Data-Driven Real-Valued Timed-Failure-Propagation-Graph Refinement for Complex System Fault Diagnosis

Gang Chen, Xinfan Lin, *Member, IEEE*, and Zhaodan Kong, *Member, IEEE*

***Abstract*—Timed Failure Propagation Graphs (TFPGs) have been widely used for the failure modeling and diagnosis of safety-critical systems. Currently most TFPGs are manually constructed by system experts, a process that can be time-consuming, error-prone, and even impossible for systems with highly nonlinear and machine-learning-based components. This letter proposes a new type of TFPGs, called Real-Valued Timed Failure Propagation Graphs (rTFPGs), designed for continuous-state systems. More importantly, it presents a systematic way of constructing rTFPGs by combining the powers of human experts and data-driven methods: first, an expert constructs a partial rTFPG based on his/her expertise; then a data-driven algorithm refines the rTFPG by adding nodes and edges based on a given set of labeled signals. The proposed approach has been successfully implemented and evaluated on three case studies.**

***Index Terms*—Failure diagnosis, signal temporal logic, spacecraft power system, timed failure propagation graphs.**

I. Introduction

TIMELY and correct detection of faults are essential for the operation of safety-critical systems. Tradition fault detection methods, such as machine learning based ones, which learn a set of fault-relevant features, and model-based ones, which construct a residual signal and then determine a residual evaluation function (based on the signal) to compare the residual with a predefined threshold [1], cannot provide the causal and temporal aspects of failure events in a wide variety of engineering systems. Since early 1990s, Timed Failure Propagation Graphs (TFPGs) have been widely used for fault diagnosis in practice, e.g., by NASA [2]. TFPGs' popularity is partly due to their directed graph formalism and their ability to describe the occurrence of failures, their direct and indirect effects, and the corresponding consequences over time [2].

a) Related Work: TFPGs are primarily constructed manually by system experts. The construction process can be time-consuming, error-prone, and even impossible for systems with highly nonlinear and machine-learning-based components. Therefore, in recent years, the automatic synthesis of TFPGs has become an increasingly active research area [3], [4], [5]. These existing methods convert the TFPG synthesis problem into either i) a timed automaton learning problem and then rely on state-of-the-art automata learning algorithms to construct the TFPG [3], or ii) a set of proof obligations and then use state-of-the-art model checkers to verify (refine if necessary) the TFPG [4], [5]. There are two main issues with existing approaches. First, they all assume that all the nodes of the to-be-synthesized TFPG are given, which is a strong assumption for many complex systems, since experts may not be able to delineate all system failures and discrepancies. Second, they can only deal with discrete-state systems, while many, if not most, realistic systems are of continuous states.

b) Contributions: This letter makes two main contributions. First, it proposes a new formalism of TFPGs, called Real-Valued Timed Failure Propagation Graphs (rTFPGs), that are capable of abstracting and diagnosing faults of continuous-state systems. Second, it proposes a data-driven method that can construct an rTFPG based on a set of signals labeled by their failure modes.

II. Real-Valued Timed Failure Propagation Graphs

A. Definitions of Signals and rTFPGs

Definition 1 (Signal): Given a discrete time domain $\mathbb{N}$, a continuous-state *signal* is a mapping $x : \mathbb{N} \rightarrow \mathbb{R}^n$. We use $x[t]$ to denote the value of signal x at time t and $x_i, i = 1, \dots, n$ to denote the i-th dimension of signal x.

We first introduce *Real-Valued Timed Failure Propagation Graphs (rTFPGs)*, based on the definition of TFPGs [6].

Definition 2 (rTFPG): An *rTFPG* is a tuple $G = \langle F, D, E, ET, DC, DP\rangle$, where (i) F is a set of failure mode nodes; (ii) D is a set of discrepancy nodes; (iii) $E \subseteq V \times V$ is

Manuscript received April 8, 2020; revised June 18, 2020; accepted July 10, 2020. Date of publication July 16, 2020; date of current version July 30, 2020. This work was supported by the Space Technology Research Institutes from NASA's Space Technology Research Grants Program under Grant 80NSSC19K1052. Recommended by Senior Editor F. Dabbene. *(Corresponding author: Zhaodan Kong.)*

The authors are with the Department of Mechanical and Aerospace Engineering, University of California at Davis, Davis, CA 95616 USA (e-mail: ggchen@ucdavis.edu; lxflin@ucdavis.edu; zdkong@ucdavis.edu).

Digital Object Identifier 10.1109/LCSYS.2020.3009932

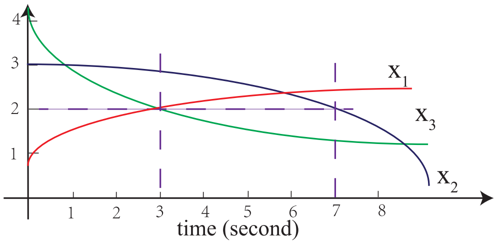

Fig. 1. An example signal x.

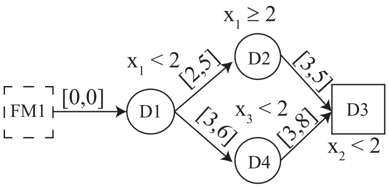

Fig. 2. An example rTFPG G. Dotted and solid boxes are failure mode node and AND node, respectively. Circles are OR nodes. $F(G) = \{FM1\}$, $D(G) = \{D1, D2, D3, D4\}$, $AND(G) = \{D3\}$, and $OR(G) = \{D1, D2, D4\}$.

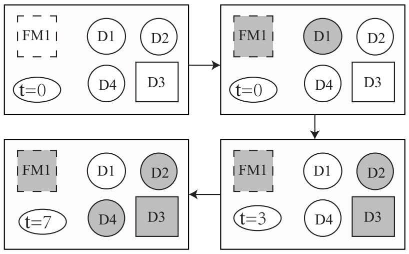

Fig. 3. The rTFPG trace π corresponding to the signal shown in Fig. 1 and the rTFPG shown in Fig. 2. The grey nodes are the ones that are active and assigned to Boolean value 1.

a set of edges with $V = F \cup D$; (iv) $ET : E \to I$ maps an edge $e \in E$ to a time interval $[t_{min}(e), t_{max}(e)] \in I$ with $t_{min}(e)$ and $t_{max}(e)$ being the minimum and maximum propagation times on the edge e; (v) $DC : D \to \{AND, OR\}$ maps a discrepancy node $d \in D$ to its discrepancy type; and (vi) DP maps a discrepancy node $d \in D$ to a predicate $\mu := f(x) \sim c \in \Phi$ over a signal x, where $f : \mathbb{R}^n \to \mathbb{R}$ is a function, $\sim \in \{<, \geq\}$, and $c \in \mathbb{R}$ a constant. We use $OR(G)$, $AND(G)$, $D(G)$, and $F(G)$ to denote the sets of OR nodes, AND nodes, discrepancy nodes, and failure mode nodes of an rTFPG G, respectively. One important feature of an rTFPG is that its edges are directed from failure mode(s) to discrepancy node(s).

One major difference between the proposed formalism of rTFPG and the existing formalism of TFPG is that the semantics of the former is defined over real-valued signals while that of the latter is defined over discrete events. On the other hand, rTFPG inherits all the desirable properties of TFPG. For instance, an rTFPG is a causal model that captures the causal and temporal aspects of failure events in a wide variety of engineering systems. Moreover, with the help of an rTFPG, a human user can comprehend the cause of failure events and as well as their temporal properties.

Example 1: Fig. 2 shows one such rTFPG. The main difference between an rTFPG and a typical TFPG is that the discrepancy nodes of an rTFPG are defined by predicates. For example, the predicate $x_1 \geq 2$ attached to the node $D2$ means that $D2$ can be activated only if $x_1 \geq 2$. The graph shows that when failure $FM1$ happens, event $D1$ will happen immediately, then event $D2$ will happen within next 2 to 5 seconds and event $D4$ will happen within the next 3 to 6 seconds. Three to five seconds after event $D2$ happens and 3 to 8 seconds after event $D4$ happens, event $D3$ will happen.

B. Semantics (Satisfaction) of rTFPG

Definition 3 (Mapping Δ_G): Given a signal x and an rTFPG G, a mapping Δ_G can be introduced to map the signal x to a discrete-state trace π by mapping $x[t]$ to $\pi[t]$ as follows: $\pi[t] := \Delta_G(x[t]) = (u_1, \ldots, u_{|F(G)|}, v_1, \ldots, v_{|D(G)|}, t)^T$, where u_i is 1 if the i^{th} failure mode node in $F(G)$ is active and 0 otherwise, v_i is 1 if the i^{th} discrepancy node in $D(G)$ is active and 0 otherwise. We call trace π an *rTFPG trace*, which represents failure propagation as a timed sequence of failure mode and discrepancy occurrences.

Example 2 (Continued): Fig. 3 illustrates the rTFPG trace $\pi := \pi[0], \ldots, \pi[8]$ corresponding to the signal x shown in Fig. 1 and the rTFPG G shown in Fig. 2. The initial state, a default $\mathbf{0}$ vector, is reset at time 0 to $\pi[0] = [1, 1, 0, 0, 0, 0]^T$ with the first 1 indicating the only fault mode node $FM1$ being active, the second 1 indicating the discrepancy node $D1$ being active (i.e., its predicate $x_1 < 2$ being satisfied), the next three zeros indicating the other three discrepancy nodes being inactive (i.e., their predicates being violated), and the last 0 being the time. The next two state $\pi[1]$ and $\pi[2]$ share the same first five elements with $\pi[0]$ but with different time t. At $t = 3$, a number of events happen: (i) $D1$'s predicate $x_1 < 2$ is not satisfied anymore, making $D1$ inactive; (ii) the predicates corresponding to $D2$ and $D3$ are now satisfied, making $D2$ and $D3$ active; therefore we have $\pi[3] = [1, 0, 1, 1, 0, 3]^T$. The remaining portion of π can be obtained similarly and easily.

With Def. 3, given an rTFPG G, we are able to map a continuous-state signal x to a discrete-state rTFPG trace π. It is important to point out that the discrete state nature of such traces allows us to use existing work on TFPGs [4] to define the conditions for a signal x to satisfy an rTFPG G. In the following, let π be the corresponding rTFPG trace of the signal x after applying the map Δ_G to x; moreover, we say $x[t] \models d \in D(G)$ if $\pi[t] \models d$, which is well defined [4], since π is a discrete-state trace and d is a node of an rTFPG.

Definition 4 (rTFPG Satisfaction): Given an rTFPG G and a set of signals S, G is *satisfiable* with respect to S if for each node d of G, there exists a signal $x \in S$, s.t. $\exists j \in \mathbb{N}, x[j] \models d$.

Example 3 (Continued): $D1$ is activated at $t = 0$ s. The satisfaction conditions for $D2$ require $D1$ keeps being active for at least 2 secs and then $D2$ is activated within 3 secs $(= 5-2 = t_{max}(e) - t_{min}(e)$ with $e = (D1, D2))$. The signal x in Fig. 1 satisfies that $D1$ is active for the first 3 secs and then $D2$ is activated (its predicate $x_1 \geq 2$ holds) at $t = 3$ s. Therefore, $D2$ is satisfied. $D4$ is satisfied similarly. The satisfaction conditions for $D3$ require that $D2$ and $D4$ keep being active for at least 3 secs and then $D3$ is activated within either 2 $(= 5 - 3$

for $e = (D2, D3)$) or 5 ($= 8 - 3$ for $e = (D4, D3)$) secs. The signal x satisfies that $D3$ is activated at $t = 7$ s. Therefore, $D3$ is satisfied and subsequently the rTFPG in Fig. 2 is satisfiable by the signal x.

In principle then, with the help of Δ_G, we are able to check the satisfaction of a continuous-state signal x with respect to an rTFPG G using existing verification tools suitable for TFPGs, e.g., Satisfiability Modulo Theories (SMT) solvers. In this letter, we will take advantage of a formalism called *Signal Temporal Logic (STL)*.

Definition 5 (STL): STL is a predicate logic defined over signals with its syntax defined as [7]: $\varphi := \mu|\neg\varphi|\varphi_1 \wedge \varphi_2|\varphi_1 \vee \varphi_2|\varphi_1\mathcal{U}_{[a,b]}\varphi_2$, where $a, b \in \mathbb{R}$; $\mu \in \Phi$ is a predicate defined the same as in Def. 2; Boolean operators $\neg$, $\wedge$, and $\vee$ are negation ("not"), conjunction ("and"), and disjunction ("or"), respectively; and temporal operator $\mathcal{U}$ stands for "until". The STL is equipped with a quantitative semantics called *robustness degree* ρ, which maps an STL formula φ and a signal x to a real value $\rho(\varphi, x)$ [7]. $\rho(\varphi, x) \geq 0$ if x satisfies φ and $\rho(\varphi, x) < 0$ if x violates φ. Therefore, the calculation of robustness degree provides us a way to check whether a signal x satisfies an STL formula φ.

Definition 6 (Activation Graph (AG)): Given a signal x and an rTFPG G, the *AG* σ of x is the subgraph of G that has been activated by the signal x. In this letter, we assume that a signal x corresponds to only one failure mode p. We call p the *label* of the signal x.

Lemma 1: Any node $d \in D(G)$ of an AG σ can be mapped to an STL formula φ_d, s.t. if a signal x activates the node d, we have $x \models \varphi_d$.

Proof: Let μ_d be the predicate of node d. Assume d has n_d direct predecessors. The STL formula φ_d can be constructed recursively as follows: if d is an *AND* node, φ_d can be written as $\wedge_{i=1}^{n_d}(\varphi_i\mathcal{U}_{[t_{min}(i),t_{max}(i)]}\mu_d)$; if d is an *OR* node, φ_d can be written as $\vee_{i=1}^{n_d}(\varphi_i\mathcal{U}_{[t_{min}(i),t_{max}(i)]}\mu_d)$; in both cases, φ_i is the STL formula of the i^{th} predecessor, $[t_{min}(i), t_{max}(i)]$ is the temporal interval of the edge directed from the i^{th} predecessor to d; φ_i can be constructed similarly as φ_d. The construction process will terminate when we have reached a fault mode node. ■

Example 4 (Continued): The AG σ of the signal x shown in Fig. 1 is the entire rTFPG G shown in Fig. 2. The node $D3$ can be mapped to an STL formula $\varphi_{D3} := (\varphi_1\mathcal{U}_{[3,5]}(x_2 < 2)) \wedge (\varphi_2\mathcal{U}_{[3,8]}(x_2 < 2))$, where $\varphi_1 = (x_1 < 2)\mathcal{U}_{[2,5]}(x_1 > 2)$ and $\varphi_2 = (x_1 < 2))\mathcal{U}_{[3,6]}(x_3 < 2)$. Any signal x activating the node $D3$ must satisfy φ_{D3}.

III. Problem Formulation

Definition 7 (rTFPG Diagnosability): Given an rTFPG G and a set of labeled signals S (with each signal x labeled by its failure mode $p_x \in F(G)$), G is *diagnosable with respect to* S if the following two conditions hold: (i) for any two signals $x', x'' \in S$ that have different labels (i.e., $p_{x'} \neq p_{x''}$), $\exists d \in \sigma_{x'}$ s.t. $x' \models \varphi_d$ and $x'' \models \neg\varphi_d$, where $\sigma_{x'}$ is the AG of the signal x'; (ii) for any two signals $x', x'' \in S$ that have the same label (i.e., $p_{x'} = p_{x''}$), $\exists d \in \sigma_{x'}$ s.t. $x' \models \varphi_d$ and $x'' \models \varphi_d$, where $\sigma_{x'}$ is the AG of the signal x'.

Algorithm 1 rTFPG Refinement Algorithm

Input: An initially satisfiable rTFPG G and a set of labeled signals $S = \cup_{p\in F(G)}S_p$, where S_p is the set of all signals labeled by the same failure mode $p \in F(G)$

Output: A G that solves Prob. 1

```
1: for each and every $p \in F(G)$ do
2:     Set $S^+ := S_p$ and $S^- := S/S^+$
3:     Refine($p, S := S^+ \cup S^-, G$)
4: Return $G$
```

The problem we are solving in this letter can be informally stated as *finding an rTFPG G that captures the failure propagation demonstrated by a set of continuous-state signals S*. Based on the definition of rTFPG diagnosability, such a problem can be formally defined as follows.

Problem 1 (rTFPG Refinement): Given an initially satisfiable rTFPG G and a set of labeled signals S (with each signal x labeled by its failure mode $p_x \in F(G)$), find another satisfiable rTFPG G' satisfying the following properties: (i) G' is diagnosable with respect to S, (ii) $F(G') = F(G)$, and (iii) $D(G) \subseteq D(G')$.

Remark 1: G, in our case, can be constructed by a system expert. One underlying assumption we are making here is the expert knows all the failure modes $F(G)$. However, the expert knows neither all the discrepancies, i.e., those nodes in $D(G')/D(G)$, nor all the edges, including those connecting (i) both nodes in $D(G')/D(G)$, (ii) one node in $D(G')/D(G)$ and the other in $D(G)$, and (iii) both nodes in $D(G)$. These edges characterize how failures propagate temporally and can be hard for experts to conceive a priori. A TFPG is equivalent to a timed automaton [3]. Since we can map an rTFPG into a TFPG by applying Δ_G, Problem 1 can be considered as an automaton learning problem and the solution proposed in this letter provides a new way to learn automata from data.

IV. Solution

A. Data-Driven rTFPG Refinement Algorithm

Alg. 1 shows the pseudo-code of our proposed algorithm to solve Prob. 1. For every failure mode $p \in F(G)$, Line 2 assigns all signals with label p as the positive example set S^+ and all the other signals as negative example set S^-; Line 3 tries to find an rTFPG G that is diagnosible with respect to $S := S^+ \cup S^-$. Specifically, the algorithm first tries to find a node $d \in D(G)$ of the current G, which is set initially to the G provided by a system expert, as well as an STL formula φ_d such that all signals in S^+ satisfy φ_d while those in S^- violate φ_d. If this can be achieved for the current G, the algorithm will move to the next failure mode; otherwise, it will refine G by using Alg. 2. The algorithm will terminate once it has checked all failure modes in $F(G)$ and subsequently found a G that can successfully diagnose all failures (thereby solving Prob. 1).

Alg. 2 shows the pseudo-code of the recursive function *Refine*(p, S, G) used in Alg. 1. Alg. 2 is provided with a set of candidate discrepancy nodes H and a set of candidate time intervals I. Each $d \in H$ is defined by its predicate

Algorithm 2 *Refine*(p, S, G)

Input: An rTFPG G with its set of edges as E, a node $p \in D(G) \cup F(G)$, a set of labeled signals $S := S^+ \cup S^-$, a set of candidate discrepancy nodes H ($H \cap D(G) = \emptyset$), and a set of candidate time intervals I

Output: A refined rTFPG G

```
1:  while $DE(p, S, G) > 0$ do
2:      for $d' \in D(G) \wedge (p, d') \in E$ do
3:          if $DE(d', S, G) \leq DE(p, S, G)$ then
4:              $Refine(d', S, G)$, break
5:      for $(d' \in D(G) \setminus mcs(p, D(G), S)) \wedge (d', p) \notin E$ do
6:          Construct $G'$ s.t. $E' := E \cup (d', p)$
7:          if $DE(p, S, G') \leq DE(p, s, G)$ then
8:              $G := G'$, $Refine(p, S, G)$
9:      for $d' \in H$ do
10:         Construct $G'$ s.t. $D(G') := D(G) \cup d'$ and $E' := E \cup e'$, where $e' := (p, d') \wedge [t_{min}(e'), t_{max}(e')] \in I$
11:         if $DE(d', S, G') < DE(p, S, G)$ then
12:             $G := G'$, $Refine(d', S, G)$, break
13:         else
14:             Construct $G''$ s.t. $D(G'') := D(G')$ and $E'' := E' \cup (v, d')$, where $v$ is a predecessor of $p$
15:             if $DE(p, S, G'') < DE(p, S, G)$ then
16:                 $G := G''$, $Refine(p, S, G)$, break
17: Return $G$
```

Algorithm 3 $DE(p, S, G)$

Input: An rTFPG G, a node $p \in D(G) \cup F(G)$, and a set of labeled signals $S := S^+ \cup S^-$

Output: The diagnosis error of node p

```
1:  $\Phi := \emptyset$
2:  if $p \in F(G)$ then
3:      $DE := 1$
4:  else
5:      while $\exists cs1, cs2 \in acs(p, D(G), S^+) \wedge (\exists d \in cs1 \cap cs2 \cap OR(G))$ do
6:          $acs(p, D(G), S^+) := (acs(p, D(G), S^+) / cs1 / cs2) \cup \{cs1 \cup cs2\}$
7:      for $cs \in acs(p, D(G), S^+)$ do
8:          Construct $G_{cs}$ s.t. $D(G_{cs}) = cs \cup \{p\}$
9:          $\Phi := \Phi \cup \{\varphi_{cs}\}$ where $\varphi_{cs}$ is the STL formula corresponding to $G_{cs}$ and $p$
10:     $DE := \min_{\varphi \in \Phi} MR(\varphi, S)$, where
```

$$MR(\varphi, S) = \begin{cases} 1, & \text{if } \exists x \in S^+, x \not\models \varphi \\ \frac{|\{x | x \in S^- \wedge x \models \varphi\}|}{|S|}, & \text{otherwise} \end{cases} \tag{1}$$

```
11: Return $DE$
```

$\mu := f(x) \sim c$, which is parameterized by $\sim \in \{\geq, <\}$ and c, and its discrepancy type, i.e., *AND* or *OR*. Each $i \in I$ is defined by an interval $[t_{min}(i), t_{max}(i)]$. In this letter, the sets of c, t_{min}, and t_{max} are discrete.

Alg. 2 uses the metric $DE(p, S, G)$, called the *diagnosis error (DE)*, to guide the refinement process, e.g., on whether to refine the current rTFPG and, if so, which discrepancy node to be added. $DE(p, S, G)$ is computed by Alg. 3. It is based on the concept of *cut-set* [8].

Definition 8 (Cut-Set): Given an rTFPG G, a node $d \in D(G)$, and a set of signals S, a set $cs \subseteq D(G) \setminus d$ is a *cut-set* of d iff there exists a signal $x \in S$, for which $\exists k \in \mathbb{N}$, s.t. $x[k] \models d$ and $\forall d' \in cs \Leftrightarrow \exists i \leq k, x[i] \models d'$. A cut-set cs of d is *minimal* iff no proper subset of cs is a cut-set. We use $acs(d, D(G), S)$ to denote all the cut-sets of d with respect to S and $mcs(d, D(G), S)$ to denote its minimal cut-set.

Lemma 2: Given an rTFPG G, a node $d \in D(G)$ and one of its cut-set $cs \in acs(d, D(G), S)$, if a signal x activates the node d, $\forall d' \in cs$, x activates the node d' as well.

Proof: According to Lemma 1, a signal x activates a node d indicates there exists an AG σ and a node $d \in \sigma$ s.t. $x \models \varphi_d$. Then, based on Def. 8, we have $\forall d' \in cs \Leftrightarrow \exists i \leq k, x[i] \models d'$. Therefore, $\forall d' \in cs$, there exists an AG σ' s.t. $\sigma' \subset \sigma$, $d' \in \sigma'$, and $x \models \varphi_{d'}$ (implying d' is activated by x as well, according to Lemma 1). ■

Example 5 (Continued): The signal x shown in Fig. 1 activates $D1$, $D2$, $D3$, and $D4$ of the rTFPG G shown in Fig. 2. Therefore, $acs(D4, D(G), S) = \{\{D1, D2, D3\}\}$ and $mcs(D4, D(G), S) = \{D1, D2, D3\}$, where $S = S^+ = x$. Note that *acs* and *mcs* are different: *acs* is the cut-set that includes all discrepancy nodes activated by the fault signals in S, while *mcs* includes discrepancy nodes activated by one signal. Therefore, *mcs* is a subset of *acs*. In this example, since there is only one signal, *acs* and *mcs* are the same.

Alg. 3 checks all the cut-sets of p and returns their best performance in the context of diagnosing S (quantified by Eqn. (1)). Line 1 checks whether p is a failure mode node. If it is, DE will be 1; otherwise, Line 5-10 will compute the DE for p. Line 5-6 find $acs(p, D(G), S^+)$, all the cut-sets of p, and merge those that share the same *OR* node. Line 8-9 construct an STL formula φ_{cs} for each and every cut-set cs of $acs(p, D(G), S^+)$. Finally, Line 10 finds the cut-set with the lowest $MR(\varphi, S)$, which is defined in Eqn. (1). $MR(\varphi, S)$ quantifies the mis-classification rate of φ in terms of $S := S^+ \cup S^-$ [9]. In our case, we use the robustness degree $\rho(\varphi, x)$ to check the satisfaction of a signal $x \in S$ with respect to an STL formula φ (see Definition 5).

Finally, let's elaborate on Alg. 2. Line 1 indicates that the algorithm will terminate if $DE(p, S, G)$ of the current rTFPG G and the current node p with respect to S is zero. Otherwise, i.e., if $DE(p, S, G)$ is positive, Line 2-16 try to decrease $DE(p, S, G)$ by exploring three options: (i) selecting a new but existing node p, (ii) adding a new edge to G, and (iii) adding a new node from the candidate discrepancy node set H and its corresponding edge(s) to G. Line 2-4 implement option (i) and try to find a direct successor d' of p that has a lower $DE(d', \Pi, G)$. If such a node can be found, the algorithm will start its refinement from d' (the algorithm is recursive in nature). Line 5-8 implement option (ii), add a new edge (d', p), where d' is inside $D(G)$ but outside $mcs(p, D(G), S)$, to the current G, resulting a new rTFPG G', and check whether G' decreases $DE(p, S, G')$. If so, the refinement will continue with G'. Line 9-16 implement option

(iii) and try to add new node(s) from H. There are two sub-options: Line 10-12 implement the first sub-option by simply adding a new node $d' \in H$ and a new edge (p, d') to the current G, resulting a new rTFPG G', and checking whether G' decreases $DE(p, S, G')$ (see Proposition 1); if so, the refinement will continue with G'; otherwise, Line 14-16 implement the second sub-option by adding another edge (v, d') to G', resulting a new rTFPG G'', where v is a predecessor of p, and checking whether G'' decreases $DE(p, S, G'')$; if so, the refinement will continue with G''.

B. Performance Guarantees

Line 10 of Alg. 2 attempts to decrease $DE(p, S, G)$ by adding a node $d' \in H$ together with an edge (p, d') to the current rTFPG G. Such a node d' is likely to exist according to the following proposition:

Proposition 1: Given an rTFPG G, a set of labeled signals $S := S^+ \cup S^-$, and a node $p \in D(G)$, let φ_p be the STL formula for p that minimizes Eqn. 1, T_p be the horizon of φ_p [10], and P^+, N^+, P^-, and N^- be the sets of correctly classified positive and negative signals and falsely classified positive and negative signals by φ_p, respectively, if either (i) $\exists k > T_p, \exists y \in N^-, min_{x \in P^+}(x_i[k] - y_i[k]) > 0$ or (ii) $\exists k > T_p, \exists y \in N^-, max_{x \in P^+}(x_i[k] - y_i[k]) < 0$, then there exists a node d' leading to a decreased $DE(p, S, G')$, where $D(G') := D(G) \cup d'$ and $E(G') := E(G) \cup (p, d')$.

Proof: It is quite straightforward that the satisfaction of all the signals in P^+ and N^+ with respect to φ_p will not be affected by the newly added node d' and edge (p, d') (i.e., the signals that are correctly classified by G will still be correctly classified by G'). If condition (i) holds, with properly chosen parameters c, t_{min}, t_{max} (e.g., if such parameters exist in H and I provided for Alg. 2), a node d' with a predicate of the form $x_i \geq c$ and an edge (p, d') with a time interval $[t_{min}, t_{max}]$ can decrease $DE(p, S, G')$ by decreasing the number of signals in N^- (in our case, $|P^-| = 0$). Similarly, if condition (ii) holds, with properly chosen parameters c, t_{min}, t_{max}, a node d' with a predicate of the form $x_i < c$ and an edge (p, d') with a time interval $[t_{min}, t_{max}]$ can decrease $DE(p, S, G')$ by decreasing the value of $|N^-|$. ■

Theorem 1: Given an initially satisfiable rTFPG G, the refined rTFPG G' obtained by using Alg. 1 is also satisfiable.

Proof: Line 8, 12, and 16 of Alg. 2 refine the current rTFPG G by adding either node(s) or edge(s). Each such refinement guarantee that (i) those nodes in $D(G)$ that are activated by signals in S^+ are still going to be activated and (ii) the newly added nodes, i.e., those in $D(G')/D(G)$, are also activated by signals in S^+. Therefore, according to Def. 4, the refined rTFPG G' obtained by using Alg. 1 is satisfiable. ■

Theorem 2: If Alg. 1 terminates, then the refined rTFPG G' obtained by using the algorithm is diagnosable with respect to the labeled signal set $S := S^+ \cup S^-$.

Proof: Let p be the last node that Alg. 1 visits before the algorithm terminates and φ_p be its corresponding STL formula. Then both conditions for the diagnosability of G' with respect to S are satisfied. (i) For two signals $x', x'' \in S$ that have different labels (i.e., $p_{x'} \neq p_{x''}$), WLOG, assume $x' \in S^+$ and $x'' \in S^-$. When Alg. 1 terminates, $DE(p, S^+ \cup S^-, G')$ is zero, implying $x' \models \varphi_p$ and $x'' \models \neg\varphi_p$. (ii) For two signals $x', x'' \in S$ that have the same label (i.e., $p_{x'} = p_{x''}$), WLOG, assume $x', x'' \in S^+$. When Alg. 1 terminates, $DE(p, S^+ \cup S^-, G')$ is zero, implying $x' \models \varphi_p$ and $x'' \models \varphi_p$. ■

With Theorems 1 and 2, we can conclude that if Alg. 1 terminates, it will return an rTFPG G' that solves Prob. 1, i.e., obtaining an rTFPG that is diagnosable with respect to S. Moreover, Alg. 1 can handle multiply failure modes in a step-by-step manner (as indicated by Line 1): for each failure mode p, the algorithm sets the signals labeled by p as positive examples and the others as negative examples.

c) Limitations: We cannot guarantee obtaining a globally optimal rTFPG (here *optimality* refers to minimizing the diagnosis error *DE*) but only a locally optimal one, given the fact that the global optimality depends on the initial graph, which is provided by the human expert. Proposition 1 shows that from this initial graph, new nodes can be added to decrease *DE*. If a node does not decrease diagnosis error *DE*, which will leads to a locally optimal graph eventually, the node will not be added. Moreover, another issue of our proposed method is its *scalability*, since, in the worst case, it the needs to check all the possible failure nodes. We will address both issues in the future, which is out of the scope of this letter.

V. Case Studies

In this section, we will validate the performance of our proposed method with three case studies, two simulated and one real. First, we apply the proposed method to an advanced diagnostics and prognostics testbed (ADAPT) developed by NASA [11]. The ADAPT emulates a spacecraft power storage and distribution system with three major components, a power generation component with two battery chargers and a solar panel, a power storage component with three sets of lead-acid batteries, and the a power distribution component with two inverters and a number of loads. Here we only consider abrupt faults, i.e., those unexpected abrupt changes in the system parameter values. Specifically, we introduce three types of abrupt faults into the system (i.e., the number of failure modes $|F(G)| = 3$): (i) adding an additive sensor bias to one of the variables of load bank 1 (*IT*), (ii) changing the capacity value of battery 1 (*L2E*), and (iii) changing the value of one of the resistances of load bank 1 (*TE*). The changes in the parameter values are bounded for all these failures. We generate 40 signals for each failure mode (20 for training and 20 for testing). Details on the three failure modes $F(G)$, the signals used in this case study S, the Python code that implements Alg. 1 to solve the rTFPG refinement problem for the case study, and the obtained rTFPG G' can be found at https://github.com/sjtugangchen/Error-Propagation-Graph.git.

We run the code on a 64bit Linux computer with a 16 core CPU at 3.8 GHz and 64GB of RAM. We set $|H| = 300$ (see Alg. 2) and set the time limit to $T_{max} = 1000$ secs, i.e., the algorithm must terminate within at most T_{max} (according to Theorem 2, if it terminates in $t < T_{max}$, a solution has been found). Fig. 4 shows the rTFPG obtained by Alg. 1 for the failure mode *L2E* (the rTFPGs for the other two failure modes

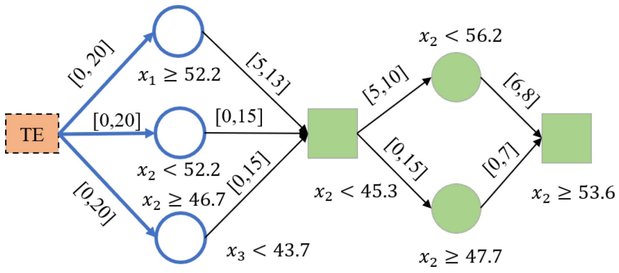

Fig. 4. The rTFPG constructed for the failure mode *TE*. The initially provided discrepancy nodes and edges are shown in blue. All the other discrepancy nodes and edges are synthesized automatically by Alg. 1.

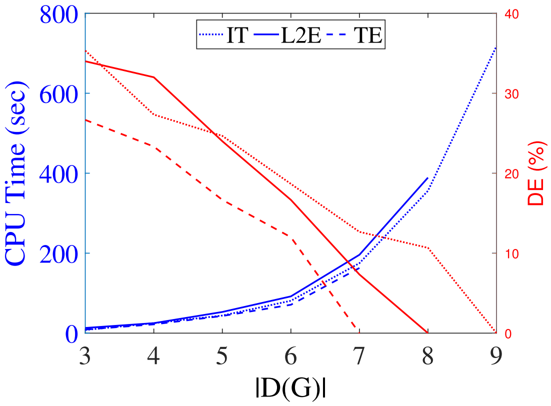

Fig. 5. CPU times and DEs for the three failure modes.

TABLE I

Comparison Results With Methods in [9], [12] (Performance Metrics Are Shown as *DE* (%)/ Time (s)/$\rho(S, \varphi)$)

Case	Proposed method	Method in [9]	Method in [12]
Iron-making	0/225/0.05	0/1024/0.13	0/256/0.12
Naval Surveillance	0/222/0.24	0/1013/0.09	0/227/0.32
ADAPT (TE)	0/205/0.19	0/985/0.12	0/245/0.23

are omitted). Fig. 5 shows the CPU times and diagnosis errors (DEs) (see Alg. 3) of the three failure modes, before Alg. 1 terminates (here we run the algorithm for each failure mode separately). It can be observed that the CPU time is roughly exponential with respect to the size of $D(G)$, which is mainly due to the for-loops inside Alg. 2. The figure also shows that all the failure modes can be diagnosed correctly, since their DEs are all zeros upon termination.

Second, we demonstrate the generalizability of our proposed method by applying it to two other existing cases, one is the naval surveillance case in [9] and the other is the cooling fault diagnosis case of an iron-making factory in [12]. Here we only use one fault for each case study, meaning that each case study solves a binary classification problem. The comparison results are shown in Table I. In the table, we use three metrics to quantify the performance of the three methods: (i) the robustness degree, defined as $\rho(S, \varphi) = \min(\min_{x \in S^+}(\rho(x, \varphi)), \min_{x \in S^-}(\rho(x, \neg\varphi)))$, where $\rho(S, \varphi)$ is the minimum robustness degree of all the signals in S, (ii) the diagnosis error *DE*, and (iii) the CPU time to reach a solution with *DE* being zero. The results in Table I show that all three methods are able to obtain a correct classifier (i.e., with *DE* being zero). However, the method proposed in this letter outperforms the other two in the term of the CPU time while obtaining a comparable robustness degree.

VI. Conclusion

This letter introduces a new formalism of TFPGs, called rTFPGs, that is suitable to model and diagnose failure propagation pertaining to continuous-state systems. Moreover, it presents a data-driven method to automatically construct such rTFPGs given a set of labeled signals. Finally, the performance of our proposed method is validated with three case studies. Given the popularity of TFPGs in safety critical systems and the proved performance of our method, we believe our paper provides a necessary foundation for many future data-driven TFPG synthesis frameworks.

References

[1] W. Xiang, J. Xiao, and M. N. Iqbal, "Robust fault detection for a class of uncertain switched nonlinear systems via the state updating approach," *Nonlinear Anal. Hybrid Syst.*, vol. 12, pp. 132–146, May 2014.

[2] S. Hayden *et al.*, "Diagnostic technology evaluation report for on-board crew launch vehicle," NASA, Washington, DC, USA, Rep. TM-2006-214552, 2006.

[3] C. Priesterjahn, C. Heinzemann, and W. Schäfer, "From timed automata to timed failure propagation graphs," in *Proc. 16th IEEE Int. Symp. Object Component Service Orient. Real Time Distrib. Comput. (ISORC)*, 2013, pp. 1–8.

[4] B. Bittner, M. Bozzano, and A. Cimatti, "Automated synthesis of timed failure propagation graphs," in *Proc. IJCAI*, 2016, pp. 972–978.

[5] M. Bozzano, A. Cimatti, M. Gario, and A. Micheli, "SMT-based validation of timed failure propagation graphs," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 3724–3730.

[6] S. Abdelwahed, G. Karsai, N. Mahadevan, and S. C. Ofsthun, "Practical implementation of diagnosis systems using timed failure propagation graph models," *IEEE Trans. Instrum. Meas.*, vol. 58, no. 2, pp. 240–247, Feb. 2009.

[7] A. Donzé and O. Maler, "Robust satisfaction of temporal logic over real-valued signals," in *Proc. Int. Conf. Formal Model. Anal. Timed Syst.*, 2010, pp. 92–106.

[8] W. E. Vesely, F. F. Goldberg, N. H. Roberts, and D. F. Haasl, "Fault tree handbook," Nucl. Regul. Commission, Washington, DC, USA, Rep. NUREG/CR-0400, 1981.

[9] Z. Kong, A. Jones, and C. Belta, "Temporal logics for learning and detection of anomalous behavior," *IEEE Trans. Autom. Control*, vol. 62, no. 3, pp. 1210–1222, Mar. 2017.

[10] J. V. Deshmukh, A. Donzé, S. Ghosh, X. Jin, G. Juniwal, and S. A. Seshia, "Robust online monitoring of signal temporal logic," *Formal Methods Syst. Design*, vol. 51, no. 1, pp. 5–30, 2017.

[11] M. J. Daigle, I. Roychoudhury, G. Biswas, X. D. Koutsoukos, A. Patterson-Hine, and S. Poll, "A comprehensive diagnosis methodology for complex hybrid systems: A case study on spacecraft power distribution systems," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 40, no. 5, pp. 917–931, Sep. 2010.

[12] G. Chen, M. Liu, and Z. Kong, "Temporal-logic-based semantic fault diagnosis with time-series data from industrial Internet of Things," *IEEE Trans. Ind. Electron.*, early access, Apr. 7, 2020, doi: 10.1109/TIE.2020.2984976.