s and D_p, where $D = D_s + D_p$.

First, the D_s actions of the other robots are sampled from the shared decisions (intentions or plannings) from the other robots, i.e. the probability distributions of action sequences, which is a compressed MCTS tree in essence (to be explained further in Section 3.4).

Second, the D_p actions of the other robots are predicted in a sufficiently-accurate and computationally-cheap method without any communication (to be explained further in Section 3.5).

Hence, the combined reward of the joint actions for guiding the MCTS tree is easy to get by sharing integrating prediction (i.e. $D = D_s + D_p$).

Algorithm 1 presents the pseudocode of the whole process of Dec-MCTS-SP. There are two loops in the algorithm.

Algorithm 1 Dec-MCTS-SP

Input: $B_{outside}$, B_{inside}, D_s, D_p
Output: a_{best}

```
1: RootNode←InitialiseMCTSTree()
2: s ← ObserveEnvironment()
3: PublishLocalObservation(s)
4: S ←ReceiveAndGetGlobalObservation()
5: while B_outside not met do
6:    X̂ ← SelectSubsetFrom(X)
7:    while B_inside not met do
8:       ExpandedNode←SD-UCTSelection(root)
9:       S_f=PredictFutureEnvironment(S)
10:      a_s=SharedActionsSampleFrom(X̂)
11:      a_p=PredictedActionsOfTeamMates(D_p,S_f)
12:      r←CalculateCombinedReward(a_s,a_p,S_f)
13:      Backpropagation(r)
14:   end while
15:   X_local ←GetLocalDistribution(RootNode)
16:   PublishLocalDistribution(X_local)
17:   X_other ←ReceiveAndUpdateDistribution()
18:   X ←GetGlobalDistribution(X_local, X_other)
19: end while
20: a_best=BestNextAction(RootNode)
21: return a_best
```

In the outside loop for budget $B_{outside}$, the planning agent shares the observation (Lines 2&3) and decision result (Lines 15-18) and grows the tree (Lines 7-14). In the inside loop for budget B_{inside}, the agent grows the tree guided by the combined reward (Line 12), which is calculated based on the shared and predicted actions.

3.3 Dealing with Uncertainty

The locations of the environment to patrol are under uncertainty. Hence, the future environment states should be predicted first (line 9 in Algorithm 1). We maintain two factored belief vectors of the information and threat states for vertex n at time step t, i.e. $\boldsymbol{b}_R^n(t)$ and $\boldsymbol{b}_I^n(t)$. They are probability distributions of threat and information states calculated by Eqn. 3 and Eqn. 4, where $v_i = n$ means that there is an agent i at location n.

$$\boldsymbol{b}_R^n(t+1) = \boldsymbol{b}_R^n(t)\boldsymbol{P}_R^n \tag{3}$$

$$\boldsymbol{b}_I^n(t+1) = \begin{cases} \boldsymbol{b}_0 & if\ v_i = n \\ \boldsymbol{b}_I^n(t)\boldsymbol{P}_I^n & if\ v_i \neq n \end{cases} \tag{4}$$

Further, we denote a feasible policy of length D at time step t as $\pi_D(t) = (\pi_{t+1}, ..., \pi_{t+D})$, which consists of D consecutive deterministic vertices (or actions). Given $\pi_D(t)$, we define the predictive expected future reward $E[\hat{R}(\pi_D(t))]$, which is the aggregate of the expected reward of each step in $\pi_D(t)$ as

$$\begin{aligned} E[\hat{R}(\pi_D(t))] = \sum_{i=1}^{D} \gamma^i (\alpha(\boldsymbol{b}_I^{\pi_{t+i}}(t+i)\boldsymbol{F} \\ - (1-\alpha)\boldsymbol{b}_R^{\pi_{t+i}}(t+i)\boldsymbol{L})) \end{aligned} \tag{5}$$

where γ is the discounted factor. $\boldsymbol{F}$ and $\boldsymbol{L}$ are two vectors denoted for the information values and threat values for different information and threat levels.

3.4 Decision Sharing with Team Mates

The various branches in the tree can represent various kinds of decisions, i.e. action sequences. The possible action sequences for all the agents are represented by $\mathcal{X}$. The most promising action sequences is restricted to a subset $\hat{\mathcal{X}} \in \mathcal{X}$. $\mathcal{X}_{local}$ and $\mathcal{X}_{other}$ mean the decisions of the local planning agent of its teammates.

We use the probability distributions of $\mathcal{X}$ to compress the tree (line 15 in Algorithm 1). First, the rewards of the branches in a constant depth will be calculated. Second, we normalize the rewards to get probability distributions. Third, the planning agent shares the probability distributions of $\mathcal{X}$ with the other agents to get global distributions (line 16-17 in Algorithm 1). By sampling from the global distributions, the planning agent can get the actions of the other agents. Finally, the selected distributions are used to grow the tree.

3.5 Decision Prediction for Team Mates

We integrate decision prediction method in Dec-MCTS-SP to reduce communication pressure and get better coordination result.

Modelling Other Agents with Heuristics-Based Policy

We use the heuristics-based methods in the context of Dec-MCTS to predict the actions of the others, the heuristics of decision (intention/planning) prediction method may have several alternatives by drawing inspiration from the work for warehouse commissioning [Claes *et al.*, 2017].

- Greedy, i.e. robots always move towards the node which has the highest reward.
- Reverse greedy from the perspective of the nodes, i.e. each location is assigned to the robot that has the best evaluation.
- Iterative greedy, i.e. we evaluate all locations for all robots and iteratively assign the currently best-evaluated location to the highest ranked agent.

Algorithm 2 shows pseudocode for the first approach. However, the choice of prediction is not limited to the above. In fact, the decision prediction method can help Dec-MCTS-SP converge faster, and you do not have to calculate a second time to predict decisions based on the same shared sampled joint actions. Besides, the $\varepsilon - Greedy$ policy can empower the Greedy to explore and decrease the predicting time.

3.6 Further Optimization

Sparse and Discounted UCT

While MCTS methods can deal with fairly large state spaces, huge state spaces are problematic. To deal with this problem, we propose to use Sparse UCT, which builds upon "Sparse Sampling" [Kearns *et al.*, 2002]. Besides, we also adopt another widely-used Discounted-UCT policy. The intuition is that the most recent rollouts are more relevant since they are obtained by sampling the most recent distributions. We combine the methods and use Sparse and Discounted UCT (SD-UCT) in Dec-MCTS-SP (Line 8 in Algorithm 1).

Algorithm 2 PredictActionsOfTeamMates

Input: $D_p, \mathcal{S}_f$
Output: $\boldsymbol{a}_p$

```
1: Initialize a_p
2: while D_p not met do
3:    while m not met do
4:       a^m_next = SelectBestAction(S_f)
5:       a_next ← add(a^m_next)
6:    end while
7: end while
8: a_p ← add(a_best)
9: return a_p
```

Incentivizing Agents by DIY Bonus

It is extremely difficult to accurately know the actions of the other agents in the future. We adopt the strategy of having a slight preference to do tasks themselves like many decentralized planning approaches [Claes *et al.*, 2017; Nijssen and Winands, 2012]. This improves the performance compared to optimising the combined reward in Eqn. 1 directly, since the revised reward is more sensitive to the planning robot itself and less affected by the uncertainty of the other robots' plans [Nijssen and Winands, 2012]. We denote $DIY - Bonus \in [0, 1]$ for the planning agent performing tasks when calculating the combined reward.

4 Experiments

4.1 Settings

We have implemented Dec-MCTS-SP on ROS (Robot Operating System) [Quigley *et al.*, 2009]. For the information gathering scenario, each agent shares their observation with the others to get a partial observation of the environment.

We use ROS nodes to simulate the uncertain environment and implement the planning processes of the robots. Several types of ROS messages are customized for the robots to observe the environment and share the information. The communication among the robots and the environment is asynchronous by using ROS-provided loosely-coupled publish/subscribe mechanism. Without loss of generality, the topological structure of all the maps in this paper are generated randomly to avoid the bias of evaluation.

We set the parameters in reward function and the value function as: the weight parameter $\alpha = 0.5$ and the discount factor $\gamma = 0.9$. The information and threat value vectors are respectively set as $\boldsymbol{F} = [0, 1, 2, 3, 4]$ and $\boldsymbol{L} = [0, 1, 2]$. The information state at each vertex in the environment has the same Markov state transition model, and the same set for the threat state transition. We benchmark against a random algorithm (Random), an auction-based algorithm (Auction). Besides, we compare our approach against the state-of-the-art approach as the baseline (Dec-MCTS).

- **Random:** The agents move in a random direction.
- **Auction:** The agents move to the adjacent with the highest value in the next several steps sequentially, which is a classical auction-based method for task allocation [Nunes and Gini, 2015].

- **Dec-MCTS:** The agents run a searching tree individually and share the observing and deciding information with each other, which serves as the baseline in this paper [Best *et al.*, 2019].
- **Dec-MCTS-SP:** Our approach that integrates sharing with prediction.

The initial locations of the agents are all at the bottom left corner of the maps. We assume each robot has a battery limit of running for 20 time steps. So, we let agents patrol continuously for 20 time steps ($h = 20$) in the uncertain environment and compare the total reward calculated by Eqn. 2. As for the other parameters for tree search of Dec-MCTS-SP and Dec-MCTS, we set the same parameters including that $B_{outside}$ is 5 and B_{inside} is 1500, which means the searching cost for Dec-MCTS-SP and Dec-MCTS is equal. D (total look-ahead steps) is 5, D_s (look-ahead steps for decision sharing) is 3, D_p (extra look-ahead for planning prediction) is 2. Besides, the DIY-Bonus is set as 0.1 empirically. In each experiment and each algorithm, we ran 100 rounds. We will present the average and standard deviation.

4.2 Results and Analysis

We ran Dec-MCTS-SP compared with Random, Auction and Dec-MCTS in the scenario as shown in Figure 2 with 36 nodes. All robots start from the left bottom corner. As the experimental results of Dec-MCTS-SP with the three heuristics are similar and are all better than the baseline (Dec-MCTS), we only present that of the first one for simplicity.

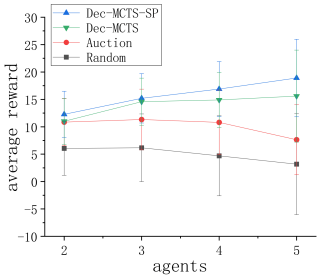

Figure 3: Average rewards of the agents by different planning approaches.

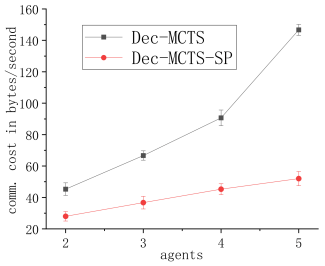

Figure 4: Communication cost compared with Dec-MCTS in bytes/seconds.

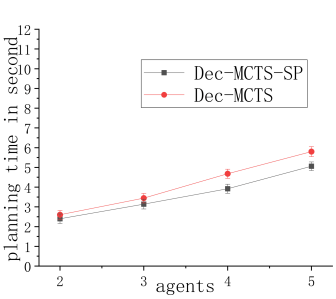

Figure 5: Average planning time compared with Dec-MCTS.

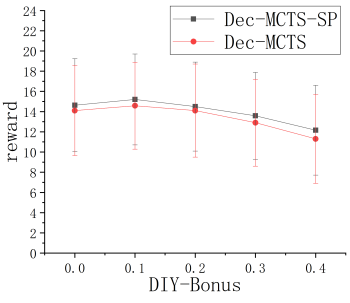

Figure 6: Performance of Dec-MCTS-SP compared with Dec-MCTS with DIY-Bonus.

Planning Accuracy with Communication Cost

Figure 3 presents the average reward of the agents by different planning approaches. Dec-MCTS-SP performs significantly better than all the other algorithms. It outperforms the Auction by more than and 60% and outperforms Dec-MCTS by more than 5% for three robots with significantly lower communication cost in bandwidth (Figure 4). Besides, the gap is gradually widening. With the increase of robots, it is normal the Random performs worse. However, the Auction performs worse, either. It is because that Auction is a greedy algorithm in essence and is prone to trap into local optimum. It is not easy for the agents to disperse to gather new information, so they may suffer from threat of the local locations continuously.

Planning Times

Figure 5 shows the average planning time of the agents by different planning approaches. In this experiment, we only evaluate and compare the two decentralized methods including Dec-MCTS-SP and Dec-MCTS. Dec-MCTS-SP has lower time cost but higher performance compared with Dec-MCTS. We can divide the reasonable decentralized planning time into three levels including "within a second", "several seconds" and "within a minute". As shown in Figure 5, the planning time can be controlled on the level of "several seconds" both for Dec-MCTS-SP and Dec-MCTS, which suits for some scenarios requiring real-time planning. With the increase of the agents, the planning time increases linearly and gently for both Dec-MCTS-SP and Dec-MCTS, which is beneficial to the scalability.

With DIY-Bonus

Figure 6 presents the performance (reward) of Dec-MCTS-SP compared with Dec-MCTS with the increase of DIY-Bonus. There are three robots in the scenario as shown in Figure 2. From the result, we can see that the performance of Dec-MCTS-SP is higher than Dec-MCTS when the DIY-Bonus increases. The performance of them begins to decline when DIY-Bonus is larger than 0.2. Dec-MCTS-SP and Dec-MCTS both have a best performance when DIY-Bonus is set to 0.1, so we set DIY-Bonus as 0.1. To sum up, the DIY-Bonus should be set carefully and appropriately.

Discussions

The results show Dec-MCTS-SP has higher reward with shorter planning time and lower communication cost compared with Dec-MCTS. We analyzed the reasons below.

First, as for the higher reward. The budgets including B_{inside} and $B_{outside}$ should be set considering real-time and online planning with the highest priority. So, the budgets for the experiments in this paper may not be enough for a Dec-MCTS tree of the depth of five ($D = D_s$) to converge. The state space for searching will increase exponentially with the growth of the depth of the tree. For the Dec-MCTS-SP tree, the tree depth is three (D_s), which is more beneficial for convergence, and thanks to the predictive look-ahead steps (D_p), the look-ahead steps is still five ($D = D_s + D_p$).

Second, as for the shorter planning time. Because it is mainly relative to the search budget (the same for Dec-MCTS-SP and Dec-MCTS) and the size of the tree (the

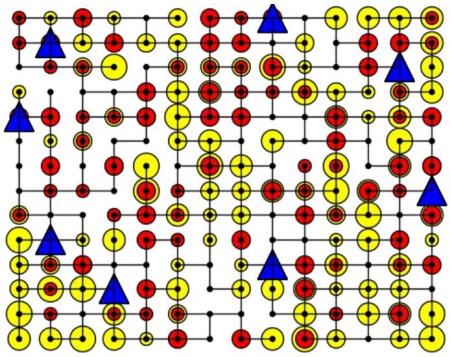

Figure 7: A larger scenario with 8 agents and 196 locations.

size of Dec-MCTS-SP tree decreases exponentially compared with Dec-MCTS due to the difference of depth).

Third, as for the lower communication cost. Because it depends on the number and length of the communicated action sequences. Apparently, because the size of the tree is extremely reduced, the shared information of Dec-MCTS-SP will decrease exponentially compared with Dec-MCTS.

Finally, Dec-MCTS-SP is general for distributed sequential decision problems, which consists of three aspects including a decision sharing method, a decision prediction method, and an integrating method of sharing and prediction.

4.3 Scalability

We demonstrate the scalability of Dec-MCTS-SP by a larger case study. Figure 7 gives a larger environment with 8 agents and 196 vertexes. The topology of the map is generated randomly. At most, we implemented the Dec-MCTS-SP method in an environment of 400 vertexes and 16 agents. The single-step planning time is about 8 seconds for 196 vertexes and about 12 seconds for 400 vertexes. Through quantitative analysis, by using Dec-MCTS-SP, the agents begin to spread out into the environment from starting point quickly and patrol sporadically in the environment instead of trapping into a small area.

5 Related Work

Communication of the robots' intentions (decisions or plannings) is critical to the coordinated behaviour of decentralized multi-robot teams, yet communication remains a bottleneck for deployed multi-robot systems [Best *et al.*, 2018]. Many planning approaches are based on the limiting assumption of perfect communication, such as Dec-MCTS [Best *et al.*, 2019], Multi-Robot Task Allocation (MRTA) [Capitan *et al.*, 2013; Nunes and Gini, 2015] and max-sum [Chen *et al.*, 2016], which makes the approaches suffer from poor scalability and not suit for realistic environment.

To solve the problem above, an alternative solution is communication-aware planning, which typically seeks to improve the available communication resources by repositioning the robots. This can be achieved by encoding communication reliability as a path planning objective [Lindhe and Johansson, 2013; Ghaffarkhah and Mostofi, 2011; Best *et al.*, 2017; Wu *et al.*, 2017]. However, the computing overhead is usually high which hinders the performance of the whole system.

Besides, some decentralized methods are based on the principle of without-communication cooperation. The cooperations are implicit, which are implied in observation or prediction. First, as for implicit cooperation by observation [Antoniades *et al.*, 2003], the agents only coordinate by observations of the environment (incl. possibly other agents). Second, as for implicit cooperation by prediction [Claes *et al.*, 2017], there is no decision sharing between agents, instead, the modeling of other agents is through predicting by several greedy heuristics, which is computationally cheap. The agents coordinated by the methods above are more tend to conflict with each other in a dense environment due to the lack of explicit decision sharing. Oppositely, we hold the idea that moderate communication is necessary for the coordinational performance.

Beyond that, planning-aware communication is a recently-proposed novel method [Best *et al.*, 2018], which is a novel planning algorithm that reasons over the value of communication messages to decide when and to whom each robot should communicate and also in the context of Dec-MCTS. However, it does not change the exponential increasing nature of the communication cost with the growth of look-ahead steps. In fact, the performance degrades compared with Dec-MCTS. We tackle the problem in a more principled method by integrating sharing and prediction with higher performance compared with Dec-MCTS. Also, while our approach is intended to be general, we address these challenges in the context of Dec-MCTS.

The information gathering problem in this paper is formulated by the framework of decentralized partially observable Markov decision process (Dec-POMDP) [Bernstein *et al.*, 2002]. It was introduced for sequential multi-agent decision-making under uncertainty, which also characterizes incomplete or partial information of the environment and other agents due to limited or no communication [Chen *et al.*, 2016].

6 Conclusions and Future Work

In this paper, we propose a decentralized planning approach called Dec-MCTS-SP, which integrates decision sharing with prediction. The experimental results show that Dec-MCTS-SP consumes lower communication cost with higher performance compared with Dec-MCTS. We will extend and evaluate the work for more complex tasks under more realistic environments in the future. Different communication network models may be used such as broadcasting or sequential communication.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (No. 91648204 and 61803375), the National Key Research and Development Program of China (No. 2017YFB1001900 and 2017YFB1301104), the Youth Talent Lifting Project (No.17-JCJQ-QT-047), and the National Science and Technology Major Project.

References

[Antoniades *et al.*, 2003] Adonis Antoniades, H Jin Kim, and Shankar Sastry. Pursuit-evasion strategies for teams of multiple agents with incomplete information. In *International Conference on Decision and Control*, volume 1, pages 756–761. IEEE, 2003.

[Bernstein *et al.*, 2002] Daniel S Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. The complexity of decentralized control of Markov Decision Processes. *Mathematics of Operations Research*, 27(4):819–840, 2002.

[Best *et al.*, 2017] Graeme Best, Wolfram Martens, and Robert Fitch. Path planning with spatiotemporal optimal stopping for stochastic mission monitoring. *IEEE Transactions on Robotics*, 33(3):629–646, 2017.

[Best *et al.*, 2018] Graeme Best, Michael Forrai, Ramgopal R Mettu, and Robert Fitch. Planning-aware communication for decentralised multi-robot coordination. In *International Conference on Robotics and Automation (ICRA)*, pages 1050–1057. IEEE, 2018.

[Best *et al.*, 2019] Graeme Best, Oliver M Cliff, Timothy Patten, Ramgopal R Mettu, and Robert Fitch. Dec-MCTS: Decentralized planning for multi-robot active perception. *The International Journal of Robotics Research*, 38(2-3):316–337, 2019.

[Capitan *et al.*, 2013] Jesus Capitan, Matthijs T J Spaan, Luis Merino, and Anibal Ollero. Decentralized multi-robot cooperation with auctioned POMDPs. *The International Journal of Robotics Research*, 32(6):650–671, 2013.

[Chen *et al.*, 2016] Shaofei Chen, Feng Wu, Lincheng Shen, Jing Chen, and Sarvapali D Ramchurn. Decentralized patrolling under constraints in dynamic environments. *IEEE Transactions on Systems, Man, and Cybernetics*, 46(12):3364–3376, 2016.

[Claes *et al.*, 2017] Daniel Claes, Frans Oliehoek, Hendrik Baier, and Karl Tuyls. Decentralised online planning for multi-robot warehouse commissioning. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pages 492–500. International Foundation for Autonomous Agents and Multiagent Systems, 2017.

[Ghaffarkhah and Mostofi, 2011] Alireza Ghaffarkhah and Yasamin Mostofi. Communication-aware motion planning in mobile networks. *IEEE Transactions on Automatic Control*, 56(10):2478–2485, 2011.

[Kartal *et al.*, 2015] Bilal Kartal, Julio Godoy, Ioannis Karamouzas, and Stephen J Guy. Stochastic tree search with useful cycles for patrolling problems. In *International Conference on Robotics and Automation (ICRA)*, pages 1289–1294. IEEE, 2015.

[Kearns *et al.*, 2002] Michael Kearns, Yishay Mansour, and Andrew Y Ng. A sparse sampling algorithm for near-optimal planning in large Markov decision processes. *Machine learning*, 49(2-3):193–208, 2002.

[Kocsis and Szepesvári, 2006] Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *European conference on machine learning*, pages 282–293. Springer, 2006.

[Lindhe and Johansson, 2013] Magnus Lindhe and Karl Henrik Johansson. Exploiting multipath fading with a mobile robot. *The International Journal of Robotics Research*, 32(12):1363–1380, 2013.

[Nijssen and Winands, 2012] Pim Nijssen and Mark H M Winands. Monte carlo tree search for the hide-and-seek game Scotland Yard. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(4):282–294, 2012.

[Nunes and Gini, 2015] Ernesto Nunes and Maria Gini. Multi-robot auctions for allocation of tasks with temporal constraints. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2110–2216, 2015.

[Oliehoek, 2018] Frans A Oliehoek. Interactive learning and decision making: Foundations, insights & challenges. In *International Joint Conferences on Artificial Intelligence (IJCAI)*, pages 5703–5708, 2018.

[Quigley *et al.*, 2009] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. ROS: an open-source Robot Operating System. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan, 2009.

[Silver *et al.*, 2016] D Silver, A. Huang, C. J. Maddison, A Guez, L Sifre, den Driessche G Van, J Schrittwieser, I Antonoglou, V Panneershelvam, and M Lanctot. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[Silver *et al.*, 2017] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, and Adrian Bolton. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, 2017.

[Wu *et al.*, 2017] Yunlong Wu, Bo Zhang, Shaoshi Yang, Xiaodong Yi, and Xuejun Yang. Energy-efficient joint communication-motion planning for relay-assisted wireless robot surveillance. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 1–9. IEEE, 2017.