Preprints of the 22nd IFAC World Congress
Yokohama, Japan, July 9-14, 2023

Failure-Aware Self-Diagnostic Task Planning under Temporal Logic Specifications ⋆

Jianing Zhao * **Shuqi Wang** * **Xiang Yin** *

* *Department of Automation, Shanghai Jiao Tong University Shanghai 200240, China*
(e-mail: {jnzhao,wangshuqi,yinxiang}@sjtu.edu.cn)

Abstract: This paper investigates the problem of task planning for high-level specifications described by linear temporal logic (LTL) formulae. Existing works on this topic mainly based on the assumption that the functionalities of the system are always correct during the execution. In this work, we consider the scenario where the system is subject to internal failures that cannot be measured directly but may be inferred by a sequence of actions. The objective is to design a *failure-aware* task plan such that (i) the system will achieve the LTL task when there is no failure; and (ii) along the designed plan, any potential failure can be detected within a bounded number of steps. We provide a framework for modeling the behavior of the system with potential internal failures. Furthermore, an effective algorithm is designed to synthesize an optimal *self-diagnostic* plan, in the form of the prefix-suffix structure, such that these two requirements are satisfied. We illustrate the proposed framework by a case study of production task planning.

Keywords: Linear Temporal Logic, Task Planning, Failure Diagnosis, Partial Observation.

1. INTRODUCTION

Task planning and decision-making in dynamic environments are the central problems in the applications of autonomous systems. In recent years, there has been growing interest in designing high-level plannings for robots in order to achieve some complex tasks described by formal languages or temporal logics such as linear temporal logic (LTL) (Kress-Gazit et al., 2018). LTL is an expressive and user-friendly language for expressing high-level requirements such as "first do something and then do something" or "repeating some actions infinitely often" (Baier and Katoen, 2008).

In the context of robot task planning, considerable works have been done in the past years on synthesizing plans for LTL tasks; see, e.g., Ulusoy et al. (2013); Kloetzer and Mahulea (2020); Shi et al. (2022); Yu et al. (2022). For example, Smith et al. (2011) provides a framework for finding an infinite plan in the prefix-suffix structure such that a surveillance task is achieved optimally. In Guo and Dimarogonas (2015), the authors consider how to redesign the plan when the environment changes. In Luo et al. (2021), sampling-based techniques are developed to enhance the scalability of the planning process. Learning-based techniques have also been developed when the environment is unknown (Cai et al., 2020).

The above mentioned works mainly focus on task planning problems for the ideal case, where robots are assumed to work correctly as desired. In practice, however, the robot itself or some of its functionalities may fail during its execution. For example, consider a surveillance robot whose task is to visit a critical region infinitely so that pictures can be taken and then been uploaded to the cloud. However, if the camera of the robot is broken during the execution, although the robot is still visiting the critical region infinitely often (and tries to take pictures), the actual task will be not achieved due to the failure. Therefore, the robot should be aware of the failure when it occurs in order to reconfigure the plan or to fix the failure component.

There have been many studies on LTL task planning under uncertainty or in the presence of adversaries (Ding et al., 2014; Niu and Clark, 2019; Ramasubramanian et al., 2020; Xie et al., 2021). In particular, a game-based model is usually used to capture the transition non-determinism of the system when taking actions. However, in general, uncertainties are different from failures. Although both of them are uncontrollable, the consequences of uncertainties can usually be observed directly, while failures are usually internal or hidden, which cannot be directly measured. One may need to choose a well-designed sequence in order to *infer* the occurrence of failure. How to achieve a desired task while maintaining the capability in detecting potential internal failures is an important but challenging task.

In this work, we formulate and solve a new failure-aware task planning problem for LTL specifications. Specifically, we consider a scenario where, in addition to the external dynamic of the system such as the mobility of the robot, the system also has an internal dynamic which is subject to failures. Furthermore, we assume that the occurrences of failures may not be observed directly by the system. Therefore, one needs to carefully design a plan such that it is (i) *task satisfying* in the sense that the LTL task is

⋆ This work was supported by the National Natural Science Foundation of China (62061136004, 62173226, 61803259) and by the National Key Research and Development Program of China (2018AAA0101700).

achieved when no failure occurs; and (ii) *self-diagnostic* in the sense that the failure can always be detected within a bounded number of steps whenever it occurs. Our contributions are twofold. First, we provide a modeling framework using transducers to capture this failure-aware task planning scenario. Second, we provide an effective algorithm for designing an optimal plan, in the prefix-suffix structure, such that it is both task satisfying and self-diagnostic.

Our works is closely related to the literature on fault diagnosis of discrete-event systems Yin and Lafortune (2017); Ma et al. (2021); Dong et al. (2023), particularly, active diagnosis Sampath et al. (1998); Yin and Lafortune (2016); Lin et al. (2017); Hu et al. (2020). However, in active diagnosis, one aim to design a feedback control strategy to detect fault rather than a failure-aware open-loop plan. Therefore, the problem setting is different from ours. Furthermore, existing works on active diagnosis do not consider ensuring high-level tasks such as LTL tasks as we considered in this work.

Basic Notations: For a set A, we denote by $|A|$ and 2^A its cardinality and its power set, respectively. A finite word σ over A is a sequence in the form of $\sigma = a_1 \cdots a_n$, where $a_i \in A$ and we denote by A^* the set of all finite words over A. Analogously, we denote by A^ω the set of all infinite words over A. Given a set of words $L \subseteq \Sigma^* \cup \Sigma^\omega$ and a word $\sigma \in L$, we denote by L/σ the set of the post-words for w in L, i.e., $L/\sigma := \{w \in \Sigma^* : \sigma w \in L\}$. We also define the prefix closure of L as $\overline{L} = \{\sigma \in \Sigma^* : \exists w \in \Sigma^* \text{ s.t. } \sigma w \in L\}$.

2. A MOTIVATING EXAMPLE

To motivate our study, let us consider an automatic manufacturing factory shown in Figure 1, where there are a team of AGVs and four types of production modules including:

(1) a stock (S) where raw materials can be loaded on the AGVs for further processing;
(2) a material processing machine (M) where raw materials can be processed, denoted by action a_0, into components commonly required by two different products;
(3) an assembly machine (A) which can assemble the processed components into two different products (Product 1 and 2) by actions a_1 and a_2, respectively;
(4) two product storages (P1/P2) where Product 1 and 2 are deposited, respectively.

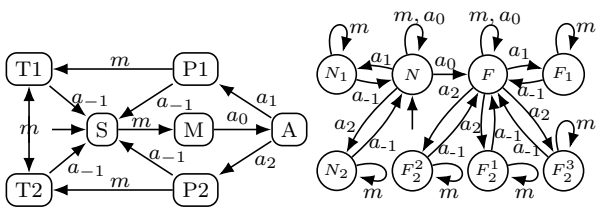
The above modules can be employed to constitute two available production lines:

$$L1 : \left(\mathrm{S} \xrightarrow{m} \mathrm{M} \xrightarrow{a_0} \mathrm{A} \xrightarrow{a_1} \mathrm{P1} \xrightarrow{a_{-1}} \mathrm{S}\right)^\omega$$

$$L2 : \left(\mathrm{S} \xrightarrow{m} \mathrm{M} \xrightarrow{a_0} \mathrm{A} \xrightarrow{a_2} \mathrm{P2} \xrightarrow{a_{-1}} \mathrm{S}\right)^\omega$$

where actions m and a_{-1} denote "move" and "back and reset" actions available for the AGVs, respectively.

Now we assume that process a_0 in the material processing machine may have small chance to result in a permanent failure of the machine. Once failure occurs, the processed material will be unqualified which will further lead to the products assembled in the assembly machine unqualified. This internal change from normal status to failure status



(a) Production Processes G (b) Transducer T

Fig. 1. Motivating example: an automatic factory

Table 1. Observation Map

	N_1	F_1	N_2	F_2^1	F_2^2	F_2^3
T_1	o_1	o_1	o_1	o_1	χ_1	χ_1
T_2	χ_2	χ_2	o_2	χ_2	o_2	χ_2

can be captured by a transducer T shown in Figure 1(b) (the formal definition will be presented in the next section). Intuitively, states with "N" and "F" denote, respectively, normal and failure statuses of the material processing machine. Furthermore, under the failure status, by using the unqualified material and taking action a_2 to assemble Product 2, the resulting product may be at three possible unqualified levels non-deterministically.

We assume that the failure status of the material processing machine cannot be measured directly. On the other hand, we can infer its status by testing the products assembled by the assembly machine. To this end, we assume that there are two test machines (T1/T2) available in the production configuration to test properties of the products. We assume that the test results are available to the user, and for qualified products and unqualified products at different levels, the test results at different test machines are shown in Table 1, where "o" and "χ" denote test results "pass" and "fail", respectively.

Now, we consider a simple production task as follows: (i) keep producing products indefinitely; and (ii) once failure occurs in the material processing machine, it can be diagnosed within at most K steps. Such a problem is referred to as the *diagnostic planning problem* in this work. Furthermore, we assume that making Product 2 is more costly than making Product 1. Therefore, the user tends to produce as many Product 1 as possible.

Clearly, if one only uses production line $L1$, then the diagnostic condition cannot be satisfied no matter how test machines T1/T2 are used. This is because both the qualified Product 1 (N_1) and the non-qualified Product 1 (F_1) have the same test results. On the other hand, one can combine production line $L2$ together with two successive tests at two different test machines for Product 2, denoted by

$$L2' : \left(\mathrm{S} \xrightarrow{m} \mathrm{M} \xrightarrow{a_0} \mathrm{A} \xrightarrow{a_2} \mathrm{P2} \xrightarrow{m} \mathrm{T2} \xrightarrow{m} \mathrm{T1} \xrightarrow{a_{-1}} \mathrm{S}\right)^\omega$$

This plan is actually *self-diagnostic* due to the fact that once the failure occurs in the material processing machine (i.e., $N \xrightarrow{a_0} F$ in T), no matter what level of unqualified Product 2 the assembly machine produces (F_2^1, F_2^2 or F_2^3), they can be distinguished by test machines T1 and T2 based on their different test results.

Furthermore, we note that the total cost incurred in production line $L2'$ is larger than that of $L1$. To reduce the

total cost while maintaining the capability in diagnosing failures, the very optimal plan is actually to execute $L1$ as much as possible within $K-5$ steps, and then to execute $L2'$ once to detect the possible failure, and repeat them infinitely often.

In what follows, we will formulate such a scenario using the notion of optimal diagnostic planning problem. First, we will provide a modeling framework to describe the execution of the system, the internal (possible failure) dynamic of the system, as well as the observation mapping. Then we will provide a formal solution to such a problem.

3. MODELING FRAMEWORK AND PROBLEM FORMULATION

3.1 Weighted Transition Systems

The environment, describing the mobility of the agent, is modeled as a weighted transition system

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

where

- X is the set of states representing different regions of the workspace;
- $x_0 \in X$ is the initial state representing the starting region of the agent;
- Act is the set of actions available in the environment;
- $\Delta : X \times Act \to X$ is the transition function such that a transition $x' = \Delta(x, a)$ means that the agent, starting from region $x \in X$, can move to region $x' \in X$ directly by taking action $a \in Act$;
- $\mathcal{AP}$ is the set of atomic propositions representing some basic properties of our interest;
- $L : X \to 2^{\mathcal{AP}}$ is a labeling function that assigns each state a set of atomic propositions;
- $w : Act \to \mathbb{R}_{\geq 0}$ is a cost function evaluating the cost incurred when the agent takes different actions.

For such map geometry, the planning problem is referred to designing a sequence of actions, $\tau = a_1 a_2 \cdots \in Act^\omega$, called a plan, which determines a unique sequence of states $X(\tau) := x_0 x_1 x_2 \cdots \in X^\omega$ such that $x_{i+1} = \Delta(x_i, a_i), i \geq 0$ and we call such $X(\tau)$ a path. The trace of an infinite path $X(\tau)$ is an infinite sequence over $2^{\mathcal{AP}}$ denoted by $L(X(\tau)) = L(x_0)L(x_1)L(x_2)\cdots$. With slight abuse of the notation, we denote by $Act(x) = \{a \in Act : \Delta(x, a)!\}$ the set of actions available at $x \in X$. For convenience, we define the set of all plans available in G as $\mathsf{Plan}^\omega(G)$; we also denote by $\mathsf{Plan}^*(G)$ the set of finite plans available in G analogously. Given a finite plan $\tau = a_1 a_2 \cdots a_n \in \mathsf{Plan}^*(G)$, we define its cost by $J(\tau) = \sum_{i=1}^{n} w(a_i)$.

3.2 Transducers and Self-Diagnostic Plans

As shown in the motivating example in Section 2, in many practical scenarios, taking different action will not only bring the agent from one region to another in the environment, but will also trigger *internal status changes* of its own. Particularly, we are interested in those statuses representing some internal failures of the agent. In general, the observation of the agent is determined together by the actual region in the environment as well as the internal state of the agent. For example, at normal or failure statuses and at different test machines, the agent will observe different result.

In order to capture such internal status change of the agent as well as the associated observation mapping, we use a *transducer* structure defined as follows:

$$T = (S, s_0, \Sigma, \xi, O, X, H)$$

where

- $S = S_N \dot\cup S_F$ is the set of the agent's internal states, where S_N and S_F are the sets of normal states and failure states,
- $s_0 \in S_N$ is the initial internal state;
- Σ is the alphabet, in particular, we have $\Sigma = Act$;
- $\xi : S \times \Sigma \to 2^S$ is the non-deterministic transition function defined by:
  - for $s \in S_N$, we have $\xi : S_N \times \Sigma \to 2^S$;
  - for $s \in S_F$, we have $\xi : S_F \times \Sigma \to 2^{S_F}$;
- O is the set of observation symbols;
- $H : S \times X \to O$ is the observation mapping.

For convenience, we denote by $\mathcal{L}^\omega(T) := \{\sigma \in \Sigma^\omega : \forall \sigma' \in \overline{\{\sigma\}} \text{ s.t. } \xi(s_0, \sigma')!\}$ the language generated by T. Here we assume $\mathsf{Plan}^\omega(G) \subseteq \mathcal{L}^\omega(T)$, which is without of generality since it simply says that any plan available in G is well defined in the transducer T. In fact, words in $\mathcal{L}^\omega(T) \setminus \mathsf{Plan}^\omega(G)$ are not of our interest due to the fact they are impossible to be executed in the environment G.

Since the internal states cannot be observed by the agent's own directly, when executing plan τ in environment G, the agent has to *infer* whether or not some failure has actually occurred based on the observation at regions of the environment. When the answer is "yes *for sure*" for any possible failures, we call such τ a *diagnostic plan*.

Before formally defining a diagnostic plan, we introduce some notations. Given a plan $\tau = a_1 a_2 \cdots \in \mathsf{Plan}^\omega(G)$, we define all its compatible runs in T by

$$\mathsf{Run}^\omega(\tau) = \{s_0 s_1 \cdots \in S^\omega : s_{i+1} \in \xi(s_i, a_{i+1}), i = 0, 1, \dots\}. \tag{1}$$

Given $\mathsf{Run}^\omega(\tau)$, we denote by $\mathsf{Run}^*(\tau)$ its prefix closure, i.e., the set of all compatible finite runs of τ in T. Given a plan $\tau \in \mathsf{Plan}^\omega(G)$ with $X(\tau) = x_0 x_1 x_2 \cdots$ being its path in environment G, for each compatible run $\rho = s_0 s_1 s_2 \cdots \in \mathsf{Run}^\omega(\tau)$ in T, we denote by

$$Obs(\rho) = H(s_0, x_0)H(s_1, x_1)H(s_2, x_2)\cdots \in O^\omega$$

the observation of ρ in G. Analogously, we define $\mathsf{Run}^*(\tau)$ and the observation of each finite run in $\mathsf{Run}^*(\tau)$. Given each $\mathsf{Run}^*(\tau)$, we have the partition

$$\mathsf{Run}^*(\tau) = \mathsf{Run}^*_N(\tau) \dot\cup \mathsf{Run}^*_F(\tau)$$

where $\mathsf{Run}^*_N(\tau)$ is the set of all normal runs such that all states visited by each $\rho \in \mathsf{Run}^*_N(\tau)$ are normal states and $\mathsf{Run}^*_F(\tau)$ is the set of faulty runs such that each run $\rho \in \mathsf{Run}^*_F(\tau)$ visited some fault states.

Now, we present the notion of diagnostic plan as follows.

Definition 1. (Self-Diagnostic Plan). Given WTS G and transducer T and a parameter $K \in \mathbb{N}$, we say a plan $\tau \in \mathsf{Plan}^\omega(G)$ is *self-diagnostic* w.r.t. T and K if any occurrence of fault can always be determined based on its observation in G within at most K-steps, i.e.,

$$\begin{aligned}&(\forall \rho \in \mathsf{Run}^*_F(\tau))(\forall w \in \mathsf{Run}^*_F(\tau)/\rho : |w| \geq K)\\ &(\forall \rho' \in \mathsf{Run}^*(\tau))[Obs(\rho w) = Obs(\rho') \Rightarrow \rho' \in \mathsf{Run}^*_F(\tau)]\end{aligned} \tag{2}$$

3.3 Linear Temporal Logic Specifications

The task of the agent is described by linear temporal logic (LTL) formulae. The syntax of general LTL formula is given as follows

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

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 $\Diamond\phi = \top U \phi$ and "always" by $\Box\phi = \neg\Diamond\neg\phi$. LTL formulae are evaluated over infinite words; the readers are referred to Baier and Katoen (2008) for the semantics of LTL. Specifically, for an infinite word $w \in (2^{\mathcal{AP}})^\omega$ over $2^{\mathcal{AP}}$, we write $w \models \phi$ if w satisfies LTL formula ϕ.

For any LTL formula ϕ, it is well known that any infinite word satisfying ϕ can be accepted by a *nondeterministic Büchi automaton* (NBA). Formally, a NBA is a 5-tuple $\mathcal{A} = (Q_B, Q_{B,0}, 2^{\mathcal{AP}}, \delta_B, F_B)$, where Q_B is the set of states; $Q_{B,0} \subseteq Q_B$ is the set of initial states; $2^{\mathcal{AP}}$ is the alphabet; $\delta_B : Q_B \times 2^{\mathcal{AP}} \to 2^{Q_B}$ is the transition function; and $F_B \subseteq Q_B$ is the set of accepting states. Given an infinite word $w = w_1 w_2 \cdots \in (2^{\mathcal{AP}})^\omega$, an *infinite run* of $\mathcal{A}$ induced by w is an infinite sequence $\pi = q_0 q_1 q_2 \cdots \in Q_B^\omega$ such that $q_0 \in Q_{B,0}$ and $q_{i+1} \in \delta_B(q_i, w_i), \forall i \geq 1$. An infinite run π is said to be accepted by $\mathcal{A}$ if $\texttt{Inf}(\pi) \cap F_B \neq \emptyset$, where we denote by $\texttt{Inf}(\pi)$ the set of states that appear infinite number of times in π. An infinite word w is said to be accepted by $\mathcal{A}$ if it induces an infinite run accepted by $\mathcal{A}$. Furthermore, given any LTL formula ϕ, there always exists an NBA over $2^{\mathcal{AP}}$ that accepts exactly all the infinite words satisfying ϕ (Vardi and Wolper, 1986).

3.4 Diagnosic Planning Problem

In the standard LTL task planning problem, it is required to find a plan $\tau \in \mathsf{Plan}^\omega(G)$ such that $L(X(\tau)) \models \phi$. Due to the structure of the accepting condition in NBA, it suffices for us to find an plan in the following form of *preffix-suffix structure* Smith et al. (2011)

$$\tau = a_1 \cdots a_k \, (a_{k+1} \cdots a_{k+m})^\omega \in \mathsf{Plan}^\omega(G)$$

with its path being

$$X(\tau) = x_0 x_1 \cdots x_k \, (x_{k+1} \cdots x_{k+m})^\omega$$

such that $L(X(\tau)) \models \phi$. Intuitively, $x_{k+1} \cdots x_{k+m}$ is the *suffix* that forms a cycle such that the agent should execute infinitely often, while $x_0 x_1 \cdots x_k$ is the *prefix* representing the transient path that leads to the cyclic path.

In our work, given such an infinite plan τ of prefix-suffix structure, we consider the weighted sum cost of its prefix and suffix, i.e.,

$$\hat{J}(\tau) = \alpha J(a_1 \cdots a_k) + (1-\alpha) J(a_{k+1} \cdots a_{k+m}) \quad (3)$$

where $\alpha \in [0,1]$ is a parameter to adjust the weights of the prefix and the suffix.

With the above preliminaries, we present the optimal self-diagnostic task planning problem formulation as follows.

Problem 1. (Optimal Self-Diagnostic Planning). Given environment of the agent modeled as G, the internal behavior of the agent captured by transducer T, and a task described by an LTL formula ϕ, design a plan $\tau \in \mathsf{Plan}^\omega(G)$ such that i) $L(X(\tau)) \models \phi$; ii) τ is self-diagnostic; iii) for any other plan $\tau' \in \mathsf{Plan}^\omega(G)$ satisfying the above two requirements, we have $\hat{J}(\tau') \geq \hat{J}(\tau)$.

4. SELF-DIAGNOSTIC PLANNING ALGORITHM

4.1 Maximally Diagnosable WTS

Given two WTSs $G = (X, x_0, Act, \Delta, \mathcal{AP}, L, w)$ and $G' = (X', x_0', Act', \Delta', \mathcal{AP}', L', w')$, we say G' is *finer* than G, denoted by $G' \sqsubseteq G$, if the followings hold:

i) $X' = X, x_0' = x_0, Act' = Act, \mathcal{AP}' = \mathcal{AP}, L' = L, w' = w$;
ii) for any $x' = \Delta'(x, a)$, we have $x' = \Delta(x, a)$.

It directly follows that $\mathsf{Plan}^\omega(G') \subseteq \mathsf{Plan}^\omega(G)$. Specifically, we are interested in a particular WTS $G' \sqsubseteq G$ satisfying the following two conditions

C1 each plan $\tau \in \mathsf{Plan}^\omega(G')$ is self-diagnostic w.r.t. T;
C2 for any G'' satisfying C1, we have $G'' \sqsubseteq G'$.

We say a WTS $G' \sqsubseteq G$ is *diagnosable* w.r.t. T if it satisfies C1 and *maximally diagnosable* w.r.t. T if it satisfies both C1 and C2.

Next, we present a sufficient and necessary condition to determine the diagnosability of a WTS. Since K is given as a design parameter in the definitions of self-diagnostic plan as well as the diagnosability of a WTS, we make some refinement on T by "counting" when failure occurs as follows.

Given a transducer $T = (S, s_0, \Sigma, \xi, O, H)$, we define a new transducer $\tilde{T} = (\tilde{S}, \tilde{s}_0, \Sigma, \tilde{\xi}, O, \tilde{H})$, where

- $\tilde{S} \subseteq S \times \{-1, 0, 1, \ldots, K\}$ is the set of states;
- $\tilde{s}_0 = (s_0, -1) \in \tilde{S}$ is the initial state;
- $\tilde{\xi} : \tilde{S} \times \Sigma \to 2^{\tilde{S}}$ is the partial transition function such that for any $\tilde{s} = (s, n), \tilde{s}' = (s', n') \in \tilde{S}, a \in \Sigma$, we have $\tilde{s}' \in \tilde{\xi}(\tilde{s}, a)$ if the followings hold:
  i) $s' \in \xi(s, a)$;
  ii)

$$n' = \begin{cases} -1, & \text{if } n = -1 \wedge s' \in S_N \\ n+1, & \text{if } \begin{matrix} 0 \leq n < K \text{ or} \\ n = -1 \wedge s' \in S_F \end{matrix} \\ K, & \text{if } n = K \end{cases} \quad (4)$$

- O is the set of observation symbols;
- $\tilde{H} : \tilde{S} \times X \to O$ is the observation map defined by: for any $(s, n) \in \tilde{S}$, we have $\tilde{H}((s, n), x) = H(s, x), \forall x \in X$.

For convenience, for each $\tilde{s} = (s, n) \in \tilde{S}$, we denote by $[\tilde{s}]_n$ the second component of $\tilde{s}$.

Definition 2. (Information State). Given a WTS G and a finite plan $\tau \in \mathsf{Plan}^*(G)$, for an observation $\alpha = Obs(\rho)$ for some $\rho \in \mathsf{Run}^*(\tau)$, the *Information State* (IS) of τ and α w.r.t. $\tilde{T}$ is defined as

$$IS(\tau, \alpha) = \left\{ s \in \tilde{S} : \begin{matrix} s \in \tilde{\xi}(\tilde{s}_0, \tau) \wedge \\ \exists \rho' \in \mathsf{Run}^*(\tau) \text{ s.t. } Obs(\rho') = \alpha \end{matrix} \right\} \quad (5)$$

We describe the dynamic of all ISs w.r.t. WTS G and transducer $\tilde{T}$ as the following structure of IS Transducer.

Definition 3. (IS Transducer). Given WTS G and transducer T, its *IS transducer* is a deterministic finite-state automaton

$$T_{IS} = (Y_{IS}, y_{IS,0}, \Sigma_{IS}, \delta_{IS})$$

where

- $Y_{IS} = X \times 2^{\tilde{S}}$ is the set of states;
- $y_{IS,0} = (x_0, \{\tilde{s}_0\}) \in Y_{IS}$ is the initial state;
- $\Sigma_{IS} = Act \times O$ is the alphabet;
- $\delta_{IS} : Q_{IS} \times Act \to Y_{IS}$ is the transition function defined by: for any $y = (x, u), y' = (x', u') \in Y_{IS}$ and $(a, o) \in \Sigma_{IS}$, we have $y' = \delta_{IS}(y, (a, o))$ if
  i) $x' = \Delta(x, a)$; and
  ii) $u' = \{s' \in \tilde{S} : \exists s' \in u \text{ s.t. } s' \in \tilde{\xi}(s, a) \wedge \tilde{H}(s', x') = o\}$

For convenience, for each $y = (x, u) \in Q_{IS}$, we denote by $[y]_{IS} = u$ the second component of y. By the construction of T_{IS}, we have the following result.

Proposition 1. Given any finite plan $\tau \in \mathsf{Plan}^*(G)$ and an observation $\alpha = o_0\alpha' = Obs(\rho)$ for some $\rho \in \mathsf{Run}^*(\tau)$, we have $IS(\tau, \alpha) = [\delta_{IS}(y_{IS,0}, (\tau, \alpha'))]_{IS}$.

Then we introduce a new structure based on which we present the sufficient and necessary condition.

Definition 4. (Diagnostic Structure). Given a WTS G, the modified transducer $\tilde{T}$ and the IS transducer T_{IS}, the *diagnostic structure* (DS) is a deterministic finite-state automaton

$$\mathcal{D} = (Q_{\mathcal{D}}, q_{\mathcal{D},0}, Act, \delta_{\mathcal{D}})$$

where

- $Q_{\mathcal{D}} \subseteq X \times 2^{\tilde{S}} \times 2^{2^{\tilde{S}}}$ is the set of states which satisfies: for any $(x, u, v) \in Q_{\mathcal{D}}$, we have
  i) v is a partition of u, i.e., $u = \dot{\cup}_{z \in v} z$;
  ii) for any $z \in v$, $s, s' \in z$, we have $\tilde{H}(s, x) = \tilde{H}(s', x)$;
- $q_{\mathcal{D},0} = (x_0, \{\tilde{s}_0\}, \{\{\tilde{s}_0\}\})$ is the initial state;
- Act is the alphabet;
- $\delta_{\mathcal{D}} : Q_{\mathcal{D}} \times Act \to Q_{\mathcal{D}}$ is the transition function defined by: for any $q = (x, u, v), q' = (x', u', v') \in Q_{\mathcal{D}}$ and any $a \in Act$, we have $q' = \delta_{\mathcal{D}}(q, a)$ if the followings hold:
  i) $x' = \Delta(x, a)$;
  ii) $u' = \bigcup_{s \in u} \tilde{\xi}(s, a)$;
  iii) $v' = \left\{ y \in 2^{\tilde{S}} : \exists z \in v, o \in O \text{ s.t. } y = [\delta_{IS}(z, (a, o))]_{IS} \right\}$.

Note that, although the domain of the third component of the state-space of $\mathcal{D}$ is $2^{2^{\tilde{S}}}$, the size of $\mathcal{D}$ is still single exponential in $\tilde{T}$. This is because, for each state $(x, u, v) \in Q_{\mathcal{D}}$, the third component v is simply a partition of the second component u.

Now, we present the sufficient and necessary condition to determine the diagnosability of a WTS G as follows.

Proposition 2. A WTS $G = (X, x_0, Act, \Delta, \mathcal{AP}, L, w)$ is diagnosable if and only if its diagnostic structure $\mathcal{D} = (Q_{\mathcal{D}}, q_{\mathcal{D},0}, Act, \delta_{\mathcal{D}})$ satisfies: for $\forall (x, u, v) \in Q_{\mathcal{D}}$, we have

$$(\forall z \in v)(\forall s, s' \in z)\,[[s]_n = K \Rightarrow [s']_n \neq -1] \qquad (6)$$

Now, based on the result of Proposition 2, we present the following algorithm to obtain the maximally diagnosable WTS $\mathcal{G} \sqsubseteq G$ from the given WTS G and transducer T.

Algorithm 1: Construct Maxi. Diagnos. WTS

Input: WTS $G = (X, x_0, Act, \Delta, \mathcal{AP}, L, w)$ and Transducer $T = (S, s_0, \Sigma, \xi, O, H)$;
Output: Maximally diagnosable WTS G^*;
Construct $\tilde{T}$ from T;
Construct T_{IS} from $\tilde{T}$ and G;
Construct DS $\mathcal{D}$ from G, $\tilde{T}$ and T_{IS};
Construct a new WTS $\hat{G} = (\hat{X}, \hat{x}_0, \hat{Act}, \hat{\Delta}, \hat{\mathcal{AP}}, \hat{L}, \hat{w})$
 where $(\hat{X}, \hat{x}_0, \hat{Act}, \hat{\mathcal{AP}}, \hat{L}, \hat{w}) \leftarrow (X, x_0, Act, \mathcal{AP}, L, w)$
 and $\hat{\xi} \leftarrow \emptyset$;
`CreateTransition`$(G, \mathcal{D}, \hat{G})$;
`Prune`$(\hat{G})$;
$G^* \leftarrow$Accessible$(\hat{G})$;
return G^*;

procedure `CreateTransition`$(G, \mathcal{D}, \hat{G})$
for $(x, u, v) \in S_{\mathcal{D}}$ **do**
  for $a \in Act$ **do**
    if $\delta_{\mathcal{D}}((x, u, v), a) = (x', u', v')$ *satisfies* (6) **then**
      $\hat{\Delta} \leftarrow \hat{\Delta} \cup \{(x, a, x')\}$;

procedure `Prune`$(\hat{G})$
while $\exists x \in \hat{X}$ s.t. $\nexists a \in Act : \hat{\xi}(x, a)!$ **do**
  Delete all such x in $\hat{G}$;

Theorem 1. The WTS $\mathcal{G}$ obtained from Algorithm 1 is the maximally diagnosable WTS, i.e., $\mathcal{G}$ satisfies conditions C1 and C2.

4.2 Optimal Self-Diagnostic Planning Algorithm

Now we tackle the diagnostic planning problem taking both the task satisfication requirement and the self-diagnostic requirement into account. Based on the result of Theorem 1, to solve Problem 1, it suffices for us to find a plan $\tau \in \mathsf{Plan}^\omega(\mathcal{G})$ such that $L(X(\tau)) \models \phi$. Naturally, we build the following product system.

Definition 5. (Product System). Given the maximally diagnosable WTS $\mathcal{G} = (X_{\mathcal{G}}, x_0, Act, \Delta_{\mathcal{G}}, \mathcal{AP}, L, w)$ and the NBA $\mathcal{A}_\phi = (Q_B, Q_{B,0}, \delta_B, F_B)$, the *product system* is defined as

$$P = \mathcal{G} \otimes \mathcal{A}_\phi = (Q_P, Q_{P,0}, Act, \Delta_P, F_P, w_P)$$

where

- $Q_P = X \times Q_B$ is the set of states;
- $Q_{P,0} = \{x_0\} \times Q_B$ is the set of initial states;
- Act is the set of actions available in environment G;
- $\Delta_P : Q_P \times Act \to Q_P$ is the transition function which is defined by: for any $p = (x, q), p' = (x', q') \in Q_P$ and any $a \in Act$, we have $p' = \Delta_P(p, a)$ if
  i) $x' = \Delta_{\mathcal{G}}(x, a)$; and
  ii) $q' \in \delta_B(q, L(x))$;
- $F_P = X \times F_B$ is the set of accepting states;
- $w_P : Act \to \mathbb{R}_{\geq 0}$ is the cost function defined by: for any $\forall a \in Act$, we have $w_P(a) = w(a)$.

Similar to WTS G, we denote by $\mathsf{Plan}^\omega(P)$ the set of all available plans in P. Based on the construction of the product system, to solve Problem 1, it suffices for us to find a plan $\tau \in \mathsf{Plan}^\omega(P)$ such that the corresponding path $Q_P(\tau)$ visits the accepting states F_P infinitely often.

Before presenting the final self-diagnostic planning algorithm, we introduce some necessary notations as follows.

Given the product system P and a set of states $Q \subseteq Q_P$, we denote by $\texttt{Reach}(Q)$ the set of states reachable from Q. Given P, we say a state $q \in Q_P$ is in a cycle of P if there exists a sequence $q_1 q_2 \cdots q_k \in Q_P^*$ such that $q_1 = q_k = q$ and $q_{i+1} \in \Delta(q_i, a_i)$ for some $a_i \in Act$ and all $i \geq 1$. We denote by $\texttt{Cycle}(P)$ the set of all states that are in some cycles of P. Furthermore, based on the accepting condition of NBA, we define the set of goal states as $\texttt{Goal}(P) = F_P \cap \texttt{Cycle}(P)$. Given two states $q, q' \in Q_P$, we denote by $\texttt{Plan}(q, q') = \{a_1 \cdots a_n \in Act^* : \exists q_0 q_1 \cdots q_n \in Q_P^* \text{ s.t. } q = q_0, q' = q_n, q_{i+1} \in \Delta_P(q_i, a_i)\}$ the set of all partial plans from q to q'. In particular, we denote by $\texttt{ShortestPlan}(q, q') = \arg\min_{\tau \in \texttt{Plan}(q,q')} J(\tau)$ the shortest partial plan from q to q', which, as a matter of fact, can be directly computed via a Dijkstra's algorithm (Bernhard and Vygen, 2008).

Algorithm 2: Optimal Self-Diagnostic LTL Planning

Input: Maxi. Diagnos. WTS $\mathcal{G}$, LTL task ϕ;
Output: Optimal plan $\tau \in \mathsf{Plan}^\omega(G)$;
Convert ϕ to NBA $\mathcal{A}_\phi$;
Construct Product System $P = \mathcal{G} \otimes \mathcal{A}_\phi$;
if $\mathit{Reach}(Q_{P,0}) \cap \mathit{Goal}(P) = \emptyset$ **then**
  return "no feasible plan";
else
  for $q_I \in Q_{P,0}$ **do**
    for $q_G \in \mathit{Reach}(Q_{P,0}) \cap \mathit{Goal}(P)$ **do**
      $\tau^{q_I, q_G} = \texttt{ShortestPlan}(q_I, q_G)$;
      $\tau^{q_G, q_G} = \texttt{ShortestPlan}(q_G, q_G)$;
  $(q_I^*, q_G^*) = \arg\min_{(q_I, q_G)} \hat{J}(\tau^{q_I, q_G}[\tau^{q_G, q_G}]^\omega)$;
  return optimal plan $\tau = \tau^{q_I^*, q_G^*}[\tau^{q_G^*, q_G^*}]^\omega$;

Now, the optimal self-diagnostic LTL planning algorithm is given by Algorithm 2, showing the technique of searching the optimal self-diagnostic plan, based on the maximally diagnosable WTS $\mathcal{G}$ obtained from Algorithm 1.

Theorem 2. For any environment described by a WTS G, the plan obtained from Algorithm 2 correctly solves the failure-aware self-diagnostic planning problem defined in Problem 2.

5. CASE STUDY

Let us go back to the motivating example in Section 2 to illustrate the proposed planning algorithm. The entire production processes can be described by a WTS $G = (X, x_0, Act, \Delta, \mathcal{AP}, L, w)$ in Figure 1(a), where $X = \{\mathrm{S}, \mathrm{M}, \mathrm{A}, \mathrm{P1}, \mathrm{P2}, \mathrm{T1}, \mathrm{T2}\}$, $Act = \{m, a_0, a_1, a_2, a_{\text{-}1}\}$, $\mathcal{AP} = \{\mathsf{stock}, \mathsf{product}\}$, and specifically, we have $L(S) = \mathsf{stock}$, $L(P1) = L(P2) = \mathsf{product}$. The transducer T and the observation map H are exactly as given in Figure 1(b) and Table 1 and for any $(s, x) \in S \times X$ that are not defined in Table 1, we set their observation by $H(s, x) = \lambda$. Therefore, a desired production line is exactly a cyclic path in G that satisfies the following LTL formula

$$\phi = \Diamond\Box\mathsf{stock} \wedge \Diamond\Box\mathsf{product}$$

Therefore, to select a failure-aware production line, it suffices to solve the self-diagnostic LTL planning problem defined in Problem 2. Without loss of generality, here we set the failure detection $K = 5$ due to the consideration of the space constraint.

To solve the self-diagnostic planning problem, we construct the diagnostic structure $\mathcal{D}$ and compute the maximally diagnosable WTS by applying Algorithm 1. Systems $\mathcal{D}$ and $\mathcal{G}$ are presented in Figures 2 and 3, respectively. Note that there is only one plan available in $\mathcal{G}$ as

$$\tau, X(\tau): \Big(\mathrm{S} \xrightarrow{m} \mathrm{M} \xrightarrow{a_0} \mathrm{A} \xrightarrow{a_2} \mathrm{P2} \xrightarrow{m} \mathrm{T2} \xrightarrow{m} \mathrm{T1} \xrightarrow{a_{\text{-}1}} \mathrm{S}\Big)^\omega$$

The NBA translated from LTL ϕ is given in Figure 4. One could easily observe that the trace of the plan $L(X(\tau))$ is accepted by NBA $\mathcal{A}_\phi$, and thus we omit the figure of the product system $P = \mathcal{G} \otimes \mathcal{A}_\phi$ here. In words, the effectiveness of our planning algorithm has been illustrated. Note that, the result here looks simple in this example because we choose $K = 5$ for the purpose of illustration. In practice, the value K can be larger and this will lead to more feasible choices in $\mathcal{G}$ for the purpose of optimization, e.g., the plan combining $L1$ and $L2'$ as we discussed in Section 2.

6. CONCLUSION

In this paper, we formulated and solved a new failure-aware task planning problem under LTL specifications. We provided a framework for modeling internal failures in addition to the external dynamic of the system in the environment. An effective algorithm was developed to synthesize an optimal plan that is both task satisfying and self-diagnostic. The proposed framework was illustrated by a case study of production lines. There are several potential future directions of our framework. First, in this work, the upper bound of the failure detection time is given as a designed parameter. In the future, we plan to further investigate the trade-off between the optimality of the plan and the failure detection time.

REFERENCES

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

Bernhard, K. and Vygen, J. (2008). Combinatorial optimization: Theory and algorithms. *Springer, Third Edition, 2005.*

Cai, M., Peng, H., Li, Z., and Kan, Z. (2020). Learning-based probabilistic LTL motion planning with environment and motion uncertainties. *IEEE Transactions on Automatic Control*, 66(5), 2386–2392.

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.

Dong, W., Yin, X., and Li, S. (2023). A uniform framework for diagnosis of discrete-event systems with unreliable sensors using linear temporal logic. *IEEE Transactions on Automatic Control*.

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.

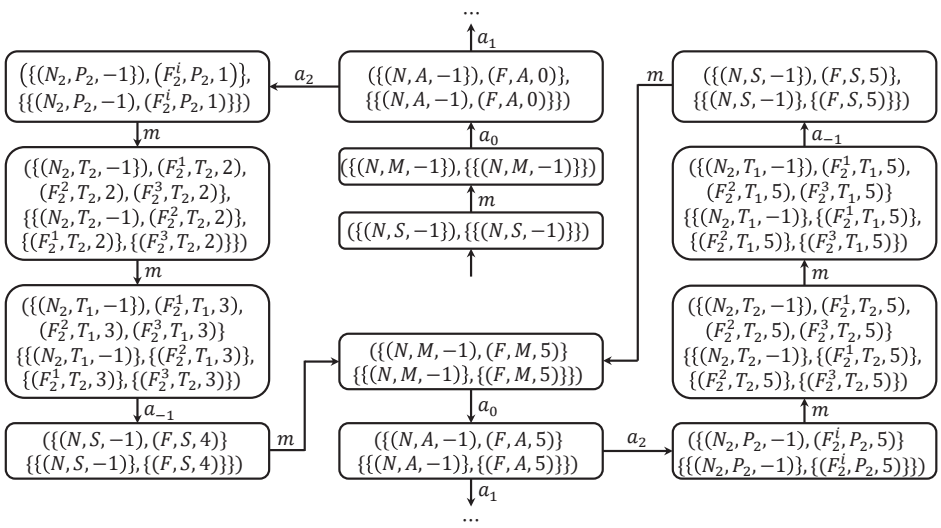


Fig. 2. Diagnostic Structure $\mathcal{D}$

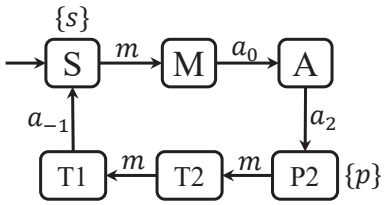


Fig. 3. Maximally Diagnosable WTS $\mathcal{G}$, where s, p denote stock and product respectively

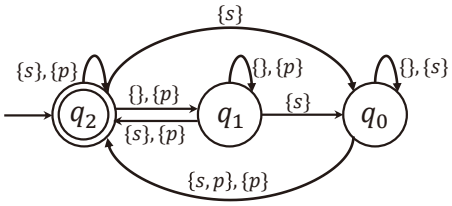


Fig. 4. NBA $\mathcal{A}_\phi$ translated from ϕ

Hu, Y., Ma, Z., and Li, Z. (2020). Design of supervisors for active diagnosis in discrete event systems. *IEEE Transactions on Automatic Control*, 65(12), 5159–5172.

Kloetzer, M. and Mahulea, C. (2020). Path planning for robotic teams based on ltl specifications and Petri net models. *Discrete Event Dynamic Systems*, 30(1), 55–79.

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.

Lin, F., Chen, W., Han, L., Shen, B., et al. (2017). N-diagnosability for active on-line diagnosis in discrete event systems. *Automatica*, 83, 220–225.

Luo, X., Kantaros, Y., and Zavlanos, M.M. (2021). An abstraction-free method for multirobot temporal logic optimal control synthesis. *IEEE Transactions on Robotics*, 37(5), 1487–1507.

Ma, Z., Yin, X., and Li, Z. (2021). Marking diagnosability verification in labeled petri nets. *Automatica*, 131, 109713.

Niu, L. and Clark, A. (2019). Optimal secure control with linear temporal logic constraints. *IEEE Transactions on Automatic Control*, 65(6), 2434–2449.

Ramasubramanian, B., Niu, L., Clark, A., Bushnell, L., and Poovendran, R. (2020). Secure control in partially observable environments to satisfy LTL specifications. *IEEE Transactions on Automatic Control*, 66(12), 5665–5679.

Sampath, M., Lafortune, S., and Teneketzis, D. (1998). Active diagnosis of discrete-event systems. *IEEE Transactions on Automatic Control*, 43(7), 908–929.

Shi, W., He, Z., Tang, W., Liu, W., and Ma, Z. (2022). Path planning of multi-robot systems with boolean specifications based on simulated annealing. *IEEE Robotics and Automation Letters*, 7(3), 6091–6098.

Smith, S.L., Tmová, 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.

Ulusoy, A., Smith, S.L., Ding, X.C., Belta, C., and Rus, D. (2013). Optimality and robustness in multi-robot path planning with temporal logic constraints. *The International Journal of Robotics Research*, 32(8), 889–911.

Vardi, M.Y. and Wolper, P. (1986). An automata-theoretic approach to automatic program verification. In *1st Symposium in Logic in Computer Science (LICS)*. IEEE Computer Society.

Xie, Y., Yin, X., Li, S., and Zamani, M. (2021). Secure-by-construction controller synthesis for stochastic systems under linear temporal logic specifications. In *2021 60th IEEE Conference on Decision and Control (CDC)*, 7015–7021. IEEE.

Yin, X. and Lafortune, S. (2016). A uniform approach for synthesizing property-enforcing supervisors for partially-observed discrete-event systems. *IEEE Transactions on Automatic Control*, 61(8), 2140–2154.

Yin, X. and Lafortune, S. (2017). On the decidability and complexity of diagnosability for labeled Petri nets. *IEEE Transactions on Automatic Control*, 62(11), 5931–5938.

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.