2024 IEEE 20th International Conference on Automation
Science and Engineering (CASE)
August 28 - September 1, 2024. Bari, Italy

2024 IEEE 20th International Conference on Automation Science and Engineering (CASE) | 979-8-3503-5851-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/CASE59546.2024.10711581

Prioritize Team Actions: Multi-Agent Temporal Logic Task Planning with Ordering Constraints

Bowen Ye, Jianing Zhao, Shaoyuan Li and Xiang Yin

***Abstract*— In this paper, we investigate the problem of linear temporal logic (LTL) path planning for multi-agent systems, introducing the new concept of *ordering constraints*. Specifically, we consider a generic objective function that is defined for the path of each individual agent. The primary objective is to find a global plan for the team of agents, ensuring they collectively meet the specified LTL requirements. Simultaneously, we aim to maintain a pre-determined order in the values of the objective function for each agent, which we refer to as the ordering constraints. This new requirement stems from scenarios like security-aware planning, where relative orders outweigh absolute values in importance. We present an efficient algorithm to solve this problem, supported by proofs of correctness that demonstrate the optimality of our solution. Additionally, we provide a case study in security-aware path planning to illustrate the practicality and effectiveness of our proposed approach.**

I. Introduction

A. Background and Motivations

In recent years, there has been a growing interest in employing *formal methods* for decision-making and task planning in autonomous systems [10], [25]. This approach involves describing design requirements using formal specifications, allowing for mathematically rigorous reasoning to automatically derive plans that ensure the overall task holds with provable guarantees. Formal methods have gained traction due to their efficiency and certifiability, leading to successful applications in various safety-critical engineering systems such as mobile robots [6], [20], [26], [27], autonomous driving [17], power systems [24], and healthcare facilities [22].

Linear temporal logic (LTL) stands out among various formal specification languages due to its user-friendly yet comprehensive nature in describing temporal requirements over linear behaviors. For example, LTL enables us to describe requirements such as "*conduct surveillance in a specific region infinitely often until detecting an object*", or "*fulfill a request within a finite number of steps once received*". In recent years, LTL has been extensively employed in path planning for mobile robots. For example, in [4], [9], [15], [19] the authors studied the LTL path planning problem to minimize the prefix-suffix cost. Efficient LTL planning algorithms have been developed to address the challenge of state space explosions [8], [14], [16], [21]. LTL path planning with unknown or partial-unknown environments has also been studied recently [7], [12], [13], [29].

In the context of LTL path planning for multi-agent systems, existing works have mainly focused on finding a globally optimal plan for all agents to minimize the objective functions such as total distance or other cost-related metrics. However, there are scenarios where the absolute value of the objective function is less important compared to the *relative order* of the objective function among all agents. One such scenario is security-aware planning [3], [5], [18], [28]. In this setting, the objective is to assign N robots to transport different items through various regions where information may be leaked to an external attacker. The attacker's strategy involves first collecting all leaked information and then targeting the agent that leaks the most data. In this context, the user may prioritize assigning the most sensitive data to the robot whose path has the least leakage among all agents. In this example, the emphasis is on the *ordering* of the leakage levels for each path rather than the absolute value of information leakage. This highlights the importance of relative comparisons in planning for security-sensitive applications.

B. Our Contributions

In this paper, we investigate the problem of LTL path planning for multi-agent systems with ordering constraints. Specifically, we introduce a generic objective function that is defined for the path of each individual agent. Our goal is to find a global plan for the team of agents, ensuring they collectively fulfill the specified LTL requirements. Simultaneously, we aim to maintain a pre-determined order in the values of the objective function for each agent, which we refer to as the ordering constraints. This new constraint is fundamentally different from the conventional cost-based requirements, as it pertains to relative rather than absolute values.

The main contributions of this work are summarized as follows. First, we formally formulate the concept of ordering constraints and their associated LTL path planning problem. Then, we propose an effective algorithm to solve this problem and prove that the obtained solution is optimal. Finally, we provide case studies in security-aware planning to illustrate the feasibility of our approach. To the best of our knowledge, such an ordering requirement has not been explored in the literature concerning LTL path planning.

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

B. Ye, J. Zhao, S. Li and X. Yin are with the Department of Automation, Shanghai Jiao Tong University and the Key Laboratory of System Control and Information Processing, the Ministry of Education of China, Shanghai 200240, China. E-mail: {yebowen1025,jnzhao,syli,yinxiang}@sjtu.edu.cn.

C. Related Works

There are several works in the literature that have addressed concepts related to the ordering constraint explored in our paper. For instance, in [18], [28], the author tackled multi-agent LTL planning tasks with security constraints, ensuring that intruders remain uncertain about whether a specific individual agent is performing crucial subtasks. A recent development is the introduction of HyperATL*, a logic extension of computation tree logic with path variables and strategy quantifiers, where ordering plays a significant role [2]. However, HyperATL* primarily focuses on strategic choices, information flow, and expressing asynchronous hyperproperties, while ordering contributes to describing information priority. Additionally, ordering requirements have been examined in the context of information transmission between network ports [23]. Nevertheless, these contexts differ from our formal setting, which pertains to temporal logic in dynamic systems.

D. Organization

The rest of the paper is organized as follows. First, we provide the formal definitions and system models in Section II. In Section II-D, we formally formulate the problems that we solve in this paper. In Section III, the solution algorithm as well as its correctness proofs are provided. A case study is presented in Section IV to illustrate the applicability of the proposed approach. Finally, we conclude this paper in Section V.

II. Preliminaries and Problem Formulation

A. System Model

We consider a multi-agent system consisting of n agents, for which we denoted by $\mathcal{I}=\{1,2,\dots,n\}$ the index set. The mobility of each agent $i\in\mathcal{I}$ is captured by a *weighted transition system* (WTS) as follows:

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

where X_i is the 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 for any $(x,x')\in\Delta_i$, the agent i can move directly to x' from x; $\mathcal{AP}_i$ is the set of atomic propositions, representing the properties of our interest; $L_i: X_i\to 2^{\mathcal{AP}_i}$ is the labeling function that assigns each state with a set of atomic propositions; $w_i:\Delta_i\to\mathbb{R}_{\geq 0}$ is the cost function such that $w_i(x,x')$ represents the cost incurred when agent i moves from x to x'.

In this work, we assume that all the agents are running synchronously, which can be implemented by using a global clock. To this end, we could capture the mobility of the entire multi-agent system using the following *global transition system* (GTS), denoted by $\mathcal{T}_g$, which is the synchronous product of each $\mathcal{T}_i$ for $i\in\mathcal{I}$.

Definition 1 (GTS): Given n weighted transition systems $\mathcal{T}_i=(X_i,x_{0,i},\Delta_i,\mathcal{AP}_i,L_i,w_i), i\in\mathcal{I}$, the global transition system $\mathcal{T}_g=\mathcal{T}_1\otimes\cdots\otimes\mathcal{T}_n$ is a 6-tuple

$$\mathcal{T}_g=(X_g,x_{0,g},\Delta_g,\mathcal{AP}_g,L_g,w_g)$$

where

- $X_g=X_1\times X_2\times\cdots\times X_n$ is the set of global states;
- $x_{0,g}=x_{0,1}\times\cdots\times x_{0,n}$ is the global initial state;
- $\Delta_g\subseteq X_g\times X_g$ is the transition relation defined by: for any $x_g=(x_1,\dots,x_n), x'_g=(x'_1,\dots,x'_n)\in X_g$, we have $(x_g,x'_g)\in\Delta_g$ if and only if for $\forall\ 1\leq i\leq n$, there is $(x_i,x'_i)\in\Delta_i$;
- $\mathcal{AP}_g=\mathcal{AP}_1\cup\cdots\cup\mathcal{AP}_n$ is the global set of atomic propositions;
- $L_g:\mathcal{AP}_g\to 2^{\mathcal{AP}_g}$ is the labeling function defined by: for any state $x_g=(x_1,\dots,x_n)\in X_g$, we have $L_g(x_g)=L_1(x_1)\cup\cdots\cup L_n(x_n)$;
- $w_g:\Delta_g\to\mathbb{R}_{\geq 0}$ is the cost function defined by: for any $x_g=(x_1,\dots,x_n), x'_g=(x'_1,\dots,x'_n)\in X_g$, if $(x_g,x'_g)\in\Delta_g$, we have $w_g(x_g,x'_g)=\sum_{i=1}^n w_i(x_i,x'_i)$.

A finite path $\tau_g=\tau_g(0)\tau_g(1)\cdots\tau_g(m)\in X_g^*$ is a finite sequence of states such that $\tau_g(0)=x_{0,g}$ and $(\tau_g(j),\tau_g(j+1))\in\Delta_g$ for any $j=0,1,\cdots,m-1$. Note that each state in X_g is an n-tuple and we denote by τ_g^i the path of agent $i\in\mathcal{I}$ in global path τ_g, i.e., $\tau_g^i=\tau_g^i(0)\tau_g^i(1)\cdots\tau_g^i(m)$, where $\tau_g^i(j)$ is the i-th component in $\tau_g(j)$. We denote by $\mathsf{Path}^*(\mathcal{T}_g)$ the set of all finite paths in $\mathcal{T}_g$. Furthermore, we define the cost of a finite path $\tau_g\in\mathsf{Path}^*(\mathcal{T}_g)$ as

$$J(\tau_g)=\sum_{j=0}^{|\tau_g|-2} w_g(\tau_g(j),\tau_g(j+1)) \tag{1}$$

where we denote by $|\tau_g|$ the length of the path τ_g. In words, it captures the total cost incurred by all agents during the execution of τ_g. For each path $\tau_g\in\mathsf{Path}^*(\mathcal{T}_g)$, we define its trace $L(\tau_g)$ as

$$L(\tau_g)=L(\tau_g(0))\cdots L(\tau_g(|\tau_g|-1)).$$

B. Task Specification

In the multi-agent system, agents are assigned with a global high-level task specification. The task is described by linear temporal logic (LTL) formulae. To be specific, 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 $\Diamond\phi=\top U\phi$. LTL formulae are evaluated over infinite words; the readers are referred to [1] for the semantics of LTL. Specifically, an infinite word $\tau\in(2^{\mathcal{AP}})^\omega$ is an infinite sequence over alphabet $2^{\mathcal{AP}}$. We write $\tau\models\phi$ if τ 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, an 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 $\tau' = \tau_1 \cdots \tau_n \in (2^{\mathcal{AP}})^*$ such that $\tau'\tau'' \models \phi$ for any $\tau'' \in (2^{\mathcal{AP}})^\omega$. We denote by $\mathcal{L}^{\phi}_{pref}$ the set of all finite good prefixes of scLTL formula ϕ.

For any scLTL formula ϕ, its good prefixes $\mathcal{L}^{\phi}_{pref}$ 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 $\tau \in \Sigma^*$ is said to be *accepted* by $\mathcal{A}$ if $f(q_0, \tau) \in Q_F$; 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 $\mathcal{L}(\mathcal{A}_\phi) = \mathcal{L}^{\phi}_{pref}$.

C. Ordering Constraints

In this work, we investigate a security problem where the multi-agent system aims to protect some behaviors that are important and do not want to be revealed by the adversary. To formulate the security requirement, we model the important behaviors by the visit to some secret states $X_S \subset X$ which are the areas that can be visited without information leakage. We also define $X_{NS} = X \setminus X_S$ the set of non-secret states. For each agent, to protect the security, it should stay in the non-secret states for as little time as possible. Then for each state $x_i \in X_i, i \in \mathcal{I}$, we define a secret labeling function l that assigns $l(x_i) = 0$ if $x_i \in X_S$ and $l(x_i) = 1$ otherwise. Then, we obtain the following modified transition systems with the secret information included.

Definition 2 (Labeling-GTS): Given a GTS $\mathcal{T}_g = (X_g, x_{0,g}, \Delta_g, \mathcal{AP}_g, L_g, w_g)$ and a set of secret states X_S, its labeling-GTS $\tilde{\mathcal{T}}_g$ is defined by:

$$\tilde{\mathcal{T}}_g = (\tilde{X}_g, \tilde{x}_{0,g}, \tilde{\Delta}_g, \mathcal{AP}_g, \tilde{L}_g, \tilde{w}_g)$$

where

- $\tilde{X}_g = X_g \times \{0, 1\}^n$;
- $\tilde{x}_{0,g} = x_{0,g} \times l_1(x^1_{0,g}) \times \cdots \times l_n(x^n_{0,g})$;
- $\tilde{\Delta}_g \subseteq \tilde{X}_g \times \tilde{X}_g$ is the transition relation defined by: for any $\tilde{x}_g = (x_g, l_1(x^1_g), \ldots, l_n(x^n_g)) \in \tilde{X}_g$ and $\tilde{y}_g = (y_g, l_1(y^1_g), \ldots, l_n(y^n_g)) \in \tilde{X}_g$, we have $(\tilde{x}_g, \tilde{y}_g) \in \tilde{\Delta}_g$ if $(x_g, y_g) \in \Delta_g$ holds;
- $\tilde{L}_g : \tilde{X}_g \to 2^{\mathcal{AP}_g}$ is the labeling function defined by: for any $\tilde{x}_g = (x_g, l_1(x^1_g), \ldots, l_n(x^n_g)) \in \tilde{X}_g$, we have $\tilde{L}_g(\tilde{x}_g) = L_g(x_g)$;
- $\tilde{w}_g : \tilde{\Delta}_g \to \mathbb{R}_{\geq 0}$ is defined by: for any $\tilde{x}_g = (x_g, l_1(x^1_g), \ldots, l_n(x^n_g)), \tilde{y}_g = (y_g, l_1(y^1_g), \ldots, l_n(y^n_g)) \in \tilde{X}_g$, we have $\tilde{w}_g(\tilde{x}_g, \tilde{y}_g) = w_g(x_g, y_g)$.

For the multi-agent system, we propose the concept of *order function* that describes the degree of the information leakage when achieving the LTL tasks.

Definition 3 (Order): Given a labeling-GTS $\tilde{\mathcal{T}}_g$, for each path $\tilde{\tau}_g \in \mathsf{Path}^*(\tilde{\mathcal{T}}_g)$, we define the order of $\tilde{\tau}_g$ for agent i as

$$d_i = \sum_{j=0}^{|\tilde{\tau}_g|-1} l(\tilde{\tau}^i_g(j)) \tag{2}$$

The order of $\tilde{\tau}_g$ is defined as $D(\tilde{\tau}_g) = (d_1, \ldots, d_n)$.

In words, $D(\tilde{\tau}_g)$ characterizes the degree of information leakage for each agent $i \in \mathcal{I}$ when executing $\tilde{\tau}_g$.

In the security-preserving problem, we aim to set the order for the mult-agent system. The order of the agents means the degree of their importance, so it's the proposed information leakage degree too. For convenience, we let the index of each agent be the importance of the information it carries and a *smaller* index means a more important information. We denote by $\mathcal{D}$ the *ordering constraint* for the multi-agent system. Then, formally we have

$$\mathcal{D} \quad \Leftrightarrow \quad \forall 1 \leq i \leq j \leq n : d_i \leq d_j \tag{3}$$

For convenience, we denote by $D(\tilde{\tau}_g) \models \mathcal{D}$ that a path $\tilde{\tau}_g$ satisfies the above ordering constraints.

D. Problem Formulation

Now, we formally present the ordering planning problem of the multi-agent system with LTL specifications as follows.

Problem 1 (Ordering Planning Problem): Given a multi-agent system modeled by a labeling-GTS $\tilde{\mathcal{T}}_g$, the task described by an LTL formula ϕ, and the order constraint given by $\mathcal{D}$ in (3), find a plan $\tilde{\tau}_g \in \mathsf{Path}^*(\tilde{\mathcal{T}}_g)$ such that

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

Remark 1: Different from the standard multi-agent LTL planning problem, we add the ordering constraints condition ii), which each agent is required to satisfy. So this allows some situation of information leaking but the system will compensate for it by some "unimportant" agents leaking actively. Compared to the former that only pursue minimum cost, our *ordering planning problem* ensures the confidentiality of important information while maintaining a low cost.

Remark 2: In the standard multi-agent LTL planning problem, we define the path that satisfies the LTL formulae with the minimum cost as the optimal one. So in our *ordering planning problem*, we define the path that satisfies the LTL formulae and the ordering constraints with the minimum cost as the optimal one.

III. Planning Algorithm

In this section, we propose an efficient algorithm to solve the multi-agent ordering planning problem.

In order to obtain the global path satisfying the LTL task formula Φ, we propose a way to "encode" automaton $\mathcal{A}_\phi$ accepting ϕ into the solution space of $\tilde{\mathcal{T}}_g$, which is detailed as follows.

Definition 4 (Product System): Given a labeling-GTS $\tilde{\mathcal{T}}_g = (\tilde{X}_g, \tilde{x}_{0,g}, \tilde{\Delta}_g, \mathcal{AP}_g, \tilde{L}_g, \tilde{w}_g)$ 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

$$\mathcal{T}_\otimes = (\Pi_\otimes, \pi_0, \Delta_\otimes, w_\otimes)$$

where:

- $\Pi_\otimes \subseteq \tilde{X}_g \times Q$ is the finite set of states;
- $\pi_0 = (\tilde{x}_{0,g}, q_0)$ is the initial state;

- $\Delta_{\otimes} \subseteq \Pi_{\otimes} \times \Pi_{\otimes}$ is the transition relation defined by: for any $\pi_{\otimes} = (x, q), \pi'_{\otimes} = (x', q') \in \Pi_{\otimes}$, we have $(\pi_{\otimes}, \pi'_{\otimes}) \in \Delta_{\otimes}$ if $(x, x') \in \tilde{\Delta}_g$ and $q' = f(q, \tilde{L}_g(x'))$;
- $w_{\otimes} : \Delta_{\otimes} \to \mathbb{R}_{\geq 0}$ is the cost function defined by: for any $\pi_{\otimes} = (x, q), \pi'_{\otimes} = (x', q') \in \Pi_{\otimes}$, we have $w_{\otimes}(\pi_{\otimes}, \pi'_{\otimes}) = \tilde{w}_g(x, x')$ if $(x, x') \in \tilde{\Delta}_g$.

The product system ensures that the movement of the multi-agent satisfies the environment constraints while not violating the LTL formula ϕ. Then we define the set of accepting states for $\mathcal{T}_{\otimes}$ as follows:

$$Goal(\mathcal{T}_{\otimes}) = \{(x, q) \in \Pi_{\otimes} : q \in Q_F\}$$

Based on the construction of $\mathcal{T}_{\otimes}$, we know that, a plan visiting the set $Goal(\mathcal{T}_{\otimes})$ satisfies the scLTL task ϕ. Also, we denote by $R(\pi_0)$ the set of all possible states reachable from π_0 in $\mathcal{T}_{\otimes}$.

Before presenting the algorithm of the multi-agent ordering planning, we first define a "check" function in Algorithm 1, which is used to determine whether or not the optimal paths obtained by solving the standard multi-agent LTL problem satisfies the ordering constraints. To be specific, we check each path by the function and if one of them satisfies $\mathcal{D}$, then this path will be returned as a feasible one, else we will return False, indicating that no paths meet the requirements.

Algorithm 1: Order Checker

Input: a set of the paths T
Output: check result (True or False) and an optimal global plan $\tilde{\tau}_g^*$

for $\tilde{\tau}_g \in T$ **do**
  calculate $D(\tilde{\tau}_g)$ and get d_i for $1 \leq i \leq n$
  if $\forall 1 \leq i \leq j \leq n, d_i \leq d_j$ **then**
    return True, $\tilde{\tau}_g^*$
return False, $\oslash$

Remark 3: In Algorithm 1 there may be more than one path satisfying the ordering LTL task, which means that the paths are the solution to Problem 1, but our algorithm only returns one of them randomly. However, in order to make our case more practical, we should get the trajectory with $\min(\sum_{i=1}^{n} d_i)$ while the size of index set of the agents is n. This means the least total leakage of the information which is defined in Section IV.

Then we can formalize the overall solution in Algorithm 2. To be specific, line 1 gets the $\tilde{\mathcal{T}}_g$ from the input and line 2 converts the task specification to the corresponding DFA $\mathcal{A}_{\phi}$. Line 3 constructs the product system of $\tilde{\mathcal{T}}_g$ and $\mathcal{A}_{\phi}$. Lines 4-5 aim to determine whether there exists a path satisfying the task specification. If agents can not reach $Goal(\mathcal{T}_{\otimes})$ from $\pi_{\otimes,0}$, return "no feasible plan", else we will get the shortest paths for all combinations in line 9-12 by utilizing the shortest path algorithm, e.g. Dijkstra's algorithm [11]. Then we will choose the optimal path set with the least cost among all feasible combinations in line 12. Then line 13-16 will check whether the path we get satisfies the order constraints, if so, return the path, else will return to line 9 but change the line 12 by adding a threshold, which means not only should we get the shortest paths in lines 10-12, but also get the second shortest paths. In line 16, the sec-min J means that we will also get the path whose cost is not minimum but is the least of those bigger than the minimum J. And while continuing, the cost will be third-min and so on.

Algorithm 2: Optimal ordering LTL plan

Input: LTL formula ϕ, labeling-GTS $\tilde{\mathcal{T}}_g$
Output: Optimal global plan τ_g

1 Get the $\tilde{\mathcal{T}}_g : (\tilde{X}_g, \tilde{x}_{0,g}, \tilde{\Delta}_g, \mathcal{AP}_g, \tilde{L}_g, \tilde{w}_g)$
2 Convert ϕ to the corresponding DFA $\mathcal{A}_{\phi} = (Q, q_0, 2^{\mathcal{AP}}, f, Q_F)$
3 Construct the product system using $\tilde{\mathcal{T}}_g$ and $\mathcal{A}_{\phi}$ $\mathcal{T}_{\otimes} = (\Pi_{\otimes}, \pi_0, \Delta_{\otimes}, w_{\otimes})$
4 Construct the $Goal(T_{\otimes})$ and get π_0 from $T_{\otimes}$
5 **if** $R(\pi_0) \cap Goal(\mathcal{T}_{\otimes}) = \emptyset$ **then**
6   return "no feasible plan";
7 **else**
8   **while** *True* **do**
9   **for** $\pi_{\otimes,F} \in Goal(\mathcal{T}_{\otimes})$ **do**
10       $\tau_{pref}(\pi_0, \pi_{\otimes,F}) = shortpath(\pi_0, \pi_{\otimes,F})$
11     $\{\tilde{\tau}_g\} = \{\Pi_g[\tau_{pref}(\pi_0, \pi^*_{\otimes,F})] \mid \pi^*_{\otimes,F} = \text{argmin}_{\pi_{\otimes,F}} J(\Pi_g[\tau_{pref}(\pi_0, \pi_{\otimes,F})]\}$
12   Flag, $\tilde{\tau}_g^*$ = Check($\{\tilde{\tau}_g\}$)
13   **if** *Flag* = *True* **then**
14   return optimal plan τ_g^*
15   **else**
16   return to line 9 while $\{\tilde{\tau}_g\} = \{\Pi_g[\tau_{pref}(\pi_0, \pi^*_{\otimes,F}) \mid \pi^*_{\otimes,F} = \arg\sec\text{ - }\min_{\pi_{\otimes,F}} J\}$

Now we summarize the main result of this work as the following theorem.

Theorem 1: Given a problem in the form of Problem 1, and we solve this problem using the Algorithm 2. It is clear that the final path we get is not only feasible but also optimal [28].

Proof.

- First we get $\{\pi_0\} \cap Goal(\mathcal{T}_{\otimes})$, which contains all paths $\tilde{\tau}_g \models \phi$;
- Then we use Dijkstra to find all of the minimum cost paths, that is the second requirement of the initial multi-agent planning, which is for any other $\tilde{\tau}'_g \in Path^*(\tilde{X}_g)$, we have $J(\tilde{\tau}_g) \leq J(\tilde{\tau}'_g)$. So we get a set of paths $\{\tilde{\tau}_g\}$;
- As in a finite graph, the number of paths in $\{\tilde{\tau}_g\}$ is finite, so we can judge whether there is a path satisfying $\mathcal{D}$ by considering the path one by one;
- If so, we return the path as our result path, because the

cost is the minimum and it satisfies the order constraints, so the path is optimal;
- If we can't find the path, then return to the second step but now we make the threshold of the finding path bigger. This is because the $J = \Sigma w$ and w is the discrete digit, so we can get the w_{min}, and then set $J' = J + w_{min}$, and go on.

In conclusion, Algorithm 2 is sound and complete in solving the optimal planning problem with ordering constraints defined in Problem 1.

IV. Case Study

In this section, we provide some typical examples to demonstrate the feasibility and the application of our algorithm. And from now on, the task with superscripts represents which agent needs to complete the task. For example, a^1 means the task a should be accomplished by agent 1. And our map is a grid map, where each grid represents a state.

We consider a military material transportation scenario, where a convoy completes a transportation task together, transporting materials from the initial point to the destinations individually, during which different vehicles in the convoy may pass through different areas for material supply or refueling. In the transportation map, there will be one or several enemy detection points. The importance of different materials varies and different materials transportation requires specialized vehicles, so the tolerance for exposure of vehicle route information varies.

Our requirement for agents route planning is to minimize the loss of the whole agents on the road as much as possible while completing the task, and to ensure that the more important the vehicle information is, the less it is exposed which means the less time it passed by the insecure area.

Here, we consider four-agent system with its LTL task. So the index of the agents is $\{1, 2, 3, 4\}$.

System model: Suppose that there are four transportation vehicles 1, 2, 3 and 4 working in a 8×8 workspace as shown in Fig.1. So we have four vehicles V_1, V_2, V_3 and V_4 and they start from their own initial position, the lower-left corner, the lower-right corner and the upper left corner of the last two respectively. At each instant,each robot can move left/right/up/down to its adjacent grid or stay at its current grid.And in this case, the cost for moving vertically or horizontally is fixed as one unit.

And each region in the working space has their own special properties:

- the initial positions(cyan region) where the four agents stay at time 0. And the red path belongs to V_1, the blue path belongs to V_2, the brown path shows the trajectory of V_3 and the black path is V_4;
- insecure areas(grey region) where the label is 1, which stands for the leakage of the information adding 1 if the agent visit it for one time;
- mining area of the rare sources(yellow region) where the agent 1 should visit for taking the source. We use a to denote it;

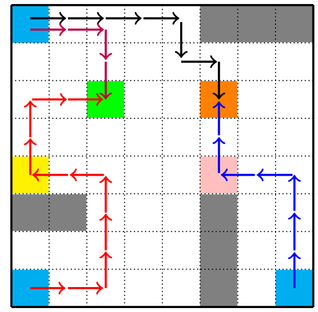

Fig. 1. The path for the ordering task planning

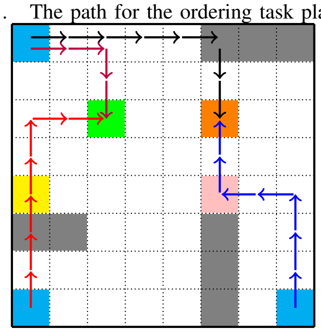

Fig. 2. The path for the minimum cost planning

- storage location for rare resources(green region) where the agent 1 need to visit after it visiting a and work there. We use b to denote it;
- places for extracting ordinary resources(pink region) where agent 2 need to visit for taking the source. We use c to denote it.
- storage location for normal resources(orange region) where the agent 2 need to visit after it visiting c and work there. We use d to denote it;
- the green region and the orange region for V_3 and V_4 is the destination where the vehicles should arrive at, sending the storage machine and working there.

Planning objective: The global task of all agents $\{V_1, V_2, V_3, V_4\}$ is to deliver the rare source and the normal source from the extracting place to the storehouse separately and taking the storage machine to the storage place of rare and ordinary resources. Formally, the overall task specification can be expressed by the following LTL formula:

$$\phi = \Diamond a^1 \wedge \Diamond b^1 \wedge (\neg b^1 U a^1) \wedge \Diamond c^2 \wedge \Diamond d^2 \wedge (\neg d^2 U c^2) \wedge \Diamond b^3 \wedge \Diamond d^4$$

which means the agent 1 must firstly arrive at region a to take the rare resource and then take it to the storage region b, nearly the same as the agent 2. And for agent 3 and 4, their task is to deliver the machine from the initial region to the storage place b and d.

Ordering constraints: As we assume that the smaller number means the more vital importance, the agents $\{V_1, V_2, V_3, V_4\}$ must satisfy the *ordering constraints* which can be interpreted into the visiting times to grey regions of V_1 is less than it of V_2, V_2 is less than V_3 and V_3 is less than V_4, which can be formulated as:

$$\mathcal{D} \quad \Leftrightarrow \quad d_1 \leq d_2 \leq d_3 \leq d_4$$

Solution: The above setting can be modeled by Problem 1. The system model can be constructed in the form of $\tilde{\mathcal{T}}_g = (\tilde{X}_g, \tilde{x}_{0,g}, \tilde{\Delta}_g, \mathcal{AP}_g, \tilde{L}_g, \tilde{w}_g)$ and eight grey regions with label $\{1\}$. Therefore the Algorithm 2 in III can be applied in this case and the synthesized plan is shown in Fig.1. Specifically,the paths in the Fig.1 are the actual paths executed by each robot that satisfy both the task specification and the ordering constraints. And the path in the Fig.2 shows that if we get the minimum cost J, we will break the order constraint which means the enemy can easily know some key information of our rare source. So that's why we added ordering constraints into the normal task planning.

TABLE I
THE COST AND ORDER WITH ORDERING CONSTRAINTS AND WITHOUT IT

	cost J	d_1	d_2	d_3	d_4
with ordering constrainst	29	0	0	0	0
without ordering constraints	25	1	0	0	1

Results: It is obvious in Table I that, despite an additional 16 percent of resources compared to the standard planning scenario, the information leakage of our ordering planning satisfies the most important requirement of minimizing information leakage, which means that the security of information is ensured as much as possible. Therefore, we claim that the multi-agent planning satisfying the ordering constraints has a better performance in protecting the security.

V. Conclusion

In this paper, we solved the problem of LTL path planning for multi-agent systems by introducing a novel ordering constraint. Unlike existing constraints that focus on absolute values, our new constraint centers on relative comparisons. We demonstrated the relevance of this new constraint in the context of information security considerations and provided illustrations through a security-aware path planning case study. In the future, we aim to further investigate the reactive control synthesis problem, taking into account uncertainties in the environment.

References

[1] Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT press, 2008.

[2] Raven Beutner and Bernd Finkbeiner. Hyperatl*: A logic for hyperproperties in multi-agent systems. *Logical Methods in Computer Science*, Volume 19, Issue 2, May 2023.

[3] Yu Chen, Shuo Yang, Rahul Mangharam, and Xiang Yin. You don't know when i will arrive: Unpredictable controller synthesis for temporal logic tasks. *IFAC-PapersOnLine*, 56(2):3591–3597, 2023.

[4] Meng Guo and Dimos V Dimarogonas. Multi-agent plan reconfiguration under local ltl specifications. *The International Journal of Robotics Research*, 34(2):218–235, 2015.

[5] Zhou He, Jiaxing Yuan, Ning Ran, and Xiang Yin. Security-based path planning of multi-robot systems by partially observed petri nets and integer linear programming. *IEEE Control Systems Letters*, 2024.

[6] Zhiyuan Huang, Weiyao Lan, and Xiao Yu. A formal control framework of autonomous vehicle for signal temporal logic tasks and obstacle avoidance. *IEEE Transactions on Intelligent Vehicles*, 2023.

[7] Yiannis Kantaros, Matthew Malencia, Vijay Kumar, and George J Pappas. Reactive temporal logic planning for multiple robots in unknown environments. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11479–11485. IEEE, 2020.

[8] Yiannis Kantaros and Michael M Zavlanos. Sampling-based optimal control synthesis for multirobot systems under global temporal tasks. *IEEE Transactions on Automatic Control*, 64(5):1916–1931, 2018.

[9] Marius Kloetzer and Cristian Mahulea. Path planning for robotic teams based on LTL specifications and petri net models. *Discrete Event Dynamic Systems*, 30(1):55–79, 2020.

[10] Hadas Kress-Gazit, Morteza Lahijanian, and Vasumathi Raman. Synthesis for robots: Guarantees and feedback for robot behavior. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:211–236, 2018.

[11] Steven M LaValle. *Planning algorithms*. Cambridge university press, 2006.

[12] Shuaiyi Li, Mengjie Wei, Shaoyuan Li, and Xiang Yin. Temporal logic task planning for autonomous systems with active acquisition of information. *IEEE Transactions on Intelligent Vehicles*, 2023.

[13] Zhiliang Li, Mingyu Cai, Shaoping Xiao, and Zhen Kan. Online motion planning with soft metric interval temporal logic in unknown dynamic environment. *IEEE Control Systems Letters*, 6:2293–2298, 2022.

[14] Ruijia Liu, Shaoyuan Li, and Xiang Yin. NNgTL: Neural network guided optimal temporal logic task planning for mobile robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[15] Zesen Liu, Meng Guo, and Zhongkui Li. Time minimization and online synchronization for multi-agent systems under collaborative temporal logic tasks. *Automatica*, 159:111377, 2024.

[16] Peng Lv, Guangqing Luo, Ziyue Ma, Shaoyuan Li, and Xiang Yin. Optimal multi-robot path planning for cyclic tasks using petri nets. *Control Engineering Practice*, 138:105600, 2023.

[17] Noushin Mehdipour, Matthias Althoff, Radboud Duintjer Tebbens, and Calin Belta. Formal methods to comply with rules of the road in autonomous driving: State of the art and grand challenges. *Automatica*, 152:110692, 2023.

[18] Weijie Shi, Zhou He, Ziyue Ma, Ning Ran, and Xiang Yin. Security-preserving multi-robot path planning for boolean specification tasks using labeled petri nets. *IEEE Control Systems Letters*, 2023.

[19] Stephen L Smith, Jana Tumova, Calin Belta, and Daniela Rus. Optimal path planning for surveillance with temporal-logic constraints. *The International Journal of Robotics Research*, 30(14):1695–1708, 2011.

[20] Daiying Tian, Hao Fang, Qingkai Yang, Zixuan Guo, Jinqiang Cui, Wenyu Liang, and Yan Wu. Two-phase motion planning under signal temporal logic specifications in partially unknown environments. *IEEE Transactions on Industrial Electronics*, 70(7):7113–7121, 2022.

[21] Cristian Ioan Vasile, Xiao Li, and Calin Belta. Reactive sampling-based path planning with temporal logic specifications. *The International Journal of Robotics Research*, 39(8):1002–1028, 2020.

[22] Matt Webster, Clare Dixon, Michael Fisher, Maha Salem, Joe Saunders, Kheng Lee Koay, Kerstin Dautenhahn, and Joan Saez-Pons. Toward reliable autonomous robotic assistants through formal verification: A case study. *IEEE Transactions on Human-Machine Systems*, 46(2):186–196, 2015.

[23] Damon Wischik, Mark Handley, and Marcelo Bagnulo Braun. The resource pooling principle. *ACM SIGCOMM Computer Communication Review*, 38(5):47–52, 2008.

[24] Zhe Xu, Agung Julius, and Joe H Chow. Energy storage controller synthesis for power systems with temporal logic specifications. *IEEE Systems Journal*, 13(1):748–759, 2017.

[25] Xiang Yin, Bingzhao Gao, and Xiao Yu. Formal Synthesis of Controllers for Safety-Critical Autonomous Systems: Developments and Challenges. *Annual Reviews in Control*, page 100940, 2024.

[26] Xinyi Yu, Weijie Dong, Shaoyuan Li, and Xiang Yin. Model predictive monitoring of dynamical systems for signal temporal logic specifications. *Automatica*, 160:111445, 2024.

[27] Xinyi Yu, Chuwei Wang, Dingran Yuan, Shaoyuan Li, and Xiang Yin. Model predictive control for signal temporal logic specifications with time interval decomposition. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 7849–7855. IEEE, 2023.

[28] Xinyi Yu, Xiang Yin, Shaoyuan Li, and Zhaojian Li. Security-preserving multi-agent coordination for complex temporal logic tasks. *Control Engineering Practice*, 123:105130, 2022.

[29] Jianing Zhao, Keyi Zhu, Mingyang Feng, and Xiang Yin. To explore or not to explore: Regret-based ltl planning in partially-known environments, 2024.