Let Others Help You: Influential Planning for Multi-Agent Systems under Temporal Logic Tasks ⋆

Bowen Ye, Yingzhu Wang, Jianing Zhao and Xiang Yin *

* *School of Automation and Intelligent Sensing, Shanghai Jiao Tong University, Shanghai 200240, China.*
E-mail: {yebowen1025,arrie.wyzzz,jnzhao,yinxiang}@sjtu.edu.cn

Abstract: In this paper, we investigate the motion planning problem for multi-agent systems under temporal logic constraints. Unlike most existing works, which assume agents are either cooperative or adversarial, we consider a new scenario called *influential planning*. Specifically, we assume there are two agents: a leader and a follower, each with their own objectives characterized by temporal logic formulas. Our objective is to design a plan for the leader such that, when the follower pursues its own objectives, the leader's objectives are also satisfied. In other words, although the leader cannot directly control the follower's behavior, it can *influence* the follower's actions by strategically synthesizing its own plan. We provide an efficient algorithm for solving this type of influential planning problem, where specifications are expressed using co-safe linear temporal logic (scLTL) formulae. Case studies are presented to illustrate the effectiveness of our framework, demonstrating how the leader's strategic planning can indirectly guide the follower's behavior to achieve desired outcomes.

Keywords: multi-agent system; co-safe LTL; discrete event; influential planning

1. INTRODUCTION

Motion and task planning is one of the central problems in autonomous systems, such as mobile robots and manipulators. In recent years, there has been growing interest in using formal languages for planning with high-level specifications, driven by the need for automated synthesis and rigorous correctness guarantees in complex safety-critical systems, e.g. Smith et al. (2011); Kress-Gazit et al. (2018); Mahulea et al. (2020); Yin et al. (2024). Among these formal languages, temporal logic, such as Linear Temporal Logic (LTL) or Signal Temporal Logic (STL), has emerged as one of the most widely used frameworks for specifying complex behaviors. By extending standard Boolean logic with temporal operators, temporal logic enables the specification of tasks that involve intricate spatial-temporal constraints on the trajectory of the system, Baier and Katoen (2008). Temporal logic planning has been extensively applied to various autonomous systems, including underwater vehicles in McMahon and Plaku (2016), aerial vehicles in Silano et al. (2021), ground vehicles in Huang et al. (2023); Kantaros et al. (2022) and collaborative robots in Yu et al. (2024).

In the multi-agent setting, temporal logic planning problems researched in Schillinger et al. (2018); Chen and Kan (2024); Cardona and Vasile (2024) require the coordination of a set of agents. Tasks can be specified either globally for the entire team or locally for individual agents. Depending on the roles of the agents in the team, most existing works on multi-agent temporal logic planning can be categorized into the collaborative setting and the adversarial setting. In the collaborative setting, it is assumed that the behavior of all agents is controllable, and their goal is to achieve a global objective. Such planning problems can be solved in a monolithic manner by considering all robots as a single system, Guo and Dimarogonas (2015); Kantaros and Zavlanos (2020), or more efficiently in a distributed manner through task decomposition and coordination researched in Kloetzer and Belta (2009); Luo and Zavlanos (2022); Chen et al. (2024). In the adversarial setting, it is further assumed that some agents are uncontrollable or even adversarial. The objective is to plan for the controllable agents in such a way that the task can be achieved robustly, regardless of the behavior of the uncontrollable or adversarial agents. This problem can, in principle, be solved using zero-sum game techniques, where the controllable agents aim to maximize the utility of task satisfaction while the adversarial agents aim to minimize it, Wolff et al. (2012); Ding et al. (2014); Fu and Topcu (2015); Pacheck and Kress-Gazit (2023).

In many cases, it may be too restrictive to strictly divide the roles of agents into purely collaborative or adversarial. For simplicity, this paper focuses on a two-agent setting, where one agent's behavior is controllable, while the other is not. Specifically, the uncontrollable agent is not necessarily fully adversarial to the controllable agent's objectives but may instead pursue its own independent goals. In such cases, although we cannot directly control the uncontrollable agent's actions, we can influence its behavior by considering its pursuit of independent objectives. Consequently, when planning for the controllable agent, we can anticipate the actions of the uncontrollable agent,

⋆ This work was supported by the National Natural Science Foundation of China (62173226, 62061136004).

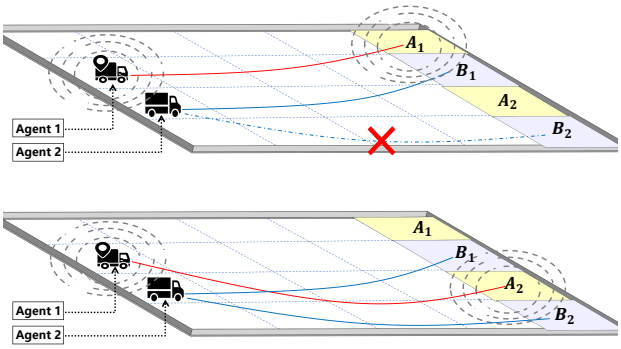

Fig. 1. A motivating example of influential planning.

enabling us to formulate less conservative plans than those designed for worst-case scenarios. We term this planning approach *influential planning*, where the controllable agent strategically interacts with the uncontrollable agent to achieve its goals while accounting for the uncontrollable agent's independent objectives.

To better motivate our study of *influential-planning*, let us consider a simple motivating example illustrated in Figure 1. We assume there are two agents moving in a grid map. The primary objective of Agent 1 is to reach Region A_1 or A_2, while the primary objective of Agent 2 is to reach Region B_1 or B_2. Additionally, Agent 1 can broadcast information to its adjacent regions, and Agent 2 needs to follow Agent 1 closely to receive the signal throughout the entire operation. Furthermore, we assume that Agent 1 has a preference for Agent 2 to arrive at B_1 rather than B_2. Although Agent 1 cannot directly control Agent 2, it can utilize Agent 2's signal pursuit behavior to influence its actions. Specifically, if Agent 1 chooses to go to Region A_1 (as shown in the top figure), Agent 2's best choice to receive the signal and reach targeted Region is to go to B_1, which aligns with Agent 1's preference. However, if Agent 1 chooses to go to Region A_2 (as shown in the bottom figure), Agent 2 is free to go to either B_1 or B_2, with the latter case being undesirable for Agent 1. This example highlights how Agent 1 can strategically influence Agent 2's behavior to achieve its own objectives without direct control.

In this paper, we formulate and solve the *influential-planning* problem for a two-agent system under temporal logic specifications. We assume that there are two agents: a controllable agent, referred to as the *leader*, and an uncontrollable agent, referred to as the *follower*. Each agent has its own task specified as a co-safe LTL formula. The leader, which is the agent we need to plan for, cannot directly control the behavior of the follower but has knowledge of the follower's task formula. The design objective is to synthesize a plan for the leader such that, for any plan the follower executes that satisfies its own task, the leader's task is also satisfied. In other words, the leader's plan must ensure that its task is achieved regardless of how the follower behaves, as long as the follower's behavior is consistent with its own task specification. This problem essentially falls into the broader category of Stackelberg games or leader-follower games within the context of game theory. Solving such problems typically involves bilevel optimization, which is generally computationally challenging, e.g. Bard (2013); Marden and Shamma (2018). To address this, we propose an efficient automata-theoretical algorithm. While we ensure the soundness of the algorithm, we prioritize practicality by sacrificing completeness to mitigate computational complexity. Finally, case studies are provided to demonstrate the effectiveness of our approach.

Our work is related to several studies in the literature. For example, optimal leader-follower control for dynamic systems has a long research history in the field of systems and control, where the objective of Basar and Selbuz (1979); Moon and Başar (2018); Ratliff et al. (2019) is to optimize the payoff of one player in the presence of another player pursuing its own utility. However, these works primarily focus on numerical payoff functions and do not consider temporal logic tasks. Recently, in Cui et al. (2025), the Stackelberg game for dynamic systems under STL specifications has been studied. The problem setting is conceptually similar to ours, as it involves a leader influencing a follower to achieve its objectives. However, the system model in Cui et al. (2025) is based on continuous nonlinear dynamic systems, whereas our work considers transition systems. As a result, the solution procedures and techniques differ significantly. There are also works in temporal logic planning where one agent must consider its implicit influence on other agents. For instance, Wang et al. (2020) studies the HyperLTL planning problem, where the correctness of a plan depends on the behavior of other agents. Similarly, Yu et al. (2022); Shi et al. (2023) investigates security-preserving temporal logic planning, where the goal is to prevent the leakage of sensitive information to other agents through information flow. However, these works assume a cooperative setting, where all agents are controllable, unlike our setting, which involves an uncontrollable follower.

The rest of the paper is organized as follows. In Section 2, we introduce the system model and review the necessary preliminaries. Section 3 presents several abstract auxiliary problems together with their solution operators, which will be used in the subsequent development. In Section 4, we formally formulate the influential planning problem addressed in this work. Section 5 then describes our solution algorithm. A case study is reported in Section 6 to illustrate the applicability of the proposed approach. Finally, Section 7 concludes the paper.

2. PRELIMINARIES

2.1 System Model

We consider a team of two agents, referred to as the leader and the follower. The mobility of each agent $i \in \{L, F\}$ is modeled by a *weighted transition system* (WTS)

$$\mathcal{T}_i = (X_i, x_{0,i}, \Delta_i, \mathcal{AP}_i, \ell_i, w_i),$$

where

- X_i is a finite set of states, representing different regions of the workspace;
- $x_{0,i}$ is the initial state representing the starting region of agent i;
- $\Delta_i \subseteq X_i \times X_i$ is the transition relation such that $(x, x') \in \Delta_i$ means that agent i can move directly to x' from x;
- $\mathcal{AP}_i$ is a set of atomic propositions, representing some basic properties of interest;

- $\ell_i : X_i \to 2^{\mathcal{AP}_i}$ is a labeling function that assigns each state with a set of atomic propositions;
- $w_i : \Delta_i \to \mathbb{R}_{\geq 0}$ is a cost function such that $w_i(x, x')$ represents the cost incurred when agent i moves from x to x'.

We assume that the two agents are running synchronously, which can be enforced by a global clock. Then the mobility of the entire multi-agent system is modeled by the *global transition system* (GTS), denoted by $\mathcal{T}$, which is the synchronous product of $\mathcal{T}_L$ and $\mathcal{T}_F$.

Definition 1. (GTS). Given two weighted transition systems $\mathcal{T}_i = (X_i, x_{0,i}, \Delta_i, \mathcal{AP}_i, \ell_i, w_i), i \in \{L, F\}$, the global transition system $\mathcal{T} = \mathcal{T}_L \otimes \mathcal{T}_F$ is a 6-tuple

$$\mathcal{T} = (X, x_0, \Delta, \mathcal{AP}, L, w),$$

where

- $X = X_L \times X_F$ is the set of global states;
- $x_0 = (x_{0,L}, x_{0,F})$ is the global initial state;
- $\Delta \subseteq X \times X$ is the transition relation defined by: for any $x = (x_L, x_F), x' = (x'_L, x'_F) \in X$, we have $(x, x') \in \Delta$ iff $(x_L, x'_L) \in \Delta_L$ and $(x_F, x'_F) \in \Delta_F$;
- $\mathcal{AP} = \mathcal{AP}_L \cup \mathcal{AP}_F$ is the global set of atomic propositions;
- $L : \mathcal{AP} \to 2^{\mathcal{AP}}$ is the labeling function defined by: for any state $x = (x_L, x_F) \in X$, we have $L(x) = L_L(x_L) \cup L_F(x_F)$;
- $w : \Delta \to \mathbb{R}_{\geq 0}$ is the cost function defined by: for any $x = (x_L, x_F), x' = (x'_L, x'_F) \in X$, if $(x, x') \in \Delta$, we have $w(x, x') = w_L(x_L, x'_L) + w_F(x_F, x'_F)$.

A finite path $\pi = \pi(0)\pi(1)\cdots\pi(m) \in X^*$ is a finite sequence of states such that $\pi(0) = x_0$ and $(\pi(j), \pi(j+1)) \in \Delta$ for any $j = 0, 1, \cdots, m-1$. Note that each state in X is an 2-tuple and we denote by π_L the path of leader and π_F the path of follower in global path π. We denote by $\mathsf{Path}^*(\mathcal{T})$ the set of all finite paths in $\mathcal{T}$. Furthermore, we define the cost of a finite path $\pi = \pi(0)\pi(1)\cdots\pi(m) \in X^*$ as

$$J(\pi) = \sum_{j=0}^{m-1} w(\pi(j), \pi(j+1)), \tag{1}$$

which captures the total cost incurred by all agents during the execution of π. For each path $\pi = \pi(0)\pi(1)\cdots\pi(m) \in X^*$, we define its trace $\ell(\pi)$ as

$$\ell(\pi) = \ell(\pi(0))\ell(\pi(1))\cdots\ell(\pi(m)).$$

2.2 Task Specification

We describe the objective of each agent by a linear temporal logic (LTL) formulae. Formally, the syntax of LTL is given as follows

$$\phi := \top \mid a \mid \phi_1 \wedge \phi_2 \mid \neg\phi \mid \bigcirc\phi \mid \phi_1 U \phi_2,$$

where $\top$ stands for the "true" predicate; $a \in \mathcal{AP}$ is an atomic proposition; $\neg$ and $\wedge$ are Boolean operators "negation" and "conjunction", respectively; $\bigcirc$ and U denote temporal operators "next" and "until", respectively. One can also derive other temporal operators such as "eventually" by $\Diamond\phi = \top U \phi$, "always" by $\Box\phi = \neg\Diamond\neg\phi$ and "implication" by $\phi_1 \to \phi_2 = \neg\phi_1 \vee \phi_2$.

LTL formulae are evaluated over infinite words; the readers are referred to Baier and Katoen (2008) for the semantics of LTL. Specifically, an infinite word $\rho \in (2^{\mathcal{AP}})^\omega$ is an infinite sequence over alphabet $2^{\mathcal{AP}}$. We write $\rho \models \phi$ if ρ satisfies LTL formula ϕ.

In this paper, we focus on a widely used fragment of LTL formulae called the *co-safe LTL* (scLTL) formulae. Specifically, a scLTL formula requires that the negation operator $\neg$ can only be applied in front of atomic propositions. Consequently, one cannot use "always" $\Box$ in scLTL. Although the semantics of LTL are defined over infinite words, it is well-known that any infinite word satisfying a co-safe LTL formula has a *finite good prefix*. Specifically, a good prefix is a finite word $\rho' = \rho(1)\cdots\rho(n) \in (2^{\mathcal{AP}})^*$ such that $\rho'\rho'' \models \phi$ for any $\rho'' \in (2^{\mathcal{AP}})^\omega$. For an finite word $\rho \in (2^{\mathcal{AP}})*$, we also write $\rho \models \phi$ if it is a good prefix and we denote by $L_{pref}(\phi)$ the set of all finite good prefixes of scLTL formula ϕ. Also, for an path π in $\mathcal{T}$, we denote by $\pi \models \phi$ if its trace satisfies ϕ.

For any scLTL formula ϕ, its good prefixes $L_{pref}(\phi)$ can be accepted by a *deterministic finite automaton* (DFA). Formally, a DFA is a 5-tuple $\mathcal{A} = (Q, q_0, \Sigma, f, Q_F)$, where Q is the finite set of states; $q_0 \in Q$ is the initial state; Σ is the alphabet; $f : Q \times \Sigma \to Q$ is a transition function; and $Q_F \subseteq Q$ is the set of accepting states. The transition function can also be extended to $f : Q \times \Sigma^* \to Q$ recursively. A finite word $\rho \in \Sigma^*$ is said to be *accepted* by $\mathcal{A}$ if $f(q_0, \rho) \in Q_F$; we denote by $\mathcal{L}(\mathcal{A})$ the set of all accepted words. Then for any scLTL formula ϕ defined over $\mathcal{AP}$, we can always build a DFA over alphabet $\Sigma = 2^{\mathcal{AP}}$, denoted by $\mathcal{A}_\phi = (Q, q_0, 2^{\mathcal{AP}}, f, Q_F)$, such that $\mathcal{L}(\mathcal{A}_\phi) = L_{pref}(\phi)$.

3. PRE-PROBLEMS AND SOLUTION

The standard LTL planning problem is to find a goal path satisfying a global task.

Problem 1. (LTL Planning Problem). Given a multi-agent system modeled by a GTS $\mathcal{T}$, a task described by an LTL formula ϕ, find find a plan $\pi \in \mathsf{Path}^*(\mathcal{T})$ such that

i) $\pi \models \phi$; and
ii) for any other $\pi' \in \mathsf{Path}^*(\mathcal{T})$, we have $J(\pi) \leq J(\pi')$.

The above planning problem can be solved using an automata-theoretical approach, which reduces the problem to a graph search over the product state space, Smith et al. (2011). We obtain an optimal solution for Problem 1 by adapting the optimal LTL planning algorithm in Ye et al. (2024): we remove its run-time checking component and refer to the resulting offline planning procedure as $SolveLTL(\mathcal{T}, \phi)$. Given a transition system $\mathcal{T}$ and an LTL formula ϕ, $SolveLTL(\mathcal{T}, \phi)$ returns a set of optimal paths set $\{\pi\}$ that satisfy ϕ. In a multi-agent setting, when the trajectory of one agent π_i is fixed, we simply condition the solver on π_i and use it to compute optimal trajectories for the remaining agents, which we denote by $SolveLTL(\mathcal{T}, \phi, \pi_i)$.

We further define a verification routine $Judge(\pi, \phi)$, which checks whether every path in the given set π satisfies the specification ϕ and returns *true* if all paths do, and *false* otherwise.

Problem 2. (Path enumeration with bounded cycle usage). Consider a directed graph

$$G = (V, E, w),$$

where V is the set of vertices, $E \subseteq V \times V$ is the set of directed edges, and $w : E \to \mathbb{R}$ is an edge-weight function. The graph may contain directed cycles. Let $s \in V$ and $e \in V$ be a designated start and end vertex, respectively.

We are interested in all finite directed paths from s to e under the following cycle constraint:

- A path is allowed to traverse directed cycles, but each directed cycle may be used at most once along a single path. Equivalently, along any such path, the same directed cycle is not traversed multiple times (i.e., the path does not repeatedly loop around the same cycle).

The goal of Problem 2 is to enumerate the set $\mathcal{P}$ of all directed $s - e$ paths in G that satisfy this "at-most-once per cycle" constraint.

Problem 3. (Vertex pruning based on path). On the same graph $G = (V, E, w)$ as in Problem 2, let $d \in V$ be a designated vertex. Let $\mathcal{P}$ denote the set of all $s - e$ paths satisfying the cycle constraint in Problem 2.

Define $\mathcal{P}_d \subseteq \mathcal{P}$ as the subset of paths that visit the vertex d at least once. Let

$$V_d = \bigcup_{p \in \mathcal{P}_d} \{\text{all vertices that appear on path } p\}$$

be the set of all vertices that lie on some path in $\mathcal{P}_d$.

The task in Problem 3 is:

- to remove from G all vertices in $V \setminus V_d$ together with their incident edges, and
- to obtain the subgraph induced by V_d, which we denote by G_d.

Equivalently, from a path-based viewpoint, the resulting graph G_d contains exactly the vertices and edges that belong to contiguous subpaths of $s - e$ paths in $\mathcal{P}$ that pass through d.

The solutions to Problems 2 and 3 can be obtained by adapting classical path-enumeration and shortest-path techniques from the literature Cherkassky et al. (1996); Bernstein (2010). In what follows, we denote the solver for Problem 2 by $EnumPaths(G, s, e)$, and for Problem 3 by $PruneByVertex(G, \{\pi\}, d)$, where $\{\pi\}$ is the output of $EnumPaths(G, s, e)$.

4. INFLUENTIAL PLANNING PROBLEM

In this work, we investigate a scenario consisting of two agents indexed as $\mathcal{I} = \{L, F\}$. The agent with index L is referred to as the *leader*, whose trajectory we plan while considering all possible behaviors of the *follower*, denoted by index F. The leader's planning must account for the follower's actions, ensuring that the leader's objectives are achieved even as the follower pursues its own goals.

Definition 2. (Collision-Free Paths). Let π_L and π_F be two finite paths of the leader and the follower, respectively, both with horizon H. We say π_L and π_F are *collision-free* if, for any $t = 0, 1, \ldots, H$, we have $\pi_L(t) \neq \pi_F(t)$. We denote by $(\pi_L, \pi_F) \models_{NC} \phi$ if $(\pi_L, \pi_F) \models \phi$ and they are collision-free.

Once the leader chooses a path π_L, the follower must select a path π_F such that a task formula ϕ can be achieved from its perspective. Additionally, we assume that the follower aims to minimize its effort, as defined by a cost function, to accomplish the task. We refer to such a plan as the *optimal response* of the follower, defined as follows.

Definition 3. (Optimal Response). Let π_L be a finite path of the leader and ϕ be a scLTL formula. We say π_F, which is a finite path of the follower, is an optimal response of the follower with respect to π_L and ϕ if

(i) $(\pi_L, \pi_F) \models_{NC} \phi$; and
(ii) for any π'_F such that $(\pi_L, \pi'_F) \models_{NC} \phi$, we have $J(\pi_F) \leq J(\pi'_F)$.

We denote by $OR_F(\pi_L, \phi)$ the set of all optimal responses of the follower with respect to π_L and ϕ.

In the influential planning setting, the leader first issues its own path, and then the follower responds. More specifically, we consider the following scenarios:

- Both the leader and the follower have their own specifications described by two scLTL formulae, ϕ_1 and ϕ_2, respectively.
- The planning process follows a leader-follower structure: the leader first chooses a path π_L, and then the follower chooses a path π_F.
- The follower is rational, meaning it only selects a plan $\pi_F \in OR_F(\pi_L, \phi)$ from the optimal response set.
- The leader must ensure that it does not block the possibility of the follower achieving ϕ_2, while simultaneously enforcing its own objective ϕ_1 strategically, regardless of the follower's rational reaction.

This framework captures the essence of *influential-planning*, where the leader indirectly influences the follower's behavior to achieve its own goals while respecting the follower's objectives. Now we formally formulate the problem that we solve in this paper.

Problem 4. (Influential LTL Planning Problem). Given a leader-follower system modeled by two weighted transition systems $\mathcal{T}_L$ and $\mathcal{T}_F$, along with their scLTL formulae ϕ_1 and ϕ_2, the goal is to find an *influential path* π_L for the leader such that:

(i) $OR_F(\pi_L, \phi_2) \neq \emptyset$; and
(ii) For any $\pi_F \in OR_F(\pi_L, \phi_2)$, we have $(\pi_L, \pi_F) \models \phi_1$.

Remark 1. We make some remarks regarding our problem formulation. First, we require that the leader should not block the possibility of the follower achieving ϕ_2. Beyond its practical meaning in collaborative settings, this requirement is also necessary for technical reasons. Without it, we would need to specify the compromised behavior of the follower if it cannot achieve its goal. Additionally, if the leader aims to ensure that $OR_F(\pi_L, \phi_2) = \emptyset$, this reduces to solving a safety game against the follower. Second, for simplicity, we do not consider the optimization problem for the leader in this formulation. That is, any path satisfying conditions (i) and (ii) is considered a feasible solution. Optimality here is solely used to model the rationality of the follower, ensuring that the follower chooses its path to minimize effort while achieving its goals.

5. PLANNING ALGORITHM

In this section, we propose an efficient algorithm to solve the influential-planning LTL problem. Firstly, to obtain a global path satisfying the LTL task formula ϕ, we need to encode the automaton $\mathcal{A}_\phi$ accepting ϕ into the solution space of the transition system $\mathcal{T}$. This encoding process is detailed as follows.

Definition 4. (Product Systems). Given a global transition system $\mathcal{T} = (X, x_0, \Delta, \mathcal{AP}, L, w)$ and an LTL formula ϕ with the corresponding DFA $\mathcal{A}_\phi = (Q, q_0, 2^{\mathcal{AP}}, f, Q_F)$, the product system is a new transition system

$$\mathcal{P} = (S, s_0, \delta_\mathcal{P}, w_\mathcal{P}),$$

where:

- $S \subseteq X \times Q$ is the finite set of states;
- $s_0 = (x_0, q_0)$ is the initial state;
- $\delta_\mathcal{P} \subseteq S \times S$ is the transition relation defined as follows: for any $s = (x, q), s' = (x', q') \in S$, we have $(s, s') \in \delta_\mathcal{P}$ if $(x, x') \in \Delta$ and $q' = f(q, L(x'))$;
- $w_\mathcal{P} : \delta_\mathcal{P} \to \mathbb{R}_{\geq 0}$ is the cost function defined by: for any $s = (x, q), s' = (x', q') \in S$, we have $w_\mathcal{P}(s, s') = w(x, x')$ whenever $(x, x') \in \Delta$.

The product system ensures that the agent's motion respects the mobility constraints of $\mathcal{T}$ while simultaneously tracking the progress of the LTL formula ϕ. We define the accepting state set of $\mathcal{P}$ as

$$Goal(\mathcal{P}) = \{(x, q) \in S : q \in Q_F\}.$$

By construction, any plan that reaches $Goal(\mathcal{P})$ corresponds to a run of $\mathcal{T}$ that satisfies the scLTL task ϕ. For a given initial product state $\pi(0) \in S$, we further denote by $R(\pi(0))$ the set of all states reachable from $\pi(0)$ in $\mathcal{P}$, which captures all feasible evolutions that are consistent with both the mobility constraints and the progression of scLTL ϕ.

We next construct a three-component product system $\mathcal{P}_{1,2}$ whose specification automaton is the synchronous product of the automata for ϕ_1 and ϕ_2, namely

$$\mathcal{A}_\phi \;=\; \mathcal{A}_{\phi_1} \times \mathcal{A}_{\phi_2}.$$

We denote by $\mathcal{P}_1$ the product system obtained when using only $\mathcal{A}_{\phi_1}$. These product constructions form the basis of the solution algorithm presented below.

Algorithm 1: Feasibility check for leader paths

Input: LTL formulas ϕ_1, ϕ_2, GTS $\mathcal{T}$, leader path set $\{\pi_L\}$
Output: optimal feasible leader path π_L

Initialize feasible set $FS_L \leftarrow \emptyset$
for *each* $\pi_L \in \{\pi_L\}$ **do**
  $\{\pi_F\} \leftarrow SolveLTL(\mathcal{T}, \phi_2, \pi_L)$
  if $Judge(\pi_L, \{\pi_F\}, \phi_1)$ **then**
    $FS_L \leftarrow FS_L \cup \{\pi_L\}$
if $FS_L = \emptyset$ **then**
  Return False, _
else
  Return True, π_L which satisfies $\forall \pi'_L \in FS_L$, $J(\pi_L) \leq J(\pi'_L)$

Algorithm 2: Feasible solution for influential problem

Input: LTL formula ϕ_1, ϕ_2, GTS $\mathcal{T}$
Output: optimal leader's path π_L

1 Set leader's feasible path set $FS_L = \emptyset$
2 Convert ϕ_i to the corresponding DFA $\mathcal{A}_{\phi_i} = (Q_i, q_{0,i}, 2^{\mathcal{AP}_i}, f_i, Q_{F,i})$, $i \in \{1, 2\}$
3 Construct the product system using $\mathcal{T}$ and $\mathcal{A}_{\phi_1}, \mathcal{A}_{\phi_2}$ $\mathcal{P}_{1,2} = (S, s_0, \delta_{\mathcal{P}_{1,2}}, w_{\mathcal{P}_{1,2}})$, and construct $\mathcal{P}_i$ with $\mathcal{T}$ and $\mathcal{A}_{\phi_i}$ for $i \in \{1, 2\}$
4 Construct the $Goal(\mathcal{P}_{1,2}), Goal(\mathcal{P}_1), Goal(\mathcal{P}_2)$ and get $\pi(0)$ from $\mathcal{P}_{1,2}$
5 **if** $R(\pi(0)) \cap Goal(\mathcal{P}_{1,2}) = \emptyset$ **then**
6 Return "no feasible plan";
7 **else**
8 Set path set $\{\pi\} = \emptyset$
9 **for** $s_{\mathcal{P}_2} \in Goal(\mathcal{P}_2)$ **do**
10 $\{\pi\} = EnumPaths(\mathcal{P}, \pi(0), s_{\mathcal{P}_2}) \bigcup \{\pi\}$
11 Set new path set $\{\pi'\} = \emptyset$
12 **for** $s_{\mathcal{P}_1} \in Goal(\mathcal{P}_1)$ **do**
13 $\{\pi'\} = \{\pi'\} \bigcup PruneByVertex(\mathcal{P}, \{\pi\}, s_{\mathcal{P}_1})$
14 $FS_L = FS_L \bigcup Solve(\phi_1, \phi_2, \mathcal{T}, \{\pi'\})$
15 Reset $\{\pi\} = \emptyset$
16 **for** $s_{\mathcal{P}_1} \in Goal(\mathcal{P}_1)$ **do**
17 $\{\pi\} = EnumPaths(\mathcal{P}, \pi(0), s_{\mathcal{P}_1}) \bigcup \{\pi\}$
18 **for** $\pi \in \{\pi\}$ **do**
19 **if** $\pi \not\models$ Pro **then**
20 Delete π from $\{\pi\}$
21 Reset path set $\{\pi'\} = \emptyset$
22 **for** $s_{\mathcal{P}_2} \in Goal(\mathcal{P}_2)$ **do**
23 $\{\pi'\} = \{\pi'\} \bigcup PruneByVertex(\mathcal{P}, \{\pi\}, s_{\mathcal{P}_2})$
24 $FS_L = FS_L \bigcup Solve(\phi_1, \phi_2, \mathcal{T}, \{\pi'\})$
25 **if** $FS_L = \emptyset$ **then**
26 Return no feasible solution
27 **else**
28 Return π_L which satisfies $\forall \pi'_L \in FS_L$, $J(\pi_L) \leq J(\pi'_L)$

We first introduce a feasibility-checking routine, summarized in Algorithm 1. Given a set of candidate leader paths π_L, the algorithm initializes an empty feasible set FS_L in line 1 and then in line 2-5 iterates over each $\pi_L \in \{\pi_L\}$. For each leader path, it calls $SolveLTL(\mathcal{T}, \phi_2, \pi_L)$ to compute the set of optimal follower paths $\{\pi_F\}$ that satisfy ϕ_2 under the fixed leader trajectory. It then invokes $Judge(\pi_L, \{\pi_F\}, \phi_1)$ to verify whether the resulting joint behavior satisfies the leader's specification ϕ_1; any leader path that passes this check is added to FS_L. After all candidates have been examined, the routine returns a feasibility flag together with the minimum-cost leader path in FS_L (if $FS_L \neq \emptyset$), and *false* otherwise. We denote this procedure by $Solve(\phi_1, \phi_2, \mathcal{T}, \pi_L)$ and employ it as a subroutine in our final algorithm.

We are now ready to present the final solution to Problem 4, as summarized in Algorithm 2. Before doing so, we define the predicate Pro used in line 19: $\pi \models$ Pro

holds if π does visit any state in $Goal(\mathcal{P}_2)$, or if, after its last visit to $Goal(\mathcal{P}_1)$, the follower's component of π does not change along the remaining suffix of the path. The main part of the algorithm is contained in lines 8–24. Conceptually, these constructions ensure that, from the follower's perspective, every minimal-cost trajectory that reaches $Goal(\mathcal{P}_2)$ is also forced to visit $Goal(\mathcal{P}_1)$. Hence any follower that optimizes its own objective for $\mathcal{P}_2$ will automatically realize the leader's objective $\mathcal{P}_1$, thereby satisfying the influential planning requirement. The proof is presented below.

Proof 1. To justify the correctness of Algorithm 2, we reduce the problem to a three-point reachability task on a finite directed graph with cycles, where each cycle is visited at most once. Both the leader L and the follower F start from a common initial state s. We assume that their evolution is synchronous, so each node in the graph can be represented as a joint state (x_L, x_F), where x_L and x_F denote the states of the leader and the follower at the same time step. The follower is required to reach a terminal state e_F corresponding to satisfaction of ϕ_2, while the leader aims to steer the joint evolution so that the system also reaches a terminal state e_L associated with ϕ_1. We distinguish two complementary cases.

In the first case, we focus on trajectories driven primarily by the follower's objective. We enumerate all paths from s to e_F and, for each such candidate, fix the leader component and resolve the follower's problem by calling $SolveLTL(\mathcal{T}, \phi_2, \pi_L)$. This yields all cost-minimal follower trajectories consistent with the fixed leader behavior. Lines 8–14 of Algorithm 2 then retain only those joint trajectories for which every cost-minimal follower path that reaches $Goal(\mathcal{P}_2)$ also visits $Goal(\mathcal{P}_1)$. Equivalently, among all feasible solutions, any trajectory attaining the follower's goal region $Goal(\mathcal{P}_2)$ must pass through $Goal(\mathcal{P}_1)$, while preserving cost optimality for F.

The second case treats the complementary situation in which the leader's target e_L is emphasized and the follower's goal e_F is encountered en route. We first enumerate all paths from s to e_L, which is feasible in a finite directed graph under the restriction that no cycle is visited more than once. From this collection, we discard all paths that either never visit e_F or visit e_F but require a nontrivial change in the follower component after the last visit to e_L. The latter are excluded because, by construction, the follower is assumed to stop moving (or remain stationary in the product system) once ϕ_2 has been satisfied. For each remaining leader path, we again fix the leader trajectory, recompute the follower's optimal response via $SolveLTL(\mathcal{T}, \phi_2, \pi_L)$, and apply the predicate $Judge$ to verify satisfaction of ϕ_1; this corresponds to lines 15–24 of Algorithm 2.

Taken together, these two cases guarantee that any global trajectory selected by Algorithm 2 (i) visits both $Goal(\mathcal{P}_2)$ and $Goal(\mathcal{P}_1)$, and (ii) does so in a way that keeps the follower's behavior cost-minimal among all feasible solutions. Therefore, the algorithm computes a solution to Problem 4 that is jointly feasible with respect to (ϕ_1, ϕ_2) and optimal from the follower's perspective.

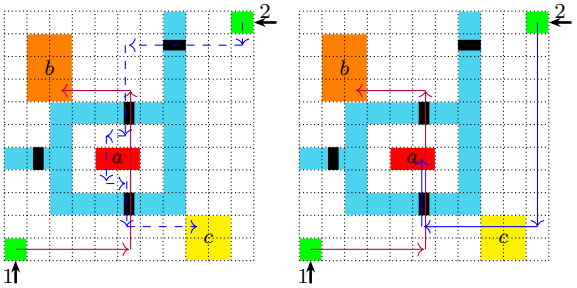

Fig. 2. A successful influential plan for the leader.

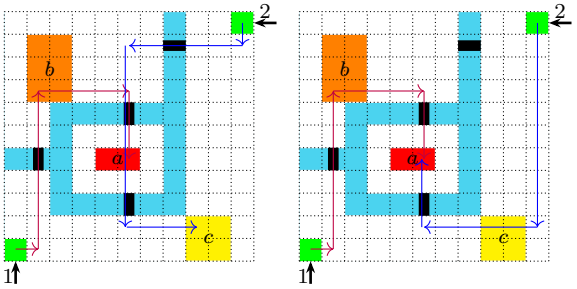

Fig. 3. An incorrect solution.

6. CASE STUDY

In this section, we provide a simple case study on delivery robots to illustrate the proposed *influential-planning problem.*

System model: We consider two delivery robots moving in a grid-world workspace, as illustrated in Figure 2. At each time instant, each robot can choose to move in one of four directions: up, down, left, or right. We assume that each movement incurs a unit cost. The blue grids represent water regions that the agents cannot move across directly. Instead, robots can only traverse the water by using the bridges, denoted by the black rectangles. The workspace consists of three regions of interest, denoted as a, b, and c, each marked with different colors in the figure. The initial locations of the leader and the follower are indicated by the green grids in the figure.

Planning Tasks: The objectives of the robots are to deliver items to different regions. Specifically:

- The leader needs to deliver items to regions a and b,
- The follower needs to deliver items to regions a and c.

Furthermore, from the leader's perspective, it prefers the follower to first deliver to region c and then to region a. However, the follower has no inherent preference for the order in which it visits these regions. Therefore, the leader cannot directly control the follower's actions but can influence its behaviors by strategically choosing its own plan.

Formally, let $\mathcal{AP} = \{a^L, b^L, c^L, a^F, b^F, c^F\}$, where a^L means that the leader is in Region a, and the same applies to the other atomic propositions. Then the tasks for the leader and the follower are specified by the following two co-safe LTL formulae, respectively:

$$\phi_1 = \Diamond a^L \wedge \Diamond b^L \wedge (\neg a^F U c^F)$$
$$\phi_2 = \Diamond a^F \wedge \Diamond c^F$$

A Correct Influential Plan: We apply Algorithm 2 to solve Problem 4 for the above setting. The solution is illustrated in Figure 2, where the red solid line in both figures denotes the leader's plan, π_L, with a total cost of 15 units.

To achieve its task ϕ_2, the follower may choose either:

- The dashed blue plan shown in the left figure, which incurs a cost of 19 units, or
- the solid blue plan shown in the right figure, which incurs a cost of 17 units.

Since the follower aims to minimize its cost, its best response plan, π_F, is the one on the right figure. In this case, the leader successfully enforces its preference on the follower's visiting order.

The intuition behind this solution is as follows: the leader strategically chooses a plan that physically occupies the shortest path available to the follower. As a result, to avoid collisions and minimize its own cost, the follower is compelled to first visit region c, aligning with the leader's desired sequence. This demonstrates how the leader can influence the follower's behavior indirectly by leveraging its own plan to shape the follower's optimal response.

An Incorrect Solution: In Figure 3, we illustrate an incorrect solution to this problem. Specifically, we consider the case where the leader chooses the red solid line as its plan π_L. In this scenario, to achieve its own task ϕ_1, the follower may choose either the plan shown in the left figure or the one in the right figure, both of which have the same cost of 17 units.

However, the plan on the left figure does not satisfy the leader's preference, as the follower first delivers to region a instead of region c. From the leader's perspective, the follower's choice between these two plans is uncontrollable, since both options have the same cost. As a result, π_L is not a correct influential plan, as it may fail to enforce the leader's desired visiting order and thus fail to achieve ϕ_1.

7. CONCLUSION

In this paper, we formulated and solved the influential planning problem for leader-follower systems under temporal logic constraints. We demonstrated that, by strategically designing plans for the leader, it can leverage the rationality of the follower to help achieve its own task. Our work makes an initial step toward addressing the influential planning problem, and there are several important directions we plan to investigate in the future. First, the current algorithm is may not the best one. In future work, we aim to design better algorithms and explore the complexity class of this problem. Second, in our current framework, we assume the follower is absolutely rational in pursuing its optimal response. We plan to relax this assumption by considering bounded rationality of the follower, making the model more realistic. Finally, we aim to investigate the fundamental limits of the influential planning problem, specifically characterizing the classes of logical specifications and environment (map) configurations for which no influential plan exists.

DECLARATION OF GENERATIVE AI AND AI-ASSISTED TECHNOLOGIES IN THE WRITING PROCESS

During the preparation of this work, we used ChatGPT to assist with polishing and editing parts of the manuscript. After using this tool, we carefully reviewed and revised all generated content as needed and take full responsibility for the final version of the paper.

REFERENCES

Baier, C. and Katoen, J.P. (2008). *Principles of model checking*. MIT press.

Bard, J.F. (2013). *Practical bilevel optimization: algorithms and applications*, volume 30. Springer Science & Business Media.

Basar, T. and Selbuz, H. (1979). Closed-loop Stackelberg strategies with applications in the optimal control of multilevel systems. *IEEE Transactions on Automatic Control*, 24(2), 166–179.

Bernstein, A. (2010). A nearly optimal algorithm for approximating replacement paths and k shortest simple paths in general graphs. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, 742–755. SIAM.

Cardona, G.A. and Vasile, C.I. (2024). Planning for heterogeneous teams of robots with temporal logic, capability, and resource constraints. *The International Journal of Robotics Research*, 43(13), 2089–2111.

Chen, Z. and Kan, Z. (2024). Real-time reactive task allocation and planning of large heterogeneous multi-robot systems with temporal logic specifications. *The International Journal of Robotics Research*, 02783649241278372.

Chen, Z., Li, L., and Kan, Z. (2024). Distributed task allocation and planning under temporal logic and communication constraints. *IEEE Robotics and Automation Letters*.

Cherkassky, B.V., Goldberg, A.V., and Radzik, T. (1996). Shortest paths algorithms: Theory and experimental evaluation. *Mathematical programming*, 73(2), 129–174.

Cui, B., Yu, X., Giua, A., and Yin, X. (2025). A stackelberg game approach for signal temporal logic control synthesis with uncontrollable agents. *arXiv preprint arXiv:2502.14585*.

Ding, X., Smith, S.L., Belta, C., and Rus, D. (2014). Optimal control of markov decision processes with linear temporal logic constraints. *IEEE Transactions on Automatic Control*, 59(5), 1244–1257.

Fu, J. and Topcu, U. (2015). Synthesis of shared autonomy policies with temporal logic specifications. *IEEE Transactions on Automation Science and Engineering*, 13(1), 7–17.

Guo, M. and Dimarogonas, D.V. (2015). Multi-agent plan reconfiguration under local ltl specifications. *The International Journal of Robotics Research*, 34(2), 218–235.

Huang, Z., Lan, W., and Yu, X. (2023). A formal control framework of autonomous vehicle for signal temporal logic tasks and obstacle avoidance. *IEEE Transactions on Intelligent Vehicles*.

Kantaros, Y., Kalluraya, S., Jin, Q., and Pappas, G.J. (2022). Perception-based temporal logic planning in un-

certain semantic maps. *IEEE Transactions on Robotics*, 38(4), 2536–2556.

Kantaros, Y. and Zavlanos, M.M. (2020). Stylus*: A temporal logic optimal control synthesis algorithm for large-scale multi-robot systems. *The International Journal of Robotics Research*, 39(7), 812–836.

Kloetzer, M. and Belta, C. (2009). Automatic deployment of distributed teams of robots from temporal logic motion specifications. *IEEE Transactions on Robotics*, 26(1), 48–61.

Kress-Gazit, H., Lahijanian, M., and Raman, V. (2018). Synthesis for robots: Guarantees and feedback for robot behavior. *Annual Review of Control, Robotics, and Autonomous Systems*, 1, 211–236.

Luo, X. and Zavlanos, M.M. (2022). Temporal logic task allocation in heterogeneous multirobot systems. *IEEE Transactions on Robotics*, 38(6), 3602–3621.

Mahulea, C., Kloetzer, M., and González, R. (2020). *Path planning of cooperative mobile robots using discrete event models*. John Wiley & Sons.

Marden, J.R. and Shamma, J.S. (2018). Game theory and control. *Annual review of control, robotics, and autonomous systems*, 1(1), 105–134.

McMahon, J. and Plaku, E. (2016). Mission and motion planning for autonomous underwater vehicles operating in spatially and temporally complex environments. *IEEE Journal of Oceanic Engineering*, 41(4), 893–912.

Moon, J. and Başar, T. (2018). Linear quadratic mean field stackelberg differential games. *Automatica*, 97, 200–213.

Pacheck, A. and Kress-Gazit, H. (2023). Physically feasible repair of reactive, linear temporal logic-based, high-level tasks. *IEEE Transactions on Robotics*, 39(6), 4653–4670.

Ratliff, L.J., Dong, R., Sekar, S., and Fiez, T. (2019). A perspective on incentive design: Challenges and opportunities. *Annual Review of Control, Robotics, and Autonomous Systems*, 2(1), 305–338.

Schillinger, P., Bürger, M., and Dimarogonas, D.V. (2018). Simultaneous task allocation and planning for temporal logic goals in heterogeneous multi-robot systems. *The international journal of robotics research*, 37(7), 818–838.

Shi, W., He, Z., Ma, Z., Ran, N., and Yin, X. (2023). Security-preserving multi-robot path planning for boolean specification tasks using labeled petri nets. *IEEE Control Systems Letters*.

Silano, G., Baca, T., Penicka, R., Liuzza, D., and Saska, M. (2021). Power line inspection tasks with multi-aerial robot systems via signal temporal logic specifications. *IEEE Robotics and Automation Letters*, 6(2), 4169–4176.

Smith, S.L., Tumova, J., Belta, C., and Rus, D. (2011). Optimal path planning for surveillance with temporal-logic constraints. *The International Journal of Robotics Research*, 30(14), 1695–1708.

Wang, Y., Nalluri, S., and Pajic, M. (2020). Hyperproperties for robotics: Planning via hyperltl. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 8462–8468. IEEE.

Wolff, E.M., Topcu, U., and Murray, R.M. (2012). Robust control of uncertain markov decision processes with temporal logic specifications. In *2012 IEEE 51st IEEE Conference on decision and control (CDC)*, 3372–3379. IEEE.

Ye, B., Zhao, J., Li, S., and Yin, X. (2024). Prioritize team actions: Multi-agent temporal logic task planning with ordering constraints. In *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*, 2840–2845. doi:10.1109/CASE59546.2024.10711581.

Yin, X., Gao, B., and Yu, X. (2024). Formal Synthesis of Controllers for Safety-Critical Autonomous Systems: Developments and Challenges. *Annual Reviews in Control*, 100940.

Yu, P., Dong, S., Sheng, S., Feng, L., and Kwiatkowska, M. (2024). Trust-aware motion planning for human-robot collaboration under distribution temporal logic specifications. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 12949–12955. IEEE.

Yu, X., Yin, X., Li, S., and Li, Z. (2022). Security-preserving multi-agent coordination for complex temporal logic tasks. *Control Engineering Practice*, 123, 105130.