Decentralized Subgoal Tree Search for Multiagent Planning without Priors or Communication

Qian Che
School of Cyber Science and Engineering
Southeast University
Nanjing, China
qche@seu.edu.cn

Yixuan Li
School of Computer Science and Engineering
Southeast University
Nanjing, China
yixuanli@seu.edu.cn

Ziyao Peng
School of Computer Science and Engineering
Southeast University
Nanjing, China
zpeng1898@seu.edu.cn

Wanyuan Wang
School of Computer Science and Engineering
Southeast University
Nanjing, China
wywang@seu.edu.cn

Yichuan Jiang
School of Computer Science and Engineering
Southeast University
Nanjing, China
yjiang@seu.edu.cn

***Abstract*—Multiagent Markov decision processes (MMDPs) provide an expressive framework for multiagent planning in stochastic domains. However, exactly solving a large MMDP is often intractable due to the exponential space of joint action. Due to the trade-off nature of trading computation time for solution quality, decentralized subgoal-based tree search (Dec-SGTS) methods have shown great success for MMDPs. Existing Dec-SGTS methods rely on the predefined subgoals and the communication between agents to perform well, which might not hold in real-world MMDP domains where there is no expert knowledge on subgoals and communication resources are limited. In this paper, we relax these assumptions that arrive at an automated and communication-free Dec-SGTS. On the one hand, we first propose an upstream guided subgoal search (UGSS) technique to exploit historical search experience for subgoal discovery. On the other hand, in order to coordinate agents' behaviors without communication, we further propose an expectation-alignment technique to proactively align agents' policies with team's expectations. Finally, we conduct extensive experiments on multirobot box-pushing tasks, and the results show that compared to multiagent planning benchmarks, the proposed communication-free Dec-SGTS method, which does not require any subgoal priors, achieves satisfactory rewards.**

Keywords—Multi-Agent Planning, Monte Carlo Tree Search, Sub-goal Search, Communication and Priors

I. INTRODUCTION

Multiagent Markov Decision Processes (MMDPs) have been widely used in the field of multiagent planning [1,2], and can model a wide range of applications ranging from mutirobot task allocation [3,4], city-wide traffic control [5], to autonomous driving in open environment [6]. Due to *the curse of history* and *the curse of dimensionality*, solving MMDPs optimally is computationally intractable. The curse of dimensionality means that in an MMDP with n agents and k actions, each state-action value reasons about a k^n global joint actions. Furthermore, the number of histories that it must evaluate is exponential in the time horizon. Due to the trade-off nature of trading computation time for solution quality, Monte-Carlo tree search (MCTS) [7] has shown great success for single agent MDP tasks, such as the game of Go [8]. Recent advances have been made in the extension of single agent MCTS to multiagent MCTS for MMDPs, focusing either on the curse of history or the curse of global joint action.

On the one hand, in order to solve the historical curse, subgoal-based MCTS (S-MCTS) can accelerate the evaluation of the historical sequence of actions by decomposing the original goal into a hierarchy of subgoals that can be addressed individually [9-13]. Primitive action sequences to achieve these subgoals can be learned to enable efficient decision making at the higher subgoal levels. On the other hand, in order to avoid the exponential space of the global joint actions, decentralized MCTS (Dec-MCTS) enables each agent to search for its own policy while considering the behavior of other agents [14]. In Dec-MCTS, the agents need to communicate and share the policy with each other in order to achieve coordination [15-18]. Recently, combining the advantages of subgoal search and decentralization, a decentralized subgoal tree search (Dec-SGTS) method has been proposed, in which each agent uses S-MCTS to make decisions in a fully decentralized way [19,20] .

Current Dec-SGTS methods rely on the predefined subgoals and the heavy communication between agents for policy sharing. However, in real-world MMDP domains, it is not readily available for experts to define a complete specification of subgoals [21-22]. Moreover, over communication may hinder the robustness of the system in the case of an unreliable network, or when the signals are noisy to infer [17, 23]. For example, in the rover exploration domain, a set of robots try to observe points of interest on the lunar surface, and the communication might not be available, but each robot must make decision based on its own local observations [24]. In this paper, Dec-SGTS is extended to a more general setting, with no prior subgoals and no communication in distributed planning. The basic idea of our approach is to use the historical search experience to discover desirable subgoals automatically, and use the expectation-alignment technique to regularize the agent's policy that match team's expectations.

Our contributions can be summarized as follows. 1) In terms of subgoal tree search without subgoal priors, an upstream-guided subgoal search (UGSS) method is proposed to obtain subgoals based on the Gaussian process and to perform subgoal tree refinement regularly to prune low-quality subgoals. 2) We propose a communication-free Dec-SGTS (CF-Dec-SGTS) for MMDPs, in which the agents use expectation alignment to align their policy (with respect to subgoal) with the team's expectations. 3) Finally, we test the proposed UGSS and CF-Dec-SGTS with an extensive empirical evaluation on multirobot box-pushing tasks. Experimental results demonstrate that compared with other decentralized MCTS benchmarks without communication, the proposed CF-Dec-SGTS can achieve the highest performance, while achieving nearly 90% of the state-of-the-art Dec-SGTS methods with communication.

In the rest of this paper, we review related work in Section 2 and formulate the problem in Section 3. We propose UGSS for subgoal tree search in Section 4 and expectation-align subgoal tree search for multiagent planning in Section 5. Finally, we evaluate our method in Section 6 and conclude in Section 7.

II. Related Work

In the related work, we mainly focus on the following areas: subgoal-based Monte-Carlo tree search (MCTS) for MDP and multiagent MCTS methods for MMDP.

A. *Subgoal-based MCTS*

MCTS is a popular approach to MDP problems and has been applied to a wide range of challenging environments [7]. One approach is to compress the search space in the temporal dimension by introducing subgoals at a coarser resolution of time, which is known as subgoal MCTS (S-MCTS) [9]. The S-MCTS divides the whole task into subtasks and abstracts the primitive actions into macro actions, allowing planners at different levels to plan independently. The high-level planner selects intermediate subgoals to achieve the final goal, while the low-level planner executes the original action sequences to achieve the subgoals.

The subgoal definition is necessary since it ensures that high-level planner has an optional set of subgoals. There are two kinds of methods for subgoal definition. Firstly, relying on extensive domain knowledge, all subgoals per state [25] and determining whether a given state is desirable as a subgoal [9, 19], can be fully known beforehand. Without the predefined subgoals, pretraining models can generate the set of subgoals through offline supervised learning [21, 22]. However, the additional dataset and offline training make it infeasible for online planning scenarios. In contrast, this paper does not assume any prior subgoal knowledge, and conducts subgoal search in the node expansion phase of MCTS, refines the historical experience and identifies the subgoal state automatically.

B. *Multiagent MCTS for MMDPs*

Due to the number of global joint actions grows exponentially, it is difficult to apply MCTS for MMDPs for trading-off exploration and exploitation. In order to reduce the search complexity, Decentralized MCTS (Dec-MCTS) enables all agents to search for their own policies while taking other agents' behaviors into consideration [14]. In order to improve coordination, agents can also communicate and share their policies with other agents [15-18]. Exploiting the advantage of subgoal MCTS, Li et al. [20] recently proposed the decentralized subgoal tree search (Dec-SGTS) method, in which each agent uses S-MCTS to make decisions in a fully decentralized manner. In order to complete collaborative tasks successfully, each agent also needs to consider teammates and may have several alternative subgoal sequences with different preferences. Therefore, in Dec-SGTS, agents need to exchange their subgoal intentions through communication. The success of current Dec-SGTS requires the heavy communication between agents for policy (or the policy distribution) sharing. In this paper, in order to extend Dec-SGTS to general multiagent collaboration scenarios without communication, we propose an expectation-alignment subgoal tree search, in which each agent can infer other agents' intentions and regulate its own policy as team's expectations.

III. Problem Formulation and Algorithm Framework

Multiagent planning problem can be modeled as a multiagent Markov decision process (MMDP), which can be defined as a tuple $F = <\mathcal{N}, \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}>$, where:

- $\mathcal{N} = \{1,2,\dots,n\}$ is the set of n agents.
- $\mathcal{S}$ is a finite set of global states s of the environment.
- $\mathcal{A} = A_1 \times \dots \times A_n$ is the set of joint actions $\vec{a} = \langle a_1, \dots, a_n \rangle$.
- $\mathcal{P}$ is the set of transition probabilities. $P: S \times A \times S' \to [0,1]$ is the probability of ending up at state S' after performing action A at state S. Transitions are assumed to be deterministic.
- $\mathcal{R}$ is the immediate joint reward function and $S \times A \to \mathrm{R}$.

In MMDP, a joint policy π: $\mathcal{S} \to \mathcal{A}$ maps the global state to the global joint action $\vec{a}$, and is equivalent to a tuple of individual policies $\pi_i: S \to A_i$. Agents interact with the environment in sequence for T periods. A finite sequence $\rho_\pi = \langle s_0, s_1, \dots, s_T \rangle$ of states generated by a policy π is called a trajectory. For any joint policy π in an MMDP, the expected discounted cumulative global reward in state s is:

$$V^\pi(s) = \mathbb{E}[\textstyle\sum_{k=0}^{T-1} \gamma^k R(s_k, \vec{a}_k, s_{k+1}) | s_0 = s] \quad (1)$$

where $\mathbb{E}$ denotes the expected value by following π, and $\gamma \in [0,1]$ is the discounted factor. The objective for solving an MMDP is to find a joint policy $\pi = \times_{1 \le i \le n} \pi_i$ that can generate a trajectory ρ_π to maximize the cumulative global reward $V^\pi(s_0)$ at the starting state s_0.

The Framework of Decentralized Subgoal Tree Search for MMDPs Without Priors or Communication. The algorithm framework consists of two parts: which address the subgoal priors (Section 4) and communication (Section 5) issues, respectively.

Part 1: Subgoal-based Monte Carlo Tree Search without Priors. In this part, we do not assume any domain knowledge about subgoals, and exploit the historical search experience to discover the desirable subgoals automatically. This kind of subgoal tree search will be used as the planning mechanism of each agent in Part 2.

Part 2: Decentralized Subgoal Tree Search without Communication. In this part, in order to achieve coordination between agents without communication, we propose an expectation alignment technique, which encourages agents to behave in ways that match team's expectations (Section 5).

IV. Subgoal-based Monte Carlo Tree Search without Priors

In this section, we first describe the background of subgoal-based Monte Carlo tree search (S-MCTS), and present the upstream-guided subgoal search technique.

A. Subgoal-based MCTS (S-MCTS)

The MCTS incrementally builds a tree by iteratively performing four stages until the computational budget is used up.

- **Selection**: Starting from the root node as current state s_0, MCTS selects the child node with the largest upper confidence bound (UCB) until reaching a leaf node s_t:

$$UCB(s_t, a_t) = Q(s_t, a_t) + c\sqrt{\frac{2\log\left(N(s_t)\right)}{N(s_t, a_t)}} \tag{1}$$

- **Expansion**: A leaf node s_t is expanded to a new node s_{t+1} after a random action a_t.
- **Rollout**: Simulating from s_{t+1} with a rollout policy $\pi_{rollout}$ until the maximum search depth H or a terminal state is reached.

- **Backpropagation**: The rollout rewards are accumulated and used to update the estimated value and visit counts of each node in the simulated path.

Subgoal-based MCTS (S-MCTS). S-MCTS abstracts subgoals from MCTS. We define an open-loop sequence of primitive actions $a_i \in A$ as a macro-action $m_t = \langle a_t, ..., a_{t+N-1}\rangle$. The macro-action space $\mathcal{M} = A^+$ is the set of all non-empty sequences. The macro-level generative model $\overline{F}: S \times \mathcal{M} \to S$ determines the subsequent state $\overline{F}(s_t, m_t) = s_{t+|m_t|}$ after a macro-action $m_t \in \mathcal{M}$ of horizon $|m_t|$ from s_t. $\mathcal{G}(s_t)$ is the subgoal set reachable from s_t by any macro-action. $\mathcal{M}(s_t, g_t)$ contains all macro-actions from state s_t and terminate in subgoal $g_t \in \mathcal{G}$ and $\mathcal{M}(s_t)$ represents all macro-actions starting from state s_t. $\mathcal{M}^*(s_t) \subseteq \mathcal{M}(s_t)$ is the set of macro-actions m_t for state s_t that locally maximizes the reward:

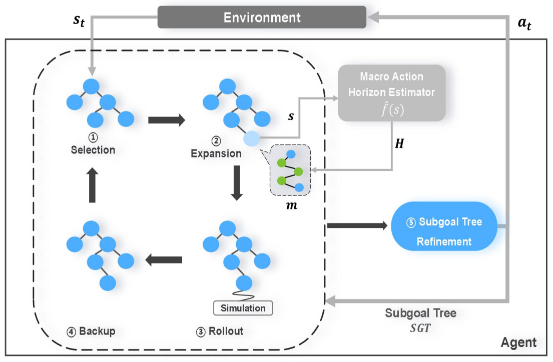


Fig. 1. The framework of UGSS

$$\mathcal{M}^*(s_t) = \left\{ \underset{m_t \in \mathcal{M}(s_t, g_t)}{\operatorname{argmax}} \overline{\mathcal{R}}(s_t, m_t) | g_t \in \mathcal{G}(s_t) \right\} \tag{3}$$

$\widehat{\mathcal{M}}(s_t) \approx \mathcal{M}^*(s_t)$ is generated incrementally by randomly sampling m_t from $\mathcal{M}(s_t)$. If a subgoal is rediscovered, the existing macro action m'_t will be replaced by m_t if $\overline{\mathcal{R}}(s_t, m_t) > \overline{\mathcal{R}}(s_t, m'_t)$. To reduce the macro action search space, macro-action coverage p_0 and tolerated error α are introduced. If we consecutively sample n macro-actions, whose subgoals are already known, and n $> log_{p_0}\alpha$, we will assume that a coverage of at least p_0 is achieved.

B. Upstream-Guided Subgoal Search (UGSS)

UGSS utilizes past planning experiences to estimate the macro-action (i.e., subgoal) horizon through Gaussian Process, which eliminates the need to predefine subgoal predicates. Besides, in UGSS, the subgoal tree is refined after several iterations to reduce the subgoal search space. The main framework of UGSS is shown in Fig.1.

Subgoal expansion with adaptive horizons. UGSS utilizes the experience in upstream planning to gradually adjust macro-action horizons. Upstream planning is defined as all steps $[1, ..., t_{cur} - 1]$ before the current step $t_{cur} \in [1, \text{T}]$. Each upstream step carries out MCTS search and may have visited the state $s = s_{t_{cur}}^{t_{leaf}}$ to expand the macro action during the process of generating the subgoal tree. The current horizon $H_{t_{cur}}^{s}$ can be calculated as the average historical macro-action horizon of the state $s = s_{t_{cur}}^{t_{leaf}}$:

$$H_{t_{cur}}^{s} = \frac{\sum_{t=1}^{t_{cur}-1} \overline{|m_t^S|}}{t_{cur}-1},\ 1 \leq t_{cur} \leq t_{leaf} \leq T \tag{4}$$

where $\overline{|m_t^S|}$ is the average horizon of macro action expanded at the state s at upstream planning step t. For continuous state space, an estimator $\hat{f}(s)$ is used to estimate the macro-action horizon $\widehat{H}$. UGSS uses the Gaussian Process (GP) [26] as the approximator. Specifically, the dataset $D_H = \{(s, H^S) | s \in TreeNodes_{upstream}\}$ is collected from upstream steps. After observing the historical state S and the corresponding horizon

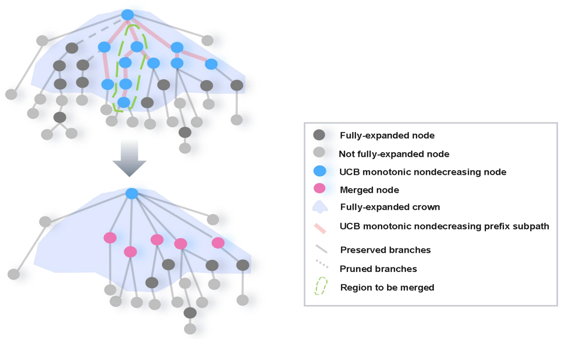


Fig. 2. Subgoal tree refinement

H^S, for a new state S_*, the corresponding horizon distribution f_* can be estimated using the posterior probability:

$$p(f_*|S_*, S, H^S) = \int p(f_*|S_*, f)p(f|S, H^S)df \quad (5)$$

$$p(f_*|S_*, S, H^S) = \mathcal{N}(f_*|\mu_*, \Sigma_*) \quad (6)$$

The mean of the distribution can be used as the estimated horizon $\widehat{H}$ and the standard deviation can be used as the uncertainty $\widehat{H_u}$:

$$\widehat{H} = \hat{f}(s) = \mu_*, \quad \widehat{H_u} = \sqrt{(\Sigma_*)} \quad (7)$$

A horizon threshold H_u is introduced to reduce the influence of uncertainty. When $\widehat{H_u} > H_u$, the estimator directly outputs $\widehat{H} = 1$, so the macro-action becomes the primitive action. Thus, by using the Gaussian process estimator $\hat{f}(s)$ to predict the boundaries of macro-action horizon $\widehat{H}$, the process of predefining subgoal predicates is eliminated.

Subgoal Tree Refinement. Due to the uncertainty of the macro-action horizons, the subgoal tree may have many dense subgoal nodes linked by short macro-actions, which leads to large subgoal search space. Therefore, subgoal tree refinement merges nodes and prunes the low-quality tree after several MCTS iterations so as to focus on evaluating high-value subgoals. The refinement is shown in Fig. 2.

- **Search.** Firstly, searching for a fully-expanded crown, which a subtree where the root node is the same with the original tree and all nodes are fully-expanded nodes.
- **Merge.** Going down from the root until reaching the leaf node of the crown. For each path, merging all nodes on the longest "UCB monotonic non-decreasing prefix subpath" $P_{prefix} = \langle s_{t_1}, m_{t_1}, s_{t_2}, m_{t_2}, ..., s_{t_k}, m_{t_k} \rangle$ into a large node. The macro-action is the primitive action trajectory represented by all edges on the subpath and the subgoal state is the terminal state.
- **Prune**. The remaining branch parts of the original tree crown that do not belong to the longest "continuous non-decreasing prefix subpath" are deleted.

Pruning ineffective branches from the subgoal tree allows subsequent iterations to focus on evaluating high-quality subgoals. The principle of "UCB monotonic non-decreasing"

Algorithm 1: Upstream-Guided Subgoal Search (UGSS)

Input: planning time limit T, MDP F, subgoal coverage p_0, tolerant error α, macro-action horizon uncertainty H_u, iteration budget b_0, refinement interval b_r.

Output: the planned original sequence of actions $\tau = \langle a_1, ..., a_T \rangle$.

```
1. Initialize the macro-action horizon experience pool $D_H \leftarrow \emptyset$;
2. for $t \leftarrow 1$ to $T$ do
3.    Initialize the subgoal tree $SGT$; $s \leftarrow s_t$
4.    for $b_n \leftarrow 1$ to $b_0$ do
5.        $s \leftarrow SGT.Selection(s)$
6.        $\widehat{H} \leftarrow \hat{f}(s, D_H)$
7.        $SGT,\ g \leftarrow SGT.AdaptiveExpand(SGT, s, p_0, \alpha, H_u)$
8.        $r \leftarrow SGT.Rollout(g, F)$
9.        $SGT \leftarrow SGT.Backup(SGT, r)$
10.       if $b_n \% b_r == 0$ do
11.           $SGT, D'_H \leftarrow SGT.Refine(SGT)$
12.           $D_H \leftarrow D_H \cup D'_H$
13.       end if
14.   end for
15.   $m_t \leftarrow SGT.BestChild(s_t)$
16.   $a_t \leftarrow m_t(s_t)$
17.   $\tau \leftarrow \tau + m_t$
18. end for
19. return $\tau$
```

makes sense since the UCB level of each node on the subpath surpasses the average of all the child nodes of its parent node.

The UGSS Algorithm. As shown in Algorithm 1, the agent builds a subgoal search Tree from the root node $s_{t_{cur}}$ in every planning step (Line 3). The refinement is executed every b_r iterations (Lines 10-13) of "selection, simulation, rollout, backpropagation" (Line 4-14). The high level planner outputs the best macro action (Line 15). The Low level planner outputs primitive actions (Line 16). The collected data for node and macro-action horizon (Line 17) are added into historical experience pool.

V. Communication-Free Decentralized Subgoal Tree Search

In this section, we propose the expectation alignment-based subgoal tree search, enabling agents to coordinate without communication. Fig. 3 shows that the framework of communication-free decentralized subgoal tree search method (CF-Dec-SGTS), which iteratively performs two phases on each agent: 1) local subgoal tree search; 2) team expectation update. The high-level planner first initializes a local Subgoal Tree (SGT), where each node represents a subgoal state. In the local Subgoal tree search (SGTS) step, the high-level planner iteratively generates SGTS based on the expectation e^i of the teammates' action intentions, and selects an optimal subgoal sequence $\mathcal{G}^{i*}$. The low-level planner will output the original action a_t^i that can be executed based on the current state and the sequence of subgoals. Each agent i infers the joint action $A_t^{(i)}$ of its teammates and updates the expectation distribution e^i of the team's planning intentions.

A. UGSS Generates Subgoal Tree

Each takes the current state as the root node and iteratively executes the four stages of MCTS: selection, expansion,

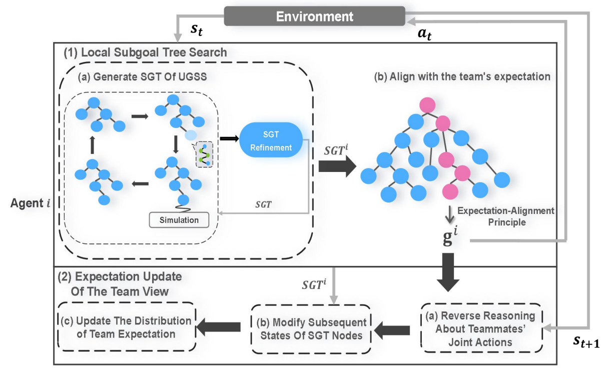

Environment
s_t
a_t
(1) Local Subgoal Tree Search
(a) Generate SGT Of UGSS
(b) Align with the team's expectation
SGT Refinement
SGT^i
SGT
Simulation
Agent i
Expectation-Alignment Principle
$\mathbf{g}^i$
(2) Expectation Update Of The Team View
SGT^i
(c) Update The Distribution of Team Expectation
(b) Modify Subsequent States Of SGT Nodes
(a) Reverse Reasoning About Teammates' Joint Actions
s_{t+1}

Fig. 3. The framework of expectation-alignment based communication-free decentralized subgoal tree search (CF-Dec-SGTS)

rollout and backpropagation. In the expansion step, the macro action duration of leaf nodes is predicted based on upstream planning data. After several MCTS iterations, a subgoal tree refinement is performed to help focus on the evaluation of high-value subgoal nodes.

Local Utility Function. We use a local utility function f^i to calculate the rewards obtained by backpropagation:

$$f^i(\mathbf{g}) := o\left(\mathbf{g}^i \cup \mathbf{g}^{(i)}\right) - o\left(\mathbf{g}^i_{\emptyset} \cup \mathbf{g}^{(i)}\right) \tag{8}$$

$o\left(\mathbf{g}^{(i)}\right)$ is the default reward of agent i without executing the subgoal sequence $\mathbf{g}^i_{\emptyset}$. Since f^i is not affected by teammates' planning, using local utility function f^i instead of the real global system reward o can accelerate the convergence of multi-agent collaboration.

B. Aligning With Team Expectation

The optimal subgoal sequence $\mathbf{g}^i := < g^i_1, g^i_2, \ldots g^i_T >$ is selected according to UCB after searching the subgoal tree. Agent i cannot get the real team expectation due to the lack of communication, so it needs to maintain a local team expectation distribution $e^i = (\tau^i, q^i)$, where $\tau^i = < s^i_1, s^i_2, \ldots, s^i_T >$ represents the expected environment state after agent i performing action at different times. $q^i \in [0,1]$ represents the probability of the trajectory.

We define expectation-align principle to replace the UCB maximum principle. The expectation-align value (EAV) for each node consists of the UCB value and the expectation distance (ED) value, which can be calculated as:

$$\begin{aligned} &EAV^i\left(s_t, m_t, s_{t+|m_t|}\right) \\ &= (1-\beta_{ED}) * UCB^i(s_t, m_t) + \beta_{ED} * ED^i\left(s_{t+|m_t|}, e^i_{t+|m_t|}\right) \\ &= (1-\beta_{ED}) * \left(Q(s_t, m_t) + c\sqrt{\frac{2\log\left(N(s_t)\right)}{N(s_t, m_t)}}\right) \\ &+ \beta_{ED} * \sum_{\left(s^i_{t+|m_t|}, q^i\right) \in e^i_{t+|m_t|}} q^i * \left\| s_{t+|m_t|} - s^i_{t+|m_t|} \right\|_2 \end{aligned} \tag{9}$$

$\beta_{ED} \in [0,1]$ is the alignment coefficient. $Q(s_t, m_t)$ is the accumulated rewards of the node. $c \in [0,1]$ is the exploration coefficient. N represents the number of visits. $s_{t+|m_t|}$ is the child node of executing the macro-action m_t from the parent node s_t. $e^i_{t+|m_t|}$ is expected state distribution of the team for agent i at $t + |m_t|$. Here, the expectation distribution e^i for agent i is initialized randomly and then updated during the team's expectation update. After updating the optimal subgoal sequence $\mathbf{g}^i$, the high-level planner outputs the optimal macro action m^i_t of the root node, and the low-level planner outputs the original action a^i_t that can be executed according to m^i_t.

Infer Teammates' Joint Actions. A_t is the executed joint action. Each agent i cannot get the real joint action $A^{(i)}_t$, but can approximate $\widetilde{A}^{(i)}_t$ based on the difference $\Delta s^{(i)}_{t+1}$ of the environment state:

$$\widetilde{A}^{(i)}_t \cong \Delta s^{(i)}_{t+1} = s_{t+1} - F(s_t, a^i_t) \tag{10}$$

$F(s_t, a^i_t)$ is the dynamic model of the environment, which calculates the next environmental state without the actions of its teammates according to the environmental state s_t and the action a^i_t. $\Delta s^{(i)}_{t+1}$ is the utility of the joint actions of the agents except agent i on the environmental state.

Update the Distribution of Team's Expectation. $A^{(i)}_t$ is the optimal joint action as these agents consider the strategies agent i may execute during planning. Therefore, after agent i deducing the joint action $\widetilde{A}^{(i)}_t$ of its teammates, it can infer the team's expectation $\tau^i_{new} = < s^i_t, s^i_{t+1}, \ldots, s^i_T >$, and update the team's expectation distribution $e^i = (\tau^i, q^i)$.

Taking $\widetilde{A}^{(i)}_t$ into consideration, traversing the path $\tau^i = < s^i_t, s^i_{t+1}, \ldots, s^i_T >$ from the root node s_t, then it updates the successor state s' of each node, and recalculates the accumulated reward R from the root node to the leaf node. The path with the largest R is the new team's expected path τ^i_{new}:

$$\tau^i_{new} = argmax_{\tau^i \in SGT} R(\tau^i) \tag{11}$$

The team's expectation distribution $e^i = (\tau^i, q^i)$ is updated after the τ^i_{new} obtained. Here, we use the distributed Lagrangian steepest descent framework:

$$\begin{aligned} q^i(\tau^i) = q^i(\tau^i) - \eta q^i(\tau^i) \Bigg[&\frac{E_q[R^i(s^i_t)] - E_q[R^i(s^i_t)|\tau^i]}{\beta} \\ &+ H(q^i) + \ln\left(q^i(\tau^i)\right) \Bigg] \end{aligned} \tag{12}$$

q^i is the distribution probability of team's expectation for agent i. η is the update step constant. E_q the average accumulated reward after t. β is the scaling factor (usually set to $= max\left(e^{-(t+1)}, \beta_d\right)$), β_d is minimum threshold). H is the entropy.

VI. Experimental Evaluation

We evaluate the proposed algorithm on the classical agent *Box-Pushing problem* [30] in a 2D grid world. As shown in Fig. 4, there are n agents and they need to complete M box-pushing

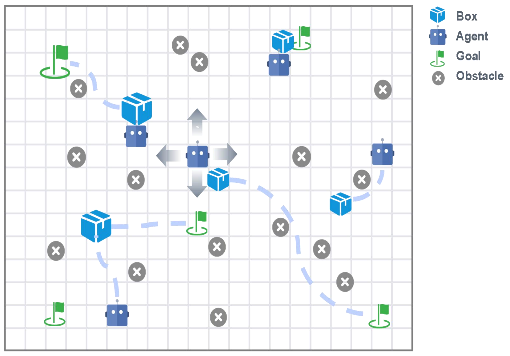


Fig. 4. Box-Pushing problem

tasks, that is, pushing each *box* to its *goal* and avoid K *obstacles* within time limit T. Each box can only be pushed toward the goal or an open field. Based on the local observation, the agent chooses among five primitive actions: *walk* or *push the box in four directions*.

A. Validating the UGSS Algorithm for Single Agent Planning

The state space S contains the locations of the agent, boxes, goals, obstacles within the agent's observation, and A is the primitive action space. R is composed of three parts, the reciprocal of the sum of the distances between all the boxes and their goals, the number of successfully pushed boxes and an indicator function about whether all the tasks are completed. All computations are performed on a 64-bit workstation with 32 GB RAM and an 8-core 3.6 GHz processor. All records are averaged over 40 instances, and use standard errors as the confidence intervals. The parameter settings of scenarios are shown in Table 1 and 2.

TABLE 1. PARAMETER SETTINGS OF DIFFERENT SINGLE AGENT SCENARIOS

Scenario Id	Number of boxes M	Number of obstacles K	Planning steps T
1	3	10	300
2	5	15	800

TABLE 2. PARAMETER SETTINGS OF UGSS

Setting Id	p_0	α	H_u	b_0	b_r
1	0.8	0.001	T/2	10000	1
2	0.8	0.001	T/2	10000	100
3	0.8	0.001	T/2	10000	500
4	0.8	0.001	T/2	10000	∞(1e9)

Comparison Methods. We compare our algorithm UGSS with the following baseline solutions: MCTS [7] and S-MCTS [9].

Average accumulated rewards. Fig.5 shows that our UGSS can achieve the highest average accumulated rewards without human experiences in both scenarios 1 and 2. Without hierarchical abstraction, MCTS wastes a lot of steps in short-distance exploration and primitive action evaluation and fails to complete the task. S-MCTS introduces subgoal search and completes all tasks. UGSS spends more time on predicting macro-action horizons in the early stage of planning as it doesn't need to predefine subgoal predicates, so it does not immediately obtain high accumulated rewards in the early stage of planning. As the GP estimator gradually obtains sufficient data to predict horizons and find the subgoals, UGSS can complete tasks of all boxes finally.

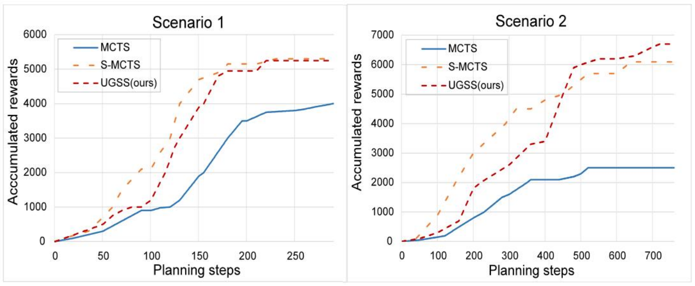


Fig. 5. Average accumulated reward for different single agent scenarios

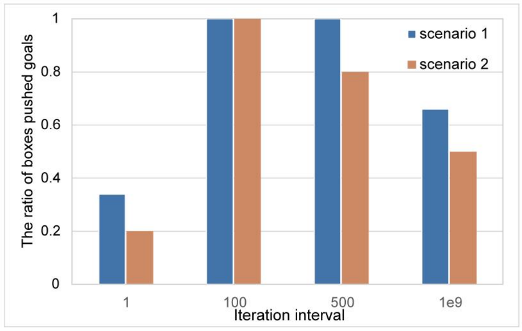
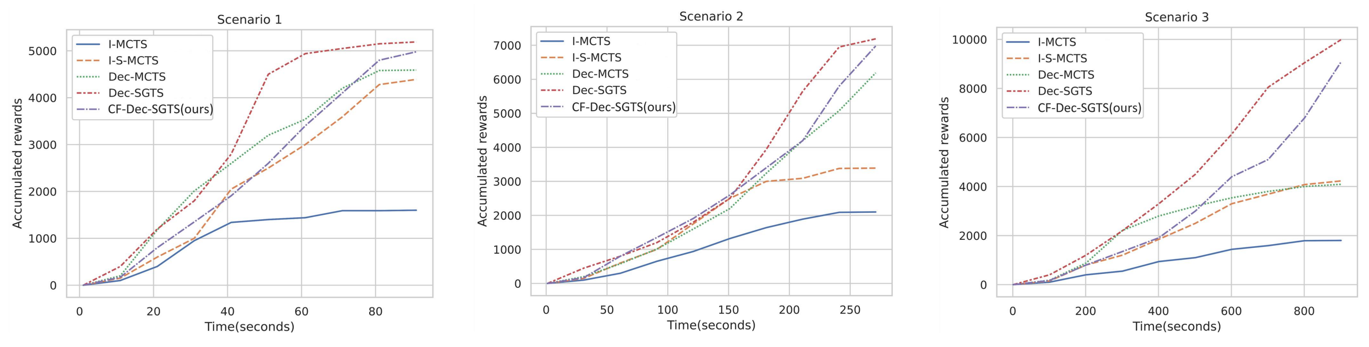


Fig. 6. Effect of iteration interval on task completion

Ablation experiments. It can be seen from Fig. 6 that UGSS with an interval of 100 iterations works best, completing nearly 100% tasks. The impact of refining intervals is more pronounced in larger scale scenarios. In scenario 2, only 80% of the tasks can be completed when the refining interval is 500. As the iteration budget b_0 is 10000, the tree can only be refined 20 times, which results in fewer and shorter macro-actions. The completion rate is only 50% for ∞ (1e9) iterations as UGSS degenerates into non-hierarchical MCTS due to no refinement. UGSS with an interval of 1 iteration works worst, which is only 20% because the node has just fully expanded but has not sufficiently evaluated, making it unreliable to merge them according to the principle of UCB monotonicity. In summary, subgoal tree refinement with appropriate interval reduces the subgoal search space and help to focus on subgoal with high quality.

B. Validating the CF-Dec-SGTS Algorithm for Multiagent Planning

Comparison Methods. We compare our algorithm communication-free Dec-SGTS (CF-Dec-SGTS) with the I-MCTS [9], I-S-MCTS [10], Dec-MCTS with communication [3], Dec-SGTS [2]. The parameters of scenarios with different scales are given in Table 3, the characteristics of different methods are given in Table 4, and the parameter of CF-DEC-SGTS method are given in Table 5.



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Fig. 7. Average accumulated reward for scenarios of different sizes

TABLE 3. PARAMETER SETTINGS OF THE DIFFERENT MULTIAGENT SCENARIOS

Id	length and width $L * W$	Number of agents N	Number of boxes M	Number of obstacles K	Planning steps T
1	600*400	3	3	10	150
2	600*400	3	5	15	400
3	1200*800	5	10	20	1000

TABLE 4. CHARACTERISTICS OF COMPARISON METHODS.

Method	Communication	Coordination	Hierarchical
I-MCTS	✗	✗	✗
I-S-MCTS	✗	✗	✓
Dec-MCTS	✓	✓	✗
Dec-SGTS	✓	✓	✓
CF-Dec-SGTS (ours)	✗	✓	✓

TABLE 5. PARAMETER SETTINGS OF CF-DEC-SGTS

Id	β_d	η	β_{ED}	p_0	α	H_u	b_0	b_r
1	0.0001	0.1	0.0	0.80	0.001	T/2	10000	100
2	0.0001	0.1	0.5	0.80	0.001	T/2	10000	100
3	0.0001	0.1	0.5	0.80	0.001	T/2	50000	500

Average accumulated rewards. Fig. 7 shows that our CF-Dec-SGTS can achieve desirable average accumulated reward in all three scenarios. Due to the lack of communication and hierarchical abstraction, I-MCTS method cannot complete the transportation task of all boxes within the specified time by independent short-distance exploration. I-S-MCTS can complete the box-pushing task in small-scale scenario 1, where each agent does not need to consider cooperation with the teammates. However, in the complex scenario 2 and large-scale scenario 3, this method cannot work since a single agent may need to push multiple boxes, and a box may also require multiple agents to cooperate. Dec-MCTS can complete all the box-pushing tasks in scenarios 1 and 2 due to communication. However, in large-scale scenario 3, this method lacks macro planning and focus on evaluating a large number of state nodes in short distances and primary action. Dec-SGTS, which combines communication and hierarchical abstraction of subgoals, is able to complete all the box-pushing tasks of different scenarios and obtain high rewards finally.

Our CF-Dec-SGTS can complete all the box-pushing tasks without communication. CF-Dec-SGTS needs to spend much time in the early exploration because it does not know the planning intention of the teammates. As it gradually estimates the expectation of the team for its state trajectory, CF-Dec-SGTS tries to align with the expectation of the team while completing its own tasks, and makes its planning stable and predictable. Therefore, CF-Dec-SGTS can finally obtain a high reward and approach 90% of method with communication.

Ablation experiments. We conduct experiments under different scale scenarios and different β_{ED} to evaluate the task completion rate. β_{ED} regulates the degree to which the selection of the best child aligns with the team's expectations, with higher values placing more emphasis on choosing a state that matches the team's expectations (making their planning intentions predictable), and lower values placing more emphasis on completing tasks independently. From Fig. 8, we can observe that expectation alignment is important for team coordination in different scenarios. The performance when the expectation alignment coefficient β_{ED}=0.5 outperforms other circumstances, and the task completion is 100%. $\beta_{ED} = 0$ means that it does not consider the team expectation alignment, and assumes that other agents perform actions randomly, so it cannot complete the task that requires cooperation. When $\beta_{ED} = 0.2$, it can complete the task under small-scale scenario 1, but due to the lack of cooperation with the teammates, it cannot complete all the box pushing tasks in large-scale scenarios. When $\beta_{ED} = 1.0$, the multi-agent cooperative box-pushing task cannot be accomplished because it only considers the state that meets the team's expectation.

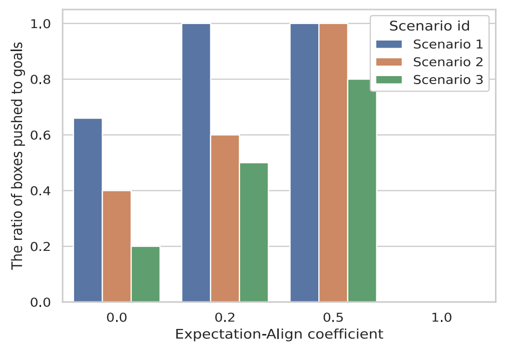


Fig. 8. Effect of expectation-align coefficient on task completion

In summary, the expectation alignment technique has a positive effect on the decentralized multiagent planning without communication. In comparison with the communication-free decentralized MCTS benchmarks (i.e., I-S-MCTS), the proposed CF-Dec-SGTS achieves a 20% increase in cumulative reward, which is nearly 90% of the cumulative reward of the optimum Dec-SGTS based on over communication.

VII. Conclusions

This paper proposes a generalized Dec-SGTS method without subgoal priors or communication between agents. During subgoal tree search of each agent, the proposed UGSS not only avoid the manual definition of the subgoal predicates, but also reduces the search space of the subgoals. During decentralized multiagent planning, each agent uses expectation alignment to align its policy with the team's expectations. The joint action of other agents and their expectations can be directly inferred from the difference information of the environmental state before and after joint-action. Integrating UGSS and expectation-alignment technique, the proposed CF-Dec-SGTS allows the agents to learn collaborative behaviors without any priors on subgoals and communication between agents. Extensive experimental results on single agent subgoal tree search and multiagent planning demonstrate that the proposed CF-Dec-SGTS can achieve multiagent cooperation and obtain desirable accumulated rewards without subgoal priors or communication.

References

1. Shushman Choudhury, Jayesh K. Gupta, Peter Morales, and Mykel J. Kochenderfer, "Scalable Anytime Planning for Multi-Agent MDPs", *AAMAS '21*, pp.341–349,2021.
2. Carlos Guestrin, Daphne Koller, Ronald Parr, "Multiagent Planning with Factored MDPs," *NIPS'01*, pp.1523-1530,2002.
3. Graeme Best, Oliver M Cliff, Timothy Patten et al., "Dec-MCTS: Decentralized Planning for Multi-Robot Active Perception", *International Journal of Robotics Research*, 38(2-3): 316-337, 2019.
4. Wanyuan Wang, Yichuan Jiang, Weiwei Wu, "Multiagent-based Resource Allocation for Energy Minimization in Cloud Computing Systems", I*EEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(2): 205-220, 2016.
5. Tianshu Chu, Jie Wang, Lara Codecà, et al., "Multi-Agent Deep Reinforcement Learning for Large-Scale Traffic Signal Control," *IEEE Transactions on Intelligent Transportation Systems*, 21(3):1086-1095, 2020.
6. Dimitrios Troullinos, Georgios Chalkiadakis, Ioannis Papamichail, and Markos Papageorgiou, "Collaborative Multiagent Decision Making for Lane-Free Autonomous Driving," *AAMAS'21*, pp.1335–1343, 2021.
7. Levente Kocsis and Csaba Szepesv´ ari, "Bandit based Monte-Carlo Planning", *ECML'06*, pp.282-293, 2006.
8. David Silver, Aja Huang, Chris J. Maddison et al, "Mastering the Game of Go with Deep Neural Networks and Tree Search," *Nature*, 529, 484–489, 2016.
9. Thomas Gabor , Jan Peter , Thomy Phan , Christian Meyer et al., "Subgoal-Based Temporal Abstraction in Monte-Carlo Tree Search", *IJCAI'19*, pp. 5562-5568, 2019.
10. Konrad Czechowski, Tomasz Odrzygóźdź, Marek Zbysiński et al, "Subgoal search for Complex Reasoning Tasks", *NeuIPS'21*, pp.624-638, 2021.
11. Aijun Bai, Siddharth Srivastava, and Stuart J. Russell "Markovian State and Action Abstractions for MDPs via Hierarchical MCTS", *IJCAI'16*, pp.3029-3039, 2016
12. Karl Pertsch, Oleh Rybkin, Frederik Ebert, "Long-Horizon Visual Planning with Goal-Conditioned Hierarchical Predictors", *NeurIPS'20*, pp.17321-17333, 2020.
13. Wang W, Jiang J, An B, et al. Toward efficient team formation for crowdsourcing in noncooperative social networks[J]. IEEE transactions on cybernetics, 47(12): 4208-4222,2016.
14. Sushmita Bhattacharya, Siva Kailas, Sahil Badyal et al., "Multiagent Rollout and Policy Iteration for POMDP with Application to Multi-Robot Repair Problems," CoRL'20, pp.1814-1828, 2020.
15. Fouad Sukkar, Graeme Best, Chanyeol Yoo et al., "Multi-Robot Region-of-Interest Reconstruction with Dec-MCTS", *ICRA'19*, pp.9101–9107, 2019.
16. Aleksander Czechowski and Frans A. Oliehoek, "Decentralized MCTS via Learned Teammate Models", *IJCAI'20*, pp.81-88, 2020.
17. Minglong Li , Wenjing Yang , Zhongxuan Cai et al., "Integrating Decision Sharing with Prediction in Decentralized Planning for Multi-Agent Coordination under Uncertainty", *IJCAI'19*, pp.450-456, 2019.
18. Qian Che, Wanyuan Wang, Fengchen Wang et al. "Structural Credit Assignment-Guided Coordinated MCTS: An Efficient and Scalable Method for Online Multiagent Planning", *AAMAS'23*, pp.543-551, 2023.
19. Karl Kurzer, Chenyang Zhou and J. Marius Zollner, "Decentralized Cooperative Planning for Automated Vehicles with Hierarchical Monte Carlo Tree Search," *IV'18*, pp.529-536, 2018.
20. Minglong Li, Zhongxuan Cai, Wenjing Yang et al, "Dec-SGTS: Decentralized Sub-Goal Tree Search for Multi-Agent Coordination", *AAAI'21*, pp.11282-11289, 2021.
21. Amy McGovern and Andrew G. Barto "Automatic Discovery of Subgoals in Reinforcement Learning using Diverse Density", *ICML'01*, pp.1-8, 2001
22. Dominik Jeurissen, Mark H.M. Winands, Chiara F. Sironi et al., "Automatic Goal Discovery in Subgoal Monte Carlo Tree Search", *CoG'21*, pp. 1-4, 2021.
23. Di Xue, Lei Yuan, Zongzhang Zhang et al., "Efficient Multi-Agent Communication via Shapley Message Value", IJCAI'22, pp.578-584, 2022.
24. Wang W, Wu G, Wu W, et al. Online collective multiagent planning by offline policy reuse with applications to city-scale mobility-on-demand systems[C]//Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems. 2022: 1364-1372.
25. Zixian Ma, Rose Wang, Fei-Fei Li, Michael S. Bernstein, Ranjay Krishna, "ELIGN: Expectation Alignment as a Multi-Agent Intrinsic Reward," NeurIPS'22, pp.1-9, 2022
26. Gaurav Dixit, Stéphane Airiau, Kagan Tumer, "Gaussian Processes as Multiagent Reward Models", *AAMAS'20,* pp.330-338, 2020.