2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Soft Task Planning with Hierarchical Temporal Logic Specifications

Ziyang Chen, Zhangli Zhou, Lin Li, and Zhen Kan

2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 979-8-3503-7770-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/IROS58592.2024.10801732

Abstract— **This works exploits soft constraints in linear temporal logic task planning to enhance the agent's capability in handling potentially conflicting or even infeasible tasks. Different from most existing works that focus on sticking to the original plan and trying to find a relaxed plan if the workspace does not permit, we augment the soft constraints to represent possible candidate sub-tasks that can be selected to fulfill the global task. Specifically, a hierarchical temporal logic specification is developed to represent LTL tasks with soft constraints and preferences. The hierarchical structure consists of an outer and inner layer, where the outer layer uses co-safe LTL to specify the task-level specifications and the inner layer specifies the low-level task-related atomic propositions via soft constraints. To cope with the hierarchical temporal logic specification, a hierarchical iterative search (HIS) algorithm is developed, which incrementally searches feasible atomic propositions and automaton states, and returns a task plan with minimum cost. Rigorous analysis shows that HIS based planning is feasible (i.e., the generated plan is applicable and satisfactory with respect to the task specification) and optimal (i.e, with minimum cost). Extensive simulation demonstrates the effectiveness of the proposed soft task planning approach.**

I. INTRODUCTION

Linear temporal logic (LTL) [1], as a formula language, has been widely applied to task representation and planning of agents [2]–[4]. However, these pre-specified constraints can be restrictive in practice. For instance, the task of cleaning a room can be accomplished by either using a vacuum or a mop. Instead of pre-specifying a particular cleaning tool, we hope the agent can choose an appropriate tool by jointly considering multiple factors, such as the workspace knowledge (e.g., the traveling distance to different cleaning tools), user preferences, etc. Hence, this work considers hard and soft constraints, where hard constraints should be enforced (e.g., avoid obstacles) while soft constraints offers the agent more flexibility in task and motion planning.

The idea of considering hard and soft constraints is not new and has been investigated in the literature (cf. [5]–[7] to name a few). Several metrics have been proposed to evaluate the task satisfaction with different soft constraints. For instance, the satisfaction of an atomic proposition is measured via time windows in [8], [9]. The works of [7], [10], [11] exploit the difference between the desired plan and the executed plan which is relaxed to meet the specification as much as possible. In [12], [13], the number of unrealized state jumps is considered as the violation of task specification. In [14], the disjunctive normal form is established and the satisfaction degree of each sub-formula is judged independently. However, these aforementioned results focus on sticking to the original plan and trying to find a relaxed plan that is mostly close to the original plan if the workspace does not permit. Few of them considers exploiting the soft constraints to represent possible candidate sub-tasks that can be selected to fulfill the global task. By introducing the preference, a variety of alternative sub-tasks are considered as soft constraints, which can be treated as a cost [15] or formulated as an optimization problem [9], [16]. Nevertheless, to facilitate the task planning, the alternative sub-tasks are still the atomic propositions labeled in the workspace, which limits the flexibility of selecting appropriate sub-tasks.

To enrich the expressivity of soft constraints and improve the flexibility of task planning, in this paper, a hierarchical temporal logic specification is developed to represent LTL tasks with soft constraints and preferences. Specifically, the hierarchical structure consists of an outer and inner layer, where the outer layer specifies the task-level specifications and the inner layer specifies the low-level task-related atomic propositions. That is, the atomic propositions of the outer formula are not directly labeled in the workspace, but can be satisfied by an inner formula with labeled atomic propositions in the workspace. The satisfaction degree of each inner formula for an outer atomic proposition are then evaluated to indicate the preference of soft constraints. To generate the task plan, the hierarchical iterative search (HIS) is developed for the proposed hierarchical temporal formula. HIS incrementally searches feasible atomic propositions and automaton states, and all feasible inner formulas are evaluated and sorted. By pruning, the search space can be effectively reduced. During planning for the outer formula, different planning results of inner formula can be provided according to different initial states, which cannot be realized by existed graph searching or optimization-based method. Extensive simulation demonstrates the effectiveness of the proposed soft task planning approach.

The contribution of this paper are summarized as follows. First, we develop a hierarchical temporal formula consisting of outer and inner temporal logic formulas, which can effectively express soft constraints and evaluate the completion of the task. Second, to cope with the hierarchical temporal formula, a hierarchical iterative search (HIS) algorithm is developed, which can effectively generate a feasible plan by searching and iteration in an incremental hierarchical tree. Third, rigorous analysis shows that the plan obtained by HIS is feasible (i.e., the generated plan is satisfactory with respect to the LTL task) and optimal (i.e., with minimum cost).

Z. Chen, Z. Zhou (Corresponding Author), L. Li, and Z. Kan are with the Department of Automation at the University of Science and Technology of China, Hefei, Anhui, China, 230026.

This work was supported in part by the National Natural Science Foundation of China under Grant 62173314 and U2013601.

II. PRELIMINARIES

As a formal language, LTL is defined over a set of atomic propositions AP with Boolean and temporal operators. The syntax of LTL is defined as:

$$\phi := true|ap|\phi_1 \wedge \phi_2|\neg\phi_1|X\phi|\phi_1 U \phi_2$$

where $ap \in AP$ is an atomic proposition, *true*, $\neg$ (negation), and $\wedge$ (conjunction) are propositional logic operators, and X (next) and U (until) are temporal operators. Other propositional logic operators such as *false*, $\vee$ (disjunction), $\rightarrow$ (implication), and temporal operators such as G (always) and F (eventually) can also be defined [1].

The word $\boldsymbol{\pi} = \pi_0\pi_1\ldots$ is an infinite sequence where $\pi_i \in 2^{AP}$, $\forall i \in \mathbb{Z}$. Given a word $\boldsymbol{\pi} = \pi_0\pi_1\ldots$, denote by $\boldsymbol{\pi}[j\ldots] = \pi_j\pi_{j+1}\ldots$ and $\boldsymbol{\pi}[\ldots j] = \pi_0\ldots\pi_j$. The semantics of LTL formulae are interpreted over $\boldsymbol{\pi}$, which are referred to [1]. As indicated in [17], given an LTL formula Φ and a word $\boldsymbol{\pi} = \pi_0\pi_1\ldots \in (2^{AP})^\omega$ satisfying $\boldsymbol{\pi} \models \phi$, $\boldsymbol{\pi}$ is said to have a good prefix if there exists $n \in \mathbb{N}$ and a truncated finite sequence $\boldsymbol{\pi}[\ldots n]$ such that $\boldsymbol{\pi}[\ldots n]\boldsymbol{\pi}[n\ldots] \models \phi$ for any infinite sequence $\boldsymbol{\pi}[n\ldots] \in (2^{AP})^\omega$. Such a formula ϕ is called a co-safe LTL formula, which can be translated into a non-deterministic finite automata (NFA) [18].

Definition 1. An NFA is a tuple $A = \{S, S_0, \Sigma, \delta, \mathcal{F}\}$, where S is a finite set of states, $S_0 \subseteq S$ is the set of initial states, $\Sigma = 2^{AP}$ is the finite alphabet, $\delta : S \times \Sigma \rightarrow 2^S$ is a transition function, and $\mathcal{F} \subseteq S$ is the set of accepting states.

Let $\Delta : S \times S \rightarrow 2^\Sigma$ denote the set of atomic propositions that enables state transitions in NFA, i.e., $\forall \pi' \in \Delta(s, s')$, $s' \in \delta(s, \pi)$. A valid run $\boldsymbol{s} = s_0s_1s_2\ldots$ of A generated by the word $\boldsymbol{\pi}$ with $\pi_{i+1} \in \Delta(s_i, s_{i+1})$ is called accepting, if $\boldsymbol{s}$ intersects with $\mathcal{F}$. A co-safe LTL formula can be translated to an NFA by the tool [19]. In this paper, NFA will be used to track the progress of the satisfaction of co-safe LTL tasks.

III. PROBLEM FORMULATION

Consider a bounded workspace $M \subset \mathbb{R}^2$. Let $L : M \rightarrow AP$ be a labeling function mapping an area in M to an executable atomic proposition and let $LM : AP \rightarrow M$ indicate the executable area of an atomic proposition. To specify robot tasks in M, co-safe LTL is employed in this work. For instance, the task of fetching a cleaning tool and then cleaning the room can be specified as a co-safe LTL formula $\phi = F(ap_1 \wedge Fap_2)$, where the mop (i.e., a cleaning tool) and the room to be cleaned in the workspace M are labeled with ap_1 and ap_2, respectively. Since the task ϕ only requires fetching a cleaning tool, if there exist multiple cleaning tools (e.g., a vacuum and a mop) stored in different locations, either fetching a vacuum or a mop before room cleaning should all satisfies the task ϕ. Hence, instead of specifying a position dependent task as in many existing works (e.g., fetching a mop by visiting the specific area labeled with ap_1), we propose to extend ap_1 with soft constraints. That is, ap_1 can be mapped to a set of areas in M to reflect all feasible candidate sub-tasks. These labeled areas can also be sorted by preference, such as using the preference measures in [15], [20].

To this end, we decouple the tasks and their executable positions by introducing a hierarchical structure to represent tasks with soft constraints and preferences, where the outer layer specifies the task while the inner layer specifies the task related executable positions. Specifically, given a set of atomic propositions AP, we construct a set of inner and outer atomic propositions AP_{in} and AP_{out}, where AP_{out} is only task dependent and AP_{in} is position dependent labeling the executable areas in the workspace, i.e., $LM : AP_{in} \rightarrow M$. Based on AP_{in} and AP_{out}, the co-safe LTL formula ϕ_{in} and ϕ_{out} can be constructed accordingly, where the outer task ϕ_{out} indicates the task specifications. Let ϕ^*_{in} denote the set of all inner tasks that satisfies the outer atomic propositions and each $\phi_{in} \in \phi^*_{in}$ represents an inner task defined over AP_{in}. That is, an $ap_{out} \in AP_{out}$ can be satisfied by completing an arbitrary inner task $\phi_{in} \in \phi^*$ and its satisfaction degree can be evaluated by $E : AP_{out} \times \phi^*_{in} \rightarrow [0, 1]$. Specifically, $E(ap_{out}, \phi_{in})$ indicates the satisfaction degree of ap_{out} if the inner task ϕ_{in} is executed. Hereafter, we denote by (ϕ_{out}, E) a hierarchical LTL formula, where ϕ_{out} is an outer formula and E is the evaluation function.

Example 1. Given the cleaning task ϕ, we define $AP_{out} = \{ap^1_{out}, ap^2_{out}\}$ and $\phi_{out} = F(ap^1_{out} \wedge Fap^2_{out})$, where ap^1_{out} and ap^2_{out} indicate the task of fetching a cleaning tool and the task of room cleaning, respectively. Suppose there are four areas of interest in the workspace, i.e., the carpet cleaner ap^1_{in}, the carpet spray ap^2_{in}, the vacuum ap^3_{in}, and the room to be cleaned ap^4_{in}. The related sub-tasks for AP_{out} are defined as follows: $\phi_1 = Fap^1_{in} \wedge Fap^2_{in}$ indicates the task of fetching the carpet cleaner and spray, $\phi_2 = Fap^3_{in}$ indicates the task of taking the vacuum, and $\phi_3 = Fap^4_{in}$ indicates the task of visiting and cleaning the room. The evaluation function is then defined as $E(ap^1_{out}, \phi_1) = 0.9$, $E(ap^1_{out}, \phi_2) = 1$, and $E(ap^2_{out}, \phi_3) = 1$, which indicates that ap^1_{out} has a soft constraint as it can be satisfied by either completing ϕ_1 or ϕ_2, and ap^2_{out} has a hard constraint as it can only be satisfied by completing ϕ_3.

The plan of a hierarchical formula (ϕ_{out}, E) in M is defined as $\Pi = (\boldsymbol{\pi}_{out}, \boldsymbol{\pi}_{in}, \boldsymbol{s}, \phi)$, where $\boldsymbol{\pi}_{out} = \pi^{out}_0\pi^{out}_1\ldots$ and $\boldsymbol{\pi}_{in} = \boldsymbol{\pi}^{in}_0\boldsymbol{\pi}^{in}_1 = \pi^0_0\pi^1_0\pi^1_1\pi^1_2\ldots$ are the words corresponding to the outer and inner propositions, respectively, where $\pi^{out}_i \in AP_{out}$ and $\pi^i_j \in AP_{in}$ indicates the jth element of $\boldsymbol{\pi}^{in}_i$, $i = 0, 1, \ldots$. Let $A_{out} = \{S_{out}, S_{out0}, \Sigma_{out}, \Delta_{out}, \mathcal{F}_{out}\}$ denote the NFA of ϕ_{out} and $\boldsymbol{s} = s_0s_1s_2\ldots$ indicates a run of A_{out}. Let $\boldsymbol{\phi} = \phi_0\phi_1\phi_2\ldots$ denote the sequence of inner tasks. Note that π_0 indicates the empty outer proposition, s_0 indicates the starting state in S_{out0}, ϕ_0 indicates the empty formulas. Specially, to connect two outer propositions π^{out}_{i-1} and π^{out}_i, the initial inner proposition π^i_0 of $\boldsymbol{\pi}^{in}_i$ is defined to map to the end proposition of $\boldsymbol{\pi}^{in}_{i-1}$, i.e., $LM(\pi^i_0) = LM(\pi^{i-1}_{|\boldsymbol{\pi}^{in}_{i-1}|})$. Then, for $i > 0$, $s_i \in S_{out}$ indicates the state of A_{out} after the execution of π^{out}_i. A plan Π satisfying (ϕ_{out}, E) in M

is denoted as $\Pi \models (\phi_{out}, E, M)$. Specifically, if $\pi_i^{out} \in \Delta(s_{i-1}, s_i)$ and $E(\pi_i^{out}, \phi_i) > 0$, $\forall i \in \mathbb{N}^+$, and $\boldsymbol{\pi}_{out}$ is accepting for ϕ_{out}, then $\Pi \models (\phi_{out}, E, M)$. Note that Π consists of a series of plan tuples $\Pi_i = (\pi_i^{out}, \boldsymbol{\pi}_i^{in}, s_i, \phi_i)$, denoted as $\Pi = \Pi_0\Pi_1\Pi_2\ldots$. The cost of Π is defined as

$$\mathsf{Cost}(\Pi) = \sum_{i\in\mathbb{N}^+} \left\{\alpha(1 - E(\pi_i^{out}, \phi_i)) + D(\boldsymbol{\pi}_i^{in})\right\}, \quad (1)$$

where

$$D(\boldsymbol{\pi}_i^{in}) = \sum_{\pi_j^i \in \boldsymbol{\pi}_i^{in}, j>0} \|LM(\pi_{j-1}^i) - LM(\pi_j^i)\|. \quad (2)$$

In (1), $\alpha(1 - E(\pi_i^{out}, \phi_i))$ evaluates the dissatisfaction cost, where $\alpha \in \mathbb{R}^+$ is a tuning weight indicating the relative importance. The term $D(\boldsymbol{\pi}_i^{in})$ is the traveling cost for inner task ϕ_i. The feasible plan with minimum cost is considered as the optimal plan in this work. Then, the planning problem for the proposed hierarchical temporal formula specification is stated as follows.

Problem 1. For an environment M and a hierarchical temporal formula specification (ϕ_{out}, E), the goal is to obtain an optimal plan $\Pi \models (\phi_{out}, E, M)$ with minimum $\mathsf{Cost}(\Pi)$.

IV. TASK PLAN

Due to undetermined inner tasks (i.e., soft constraints), Problem 1 cannot be solved by graph-search based algorithms as they require a complete and deterministic product automaton. To address this challenge, a hierarchical iterative search (HIS) is developed to incrementally construct the plan Π. As shown in Fig. 1, HIS first searches the NFA states and atomic propositions for ϕ_{out} to construct a tree to track the task progress. Based on the current system states and selected $ap_{out} \in AP_{out}$, HIS further searches inner formulas $\phi_{in} \in \phi^*$ satisfying $E(ap_{out}, \phi_{in}) > 0$. The plan for ϕ_{in} can be obtained by task planner in [21]–[23] and the system state and cost are updated accordingly. The feasible plan can then be obtained from the tree to guide the motion of robot.

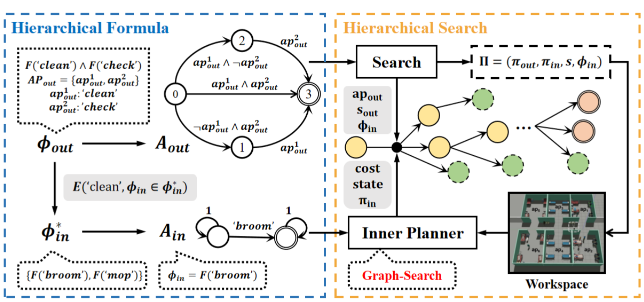

Fig. 1. The framework of hierarchical iterative search (HIS). Given an outer formula ϕ_{out}, a set of inner formula ϕ_{in} and the evaluation function E, the hierarchical formula (ϕ_{out}, E) can be constructed. Based on the NFA A_{out}, the feasible outer atomic propositions and inner task can be searched and construct a node to indicate the task progress. Then, based on the searched inner task, the inner planner generates the plan with minimum cost, which guides the iteration of system states and pruning of nodes. After search, a plan with minimum cost for outer task will be obtained as Π.

The searching tree $T = \{d_0, d_1, d_2, \ldots\}$ is defined as a set of nodes $d_i = \{s_i^T, \pi_i^{Tout}, \boldsymbol{\pi}_i^{Tin}, \phi_i^T, f_i, p_i, c_i\}$ where

- $s_i^T \in S_{out}$ is the NFA state corresponding to ϕ_{out};
- $\pi_i^{Tout} \in AP_{out}$ is an atomic proposition of ϕ_{out};
- $\boldsymbol{\pi}_i^{Tin}$ indicates the sub-tasks performed for π_i^{Tout};
- ϕ_i^T is the selected inner formula for π_i^{Tout};
- $f_i \in T$ is the father node of d_i.
- p_i is the robot position after performing π_i^{Tout};
- c_i is the cost after performing π_i^{Tout}.

Based on the tree T, the planner HIS is developed and Alg. 1 outlines how the tree T is constructed. After generating the NFA A_{out} of ϕ_{out}, T is first initialized as the root node d_0 with $\mathsf{Flag}(d_0) = 0$ indicating that d_0 has not generated child nodes yet. The algorithm ends if all nodes have generated child nodes. For node d_i satisfying $\mathsf{Flag}(d_i) = 0$, the NFA state s and ap_{out} satisfy $ap_{out} \in \Delta(s_i^T, s)$. Let $S_{pre}(d_i)$ be the set of searched NFA states in the node path from d_0 to d_i, which satisfies $S_{pre}(d_i) = S_{pre}(f_i) \cup \{s_i^T\}$ and $S_{pre}(d_0) = \emptyset$. To reduce the search space, HIS only selects unexplored NFA states for d_i, i.e., $\forall s \in S_{out} - S_{pre}(d_i)$. The inner formula $\phi_{in} \in \phi_{in}^*$ satisfying $E(ap_{out}, \phi_{in}) > 0$ is selected as the performed sub-task for ap_{out}. The function Generate is invoked to generate the set of child nodes d_{sub}^*, which are then added to the tree T. After all nodes have generated their child nodes, the node d_j that completes ϕ_{out} with minimum cost will be selected. The node path from d_0 to d_j, denoted as $P(d_j, T) = d_{j_0}d_{j_1}\ldots d_{j_n}$, can be traced in T, where $d_{j_0} = d_0$, $d_{j_n} = d_j$, and $\forall i \in [n]$, $f_{j_i} = d_{j_{i-1}}$. For each node $d_i \in d^*$, $(\pi_i^{Tout}, \boldsymbol{\pi}_i^{Tin}, s_i^T, \phi_i^T)$ can be constructed as a plan tuple Π_i and all plan tuples can be constructed as the plan Π, denoted by function $\mathsf{Plan}(d^*)$. Finally, we obtain a feasible plan Π i.e., $\Pi \models (\phi_{out}, E, M)$.

Algorithm 1: Hierarchical Iterative Search

Input: ϕ_{in}^*, ϕ_{out}, E, M
Output: Π

```
Convert φ_out to NFA A_out;
Initialize the T = {d_0}, Flag(d_0) = 0;
while 1 do
    if Tra(d) = 1, ∀d ∈ T_D, then
        break;
    end
    for d_i ∈ T, s.t. Flag(d) = 0 do
        for s ∈ S_out − S_pre(d_i), ap_out ∈ AP_out,
          φ_in ∈ φ*_in, ap_out ∈ Δ(s_i^T, s), E(ap_out, φ_in) > 0
          do
            d*_sub = Generate(d_i, s, ap_out, φ_in, M, T);
            Add new nodes d*_sub into T;
        end
        Flag(d) = 1;
    end
end
Select d_j satisfied that s_j^T ∈ F_out and c_j ≤ c_i, ∀d_i ∈ T
  satisfying s_i^T ∈ F_out;
d* = P(d_j, T);
Π = Plan(d*);
Return Π
```

In Alg. 1, function Generate is developed to construct the set of child nodes d_{sub}^*. First, for each inner subtask $\pi \in AP_{in}$, the plan $\boldsymbol{\pi}^{in}$ for ϕ_{in} with minimum cost can be obtained by existing methods [21], [24]. Let $\boldsymbol{\pi}^{in} = \mathsf{Planner}(\phi_{in}, M, p_i, LM(\pi))$ be the generated plan, which satisfies the initial position $LM(\boldsymbol{\pi}_0^{in}) = p_i$ and the end position $LM(\boldsymbol{\pi}_n^{in}) = LM(\pi)$, $n = |\boldsymbol{\pi}^{in}|$. Given the plan $\boldsymbol{\pi}^{in}$, the child node d_{sub} can be initialized by the searched NFA state $s \in S_{out}$, the atomic proposition $ap_{out} \in AP_{out}$, the inner formula $\phi_{in} \in \phi_{in}^*$, and the plan $\boldsymbol{\pi}^{in}$. Then, the system can be updated from the initial position p_i to the final position of $\boldsymbol{\pi}^{in}$, i.e., $p = LM(\boldsymbol{\pi}_n^{in})$, $n = |\boldsymbol{\pi}^{in}|$. The cost c is determined by (1), where $\alpha(1 - E(ap_{out}, \phi_{in}))$ indicates the dissatisfaction cost caused by ϕ_{in} and $\sum_{\pi_j \in \boldsymbol{\pi}^{in}} \|LM(\pi_{j-1}) - LM(\pi_j)\|$ indicates the traveling cost. The tree can be further pruned to reduce the search space. If there exists a node $d_j \in T$ with the same NFA state and smaller cost, the child node d_{sub} will not generate its child nodes by setting $\mathsf{Flag}(d_{sub}) = 1$. If there exists a node $d_j \in T$ with the same NFA state and larger cost, then for each node d_k expanded by d_j are set $\mathsf{Flag}(d_k) = 1$. It guarantees that, for each NFA state $s \in S_{out}$, only one node with NFA state s can generate the child nodes, which reduces the searchable space without affecting the feasibility of plan. For the node with $s^T \in \mathcal{F}_{out}$, which indicates that ϕ_{out} has been completed, we set $\mathsf{Flag}(d_{sub}) = 1$ to terminate the expansion of the node d_{sub}. Finally, d_{sub} will be added into d_{sub}^* for the next expansion in Alg. 1.

Algorithm 2: Generate

Input: d_i, s, ap_{out}, ϕ_{in}, M, T
Output: d_{sub}^*

```
Initialize d*_sub = ∅;
for π ∈ AP_in and ∃π^in = Planner(φ_in, M, p_i, LM(π)) do
    Initialize d_sub = {s^T, π^out, π^in, φ^T, f, p, c};
    s^T = s, π^out = ap_out, φ^T = φ_in, f = d_i;
    p = LM(π^in_n), n = |π^in|;
    c = c_i + α(1 − E(ap_out, φ_in)) +
        Σ_{π_j ∈ π^in} ||LM(π_{j−1}) − LM(π_j)||;
    for d_j in T do
        if c ≥ c_j, s^T = s^T_j and p = p_j then
            Flag(d_sub) = 1;
        end
        if c < c_j, s^T = s^T_j and p = p_j then
            ∀d_k satisfying d_j ∈ P(d_k, T), Flag(d_k) = 1;
        end
    end
    if s^T ∈ F_out then
        Flag(d_sub) = 1;
    end
    Add d_sub into d*_sub;
end
Return d*_sub;
```

V. ALGORITHM ANALYSIS

This section shows that the plan Π generated by HIS is feasible, and optimal. The optimality indicates that Π has the minimum cost for Problem 1.

Theorem 1. *Given M and a hierarchical formula (ϕ_{out}, E), the generated plan Π by HIS is guaranteed to be feasible.*

Proof. Consider a node $d_i \in T$ and suppose the associated node path is $P(d_i, T)$. By Alg. 1, $\forall d_{i_j} \in P(d_i, T)$, there exists $\pi_{i_j}^{Tout} \in \Delta(s_{i_{j-1}}^T, s_{i_j}^T)$ satisfying $E(\pi_{i_j}^{Tout}, \phi_{i_j}) > 0$. Therefore, for $\Pi_j = (\pi_j^{Tout}, \boldsymbol{\pi}_j^{Tin}, s_j^T, \phi_j^T)$, $j \in [|P(d_i, T)|]$, there exists $\pi_j^{Tout} \in \Delta(s_{j-1}^T, s_j^T)$ satisfying $E(\pi_i^{Tout}, \phi_i^T) > 0$. Since the selected node d_j satisfies $s_j^T \in \mathcal{F}_{out}$, $\boldsymbol{\pi}_{out}$ is an accepting word for ϕ_{out}, i.e., $\Pi \models (\phi_{out}, M)$. □

Lemma 1. *The generated plan Π by HIS does not have the same NFA states and the pruned nodes.*

Proof. We first show that Π does not have the same NFA states. Suppose that there exists a feasible plan $\Pi = \Pi_0\Pi_1 \ldots \Pi_n = (\boldsymbol{\pi}_{out}, \boldsymbol{\pi}_{in}, \boldsymbol{s}, \boldsymbol{\phi})$ with minimum cost $\mathsf{Cost}(\Pi)$. If there exists $s_i = s_j$, $i < j$, let $\Pi^{new} = \Pi_0\Pi_1 \ldots \Pi_i\Pi_{j+1}\Pi_{j+2} \ldots \Pi_n = (\boldsymbol{\pi}_{out}^{new}, \boldsymbol{\pi}_{in}^{new}, \boldsymbol{s}^{new}, \boldsymbol{\phi}^{new})$ be a new plan. The word sequences of Π and Π^{new} are denoted as $\boldsymbol{\pi}_{in}[i \ldots j+1] = \pi_{n_i}^i \ldots \pi_{n_i'}^i \pi_{n_{i+1}}^{i+1} \ldots \pi_{n_j'}^j \pi_{n_{j+1}}^{j+1} \ldots \pi_{n_{j+1}'}^{j+1}$ and $\boldsymbol{\pi}_{in}^{new}[i \ldots j+1] = \pi_{n_i}^i \ldots \pi_{n_i'}^i \pi_{n_{j+1}}^{j+1} \ldots \pi_{n_{j+1}'}^{j+1}$. As $\sum_{\pi_k \in \pi_{n_i'}^i \ldots \pi_{n_{j+1}}^{j+1}} \|LM(\pi_k) - LM(\pi_{k-1})\| \geq \|LM(\pi_{n_{j+1}}^{j+1}) - LM(\pi_{n_i'}^i)\|$ and $\forall \pi_k^{out} \in \boldsymbol{\pi}_{out}[i \ldots j+1]$, $\alpha(1 - E(\pi_k^{out}, \phi_k)) \geq 0$, there exists $\mathsf{Cost}(\Pi_i\Pi_{j+1}) \leq \mathsf{Cost}(\Pi_i\Pi_{i+1} \ldots \Pi_j\Pi_{j+1})$. Therefore, there exists $\mathsf{Cost}(\Pi^{new}) \leq \mathsf{Cost}(\Pi)$ which conflicts with hypothesis. Hence, the same NFA state does not exist in Π.

We then show that Π does not have the pruned nodes. Suppose there exist $d_i, d_j \in T$ satisfying $d_i = d_j$, $p_i = p_j$, and $c_i \leq c_j$. If $d_j \in d^*$, for the optimal plan $\Pi = \Pi_0\Pi_1 \ldots \Pi_n = (\boldsymbol{\pi}_{out}, \boldsymbol{\pi}_{in}, \boldsymbol{s}, \boldsymbol{\phi})$, there exists $s_j^T = s_k \in \boldsymbol{s}$. Then, there exists another plan $\Pi_{new} = \mathsf{Plan}(d_i)\Pi_{k+1} \ldots \Pi_n$ that satisfies $\mathsf{Cost}(\Pi) - \mathsf{Cost}(\Pi_{new}) = c_j - c_i \geq 0$. Therefore, Π is not the optimal plan and thus the pruned nodes are not in Π. □

Theorem 2. *Given M and a hierarchical formula (ϕ_{out}, E), the generated plan Π by HIS is guaranteed to be optimal.*

Proof. Suppose there exists a feasible plan $\Pi_{min} = \Pi_0\Pi_1 \ldots \Pi_n = (\boldsymbol{\pi}_{out}, \boldsymbol{\pi}_{in}, \boldsymbol{s}, \boldsymbol{\phi})$ with minimum cost $\mathsf{Cost}(\Pi)$. For the outer atomic proposition π_i^{out} and its inner task ϕ_i, there may exists several feasible inner plans, which include the inner plan $\boldsymbol{\pi}_i^{in} = \pi_0^i\pi_1^i \ldots \pi_{n_i}^i$. If there exists $\boldsymbol{\pi}_x^{in} = \pi_0^x\pi_1^x \ldots \pi_{n_x}^x$ with larger traveling cost and the same end position, i.e. $D(\boldsymbol{\pi}_i^{in}) < D(\boldsymbol{\pi}_x^{in})$ and $LM(\pi_{n_i}^i) = LM(\pi_{n_x}^x)$, then $\boldsymbol{\pi}_x^{in}$ will be ignored by Alg. 2. As $\boldsymbol{\pi}_i^{in}$ and $\boldsymbol{\pi}_x^{in}$ generate the same system and task states, if the inner plan $\boldsymbol{\pi}_i^{in}$ is replaced by $\boldsymbol{\pi}_x^{in}$, the whole cost must be larger. Therefore, the part of optimal plan $\boldsymbol{\pi}_i^{in} \in \boldsymbol{\pi}_{in} \in \Pi_{min}$ can always be searched by Alg. 2 and the pruning for $\boldsymbol{\pi}_x^{in}$ does not affect the optimality. Since the search method in Alg 1 without pruning in Alg 2 can identify all feasible plans, it must contain Π_{min}. By Lemma 1, the pruned nodes are not in the optimal plan. The nodes that construct Π_{min} are still in the tree and will be selected in Alg 1. Hence, HIS guarantees the optimality. □

After pruning in Alg. 2, no nodes with the same NFA states and system states (position) exist in Π. Since we only consider the areas of interest, the number of system states is equal to the size of AP_{out}. Therefore, the upper bound of number of nodes in T is $|AP_{out}| \times |S_{out}|$. As indicated in Alg. 1, there are no nodes with the same NFA states in the node sequence from the root node to a leaf node. Therefore, the tree searches for $|S_{out}|$ times at most. As mentioned before, there are at most $|S_{out}| \times |AP_{out}|$ nodes in the tree and at most $|S_{out}| \times |AP_{out}| \times |\phi^*_{in}|$ nodes can be generated in each searching. Therefore, the time complex for $|S_{out}|$, $|AP|$, $|\phi^*_{in}|$ are $O(n^3)$, $O(n^2)$, $O(n)$, respectively.

VI. SIMULATION RESULTS

In the simulation, LTL2STAR is used to convert LTL formula to NFA [19]. Python 3.8, Ubuntu 18.4 and ROS melodic are applied in the system simulation.

A. Numerical Simulations

We first evaluate the performance of HIS with hard and soft constraints. Consider a hierarchical temporal formula $\phi_{out} = Fap^1_{out} \wedge Fap^2_{out} \wedge Fap^3_{out}$. The set of inner formulas is $\phi^*_{in} = \{Fap^1_{in}, Fap^2_{in}, Fap^3_{in}, Fap^4_{in}, F(ap^4_{in} \wedge Fap^3_{in})\}$ with $AP_{in} = \{ap^1_{in}, ap^2_{in}, ap^3_{in}, ap^4_{in}\}$. For ap^1_{out} and ap^2_{out}, we define $E(ap^1_{out}, Fap^1_{in}) = 1$, $E(ap^1_{out}, \phi_{in}) = 0$, $\forall \phi_{in} \neq Fap^1_{in}$, $E(ap^2_{out}, Fap^2_{in}) = 1$, $E(ap^2_{out}, \phi_{in}) = 0$ $\forall \phi_{in} \neq Fap^2_{in}$, then ap^1_{out} and ap^2_{out} are considered as hard constraints. For ap^3_{out}, we define $E(ap^3_{out}, Fap^3_{in}) = 0.8$, $E(ap^3_{out}, Fap^4_{in}) = 0.6$, and $E(ap^3_{out}, F(ap^4_{in} \wedge Fap^3_{in})) = 1$, then ap^3_{out} has soft constraints. For different workspaces in Fig. 2, HIS generates the optimal plan as follows. In Fig. 2(a), there is no ap^4_{in} in the workspace. As ap^3_{out} has a soft constraint, there is a feasible plan returned by HIS, i.e., $\boldsymbol{\pi}_{in} = ap^1_{in}ap^2_{in}ap^3_{in}$, $\phi^1_{in} = Fap^1_{in}$, $\phi^2_{in} = Fap^2_{in}$, $\phi^3_{in} = Fap^3_{in}$. In Fig. 2(b), there is no ap^1_{in} in the workspace. As ap^1_{out} has a hard constraint with ap^1_{in}, there is no feasible plan returned by HIS. In Fig. 2(c), as ap^3_{out} has a soft constraint with preference, HIS will search all feasible inner formulas with ap^3_{out}. For different tuning weight α, HIS will select different plans according to dissatisfaction costs and travelling costs. If $\alpha \gg \max_{ap^i_{in}, ap^j_{in} \in AP_{in}} \{\|LM(ap^i_{in}) - LM(ap^j_{in})\|\}$, then HIS will select the plan with smaller dissatisfaction cost. The obtained plan is $\boldsymbol{\pi}_{in} = ap^1_{in}ap^2_{in}ap^4_{in}ap^3_{in}$, $\phi^1_{in} = Fap^1_{in}$, $\phi^2_{in} = Fap^2_{in}$, $\phi^3_{in} = F(ap^4_{in} \wedge Fap^3_{in})$. If $\alpha = 0$, then HIS will select the plan with smaller traveling cost. The obtained plan is $\boldsymbol{\pi}_{in} = ap^1_{in}ap^2_{in}ap^4_{in}$, $\phi^1_{in} = Fap^1_{in}$, $\phi^2_{in} = Fap^2_{in}$, $\phi^3_{in} = Fap^4_{in}$.

B. Performance of task plan

We then evaluate the performance of HIS in terms of the solution time in finding an optimal plan. The workspace M consists of 8 areas of interest and a mobile agent. The atomic task $ap^i_{out}, i = 1, \ldots, 8$, represents the task of visiting area i, respectively. The inner formula set $\phi^*_{in} = \{ap^1_{in}, \ldots, ap^8_{in}\}$ and evaluation function E are fixed, which satisfy $E(ap^i_{out}, ap^i_{in}) = 1$ and $E(ap^i_{out}, ap^j_{in}) = 0$, $\forall j \neq i$. The areas of interest and the initial positions of agents are randomly deployed and the task specification with different

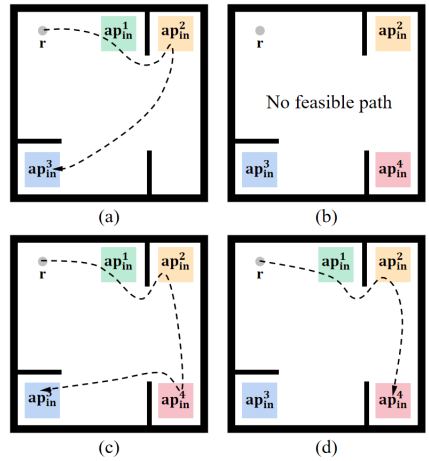

Fig. 2. The task plans in different environments, where the grey dot indicates the robot and the dashed line indicates the trajectory. The areas of interest are labeled with ap^1_{in}, ap^2_{in}, ap^3_{in}, and ap^4_{in}, respectively. The paths indicate the obtained plans under different environments and settings.

number of outer atomic propositions and NFA states are tested in 10 random environments. The average solution time of 10 runs is listed in Tab I. The solution time is approximately linearly proportional to $|S_{out}|^2$, which is smaller than the upper bound of time complexity $O(n^3)$ in Sec. V.

TABLE I
SOLUTION TIME FOR DIFFERENT SETTINGS

$\|AP_{out}\|$	$\|S_{out}\|$	Time (/s)	$\|AP_{out}\|$	$\|S_{out}\|$	Time (/s)
4	16	0.00805	6	56	0.0641
5	32	0.0261	6	60	0.0888
6	18	0.00808	6	64	0.0955
6	33	0.0321	7	96	0.187
6	40	0.0400	7	128	0.323
6	48	0.0469	8	192	0.690
6	52	0.0626	8	256	1.60

C. Experimental Simulations

Consider a warehouse environment as shown in Fig. 3(a), which consists of a storeroom and three warehouses. The agent is required to inspect the warehouses, clean the storeroom, and report to the staff in a warehouse, i.e., $\phi_{out} = Fap^1_{out} \wedge Fap^2_{out} \wedge Fap^3_{out}$, where ap^1_{out} indicates the cleaning task, ap^2_{out} indicates the inspection task, and ap^3_{out} indicates the report task. The inner sub-tasks are defined as $AP_{in} = \{ap^1_{in}, ap^2_{in}, ap^3_{in}, ap^4_{in}, ap^5_{in}, ap^6_{in}\}$, where ap^i_{in}, $i \in \{1, 2, 3\}$ indicates the ith warehouse, ap^4_{in} indicates the vacuum in warehouse 2, ap^5_{in} indicates the trash in warehouse 3, ap^6_{in} indicates cleaning the storeroom.

Based on the outer and inner sub-tasks, the evaluation function is defined as follows. For the cleaning task ap^1_{out}, we define $E(ap^1_{out}, F(ap^4_{in} \wedge Fap^6_{in})) = 1$, $E(ap^1_{out}, F(ap^5_{in} \wedge Fap^6_{in})) = 0.8$, which is considered as soft constraints with preference. For the inspection task ap^2_{out}, we define $E(ap^2_{out}, Fap^1_{in} \wedge Fap^2_{in} \wedge Fap^3_{in}) = 1$, which is considered as a hard constraint. Finally, for the report task ap^3_{out}, we define $E(ap^1_{out}, Fap^1_{in}) = E(ap^1_{out}, Fap^2_{in}) = 1$, which

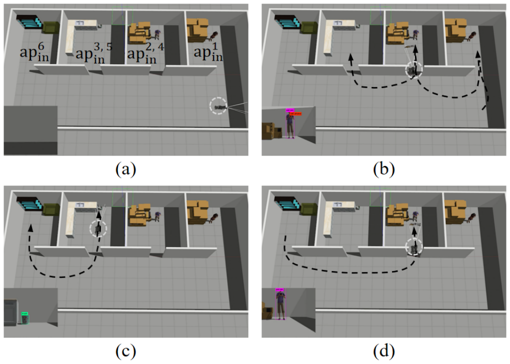

Fig. 3. The simulation results. In (a), initial environment constructed in Gazebo with 6 inner atomic proposition, denoted as ap_1-ap_6, respectively. In (b), the agent is executing ap_{in}^2 for ϕ_1. As finding ap_{in}^4 can not be executed, the plan will be revised. In (c), after executing ϕ_1, the agent continues to perform ϕ_2. In (d), the agent has arrived at warehouse 2 and completed the whole task ϕ_{out}.

is also considered as a soft constraint without preference. Suppose all inner atomic propositions are feasible initially and the agent may change the plan if the inner sub-tasks are found infeasible later. The coefficient α is set larger than $\max_{ap_{in}^i, ap_{in}^j \in AP_{in}} \{\|LM(ap_{in}^i) - LM(ap_{in}^j)\|\}$, i.e., HIS will select the plan with higher satisfaction degree.

The simulation results are shown in Fig. 3. A feasible plan is obtained by HIS as $\Pi = (\boldsymbol{\pi}_{out}, \boldsymbol{\pi}_{in}, \boldsymbol{s}, \boldsymbol{\phi})$, where $\boldsymbol{\pi}_{out} = ap_{out}^2 ap_{out}^1 ap_{out}^3$, $\boldsymbol{\pi}_{in} = ap_1^{in} ap_2^{in} ap_3^{in} ap_4^{in} ap_6^{in} ap_2^{in}$, $\phi_1 = Fap_{in}^1 \wedge Fap_{in}^2 \wedge Fap_{in}^3$, $\phi_2 = F(ap_{in}^4 \wedge Fap_{in}^6))$, $\phi_3 = Fap_{in}^2$. In Fig. 3(b), after arriving warehouse 2, the robot does not find the vacuum, i.e., ϕ_2 can not be executed. The task is then re-planed as $\boldsymbol{\pi}_{in} = ap_1^{in} ap_2^{in} ap_3^{in} ap_5^{in} ap_6^{in} ap_2^{in}$, $\phi_2 = F(ap_{in}^5 \wedge Fap_{in}^6))$. In Fig. 3(c), since it detects the trash in warehouse 3, $\boldsymbol{\phi}$ and $\boldsymbol{\pi}_{in}$ are executable and then the agent will execute ϕ_2. In Fig 3(d), the agent completes the last task ϕ_3 at warehouse 2, as it has smaller cost than performing in warehouse 1. The simulation video is provided[1].

VII. CONCLUSIONS

A hierarchical temporal formula is developed in this work to facilitate task planning with soft constraints. The developed search method HIS can incrementally present the task progress and efficiently generate a feasible and optimal plan. Simulation results demonstrate the expressiveness of the proposed formula and the effectiveness of the solution method. Additional research will consider extending the proposed method to multi-agent systems.

REFERENCES

[1] C. Baier and J.-P. Katoen, *Principles of model checking*. MIT press, 2008.

[2] Z. Liu, B. Wu, J. Dai, and H. Lin, "Distributed communication-aware motion planning for multi-agent systems from stl and spatel specifications," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE, 2017, pp. 4452–4457.

[3] M. Kloetzer and C. Belta, "Automatic deployment of distributed teams of robots from temporal logic motion specifications," *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 48–61, 2009.

[4] M. Guo and D. V. Dimarogonas, "Multi-agent plan reconfiguration under local ltl specifications," *The International Journal of Robotics Research*, vol. 34, no. 2, pp. 218–235, 2015.

[5] M. Cai, S. Xiao, B. Li, Z. Li, and Z. Kan, "Reinforcement learning based temporal logic control with maximum probabilistic satisfaction," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 806–812.

[6] H. Rahmani and J. M. O'Kane, "What to do when you can't do it all: Temporal logic planning with soft temporal logic constraints," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 6619–6626.

[7] M. Cai, H. Peng, Z. Li, H. Gao, and Z. Kan, "Receding horizon control-based motion planning with partially infeasible ltl constraints," *IEEE Control Systems Letters*, vol. 5, no. 4, pp. 1279–1284, 2020.

[8] Z. Li, M. Cai, S. Xiao, and Z. Kan, "Online motion planning with soft metric interval temporal logic in unknown dynamic environment," *IEEE Control Systems Letters*, vol. 6, pp. 2293–2298, 2022.

[9] S. Ahlberg and D. V. Dimarogonas, "Human-in-the-loop control synthesis for multi-agent systems under hard and soft metric interval temporal logic specifications," in *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2019, pp. 788–793.

[10] M. Cai, S. Xiao, Z. Li, and Z. Kan, "Optimal probabilistic motion planning with potential infeasible ltl constraints," *IEEE Transactions on Automatic Control*, vol. 68, no. 1, pp. 301–316, 2021.

[11] M. Guo and D. V. Dimarogonas, "Reconfiguration in motion planning of single-and multi-agent systems under infeasible local ltl specifications," in *52nd IEEE Conference on Decision and Control*. IEEE, 2013, pp. 2758–2763.

[12] B. Lacerda, F. Faruq, D. Parker, and N. Hawes, "Probabilistic planning with formal performance guarantees for mobile service robots," *The International Journal of Robotics Research*, vol. 38, no. 9, pp. 1098–1123, 2019.

[13] M. Guo and M. M. Zavlanos, "Probabilistic motion planning under temporal tasks and soft constraints," *IEEE Transactions on Automatic Control*, vol. 63, no. 12, pp. 4051–4066, 2018.

[14] Dimitrova, Rayna and Ghasemi, Mahsa and Topcu, Ufuk, "Reactive synthesis with maximum realizability of linear temporal logic specifications," *Acta Informatica*, vol. 57, pp. 107–135, 2020.

[15] M. Guo, S. Andersson, and D. V. Dimarogonas, "Human-in-the-loop mixed-initiative control under temporal tasks," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6395–6400.

[16] P. Halder and M. Althoff, "Minimum-violation velocity planning with temporal logic constraints," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2022, pp. 2520–2527.

[17] B. Lacerda, D. Parker, and N. Hawes, "Optimal and dynamic planning for markov decision processes with co-safe ltl specifications," in *IEEE Int. Conf. Intell. Robot. Syst.* IEEE, 2014, pp. 1511–1516.

[18] K. Cho, J. Suh, C. J. Tomlin, and S. Oh, "Cost-aware path planning under co-safe temporal logic specifications," *IEEE Robot. Autom. Lett.*, vol. 2, no. 4, pp. 2308–2315, 2017.

[19] P. Gastin and D. Oddoux, "Fast ltl to büchi automata translation," in *International Conference on Computer Aided Verification*. Springer, 2001, pp. 53–65.

[20] Z. Zhou, Z. Chen, M. Cai, Z. Li, Z. Kan, and C.-Y. Su, "Vision-based reactive temporal logic motion planning for quadruped robots in unstructured dynamic environments," *IEEE Trans. Ind. Electron.*, 2023.

[21] L. Li, Z. Chen, H. Wang, and Z. Kan, "Fast task allocation of heterogeneous robots with temporal logic and inter-task constraints," *IEEE Robotics and Automation Letters*, 2023.

[22] S. Li, D. Park, Y. Sung, J. A. Shah, and N. Roy, "Reactive task and motion planning under temporal logic specifications," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 12 618–12 624.

[23] S. L. Smith, J. Tumova, C. Belta, and D. Rus, "Optimal path planning for surveillance with temporal-logic constraints," *The International Journal of Robotics Research*, vol. 30, no. 14, pp. 1695–1708, 2011.

[24] C. I. Vasile, X. Li, and C. Belta, "Reactive sampling-based path planning with temporal logic specifications," *The International Journal of Robotics Research*, vol. 39, no. 8, pp. 1002–1028, 2020.

[1] `https://youtu.be/wBBUaeTldgU`