Dec-SGTS: Decentralized Sub-Goal Tree Search for Multi-Agent Coordination –Supplementary Material

We report additional material to supplement our work. In Section 1, we present the formalisations and proofs of the propositions, which provide Dec-SGTS with solid theoretical basis. In Section 2, we clarify the parameter settings of the experiments for reproducibility.

1. Formalisations and Proofs of the Proposed Propostions for Dec-SGTS

In this section, we provide a detailed theoretical analysis of Dec-SGTS, which is an anytime, hierarchical and decentralized approach to multi-agent coordination with three key algorithmic components: (1) D-UCT: the sub-goal tree search is designed to perform long-horizon planning for single-agent subgoal seqeunces while considering the changing plans of the other teammate agents. (2) Tree Expansion: the sub-goal tree is expanded with several children of neighbor successive subgoal states along with finding the best subgoal-pair distances and policies. (3) Intention Update: the product distribution optimization is designed to directly optimize the joint multi-agent plan while being restricted to a small subset of possible subgoal sequences.

Proof of Proposition 1 for D-UCT

Proposition 1. *Although the joint probability distribution, i.e. q_n, is changing (and converging), D-UCT maintains an exploration–exploitation trade-off for child selection and achieves a polynomial convergence rate.*

Proof. Gariver et al. first studied a specific type of non-stationary, switching bandit problem using Discounted Upper Confidence Bounds (D-UCB) (Garivier and Moulines 2011). It has been applied to Multi-Armed Bandit (MAB) problem for solving the *exploration-exploitation dilemma* with non-stationary and unknown distributions. Best et al. extend the Multi-Armed Bandit problem to Decentralized Monte Carlo Tree Search (Dec-MCTS) and propose Discounted Upper Confidence Bounds for Tree (D-UCT), where the joint probability distribution is changing and converging (cf. the proof of Proposition 3). Following the theoretical analysis and conclusion of Gariver et al. (Garivier and Moulines 2011) and Best et al. (Best et al. 2019), we justify Proposition 2 as below. Consider the algorithm D-UCT running on a Tree T of depth D and branching factor K. The payoff distributions of the leaf nodes are independently distributed and can change at breakpoints. The breakpoint means the distribution q_n changes abruptly when the tree search is going on. Then, under some reasonable assumption (Assumption 2 in (Best et al. 2019)), Best et al. prove that the bias of the payoff at the root node is $O(KD\log(t)\sqrt{E[\Upsilon_t]/t})$, where t is the round that the bandit arm is pulled, that is, the search time of the SGTS till now. Υ_t denotes the number of breakpoints at time t. $E[\Upsilon_t]$ denotes the expected number of breakpoints at time t. Further, the probability of failure at the root node becomes zero as t grows large. Best et al. give some reasonable assumption of state conditions for the breakpoint sequence $\{E[\Upsilon_t]\}_t$ (Assumption 2 in (Best et al. 2019)), and they can ensure these assumptions are satisfied by selecting appropriate value for the sample period b_1 (Line 5 in Alg. 1). Here, we provide a concrete example definition for b_1. Recall that n is the number of times $\hat{\mathcal{G}}_n$ is changed (Line 2 in Alg. 1) and b_1 is the number of calls to local intention update with sample space $\hat{\mathcal{G}}_n$ (Line 5 in Alg. 1). Let $b_2 > 0$ denotes the fixed number of iterations for local sub-goal tree search (Line 7 in Alg. 1). For this example, we let $b_1 = 1/\lfloor at^{-2} \rfloor$ and, therefore, $n = \lceil a(1-t^{-1}) \rceil$ where $a > c$. Therefore, the expected upper bound on breakpoints $E[\Upsilon_t] = n/c$ and , thus, $\lim_{t\to\infty} E[\Upsilon_t] = a/c$. This ensures D-UCT holds a convergence of failure probability and the bias of the root node is $O(KD\log(t)\sqrt{t})$. Therefore, D-UCT maintains an exploration–exploitation trade-off and achieves a polynomial convergence rate, even in problems where the reward distributions are changing, such as in Dec-SGTS.

Proof of Proposition 2 for Tree Expansion

Proposition 2. *Given a proper σ, Algorithm 2 can find the best subgoal-pair distances and policies asymptotically when expanding the tree with successive subgoal states.*

Proof. We can view the primitive action sequence from a subgoal to another subgoal as a macro-action at primitive time step t, which is defined as m_t. Therefore the macro reward of agent i at primitive time step t can be described as $\overline{r}(s_t, m_t) = \sum_{k=0}^{|m_t|-1} r(s_{t+k}, a_{t+k})$. Thus the optimal set of macro-actions m_t for state s_t which maximizes the reward

$\overline{\mathcal{R}}(s_t, m_t)$ can be defined by:

$$\mathcal{M}^*(s_t) = \{ \underset{m_t \in \mathcal{M}(s_t, s_{t'})}{\arg\max} \overline{r}(s_t, m_t) \} \quad (1)$$

where $\mathcal{M}(s_t, g_t)$ denotes the macro-actions from the subgoal state s_t to another subgoal state $s_{t'}$, $g(s_t) = 1$ and $g(s_{t'}) = 1$. Given state s_t, $\mathcal{M}^*(s_t)$ can be approximated with $\hat{\mathcal{M}}(s_t)$. $\hat{\mathcal{M}}(s_t) \approx \mathcal{M}^*(s_t)$ is generated incrementally by randomly sampling a macro-action m_t from $\mathcal{M}(s_t)$. If we get a better reward $\overline{r}(s_t, m_t)$, we can get a better subgoal-pair distances and policy between the initial and terminal subgoal states of the macro action. The state space of the subgoal states is a finite space. The size of the space is decided by the domain. We store the best evaluation in every trial, i.e. elitism preservation policy. As the trial times increase, we can asymptotically reach the best distances and policies of the subgoal-pairs. In the context of Bernoulli Trial, we can justify why we can asymptotically optimize the reward and get the best subgoal-pair evaluation in other words. Action coverage count c is used for measuring the degree of action coverage for state s_t (Line 1 in Alg. 2). The action coverage threshold σ is calculated with $\log_{p_0} \alpha$ by hypothesis testing, where p_0 is action coverage, $\alpha \in [0, 1]$ is the tolerated error. The sampling can be seen as a Bernoulli Trial. Let $\mathbf{Y} =< Y_1, ..., Y_m >$ be a Bernoulli process based on Bernoulli distribution $\mathcal{B}_p$ with $Y_j \sim \mathcal{B}_p$. When $Y_j = 1$, the sampled macro-action is rediscovered, and $Y_j = 0$ otherwise. To test if $p \geq p_0$, we get action coverage threshold $\sigma = \log_{p_0} \alpha$. Given a tolerated error, e.g. $\alpha = 0.05$, when σ increases, the action coverage will grow larger. According to the original analysis of Bernoulli Trial, when the action coverage is large enough, we can get the best subgoal-pair distances and policy. Finally, we can summarize that given a proper σ, Algorithm 2 can find the best subgoal-pair distances and policies asymptotically when expanding the tree with successive subgoal states.

Proof of Proposition 3 for Intention Update

Proposition 3. *Given an appropriate subset $\hat{\mathcal{G}}_n^i \in \mathcal{G}_n^i$, the algorithm by iteratively implementing the subgoal intention update asymptotically converges to the joint distribution q_n that optimizes the joint subgoal sequences.*

Proof. Following the analysis of Best et al. (Best et al. 2019), we justify Proposition 3 as below. In the intention update phase of Dec-SGTS, in fact, q_n is an approximation to the best joint distribution p_n. The most likely p_n can be found by minimizing the maxent Lagrangian (Best et al. 2019). In that formulation, pq KL divergence is the divergence from q_n to the optimal joint distribution p_n. The approach to solving this formulation is essentially a decentralized gradient descent method over the space of q_n. Consider the situation where, at each iteration n, we choose a subset $\hat{\mathcal{G}}_n^i \in \mathcal{G}_n^i$ for each agent. In fact, this approach is equivalent to Monte Carlo sampling of the expected utility and thus the biased estimator is consistent (asymptotically converges to $E[f^i]$). The consistency has no connection with the design of the utility function f, it only requires all agents use the same f. So the subgoal-based utility function in Dec-SGTS will not influence the consistency. For tractable computation and faster convergence, in Dec-SGTS, we modify the random selection by choosing a sparse set of strategies $\hat{\mathcal{G}}_n$ with the highest expected utility (Sparse Representation). Although this does not ensure we sample the entire domain $\mathcal{G}$ asymptotically, in practice $q_n(\mathcal{G}^n)$ is a reasonably accurate representation of $q_n(\mathcal{G})$, and therefore this gives us an approximation to importance sampling (Wolpert, Strauss, and Rajnarayan 2006). Variants of the Intention Update has been shown to converge to a distribution that locally minimizes the pq KL divergence under reasonable assumptions, such as an appropriate cooling schedule for β (Wolpert and Bieniawski 2004). It means that given an appropriate subset $\hat{\mathcal{G}}_n^i \in \mathcal{G}_n^i$, the algorithm by iteratively implementing the subgoal intention update asymptotically converges to to the best joint distribution that optimizes the joint subgoal sequences.

2. Parameter Settings

We use a platform of 12 cores, 3.7 GHz and 16 GB Memory. The algorithms are implemented by Python. Rospy is leveraged as a middleware library to enable the agents to communicate (Quigley et al. 2009). Each agent is a ROS node running by an independent process. We design a message type using MDS (Message Description Specification) of ROS (Quigley et al. 2009) composed of four fields including agent ID, time step, the probability distribution over subgoal sequences and the subgoal-pair distances in those subgoal pairs, which represents current subgoal intention.

Except for the parameter settings mentioned in the main body of our paper, we set the other parameters as follows. The action coverage threshold $\sigma = 40$, the horizon $h = 60$, the discounted factor of D-UCT $\gamma = 0.99$, the number of paths in sparse representation of current tree $|\hat{\mathcal{G}}^i| = 10$. The budgets for the loops in Alg. 1 are set as $b_2 = 10$, $b_1 = 100$. We set b_0 arbitrarily, e.g. 10000000. The algorithms in our experiments are all anytime, which means that we can get a meaningful result whenever we stop the program from running. Since the running time can be controlled manually, so it is enough to set b_0 as a large number. In intention update function (Eqn. 3), we set the designated step size $\eta = 0.1$. We let β gradually decrease, that is, $\beta = e^{-(m+1)}$, if $\beta < 0.0001$, we let $\beta = 0.0001$, where m is the iteration number (Line 5 in Alg. 1). We set the same parameter settings for Dec-SGTS, Dec-MCTS, S-MCTS and MCTS.

References

Best, G.; Cliff, O. M.; Patten, T.; Mettu, R. R.; and Fitch, R. 2019. Dec-MCTS: Decentralized planning for multi-robot active perception. *The International Journal of Robotics Research (IJRR)* 38(2-3): 316–337.

Claes, D.; Oliehoek, F.; Baier, H.; and Tuyls, K. 2017. Decentralised online planning for multi-robot warehouse commissioning. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 492–500.

Garivier, A.; and Moulines, E. 2011. On upper-confidence bound policies for switching bandit problems. In *Proceedings of the International Conference on Algorithmic Learning Theory (ALT)*, 174–188. Springer.

James, S.; Konidaris, G.; and Rosman, B. S. 2017. An analysis of monte carlo tree search. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Li, M.; Yang, W.; Cai, Z.; Yang, S.; and Wang, J. 2019. Integrating decision sharing with prediction in decentralized planning for multi-agent coordination under uncertainty. In *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI)*, 450–456. AAAI Press.

Nunes, E.; and Gini, M. 2015. Multi-robot auctions for allocation of tasks with temporal constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Quigley, M.; Conley, K.; Gerkey, B.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; and Ng, A. Y. 2009. ROS: an open-source Robot Operating System. In *IEEE International Conference on Robotics and Automation (ICRA) workshop on open source software*, volume 3, 5. Kobe, Japan.

Wolpert, D. H.; and Bieniawski, S. 2004. Distributed control by lagrangian steepest descent. In *Proceedings of the IEEE Conference on Decision and Control (CDC)*, 1562–1567. IEEE.

Wolpert, D. H.; Strauss, C. E.; and Rajnarayan, D. 2006. Advances in distributed optimization using probability collectives. *Advances in Complex Systems* 9(04): 383–436.