Article

Robust Satisfaction of Metric Interval Temporal Logic Objectives in Adversarial Environments

Luyao Niu [1]*, Bhaskar Ramasubramanian [2], Andrew Clark [3], and Radha Poovendran[1]

[1] Network Security Lab, Department of Electrical and Computer Engineering, University of Washington; luyaoniu @uw.edu, rp3@uw.edu
[2] Electrical and Computer Engineering, Western Washington University; ramasub@wwu.edu
[3] Department of Electrical and Systems Engineering, Washington University in St. Louis; andrewclark@wustl.edu
* Correspondence: luyaoniu@uw.edu (L.N.)

Abstract: This paper studies the synthesis of controllers for cyber-physical systems (CPSs) that are required to carry out complex time-sensitive tasks, in the presence of an adversary. The time-sensitive task is specified as a formula in metric interval temporal logic (MITL). CPS that operate in adversarial environments have typically been abstracted as stochastic games (SGs). However, since traditional SG models do not incorporate a notion of time, they cannot be used in a setting where the objective is time-sensitive. To address this, we introduce durational stochastic games (DSGs). DSGs generalize SGs to incorporate a notion of time and model the adversary's abilities to tamper with the control input (actuator attack) and manipulate timing information perceived by the CPS (timing attack). We define notions of spatial, temporal, and spatio-temporal robustness to quantify the amounts by which system trajectories under the synthesized policy can be perturbed in space and in time without affecting satisfaction of the MITL objective. In the case of an actuator attack, we design computational procedures to synthesize controllers that will satisfy the MITL task along with a guarantee on its robustness. In the presence of a timing attack, we relax the robustness constraint to develop a value iteration-based procedure to compute the CPS policy as a finite state controller to maximize the probability of satisfying the MITL task. A numerical evaluation of our approach on a signalized traffic network is presented to illustrate our results.

Keywords: MITL specification; control synthesis; adversary; robustness; Stackelberg game

1. Introduction

Cyber-physical systems (CPSs) are playing increasingly important roles in multiple applications, including autonomous vehicles, robotics, and advanced manufacturing [1]. In many of these applications, the CPS is expected to satisfy complex time-critical objectives in dynamic environments with autonomy. An example is a scenario where a drone has to periodically surveil a target region in its environment. One way to specify requirements on the CPS behavior is through a temporal logic framework [2] like metric interval temporal logic (MITL) or signal temporal logic (STL). Verification of satisfaction of the temporal logic objective can then be achieved by applying principles from model checking [2,3] to a finite transition system that abstracts the CPS [4–7]. Solution techniques to verify such an objective usually return a 'yes/no' output, indicating if the behavior of the CPS will satisfy the desired task and if it is possible to synthesize a control policy to satisfy this objective.

However, such binary-valued verification results may not be adequate when an adversary can inject inputs that affect the behavior of the CPS. Small perturbations can result in significantly large changes in the output of a CPS, and can lead to violations of the desired task. The authors of [8,9] defined a notion of *robustness degree* to quantify the extent to which a CPS could tolerate deviations from its nominal behavior without resulting in violation of the desired specification.

Citation: Niu, L.; Ramasubramanian, B.; Clark, A.; Poovendran, R. Robust Satisfaction of Metric Interval Temporal Logic Objectives in Adversarial Environments. *Games* **2023**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published:

For time-critical CPS, an adversary could launch attacks on clocks of the system (by timing attack) and the inputs to the system (by actuator attack). In the latter case, stochastic games (SGs) have been used to model the interaction between the CPS and adversary [10]. However, SGs do not include information about the time taken for a transition between two states. To bridge this gap, we introduce durational stochastic games (DSGs). In addition to transition probabilities between states under given actions of the CPS and adversary, a DSG encodes the time taken for the transition as a probability mass function. Although DSGs present a modeling formalism for time-critical objectives, they introduce an additional attack surface that can be exploited by an adversary.

In this paper, we synthesize controllers to satisfy an MITL specification which can be represented by a deterministic timed Büchi automaton with a desired robustness guarantee. The robustness guarantee quantifies how sensitive the synthesized policy (that satisfies the MITL task) will be to disturbances and adversarial inputs. The adversary is assumed to have the following abilities: it can tamper with the input to the defender through an actuator attack [11], and it can affect the time index observed by the CPS by effecting a timing attack [12]. An actuator attack could steer the DSG away from a target set of states, while a timing attack will prevent it from satisfying the objective within the specified time interval.

To address perturbations originating from different attack surfaces (timing information and system inputs), we develop three notions of robustness, namely spatial, temporal, and spatio-temporal robustness. The spatial robustness is defined over discrete timed words, and quantifies the maximum perturbation that can be tolerated by timed words so that the desired tasks can still be satisfied in the absence of timing attacks. The temporal robustness characterizes the maximum timing perturbation that can be tolerated by a CPS such that the given MITL objective will not be violated. We introduce a notion of spatio-temporal robustness that unifies the concepts of spatial and temporal robustness. Using these three notions of robustness, we develop algorithms to estimate them and compute controllers for CPSs to guarantee that the given MITL objective can be satisfied with desired robustness guarantee. This paper makes the following contributions.

- We introduce durational stochastic games (DSGs) to model the interaction between the CPS which has to satisfy a time-critical objective and an adversary who can initiate actuator and timing attacks.
- We define notions of spatial, temporal, and spatio-temporal robustness which quantify the robustness of system trajectories to spatial, temporal, and spatio-temporal perturbations, respectively, and present computational procedures to estimate them. We design an algorithm to compute a policy for the CPS (defender) with robustness guarantee when the adversary is limited to effecting only actuator attacks.
- We demonstrate that the defender cannot estimate the spatio-temporal robustness correctly when the adversary can initiate both actuator and timing attacks. We relax the robustness constraints in such cases and present a value iteration based procedure to compute the defender's policy, represented as a finite state controller, to maximize the probability of satisfying the MITL objective.
- We evaluate our approach on a signalized traffic network. We compare our approach with two baselines, and show that it outperforms both baselines.

The remainder of this paper is organized as follows. Section 2 discusses related work. Section 3 gives background on MITL and deterministic timed Büchi automata. We define the DSG and notions of robustness in Section 4 and formally state the problem of interest. Sections 5 and 6 respectively present our results when the adversary is limited to initiating only actuator attacks and when it can effect both actuator and timing attacks. Experimental results are presented in Section 7. Section 8 concludes the paper.

2. Related Work

For a single agent, semi-Markov decision processes (SMDPs) [13] can be used to model Markovian dynamics where time taken for transitions between states is a random variable.

SMDPs have been used in production scheduling [14] and optimization of queues [15]. Stochastic games (SGs) generalize MDPs when there is more than one agent taking an action [16]. SGs have been widely adopted to model strategic interactions between CPS and adversaries. For example, a zero-sum SG was formulated in [17] to allocate resources to protect power systems against malicious attacks. Two SGs were developed in [18] to detect intrusions to achieve secret and reliable communications. The satisfaction of complex objectives modeled by linear temporal logic (LTL) formulae for zero-sum two-player SGs was presented in [10], where the authors synthesized controllers to maximize the probability of satisfying the LTL formula. However, this approach will not apply when the system has to satisfy a time-critical specification and the adversary can launch a timing attack.

Timed automata (TA) [3] attach finitely many clock constraints to each state. A transition between any two states will be influenced by the satisfaction of clock constraints in the respective states. There has been significant work in the formulation of timed temporal logic frameworks, a detailed survey of which is presented in [19]. Metric interval temporal logic (MITL) [20] is one such fragment that allows for the specification of formulae that explicitly depend on time. Moreover, an MITL formula can be represented as a TA [20,21] that will have a feasible path in it if and only if the MITL formula is true.

Control synthesis under metric temporal logic constraints was studied for motion planning applications in [6,7,22,23]. The authors of [22] considered a vehicle routing problem to meet MTL specifications by solving a mixed integer linear program. Timed automaton-based control synthesis under a subclass of MITL specifications was studied in [6,7]. Cooperative task planning of multi-agent system under MITL specifications was studied in [24]. In comparison, we consider actions of an adversarial player, whose objective is opposite to that of the defender. This leads to a modeling of the interaction between the adversary and defender as an SG. Moreover, they limited their focus to a certain fragment of MITL, while this paper offers a generalized treatment to arbitrary MITL formulae.

Finite state controllers (FSCs) were used to simplify the policy iteration procedure for POMDPs in [25]. The satisfaction of an LTL formula of a POMDP was presented in [26]. This was extended to the case with an adversary, who also only had partial observation of the environment, and whose goal was to prevent the defender from satisfying the LTL formula in [27,28]. These treatments, however, did not account for the presence of timing constraints on the satisfaction of a temporal logic formula.

Control synthesis for control systems under disturbances with robustness guarantees has been extensively studied [29–32]. Such robustness guarantees can be categorized as a notion of spatial robustness. Robust satisfaction of temporal logic tasks have been studied for signal monitoring and property verification. A notion of robustness degree for continuous signals was defined in [8] by computing a distance between the given timed behavior and the set of behaviors that satisfy a property expressed in temporal logic. Our notion of spatial robustness is defined over discrete timed words using the Levenshtein distance, which distinguishes our approach from [8]. The robustness degree between two LTL formulae was introduced in [33]. The authors of [34] adopted a different approach and used the weighted edit distance to quantify a measure of robustness. The notion of temporal robustness was also investigated in [9]. There are three differences between our definition of temporal robustness and that in [9]. First, the temporal robustness in [9] is defined for a specific trace. In our framework, since the DSG is not deterministic, there could be multiple traces that satisfy the MITL objective under the defender and adversary policies. Therefore, we define temporal robustness with respect to policies of the defender and adversary and the MITL specification. Second, the temporal robustness of a real-valued signal is computed as the maximum amount of time units by which we can shift on the rising/falling edge of a 'characteristic function' in [9]. In comparison, we work with discrete timed words. Finally, our work considers the presence of an adversary while [9] assumes a single agent. Robust control under signal temporal logic (STL) formulae has been studied based on notions of space robustness [35,36] and temporal robustness [37,38]. These works did not consider the presence of an adversary.

A preliminary version of this paper [39] synthesized policies to satisfy MITL objectives under actuator and timing attacks without robustness guarantees. In this paper, we define three robustness degrees, and develop algorithms to compute these quantities. We show that any defender policy that gives positive robustness degree is an almost-sure satisfaction policy, which is stronger than quantitative satisfaction policies synthesized in [39].

3. MITL and Timed Automata

We introduce the syntax and semantics of metric interval temporal logic, and its equivalent representation as a timed automaton. We use $\mathbb{R}, \mathbb{R}_{\geq 0}, \mathbb{N}, \mathbb{Q}_{\geq 0}$ to denote the sets of real numbers, non-negative reals, positive integers, and non-negative rationals. Vectors are represented by bold symbols. The comparison between vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ is element-wise, and $\mathbf{v}(i)$ denotes the i-th element of $\mathbf{v}$.

Given a set of atomic propositions Π, a metric interval temporal logic (MITL) formula is defined inductively as

$$\varphi := \top|\pi|\neg\varphi|\varphi_1 \wedge \varphi_2|\varphi_1 \mathcal{U}_I \varphi_2,$$

where $\pi \in \Pi$ is an atomic proposition, and I is a non-singular time interval with integer end-points. MITL admits derived operators like 'constrained eventually' ($\Diamond_I \varphi := \top \mathcal{U}_I \varphi$) and 'constrained always' ($\Box_I \varphi := \neg(\Diamond_I \neg\varphi)$). Throughout this paper, we assume that I is bounded. We further rewrite the given MITL formula in the negation normal form so that negations appear only in front of atomic propositions. We augment the atomic proposition set Π so that any atomic proposition π and its negation $\neg\pi$ are both included in Π.

We focus on the pointwise MITL semantics [40]. A *timed word* is an infinite sequence $\rho = (a_0, t_0)(a_1, t_1)\ldots$, where $a_i \in 2^{\Pi}$, $t_i \in \mathbb{R}_{\geq 0}$ is the time index with $t_{i+1} > t_i \ \forall i \geq 0$. We denote $a_0, a_1, \cdots$ as a word over Π, and $t_0, t_1, \cdots$ as a time sequence. With $\rho(i) = (a_i, t_i)$, we define: $\textsc{untime}(\rho) := a_0, a_1, \cdots$, and $\textsc{val}(\rho) := t_0, t_1, \cdots$. We interpret MITL formulae over timed words as follows.

Definition 1 (MITL Semantics)**.** *Given a timed word ρ and an MITL formula φ, the satisfaction of φ at position j, denoted as $(\rho, j) \models \varphi$, is defined inductively as follows:*

1. *$(\rho, j) \models \top$ if and only if (iff) (ρ, j) is true;*
2. *$(\rho, j) \models \pi$ iff $\pi \in a_j$;*
3. *$(\rho, j) \models \neg\varphi$ iff (ρ, j) does not satisfy φ;*
4. *$(\rho, j) \models \varphi_1 \wedge \varphi_2$ iff $(\rho, j) \models \varphi_1$ and $(\rho, j) \models \varphi_2$;*
5. *$(\rho, j) \models \varphi_1 \mathcal{U}_I \varphi_2$ iff $\exists k \geq j$ such that $(\rho, k) \models \varphi_2$, $t_k - t_j \in I$, and $(\rho, m) \models \varphi_1$ holds for all $j \leq m < k$.*

We denote $\rho \models \varphi$ if $(\rho, 0) \models \varphi$. The satisfaction of an MITL formula can be equivalently associated to accepting words of a timed Büchi automaton (TBA) [20]. Let $C = \{c_1, \cdots, c_M\}$ be a finite set of clocks. Define a set of clock constraints $\Phi(C)$ over C as $\xi = \top|\bot|c \bowtie \delta|\xi_1 \wedge \xi_2$, where $\bowtie \in \{\leq, \geq, <, >\}$, $c, c' \in C$ are clocks, and $\delta \in \mathbb{Q}$ is a non-negative rational number. In this paper, we focus on a subclass of MITL formulae that can be equivalently represented as deterministic timed Büchi automaton defined as follows.

Definition 2 (Deterministic Timed Büchi Automaton [3])**.** *A deterministic timed Büchi automaton (DTBA) is a tuple $\mathcal{A} = (Q, 2^{\Pi}, q_0, C, \Phi(C), E, F)$, where Q is a finite set of states, 2^{Π} is an alphabet over atomic propositions in Π, q_0 is the initial state, $E \subseteq Q \times Q \times 2^{\Pi} \times 2^{C} \times \Phi(C)$ is the set of transitions, and $F \subseteq Q$ is the set of accepting states. A transition $< q, q', a, C', \phi > \in E$ if $\mathcal{A}$ enables the transition from q to q' when a subset of atomic propositions $a \in 2^{\Pi}$ and clock constraints $\phi \in \Phi(C)$ evaluate to true. The clocks in $C' \subseteq C$ are reset to zero after the transition.*

We present the DTBA representing MITL formula $\varphi = \Diamond_{[2,3]}\pi$ as an example in Fig. 1. In this figure, the states Q and transitions E are represented by circles and arrows, respectively. Here the initial state is q_0. The set of accepting states is $F = \{q_2\}$. Consider the transition from initial state q_0 to state q_2. The transition $< q_0, q_2, \pi, c, \phi >$ can take place

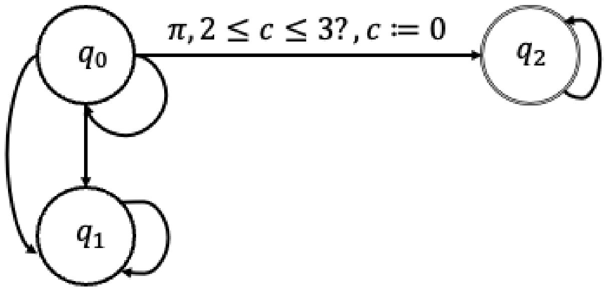

Figure 1. The deterministic timed Büchi automaton (DTBA) representing metric interval temporal logic formula $\varphi = \diamond_{[2,3]}\pi$. The states and transitions of the DTBA are represented by circles and arrows, respectively. The initial state of this DTBA is q_0 and the accepting state is q_2. The formula φ can be satisfied if DTBA reaches state q_2.

if atomic proposition π is evaluated to be true and clock constraint $\phi(c)$ defined on clock c satisfies $2 \leq c \leq 3$. Furthermore, the clock c is reset to zero after the transition.

Given the set of clocks C, $\mathbf{v} : C \mapsto V$ is the *valuation* of C, where $V \subseteq \mathbb{Q}^{|C|}$. Let $\mathbf{v}(c)$ be the valuation of clock $c \in C$. We say $\mathbf{v} = \mathbf{0}$ if $\mathbf{v}(c) = 0$ for all $c \in C$. Given $\delta \in \mathbb{R}_{\geq 0}$, we let $\mathbf{v} + \delta := [\mathbf{v}(1) + \delta, \cdots, \mathbf{v}(|C|) + \delta]^T$. A *configuration* of $\mathcal{A}$ is a pair $(q, \mathbf{v})$, where $q \in Q$ is a state of $\mathcal{A}$. Suppose a transition $< q, q', a, C', \xi >$ is taken after δ time units. Then the DTBA is transited from configuration $(q, \mathbf{v})$ to $(q', \mathbf{v} + \delta)$ such that $\mathbf{v} + \delta \models \xi$, $\mathbf{v}'(c) = \mathbf{v}(c) + \delta$ for all $c \notin C'$, and $\mathbf{v}'(c) = 0$ for all $c \in C'$. We denote the transition between these configurations as $(q, \mathbf{v}) \xrightarrow{a,\delta} (q', \mathbf{v} + \delta)$. A *run* of $\mathcal{A}$ is a sequence of such transitions between configurations $\beta := (q_0, \mathbf{v}_0) \xrightarrow{a_0,\delta_0} (q_1, \mathbf{v}_1) \cdots$. A feasible run β on $\mathcal{A}$ is *accepting iff* it intersects with F infinitely often.

4. Problem Setup and Formulation

In this section, we propose *durational stochastic games* that generalize stochastic games, and present the defender and adversary models in terms of information available to them. We then define three robustness degrees, and state the problem of interest.

4.1. Environment, Defender, and Adversary Models

We introduce durational stochastic games as a generalization of stochastic games [10]. Different from SGs, DSGs model (i) timing information for transitions between states, and (ii) an attack surface resulting from the timing information. An SG is defined as follows.

Definition 3 (Stochastic Game). *A (labeled) stochastic game $\mathcal{SG}$ is a tuple $\mathcal{SG} = (S, U_C, U_A, Pr, \Pi, \mathcal{L})$, where S is a finite set of states, U_C is a finite set of actions of the defender, U_A is a finite set of actions of an adversary, $Pr : S \times U_C \times U_A \times S \to [0,1]$ is a transition function where $Pr(s, u_C, u_A, s')$ is the probability of a transition from state s to state s' when the defender takes action u_C and the adversary takes action u_A. Π is a set of atomic propositions. $L : S \to 2^{\Pi}$ is a labeling function mapping each state to a subset of propositions in Π.*

The SG in Definition 3 cannot be used to verify satisfaction of an MITL objective since it does not include a notion of time. We define *durational stochastic games* to bridge this gap. DSGs incorporate a notion of time taken for a transition between states, and also models the ability of an adversary to modify this timing information.

Definition 4 (Durational Stochastic Game). *A (labeled) durational stochastic game (DSG) is a tuple $\mathcal{G} = (S_{\mathcal{G}}, s_{\mathcal{G},0}, U_C, U_A, Inf_{\mathcal{G},C}, Inf_{\mathcal{G},A}, Pr_{\mathcal{G}}, T_{\mathcal{G}}, \Pi, L, Cl)$. $S_{\mathcal{G}}$ is a finite set of states, $s_{\mathcal{G},0}$ is the initial state, U_C, U_A are finite sets of actions. $Inf_{\mathcal{G},C} : S_{\mathcal{G}} \times \mathbb{R}_{\geq 0} \mapsto (S_{\mathcal{G}} \times \mathbb{R}_{\geq 0})^*$ and $Inf_{\mathcal{G},A} : S_{\mathcal{G}} \times \mathbb{R}_{\geq 0} \mapsto (S_{\mathcal{G}} \times \mathbb{R}_{\geq 0} \times U_C)^*$ are information sets of the defender and adversary, respectively, where $(\cdot)^*$ is the Kleene operator. $Pr_{\mathcal{G}} : S_{\mathcal{G}} \times U_C \times U_A \times S_{\mathcal{G}} \mapsto [0,1]$ encodes*

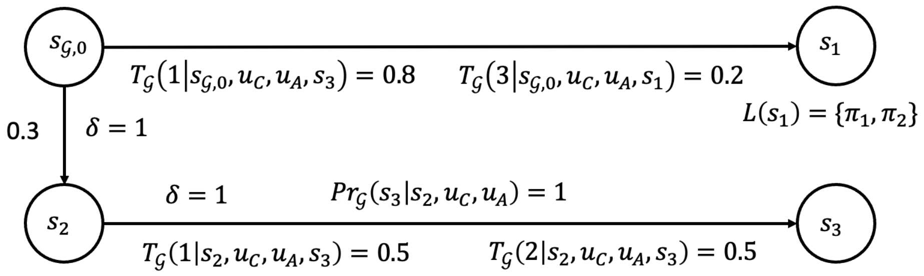
$s_{\mathcal{G},0}$
s_1
$T_{\mathcal{G}}(1|s_{\mathcal{G},0}, u_C, u_A, s_3) = 0.8$
$T_{\mathcal{G}}(3|s_{\mathcal{G},0}, u_C, u_A, s_1) = 0.2$
$L(s_1) = \{\pi_1, \pi_2\}$
0.3
$\delta = 1$
$\delta = 1$
$Pr_{\mathcal{G}}(s_3|s_2, u_C, u_A) = 1$
s_2
s_3
$T_{\mathcal{G}}(1|s_2, u_C, u_A, s_3) = 0.5$
$T_{\mathcal{G}}(2|s_2, u_C, u_A, s_3) = 0.5$

Figure 2. This figure presents an example of a DSG consisting of 4 states, denoted as $S_{\mathcal{G}} = \{s_{\mathcal{G},0}, s_1, s_2, s_3\}$. The transition probabilities $Pr_{\mathcal{G}}$ and the probability mass function $T_{\mathcal{G}}$ for some transitions are given in the figure. The labeling function L for state s_1 is given as $L(s_1) = \{\pi_1, \pi_2\}$.

$Pr_{\mathcal{G}}(s'_{\mathcal{G}}|s_{\mathcal{G}}, u_C, u_A)$, the transition probability from state $s_{\mathcal{G}}$ to $s'_{\mathcal{G}}$ when the controller and adversary take actions u_C and u_A. $T_{\mathcal{G}} : S_{\mathcal{G}} \times U_C \times U_A \times S_{\mathcal{G}} \times \Delta \mapsto [0,1]$ is a probability mass function. $T_{\mathcal{G}}(\delta|s_{\mathcal{G}}, u_C, u_A, s'_{\mathcal{G}})$ denotes the probability that a transition from $s_{\mathcal{G}}$ to $s'_{\mathcal{G}}$ under actions u_C and u_A takes $\delta \in \Delta$ time units, where Δ is a finite set of time units that each transition of DSG can possibly take to complete. Π is a set of atomic propositions. $L : S_{\mathcal{G}} \mapsto 2^{\Pi}$ is a labeling function that maps each state to atomic propositions in Π that are true in that state, and Cl is a finite set of clocks.

The set of admissible actions that can be taken by the defender (adversary) in a state $s \in S_{\mathcal{G}}$ is denoted $U_C(s)$ $(U_A(s))$. A path on $\mathcal{G}$ is a sequence of states $w := s_0 \xrightarrow[\delta_0]{u_{C,0}, u_{A,0}} s_1 \ldots \xrightarrow[\delta_i]{u_{C,i}, u_{A,i}} s_{i+1} \ldots$ such that $s_0 = s_{\mathcal{G},0}$, $Pr_{\mathcal{G}}(s_{i+1}|s_i, u_{C,i}, u_{A,i}) > 0$, and $T_{\mathcal{G}}(\delta|s_i, u_{C,i}, u_{A,i}, s_{i+1}) > 0$ for some $u_{C,i} \in U_C(s_i)$, $u_{A,i} \in U_A(s_i)$, and $\delta \in \Delta$ for all $i \geq 0$. Consider the DSG with $S_{\mathcal{G}} = \{s_{\mathcal{G},0}, s_1, s_2, s_3\}$, $U_C = \{u_C\}$, and $U_A = \{u_A\}$ presented in Fig. 2 as an example. We have that $w = s_0 \xrightarrow[1]{u_C, u_A} s_2 \xrightarrow[1]{u_C, u_A} s_3$ is a finite path. We denote the set of finite (infinite) paths by $(S_{\mathcal{G}} \times \mathbb{R}_{\geq 0})^*$ $((S_{\mathcal{G}} \times \mathbb{R}_{\geq 0})^{\omega})$. Given a path w, $L(w) := L(s_{\mathcal{G},0}), L(s_1), \ldots,$ is the sequence of atomic propositions corresponding to states in w. The sequence of state-time tuples in w is obtained as $(s_0, k_0), (s_1, k_1), \ldots$, where $k_i + \delta_i = k_{i+1}$, $i = 0, 1, \ldots$.

For the defender, a *deterministic policy* $\mu : (S_{\mathcal{G}} \times \mathbb{R}_{\geq 0})^* \mapsto U_C$ is a map from the set of finite paths to its actions. A *randomized policy* $\mu : (S_{\mathcal{G}} \times \mathbb{R}_{\geq 0})^* \mapsto \mathcal{D}(U_C)$ maps the set of finite paths to a probability distribution over its actions. A policy is *memoryless* if it only depends on the the most recent state.

Consider a path w in $\mathcal{G}$. At a state s, the information set of the defender is $Inf^{w}_{\mathcal{G},C}(s, k_C) := \{(s_{\mathcal{G},0}, 0), \ldots, (s, k_C)\}$, where k_C is the time perceived by the defender when it reaches s along w. For example, given the finite path $w = s_0 \xrightarrow[1]{u_C, u_A} s_2 \xrightarrow[1]{u_C, u_A} s_3$ for the DSG presented in Fig. 2, information set $Inf^{w}_{\mathcal{G},C}(s_2, k_C) = \{(s_{\mathcal{G},0}, 0), (s_1, 1)\}$. For the adversary, $Inf_{\mathcal{G},A}(s, k_A) := \{(s_{\mathcal{G},0}, 0), \ldots, (s, k_A)\} \cup \{\mu\}$, where k_A is the time observed by the adversary at s, and μ is the defender's policy. Information sets of the defender and adversary are given by $Inf_{\mathcal{G},C}(s, k_C) := \bigcup_{w} Inf^{w}_{\mathcal{G},C}(s, k_C)$ and $Inf_{\mathcal{G},A}(s, k_A) := \bigcup_{w} Inf^{w}_{\mathcal{G},A}(s, k_A)$.

We assume that the initial time is 0, and this is known to both agents. The adversary having knowledge of the policy μ committed to by the defender introduces an asymmetry between the information sets of the two agents. We note that although the adversary is aware of the defender's randomized policy, it does not know the exact action u_C. This is also known as the *Stackelberg setting* in game theory. We assume a concurrent Stackelberg setting in that both the defender and adversary take their actions at each state simultaneously.

The solution concept to a Stackelberg game is Stackelberg equilibrium, which is defined as follows.

Definition 5 (Stackelberg Equilibrium [16])**.** *A tuple $(\mu, (\tau, \gamma))$ is a* Stackelberg equilibrium *if $\mu = \arg\max_{\mu'} Q_C(\mu', BR(\mu'))$, where $Q_C(\mu, (\tau, \gamma))$ and $Q_A(\mu, (\tau, \gamma))$ are the expected utilities* *of the defender and adversary under policies μ and (τ, γ), respectively, and $BR(\mu') = \{(\tau, \gamma) : (\tau, \gamma) = \arg\max_{(\tau',\gamma')} Q_A(\mu', (\tau', \gamma'))\}$.*

If $BR(\mu')$ contains multiple adversary policies, the defender will arbitrarily pick one. During an *actuator attack*, the adversary can manipulate state transitions in $\mathcal{G}$ since its actions u_A will influence the transition probabilities $Pr_{\mathcal{G}}$. The adversary could also exploit the attack surface that will be introduced as a consequence of including timing information. We term this a *timing attack*. In this paper, we consider the worst-case scenario, and assume that the adversary knows the correct time index at each time k. However, it can manipulate timing information perceived by the defender through $T_{\mathcal{G}}$. Thus, the time index k_C perceived by the defender need not be the same as that known to the adversary, k_A.

The adversary launches actuator and timing attacks through attack policies. An *actuator attack policy* $\tau : (S_{\mathcal{G}} \times \mathbb{R}_{\geq 0})^* \mapsto U_A$ specifies the action taken by the adversary, given the set of finite paths. A *timing attack policy* $\gamma : V \times V \mapsto [0, 1]$ takes as its input the correct clock valuation, and yields a probability distribution over clock valuations. This models the ability of the adversary to manipulate clock valuations. For an intelligent adversary, it should launch the timing attack such that resulting sequence of clock valuations is monotone when the clocks are not reset. The reason is such non-monotone clock valuations informs the defender the presence of timing attack, and thus the defender can simply ignore the perceived clock valuations.

4.2. *Definitions of Robustness Degree*

In this subsection, we define three robustness degrees defined with respect to policies on the DSG $\mathcal{G}$.

4.2.1. Spatial Robustness

The spatial robustness, denoted as $\chi_s^{\varphi}(\mu, \tau, \gamma)$, represents the minimum distance between any accepting (resp. non-accepting) path on the DSG induced by policies μ and (τ, γ) and the language of the MITL specification, without regard to the timing information. We define the spatial robustness using the Levenshtein distance, which is used to measure the distance between strings [41].

Definition 6 (Levenshtein Distance [41])**.** *The Levenshtein distance between sequences of symbols w_1 and w_2, denoted $d_L(w_1, w_2)$, is the minimum number of edit operations (insertions, substitutions, or deletions) that can be applied to w_1 so that w_1 can be converted to w_2.*

Consider timed words $w_1 = (q_0, 0)(q_1, 1)(q_2, 2)\ldots$ and $w_2 = (q_0, 0)(q_1', 1)(q_2, 2)\ldots$ that differ at position 1, where $q_1 \neq q_1'$. Then, $d_L(w_1, w_2) = 1$, since w_1 can be converted to w_2 by substituting q_1 with q_1'. Relying on the Levenshtein distance in Definition 6, we define the *spatial robustness* $\chi_s^{\varphi}(\mu, \tau, \gamma)$ for policies μ and (τ, γ) on a DSG $\mathcal{G}$ with respect to MITL formula φ as

$$\chi_s^{\varphi}(\mu, \tau, \gamma) = \begin{cases} \min_{w_1 \in \mathcal{B}_{\mathcal{G}}^{\mu\tau\gamma}, w_2 \notin \mathcal{L}} d_L(w_1, w_2), \text{ if } \mathcal{B}_{\mathcal{G}}^{\mu\tau\gamma} \subseteq \mathcal{L}; \\ -\min_{w_1 \in \mathcal{B}_{\mathcal{G}}^{\mu\tau\gamma}, w_2 \in \mathcal{L}} d_L(w_1, w_2), \text{ otherwise.} \end{cases} \tag{1}$$

In Equation (1), $\mathcal{B}_{\mathcal{G}}^{\mu\tau\gamma}$ is the set of paths enabled on $\mathcal{G}$ under policies μ and (τ, γ), and $\mathcal{L}$ contains the set of paths on $\mathcal{G}$ that satisfy φ. We note that since $d_L(\cdot, \cdot) \geq 0$, any path $w \in \mathcal{B}_{\mathcal{G}}^{\mu\tau\gamma}$ synthesized under policies μ and τ that satisfies φ will result in $\chi_s^{\varphi}(\mu, \tau, \gamma) > 0$. If, for some $w \in \mathcal{B}_{\mathcal{G}}^{\mu\tau\gamma}$, $w \notin \mathcal{L}$, then $\chi_s^{\varphi}(\mu, \tau, \gamma) \leq 0$.

4.2.2. Temporal Robustness

The temporal robustness $\chi_t^{\varphi}(\mu, \tau, \gamma)$ captures the maximum time units by which any accepting path synthesized under policies μ and (τ, γ) can be temporally perturbed so that the MITL formula φ is not violated. Given an accepting run w and $k \in \mathbb{Q}$, we let $\textsc{val}(w) + k := \mathbf{v}_0 + k, \mathbf{v}_1 + k, \ldots$. We define the left temporal robustness $\chi_t^{\varphi,-}(\mu,\tau,\gamma)$ and right temporal robustness $\chi_t^{\varphi,+}(\mu,\tau,\gamma)$ as:

$$\chi_t^{\varphi,-}(\mu,\tau,\gamma) = \max \bigcap_{w \in \mathcal{B}_{\mathcal{G}}^{\mu\tau\gamma}} \{k | w' \models \varphi \ \forall w' \text{ s.t. } 0 \le \textsc{val}(w) - \textsc{val}(w') \le k \in \mathbb{Q}\}, \quad (2)$$

$$\chi_t^{\varphi,+}(\mu,\tau,\gamma) = \max \bigcap_{w \in \mathcal{B}_{\mathcal{G}}^{\mu\tau\gamma}} \{k | w' \models \varphi \ \forall w' \text{ s.t. } 0 \le \textsc{val}(w') - \textsc{val}(w) \le k \in \mathbb{Q}\}. \quad (3)$$

The left (right) temporal robustness $\chi_t^{\varphi,-}(\mu,\tau,\gamma)$ ($\chi_t^{\varphi,+}(\mu,\tau,\gamma)$) indicates that an accepting run w induced by μ and (τ,γ) can be perturbed up to k time units to the left (right) without violating φ. These definitions also ensure that any perturbation smaller than $\chi_t^{\varphi,-}(\mu,\tau,\gamma)$ or $\chi_t^{\varphi,+}(\mu,\tau,\gamma)$ will not violate φ. The temporal robustness is then:

$$\chi_t^{\varphi}(\mu,\tau,\gamma) = \begin{cases} \min\{\chi_t^{\varphi,-}(\mu,\tau,\gamma), \chi_t^{\varphi,+}(\mu,\tau,\gamma)\}, \text{ if } \mathcal{B}_{\mathcal{G}}^{\mu\tau\gamma} \subseteq \mathcal{L} \\ \Lambda, \text{ otherwise} \end{cases}, \quad (4)$$

where Λ is a symbol indicating that policies μ and (τ,γ) can lead to non-accepting runs.

4.2.3. Spatio-temporal Robustness

We define the spatio-temporal robustness $\chi^{\varphi}(\mu,\tau,\gamma)$ to unify notions of spatial and temporal robustness as:

$$\chi^{\varphi}(\mu,\tau,\gamma) = I(\chi_s^{\varphi}(\mu,\tau,\gamma) \ge \epsilon_s)\chi_t^{\varphi}(\mu,\tau,\gamma), \quad (5)$$

where $I(\chi_s^{\varphi}(\mu,\tau,\gamma) \ge \epsilon_s)$ is an indicator function that equals to 1 if $\chi_s^{\varphi}(\mu,\tau,\gamma) \ge \epsilon_s$ and -1 otherwise. In other words, the spatio-temporal robustness $\chi^{\varphi}(\mu,\tau,\gamma)$ captures the maximum time units by which any accepting run can be perturbed without violating the MITL specification φ, given a desired spatial robustness ϵ_s, under policies μ and (τ,γ). Note that when the spatio-temporal robustness is $-\Lambda$, we have that policies μ and (τ,γ) lead to non-accepting runs.

4.2.4. Robust MITL Semantics

Given the spatio-temporal robustness in Equation (5), we can use a real-valued function $\zeta^{\varphi}(\rho,j)$ to reason about the satisfaction of φ such that $(\rho,j) \models \varphi \equiv \zeta^{\varphi}(\rho,j) > 0$.

Definition 7 (Robust MITL Semantics). *Let ρ be a timed word. We define a real-valued function $\zeta^{\varphi}(\rho,j)$ such that the satisfaction of an MITL formula φ at position j by a timed word ρ, written $(\rho,j) \models \varphi := \zeta^{\varphi}(\rho,j) > 0$, can be recursively defined as:*

1. $\zeta^{\varphi}(\rho,j) = f(\rho,j)$;
2. $\zeta^{\varphi_1 \wedge \varphi_2}(\rho,j) = \min\{\zeta^{\varphi_1}(\rho,j), \zeta^{\varphi_2}(\rho,j)\}$;
3. $\zeta^{\varphi_1 \vee \varphi_2}(\rho,j) = \max\{\zeta^{\varphi_1}(\rho,j), \zeta^{\varphi_2}(\rho,j)\}$;
4. $\zeta^{\varphi_1 \mathcal{U}_{[a,b]} \varphi_2}(\rho,j) = \max_{t' \in [j+a, j+b]}\{\min\{\zeta^{\varphi_2}(\rho,t'), \min_{t'' \in [j,t']} \zeta^{\varphi_1}(\rho,t'')\}\}$.

where $f(\rho,j) = I(\min_{w \notin \mathcal{L}} d_L(\rho,w) \ge \epsilon_s)\bar{k}$, and $\bar{k} = \max\{k | (\rho',j) \models \varphi \ \forall \rho'$ s.t. $0 \le |\textsc{val}(\rho) - \textsc{val}(\rho')| \le k\}$.

4.3. Problem Statement

Before formally stating the problem of interest, we prove a result which shows that a defender's policy that provides positive spatio-temporal robustness satisfies the MITL objective φ with probability one.

Proposition 1. *Given an MITL objective φ and policies μ and (τ,γ), the spatio-temporal robustness $\chi^{\varphi}(\mu,\tau,\gamma) > 0$ implies almost-sure satisfaction of φ under the agent policies when there is no timing attack.*

Proof. The proof of this result is deferred to Appendix B. □

Given Proposition 1, we formally state our problem:

Problem 1 (Robust Policy Synthesis for Defender). *Given a DSG $\mathcal{G}$ and an MITL formula φ, compute an almost-sure defender policy. That is, compute μ such that $\chi^{\varphi}(\mu,\tau,\gamma) \geq \epsilon_t$, where $(\tau,\gamma) \in BR(\mu)$.*

5. Solution: Only Actuator Attack

We present a solution to robust policy synthesis for defender as described in Problem 1, assuming that the adversary only launches an actuator attack. We construct a *product DSG* $\mathcal{P}$ from DSG $\mathcal{G}$ and DTBA $\mathcal{A}$. We present procedures to evaluate the spatio-temporal robustness, and compute an optimal policy for the defender on $\mathcal{P}$.

5.1. Product DSG

In the following, we give the definition of product DSG.

Definition 8 (Product Durational Stochastic Game). *A PDSG $\mathcal{P}$ constructed from a DSG $\mathcal{G}$, DTBA $\mathcal{A}$, and clock valuation set V is a tuple $\mathcal{P} = (S, s_0, U_C, U_A, Inf_C, Inf_A, Pr, Acc)$. $S = S_{\mathcal{G}} \times Q \times V$ is a finite set of states, $s_0 = (s_{\mathcal{G},0}, q_0, \mathbf{0})$ is the initial state, U_C, U_A are finite sets of actions. Inf_C, Inf_A are information sets of the defender and adversary. $Pr : S \times U_C \times U_A \mapsto \mathcal{S}$ encodes $Pr((s',q',\mathbf{v}')|(s,q,\mathbf{v}),u_C,u_A)$, the probability of a transition from state $(s,q,\mathbf{v})$ to $(s',q',\mathbf{v}')$ when the defender and adversary take actions u_C and u_A. The probability*

$$Pr\big((s',q',\mathbf{v}')|(s,q,\mathbf{v}),u_C,u_A\big) := T_{\mathcal{G}}(\delta|s,u_C,u_A,s')Pr_{\mathcal{G}}(s'|s,u_C,u_A) \tag{6}$$

if and only if $(q,\mathbf{v}) \xrightarrow{L(s'),\delta} (q',\mathbf{v}')$, and zero otherwise. $Acc = S_{\mathcal{G}} \times F \times V$ is a finite set of accepting states.

The following result shows that the transition probability of $\mathcal{P}$ is well-defined.

Proposition 2. *The function $Pr(\cdot)$ satisfies $Pr((s',q',\mathbf{v}')|(s,q,\mathbf{v}),u_C,u_A) \in [0,1]$ and*

$$\sum_{(s',q',\mathbf{v}')} Pr\big((s',q',\mathbf{v}')|(s,q,\mathbf{v}),u_C,u_A\big) = 1. \tag{7}$$

Proof. The proof is presented in Appendix B. □

We write $\mathfrak{s}$ to represent a state $(s,q,\mathbf{v})$ in PDSG $\mathcal{P}$. We denote the clock valuation of $\mathfrak{s}$ by $Time(\mathfrak{s})$. In the sequel, we compute a set of states called *generalized accepting maximal end components* (GAMECs) of $\mathcal{P}$. Any state $\mathfrak{s}$ in GAMECs satisfies that the successor state $\mathfrak{s}'$ also belogns to GAMECs under any policy committed by the defender, regardless of the actions taken by the adversary. Therefore, for a path that stays within GAMECs, it is guaranteed that the path corresponds to a run that intersects with F infinitely many times, and thus the path satisfies specification φ. We can thus translate the problem of satisfying φ to the problem of reaching GAMECs, under any adversary action. The set $\mathcal{C} = \{\mathfrak{s}|\mathfrak{s}$ belongs to some GAMEC$\}$ can be computed using the procedure COMPUTE_GAMEC($\mathcal{P}$) in Algorithm 1. The idea is that at each state, we prune the defender's admissible action set by retaining only those actions that ensure state transitions in $\mathcal{P}$ will remain within GAMECs, under any adversary action.

The procedure Compute_GAMEC($\mathcal{P}$) presented in Algorithm 1 takes the product DSG $\mathcal{P}$ as its input, and returns set $\mathcal{C}$. The algorithm iteratively updates $\mathcal{C}$ by removing a set of

Algorithm 1 Computing the set of GAMECs $\mathcal{C}$.

```
1: procedure COMPUTE_GAMEC($\mathcal{P}$)
2:     Input: PDSG $\mathcal{P}$
3:     Output: Set of GAMECs $\mathcal{C}$
4:     Initialization:$D(\mathfrak{s}) \leftarrow U_C(\mathfrak{s}) \forall \mathfrak{s}; \mathcal{C} \leftarrow \emptyset; \mathcal{C}_{temp} \leftarrow \{S\}$
5:     repeat
6:         $\mathcal{C} \leftarrow \mathcal{C}_{temp}, \mathcal{C}_{temp} \leftarrow \emptyset$
7:         for $N \in \mathcal{C}$ do
8:             $R \leftarrow \emptyset$
9:             Let $SCC_1, \cdots, SCC_n$ be the set of strongly connected components of underlying digraph $G_{(N,D)}$
10:            for $i = 1, \cdots, n$ do
11:                for each state $\mathfrak{s} \in SCC_i$ do
12:                    $D(\mathfrak{s}) \leftarrow \{u_C \in U_C(\mathfrak{s}) | \mathfrak{s}' \in N, Pr(\mathfrak{s}'|\mathfrak{s}, u_C, u_A) > 0, \forall u_A \in U_A(\mathfrak{s})\}$
13:                    if $D(s) = \emptyset$ then
14:                        $R \leftarrow R \cup \{\mathfrak{s}\}$
15:                    end if
16:                end for
17:            end for
18:            while $R \neq \emptyset$ do
19:                dequeue $\mathfrak{s} \in R$ from $R$ and $N$
20:                if $\exists \mathfrak{s}' \in N$ and $u_C \in U_C(\mathfrak{s}')$ such that $Pr(\mathfrak{s}|\mathfrak{s}', u_C, u_A) > 0$ for some $u_A \in U_A(\mathfrak{s}')$ then
21:                    $D(\mathfrak{s}') \leftarrow D(\mathfrak{s}') \setminus \{u_C\}$
22:                    if $D(\mathfrak{s}') = \emptyset$ then
23:                        $R \leftarrow R \cup \{\mathfrak{s}'\}$
24:                    end if
25:                end if
26:            end while
27:            for $i = 1, \cdots, n$ do
28:                if $N \cap SCC_i \neq \emptyset$ then
29:                    $\mathcal{C}_{temp} \leftarrow \mathcal{C}_{temp} \cup \{N \cap SCC_i\}$
30:                end if
31:            end for
32:        end for
33:    until $\mathcal{C} = \mathcal{C}_{temp}$
34:    for $N \in \mathcal{C}$ do
35:        if $Acc_{\mathcal{G}} \cap N = \emptyset$ then
36:            $\mathcal{C} \leftarrow \mathcal{C} \setminus N$
37:        end if
38:    end for
39:    return $\mathcal{C}$
40: end procedure
```

states R. R includes any state $\mathfrak{s}$ that is in some strongly connected component (SCC) and has an empty admissible defender action set (line 13). R also includes states $\mathfrak{s}'$ from which $\mathcal{P}$ can be steered to R under some adversary action (line 20). Lines 35-37 verify accepting conditions defined by the DTBA. The termination of Algorithm 1 is given by the following Proposition.

Proposition 3. *Algorithm 1 terminates in a finite number of iterations.*

Proof. The proof of this proposition is given in Appendix B. □

5.2. Evaluating Spatial Robustness

From Equation (1), evaluating the spatial robustness is equivalent to computing the Levenshtein distance between paths on the DSG synthesized under policies μ and (τ, γ) and $\mathcal{L}$. This is equivalent to computing the Levenshtein distance between two automata, where the first automaton $\mathcal{P}^{\mu\tau\gamma}$ is the PDSG induced by policies μ and (τ,γ). The second automaton is $\bar{\mathcal{A}}$, the DTBA representing $\neg\varphi$. We adopt the approach proposed in [42] to compute the Levenshtein distance between $\mathcal{P}^{\mu\tau\gamma}$ and $\bar{\mathcal{A}}$.

We first construct a DSG $\mathcal{G}^{\mu\tau\gamma}$ from the original DSG $\mathcal{G}$. Given policies μ and (τ,γ), we retain only those transitions such that $Pr_{\mathcal{G}}(s'|s,u_C,u_A) > 0$, $T_{\mathcal{G}}(\delta|s,u_C,u_A,s') > 0$ for some δ, $\mu(s,u_C) > 0$, and $\tau(s,u_A) > 0$, and remove all other transitions. We augment the alphabet of DTBA $\mathcal{A}$ as $2^{\Pi} \cup \{null\}$, where *null* is a symbol that will be used to indicate deletion and insertion operations. The alphabet of $\bar{\mathcal{A}}$ is also augmented to include *null*. The PDSG $\mathcal{P}^{\mu\tau\gamma}$ in Definition 8 can be constructed from $\mathcal{G}^{\mu\tau\gamma}$ and $\mathcal{A}$. Given $\mathcal{P}^{\mu\tau\gamma}$ and $\bar{\mathcal{A}}$, we construct $\hat{\mathcal{P}} := \mathcal{P}^{\mu\tau\gamma} \times \bar{\mathcal{A}}$. Following [42], we construct a *weighted transducer* to capture the cost associated to each edit operation (assumed = 1). We assign a cost $c((s,q,\mathbf{v},\bar{q}),(s',q',\mathbf{v}',\bar{q}'))$ to each transition from state $(s,q,\mathbf{v},\bar{q})$ to $(s',q',\mathbf{v}',\bar{q}')$ in $\hat{\mathcal{P}}$. In particular, $c((s,q,\mathbf{v},\bar{q}),(s',q',\mathbf{v}',\bar{q}')) = 1$ if $L(s')$ is not the same as the label of the transition from $\bar{q}$ to $\bar{q}'$ in $\bar{\mathcal{A}}$. We can then apply a shortest path algorithm on $\hat{\mathcal{P}}$ from the initial state $(s_0,q_0,\mathbf{0},\bar{q}_0)$ to the union of the GAMECs of $\hat{\mathcal{P}}$ to compute the minimum Levenshtein distance. The correctness of this approach follows from [42, Thm. 2].

The computational complexity of calculating the spatial robustness for any given policies μ and (τ,γ) is $O((|2^{\Pi}|+1)^2|\mathcal{P}^{\mu\tau\gamma}||\bar{\mathcal{A}}|)$, where $|\mathcal{P}^{\mu\tau\gamma}|$ and $|\bar{\mathcal{A}}|$ are the sizes of $\mathcal{P}^{\mu\tau\gamma}$ and $\bar{\mathcal{A}}$, respectively [42].

Algorithm 2 Evaluate Temporal Robustness.

```
1: procedure TEMPORAL($\varphi, \mathfrak{s}, \delta, M(\mathfrak{s}')$)
2:     Input: MITL formula $\varphi$, current state $\mathfrak{s}$, time duration $\delta$, indicator function $M(\mathfrak{s}')$
3:     Output: Temporal robustness $\chi_t^{\varphi}(\mu,\tau,\gamma)$
4:     if $\varphi = \pi$ then
5:         $left\_temp \leftarrow \min \bigcup_{\mathfrak{s}''}\{Time(\mathfrak{s}'') - Time(\mathfrak{s})\}$, where $\mathfrak{s}''$ is reachable from $\mathfrak{s}$
6:         $right\_temp \leftarrow \min \bigcup_{\mathfrak{s}''}\{\overline{V} - Time(\mathfrak{s}'')\}$, where $\mathfrak{s}''$ is reachable from $\mathfrak{s}$
7:         return $\min\{left\_temp, right\_temp\}$
8:     else if $\varphi = \phi_1 \wedge \phi_2$ then
9:         $r_1 \leftarrow$ TEMPORAL($\phi_1, \mathfrak{s}, \delta, M(\mathfrak{s}')$)
10:        $r_2 \leftarrow$ TEMPORAL($\phi_2, \mathfrak{s}, \delta, M(\mathfrak{s}')$)
11:        return $\min\{r_1, r_2\}$
12:    else if $\varphi = \phi_1 \vee \phi_2$ then
13:        $r_1 \leftarrow$ TEMPORAL($\phi_1, \mathfrak{s}, \delta, M(\mathfrak{s}')$)
14:        $r_2 \leftarrow$ TEMPORAL($\phi_2, \mathfrak{s}, \delta, M(\mathfrak{s}')$)
15:        return $\max\{r_1, r_2\}$
16:    else if $\varphi = \phi_1 \mathcal{U}_I \phi_2$ then
17:        if $M(\mathfrak{s}') = 0$ then
18:            $r_1 \leftarrow \min \bigcup_{\mathfrak{s}',\delta,\delta'} \{$TEMPORAL$(\phi_1, \mathfrak{s}, \delta, M(\mathfrak{s}')), b^{\mu\tau\gamma}(\mathfrak{s},\mathfrak{s}')$TEMPORAL$(\phi_1 \mathcal{U}_{I-\delta} \phi_2,$
               $\mathfrak{s}', \delta', M(\mathfrak{s}''))\}$
19:        else
20:            $r_1 \leftarrow \min \bigcup_{\mathfrak{s}'} \{(Time(\mathfrak{s}') - \underline{I}), (\overline{I} - Time(\mathfrak{s}'))\}$
21:        end if
22:        if $0 \in I$ then
23:            $r_2 \leftarrow$ TEMPORAL($\phi_2, \mathfrak{s}, \delta, M(\mathfrak{s}')$)
24:        else
25:            $r_2 \leftarrow -\infty$
26:        end if
27:        return $\max\{r_1, r_2\}$
28:    end if
29: end procedure
```

5.3. Evaluating Temporal Robustness

In this subsection, we present a procedure to evaluate the temporal robustness. We introduce some notation. For a time interval I, we use $\underline{I}$ and $\overline{I}$ to represent its lower and upper bounds. The upper bound of the clock valuation set is denoted $\overline{V}$. The indicator function $M(\mathfrak{s})$ takes value 1 if $\mathfrak{s}$ is in GAMEC, and 0 otherwise. A state $\mathfrak{s}'$ is said to be a neighboring state of $\mathfrak{s}$ if $Pr(\mathfrak{s}'|\mathfrak{s}, u_C, u_A) > 0$ for some u_C and u_A such that $\mu(\mathfrak{s}, u_C) > 0$ and $\tau(\mathfrak{s}, u_A) > 0$. Given the policies of the defender and adversary, we define

$$b^{\mu\tau\gamma}(\mathfrak{s}, \mathfrak{s}') := \begin{cases} 1 & \text{if } \mathfrak{s}' \text{ is a neighboring state of } \mathfrak{s} \\ -\infty & \text{otherwise} \end{cases}.$$

The procedure TEMPORAL$(\varphi, \mathfrak{s}, \delta, M(\mathfrak{s}'))$ presented in Algorithm 2 computes the left and right temporal robustness with respect to the MITL objective φ. The left and right temporal robustness of π can be computed by searching over a directed graph representation of the product DSG. The algorithm determines the temporal robustness of φ following the robust MITL semantics (Definition 7) by simple algebraic computations over the temporal robustness of all atomic propositions in φ.

We detail the working of Algorithm 2, which is a recursive procedure to compute the temporal robustness. It takes an MITL formula φ, current state $\mathfrak{s}$, time duration δ, and an indicator function $M(\mathfrak{s}')$ as its inputs. If $\varphi = \pi$, then Algorithm 2 computes the minimum left temporal robustness (*Line 5*) and right temporal robustness (*Line 6*), respectively. The minimum of these quantities is returned as the temporal robustness. From the robust MITL semantics, Algorithm 2 returns the minimum (maximum) temporal robustness when φ is a conjunction (disjunction). When $\varphi = \phi_1 \mathcal{U}_I \phi_2$, the robustness is computed following *Lines 16–27*. Here, $I - t := \{t' - t | t' \in I\}$. Since we focus on the worst-case robustness, we compute the minimum value over times δ and neighboring states $\mathfrak{s}'$ in *Line 18*. We establish the correctness of Algorithm 2 as follows.

Theorem 1. *Given a PDSG with initial state* $\mathfrak{s}_0$*, MITL formula* φ*, and policies* μ *and* τ*, suppose Algorithm 2 returns* $\epsilon \geq 0$*. Then, any run on the PDSG synthesized under policies* μ *and* τ *can be temporally perturbed by* $\hat{\epsilon} \in [0, \epsilon]$ *without violating* φ*.*

Proof. The proof is presented in Appendix B. □

The complexity of Algorithm 2 is $O(|cl(\varphi)|(|S| + |Pr|))$, where $|cl(\varphi)|$ is the size of the closure of formula φ, $|Pr|$ is the number of nonzero elements in matrix Pr.

5.4. Evaluating Spatio-temporal Robustness

Table 1. Computational complexities of evaluating spatial and temporal robustness when policies are given. $|\mathcal{P}^{\mu\tau\gamma}|$ is the size of the product DSG $\mathcal{P}^{\mu\tau\gamma}$ induced by policies μ and (τ, γ). $|\bar{\mathcal{A}}|$ is the size of the timed Büchi automaton of MITL specification $\neg\varphi$. $|cl(\varphi)|$ denotes the size of the closure of φ, and $|Pr|$ is the number of nonzero elements in matrix Pr. The complexity of Algorithm 3 is $(S) + (T)$.

Robustness	**Complexity**
Spatial (S)	$O((\lvert 2^{\Pi}\rvert + 1)^2 \lvert\mathcal{P}^{\mu\tau\gamma}\rvert \lvert\bar{\mathcal{A}}\rvert)$
Temporal (T)	$O(\lvert cl(\varphi)\rvert(\lvert S\rvert + \lvert Pr\rvert))$

We use the results of the previous two subsections to compute the spatio-temporal robustness using the procedure ROBUST$(\varphi, \mathfrak{s}, \delta, M(\mathfrak{s}'), \epsilon_s)$ presented in Algorithm 3. From Equation (5), when the spatial robustness is above ϵ_s, Algorithm 3 returns the temporal robustness. Otherwise, it returns the negative value of the temporal robustness. The complexity of Algorithm 3 is $O(|cl(\varphi)|(|S| + |Pr|) + (|2^{\Pi}| + 1)^2 |\mathcal{P}^{\mu\tau\gamma}||\bar{\mathcal{A}}|)$. Table 1 summarizes the computational complexities of evaluating the spatial and temporal robustness.

Algorithm 3 Evaluate Spatio-temporal Robustness.

```
1: procedure ROBUST($\varphi, \mathfrak{s}, \delta, M(\mathfrak{s}'), \epsilon_s$)
2:     Input: MITL formula $\varphi$, current state $s$, time duration $\delta$, indicator function $M(\mathfrak{s}')$
3:     Output: Spatio-temporal robustness $\chi^{\varphi}(\mu, \tau, \gamma)$
4:     if $\varphi = \top$ then
5:         return $\infty$
6:     else if $\varphi = \bot$ then
7:         return $-\infty$
8:     else
9:         if SPATIAL($\varphi, s$) $\geq \epsilon_s$ then
10:            return TEMPORAL($\varphi, \mathfrak{s}, \delta, M(\mathfrak{s}')$)
11:        else
12:            return $-$TEMPORAL($\varphi, \mathfrak{s}, \delta, M(\mathfrak{s}')$)
13:        end if
14:    end if
15: end procedure
```

5.5. Control Policy Synthesis 430

In this subsection, we compute a control policy that solves robust policy synthesis for 431 defender in Problem 1 when there is no timing attack. 432

Algorithm 4 Robust Control Policy Synthesis for Defender.

```
1: procedure POLICY_SYNTHESIS($\mathcal{P}$, $\varphi$)
2:     Input: Product DSG $\mathcal{P}$, MITL formula $\varphi$
3:     Output: Control policy $\mu$
4:     Initialization: Iteration index $k \leftarrow 1$. Initialize $\mu^k(\mathfrak{s}, u_C) \leftarrow \frac{1}{|U_C(\mathfrak{s})|}$ for all $\mathfrak{s}$ and $u_C \in U_C(\mathfrak{s})$,
   and compute adversary policy $(\tau^k, \gamma^k) \in \mathcal{BR}(\mu^k)$. Let $\mathcal{E}_s, \mathcal{E}_t \leftarrow \emptyset$.
5:     while true do
6:         Compute spatio-temporal robustness $\chi^{\varphi}(\mu^k, \tau^k, \gamma^k) = $ ROBUST($\varphi, \mathfrak{s}_0, \delta, M(\mathfrak{s}')$).
7:         if $\chi^{\varphi}(\mu^k, \tau^k, \gamma^k) \geq \epsilon_t$ then
8:             return $\mu^k$
9:         else if $0 \leq \chi^{\varphi}(\mu^k, \tau^k, \gamma^k) < \epsilon_t$ then
10:            $\mathcal{E}_t \leftarrow \mathcal{E}_t \cup \{\mathfrak{s} : $ ROBUST($\varphi, \mathfrak{s}, \delta, M(\mathfrak{s}')$) $< \epsilon_t\}$
11:            for $\mathfrak{s} \in \mathcal{E}_t$ do
12:                Let $U_C(\mathfrak{s}') \leftarrow U_C(\mathfrak{s}') \setminus \{u_C : \mu^k(\mathfrak{s}', u_C) > 0, Pr^{\mu^k \tau^k}(\mathfrak{s}', \mathfrak{s}) > 0\}$ for all $\mathfrak{s}' \notin \mathcal{E}_t \cup \mathcal{E}_s$
13:                if $U_C(\mathfrak{s}') = \emptyset$ then
14:                    $\mathcal{E}_t \leftarrow \mathcal{E}_t \cup \{\mathfrak{s}'\}$
15:                end if
16:            end for
17:        else
18:            $\mathcal{E}_s \leftarrow \mathcal{E}_s \cup \{\mathfrak{s} : $ ROBUST($\varphi, \mathfrak{s}, \delta, M(\mathfrak{s}')$) $< 0\}$
19:            for $\mathfrak{s} \in \mathcal{E}_s$ do
20:                Let $U_C(\mathfrak{s}') \leftarrow \{u_C | \mu^k(\mathfrak{s}', u_C) > 0, Pr^{\mu^k \tau^k}(\mathfrak{s}', \mathfrak{s}) > 0\}$
21:                if $U_C(\mathfrak{s}') = \emptyset$ then
22:                    $\mathcal{E}_s \leftarrow \mathcal{E}_s \cup \{\mathfrak{s}'\}$
23:                end if
24:            end for
25:            Update defender's policy $\mu^{k+1}(\mathfrak{s}', u_C) \leftarrow \frac{1}{|U_C(\mathfrak{s}')|}$ for all $\mathfrak{s}'$ and $u_C \in U_C(\mathfrak{s}')$
26:            if GAMEC is not reachable from initial state $\mathfrak{s}_0$ then
27:                return message "failure" indicating no solution is found
28:                Break
29:            end if
30:        end if
31:        Let $k \leftarrow k + 1$.
32:    end while
33: end procedure
```

From Proposition 1, solving robust policy synthesis for defender in Problem 1 is equivalent to finding a defender policy so that the spatio-temporal robustness exceeds a desired threshold. This procedure is named as POLICY_SYNTHESIS($\mathcal{P}, \varphi$) and is presented in Algorithm 4. We initialize a policy μ^k, $k = 1$ (*Line 4*). We also define sets of states $\mathcal{E}_t$ and $\mathcal{E}_s$ that will indicate states/transitions that lead to violations of temporal and spatial robustness. We then compute the best response to μ^k as (τ^k, γ^k), evaluate the spatio-temporal robustness $\chi^\varphi(\mu^k, \tau^k, \gamma^k)$. If $\chi^\varphi(\mu^k, \tau^k, \gamma^k) \geq \epsilon_t$, then we synthesize the policy μ^k returned in *Line 6*. If $0 \leq \chi^\varphi(\mu^k, \tau^k, \gamma^k) < \epsilon_t$, then the spatial robustness exceeds ϵ_s, but the temporal robustness is below ϵ_t. In this case, we eliminate defender actions u_C that steer the PDSG into states $\mathfrak{s}$ in $\mathcal{E}_t$ with positive probability thereby causing violation of the temporal robustness constraint. If $\chi^\varphi(\mu^k, \tau^k, \gamma^k) < 0$ (*Line 17*), then the spatial robustness constraint is violated. In this case, we eliminate defender actions that steer the system into states in $\mathcal{E}_s$. If no state in GAMEC is reachable from the initial state $\mathfrak{s}_0$ of the product DSG $\mathcal{P}$, then the procedure POLICY_SYNTHESIS($\mathcal{P}, \varphi$) presented in Algorithm 4 reports failure, indicating that no solution is found for robust policy synthesis for defender in Problem 1 and terminates. We establish the converge of Algorithm 4 as follows.

Theorem 2. *Algorithm 4 terminates within a finite number of iterations.*

Proof. The proof of this theorem is presented in Appendix B. □

In the worst case, we have that Algorithm 4 updates $\hat{U}_C = \emptyset$ with at most $|S| \times |U_C|$ number of iterations. Thus the complexity of Algorithm 4 is $O(|S| \times |U_C|)$. We further present the optimality of the policy found by Algorithm 4 in the following theorem.

Theorem 3. *If Algorithm 4 returns a defender's policy, denoted as μ^*, then the problem of robust policy synthesis for defender in Problem 1 is feasible. Moreover, the defender's policy μ^* is an optimal solution to Problem 1.*

Proof. The proof is presented in Appendix B. □

The soundness of Algorithm 4 is given below:

Corollary 1. *Algorithm 4 is sound but not complete. That is, any control policy returned by Algorithm 4 guarantees probability one of satisfying the given MITL specification, but we cannot conclude that there exists no solution to the problem if Algorithm 4 returns no solution.*

6. Solution: Actuator and Timing Attacks

In this section, we present a solution under both actuator attack and timing attacks.

Compared to the case when there is no timing attack, we make the following observations. The evaluation of spatial robustness remains unchanged when the adversary can initiate both actuator and timing attacks. Second, the evaluation of temporal robustness can become inaccurate during a timing attack. This is because timing information perceived by the defender can be arbitrarily manipulated by the adversary. As a result, the defender will not be able to evaluate the temporal robustness, and hence the spatio-temporal robustness during a timing attack. Finally, since the defender cannot accurately evaluate the temporal robustness, Proposition 1 will not hold during a timing attack. In the following, we relax the problem of robust synthesis for defender in Problem 1, and try to compute a defender policy such that the probability of satisfying the φ is maximized in the presence of actuator and timing attacks. The reason the defender can evaluate the probability of satisfying φ is that it knows the transition probability $Pr_{\mathcal{G}}$ and probability mass function $T_{\mathcal{G}}$. Thus, it can determine the expected probability and time of reaching each state, given policies of the defender and adversary. The relaxed problem is:

Problem 2 (Policy Synthesis for Defender)**.** *Given a DSG $\mathcal{G}$ and an MITL objective φ, compute a defender's policy such that the probability of satisfying φ is maximized, and adversary policy (τ,γ) is the best response to control policy μ. That is $\max_\mu \mathbb{P}^\varphi(\mu,\tau,\gamma)$, where $(\tau,\gamma)\in BR(\mu)$.* 478 479 480

Since the timing information perceived by the defender has been manipulated by the adversary, the defender has limited knowledge of the current time. Even in this case, it can still detect unreasonable time sequences, e.g., a time sequence that is not monotonic. To recover from the deficit of timing information, we represent the defender's policy using a finite state controller, which enables the defender to track the estimated time. 481 482 483 484 485

Definition 9 (Finite State Controller [25])**.** *A finite state controller (FSC) is a tuple $\mathcal{F}=(Y,y_0,\mu)$, where $Y=\Lambda\times\{0,1\}$ is a finite set of internal states, Λ is a set of estimates of clock valuations, the set $\{0,1\}$ indicates if a timing attack has been detected (1) or not (0). y_0 is the initial internal state. μ is the defender policy, given by:* 486 487 488 489

$$\mu=\begin{cases}\mu_0: Y\times S\times Y\times U_C\mapsto[0,1], \text{ if } \mathcal{H}_0 \text{ holds;}\\ \mu_1: Y\times S_{\mathcal{G}}\times Q\times Y\times U_C\mapsto[0,1], \text{ if } \mathcal{H}_1 \text{ holds,}\end{cases}$$

where μ_0 and μ_1 respectively denote the control policies that will be executed when hypothesis $\mathcal{H}_0$ or $\mathcal{H}_1$ holds. 490 491

For an FSC as given in Definition 9, hypothesis $\mathcal{H}_0$ represents the scenario where no timing attack is detected by the defender, while $\mathcal{H}_1$ represents the scenario where a timing attack is detected. In the FSC, the defender's policy specifies the probability of reaching the next internal state by taking an action u_C, given the current state of DSG, the detection result of timing attack, and the state of DTBA. 492 493 494 495 496

Algorithm 5 Computing an optimal control policy.

```
 1: procedure CONTROL_SYNTHESIS($\mathcal{Z}$)
 2:     Input: Global DSG $\mathcal{P}$
 3:     Output: value vector $\mathbf{Q}$
 4:     Initialization: $\mathbf{Q}^0 \leftarrow \mathbf{0}$, $\mathbf{Q}^1(\mathfrak{s}) \leftarrow 1$ for $\mathfrak{s}\in Acc$, $\mathbf{Q}^1(\mathfrak{s}) \leftarrow 0$ otherwise, $k \leftarrow 0$
 5:     while $\max\{|\mathbf{Q}^{k+1}(\mathfrak{s})-\mathbf{Q}^k(\mathfrak{s})| : \mathfrak{s}\in S\} > \epsilon$ do
 6:         $k \leftarrow k+1$
 7:         for $\mathfrak{s}\notin Acc$ do
 8:             $\mathbf{Q}^{k+1}(\mathfrak{s}) \leftarrow \max_{\mu}\min_{\tau,\gamma}\Big\{\sum_{u_C}\sum_{u_A}\sum_{(\mathfrak{s}',y)}\tau((\mathfrak{s},y),u_A)\gamma(\mathbf{v}'',\mathbf{v})\mathbf{Q}((\mathfrak{s}',y'))$
                $\cdot Pr_{\mathcal{Z}}((\mathfrak{s}',y')|(\mathfrak{s},y),u_C,u_A)\Big\}$
 9:         end for
10:     end while
11:     return $\mathbf{Q}^k$
12: end procedure
```

To capture the state evolutions of DSG, DTBA, and FSC, we construct a *global DSG*. 497

Definition 10 (Global DSG (GDSG))**.** *A GDSG is a tuple $\mathcal{Z}=(S_{\mathcal{Z}},s_{\mathcal{Z},0},U_C,U_A,Inf_{\mathcal{Z},C},Inf_{\mathcal{Z},A},Pr_{\mathcal{Z}},Acc_{\mathcal{Z}})$, where $S_{\mathcal{Z}}=S\times Y$ is a finite set of states, $s_{\mathcal{Z},0}=(s_0,q_0,\mathbf{0},y_0)$ is the initial state. U_C and U_A are finite sets of actions and $Inf_{\mathcal{Z},C}$ and $Inf_{\mathcal{Z},A}$ are the information sets of the defender and adversary respectively. $Pr_{\mathcal{Z}}: S_{\mathcal{Z}}\times U_C\times U_A\times S_{\mathcal{Z}}\mapsto[0,1]$ is a transition function where $Pr_{\mathcal{Z}}((s',q',\mathbf{v}',y')|(s,q,\mathbf{v},y),u_C,u_A)$ is the probability of a transition from state $(s,q,\mathbf{v},y)$ to $(s',q',\mathbf{v}',y)$ when the defender and adversary take actions u_C and u_A respectively. The transition probability is given by*

$$Pr_{\mathcal{Z}}((s',q',\mathbf{v}',y')|(s,q,\mathbf{v},y),u_C,u_A)$$

$$= \begin{cases} \sum_{\mathbf{v}''} \gamma(\mathbf{v}''|\mathbf{v})\mu_0(y', u_C|s, q, \mathbf{v}'', y) Pr((s', q', \mathbf{v}')|(s, q, \mathbf{v}), u_C, u_A), & \text{if } \mathcal{H}_0 \text{ holds;} \\ \mu_1(y', u_C|s, q, y) T_{\mathcal{G}}(\delta|s, u_C, u_A, s') Pr_{\mathcal{G}}(s'|s, u_C, u_A), & \text{if } \mathcal{H}_1 \text{ holds;} \end{cases}$$

$Acc_{\mathcal{Z}} = Acc \times Y$ is the set of accepting states.

Consider the global DSG. Let $\mathbf{Q} \in \mathbb{R}^{|S_{\mathcal{Z}}|}$ be the probability of satisfying φ. Then $\mathbf{Q}$ can be computed from Proposition 4. A proof is presented in [39].

Proposition 4. *Let $\mathbf{Q} := \max_{\mu} \min_{\tau,\gamma} \mathbb{P}(\varphi)$ be the probability of satisfying φ. Then,*

$$\mathbf{Q}((\mathfrak{s}, y)) = \max_{\mu} \min_{\tau,\gamma} \sum_{u_C} \sum_{u_A} \sum_{(\mathfrak{s}', y)} \tau((\mathfrak{s}, y), u_A) \mathbf{Q}((\mathfrak{s}', y')) \cdot Pr_{\mathcal{Z}}\big((\mathfrak{s}', y')|(\mathfrak{s}, y), u_C, u_A\big), \ \forall(\mathfrak{s}, y).$$

Moreover, the value vector is unique.

We use the procedure CONTROL_SYNTHESIS($\mathcal{Z}$) presented in Algorithm 5 to compute the policy μ. Guarantees on its termination is presented in [39]. We finally remark on the complexity of Algorithm 5. We first make the following relaxation to Line 5 of Algorithm 5 so that $\mathbf{Q}^{k+1}((\mathfrak{s}, y))$ is updated if the following holds

$$\max_{\mu} \min_{\tau,\gamma} \sum_{u_C} \sum_{u_A} \sum_{(\mathfrak{s}', y)} \tau((\mathfrak{s}, y), u_A) \mathbf{Q}((\mathfrak{s}', y')) \cdot Pr_{\mathcal{Z}}\big((\mathfrak{s}', y')|(\mathfrak{s}, y), u_C, u_A\big) \geq (1+\epsilon)\mathbf{Q}^k((\mathfrak{s}, y)).$$

Then, Algorithm 5 converges to some $\mathbf{Q}^{k+1}(\mathfrak{s}, y)$ satisfying $\|\mathbf{Q}^{k+1}(\mathfrak{s}, y) - \mathbf{Q}^k(\mathfrak{s}, y)\|_\infty < \epsilon$ within $|S_{\mathcal{Z}}| \max_{(\mathfrak{s},y)}\{\log(1/\mathbf{Q}^0((\mathfrak{s}, y)))/\log(1+\epsilon)\}$ iterations, where parameter $\mathbf{Q}^0((\mathfrak{s}, y)))$ is the smallest value of $\mathbf{Q}^k((\mathfrak{s}, y)))$ for $k = 0, 1, \ldots$. Further, Line 8 of Algorithm 5 can be solved using a linear program in polynomial time, denoted as f. Combining these arguments, the complexity of Algorithm 5 is $|S_{\mathcal{Z}}| f \max_{(\mathfrak{s},y)}\{\log(1/\mathbf{Q}^0((\mathfrak{s}, y)))/\log(1+\epsilon)\}$.

7. Case Study

In this section, we present a numerical case study on a signalized traffic network. The case study was implemented using MATLAB on a Macbook Pro with a 2.6GHz Intel Core i5 CPU and 8GB RAM.

7.1. Signalized Traffic Network Model

We consider a signalized traffic network [43] consisting of 5 intersections and 12 links under the remote control of a transportation management center (TMC). A representation of the signalized traffic network is shown in Fig. 3. We briefly explain how a DSG from Definition 4 can model the network. Each DSG state models the total number of vehicles on a link in the network. Transitions between the states in the DSG models the flow of vehicles. Since vehicle capacity of a link is finite, the number of states in the DSG will be finite. The defender's action set represents that the TMC can actuate a link by issuing a 'green signal' on outgoing intersections of that link. Conversely, the TMC can block a link by issuing a 'red signal'. The TMC is assumed to control the traffic network over an unreliable wireless channel. Thus, an intelligent adversary can launch man-in-the-middle attacks to tamper with the traffic signal issued by the TMC, or manipulate observations of the TMC. In particular, the adversary can initiate an actuator attack to change the traffic signal and a timing attack to manipulate the time stamped measurement (number of vehicles at each link along with the time index) perceived by the TMC.

The TMC is given one of the following objectives: (i) number of vehicles at link 4 is eventually below 10 before deadline $d = 6$: $\varphi_1 = \Diamond_{[0,6]}(x_4 \leq 10)$; (ii) number of vehicles at links 3 and 4 are eventually below 10 before $d = 6$: $\varphi_2 = \Diamond_{[0,6]}((x_3 \leq 10) \wedge (x_4 \leq 10))$; (iii) number of vehicles at links 3, 4 and 5 are eventually below 10 before $d = 6$: $\varphi_3 = \Diamond_{[0,6]}((x_3 \leq 10) \wedge (x_4 \leq 10) \wedge (x_5 \leq 10))$. Spatial and temporal robustness thresholds are

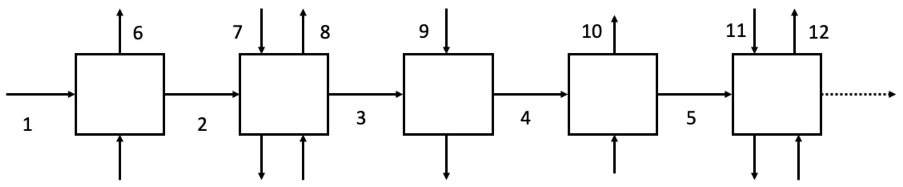

Figure 3. Representation of a *signalized traffic network* consisting of 5 intersections and 12 links.

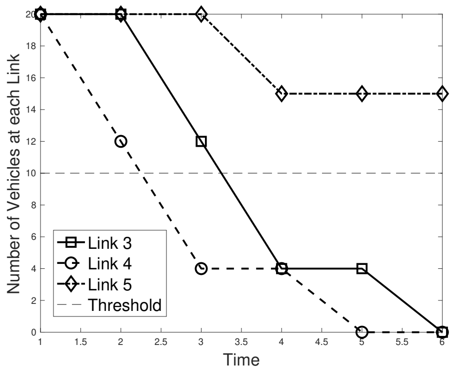

Figure 4. A sample of the number of vehicles on links 3, 4, and 5 over time using our proposed approach. In this realization, the number of links on link 5 is above the threshold.

set to $\epsilon_s = 1$ and $\epsilon_t = 1$. We compare our approach with two baselines. In Baseline 1, the TMC periodically issues green signals. In the second, In Baseline 2, the TMC always issues green signal for links $3, 4, 5$ to greedily minimize the number of vehicles on these links.

7.2. Numerical Results

In the following, we present the numerical results using our proposed approach and the two baselines.

We first report the results when the adversary only launches actuator attack, and the TMC is given specification φ_1. We compute a control policy using Algorithm 4. A sample sequence of traffic signals is presented in Table 2. By Proposition 1 and Corollary 1, the MITL specification φ_1 is satisfied with probability one.

Table 2. Sample sequence of traffic lights realized at each intersection for the MITL specification $\varphi_1 = \Diamond_{[0,6]}(x_4 \leq 10)$. The letters 'R' and 'G' represent 'red' and 'green' signals.

	Intersection				
Time	1	2	3	4	5
1	G	R	R	G	R
2	R	R	G	G	R
3	R	G	G	G	R
4	R	R	R	R	G
5	R	G	G	G	R
6	G	G	G	R	G

We then consider an adversary that launches both actuator and timing attacks. Suppose the TMC is equipped with an FSC with 5 states. We show results of our approach using Algorithm 5 in Fig. 4. In this example, φ_3 is violated since the number of vehicles on link 5 exceeds the threshold 10. We also give the probabilities of satisfying each MITL specification using Algorithm 5. Specifications φ_1, φ_2, and φ_3 are satisfied with probabilities .7000, .6857, and .4390, respectively.

We assume that the TMC commits to deterministic policies in both baselines. In Baseline 1, the adversary launches actuator attacks when the TMC issues a green signal, and does not attack when it issues a red signal. In Baseline 2, the adversary always launches an actuator attack. In both baselines, the adversary launches a timing attack at each time instant to delay the TMC's observation. As a consequence, both baselines have *zero probability* of satisfying φ_1, φ_2, or φ_3.

The DSG in our experiments had 232 states. For φ_1, the GAMEC of the product DSG had 400 states. For φ_2 and φ_3, the GAMEC had 160 and 80 states respectively. The computation time of Algorithm 4 for φ_1 was 264 seconds. Algorithm 5 took 720 seconds.

8. Conclusion and Future Work

In this paper, we proposed methods to synthesize controllers for cyber-physical systems to satisfy metric interval temporal logic (MITL) tasks in the presence of an adversary while additionally providing robustness guarantees. We considered the fragment of MITL formulae that can be represented by deterministic timed Büchi automata. The adversary could initiate actuator and timing attacks. We modeled the interaction between the defender and adversary using a durational stochastic game (DSG). We introduced three notions of robustness degree- spatial robustness, temporal robustness, and spatio-temporal robustness, and presented procedures to estimate these quantities, given the defender and adversary's policies and current state of the DSG. We further presented a computational procedure to synthesize the defender's policy that provided a robustness guarantee when the adversary could only initiate an actuator attack. A value iteration based procedure was given to compute a defender's policy to maximize the probability of satisfying the MITL goal. A case study on a signalized traffic network illustrated our approach.

DSGs can be adopted to model interactions between a defender and adversary across various application domains with time-sensitive constraints. Examples include time-sensitive motion planning of drones, product scheduling of industrial control systems, and time-sensitive message transmission in wireless communication in the presence of adversaries. For future work, we will generalize our definition of DSG to broaden its applications. We will generalize DSG to address partial observations by the CPS and adversary. We will additionally investigate the scenarios where the adversary is nonrational and may not perform its best response to the strategies committed by defender.

Author Contributions: Conceptualization, L.N., B.R., A.C., and R.P.; Methodology, L.N., B.R. A.C., and R.P.; Software, L.N. and B.R.; Validation, B.R.; Formal analysis, L.N., B.R., and A.C.; Writing – original draft, L.N. and B.R.; Writing – review & editing, A.C. and R.P.; Supervision, R.P.; Project administration, R.P..

Funding: This work was supported by the Office of Naval Research grant N00014-20-1-2636, National Science Foundation grants CNS 2153136 and CNS 1941670, and Air Force Office of Scientific Research grants FA9550-20-1-0074 and FA9550-22-1-0054.

Informed Consent Statement: Not applicable.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A Summary of Notations

This appendix summarizes the notations used in this paper, as presented in Table A1.

Table A1. This table gives a list of notation and symbols used in this paper.

Variable Notation	Interpretation
φ	MITL formula
ρ	Timed word
$\mathcal{A}$	Deterministic timed Büchi automaton (DTBA)
$\mathbf{v}$	Clock valuation
β	Run of DTBA
$\mathcal{G}$	Durational stochastic game (DSG)
μ	Defender's policy
τ	Actuator attack policy by the adversary
γ	Timing attack policy by the adversary
$\chi_s^\varphi(\mu,\tau,\gamma)$	Spatial robustness
$\chi_t^\varphi(\mu,\tau,\gamma)$	Temporal robustness
$\chi^\varphi(\mu,\tau,\gamma)$	Spatio-temporal robustness
$\mathcal{P}$	Product durational stochastic game
$\mathcal{F}$	Finite state controller (FSC)
$\mathcal{Z}$	Global durational stochastic game (GDSG)
$\mathcal{C}$	Set of generalized accepting maximal end components (GAMECs)

Appendix B Proofs of Technical Results 591

In this appendix, we present the proofs of all the technical results. 592

Proof of Proposition 1. From Equation (4), $\chi_t^\varphi(\mu,\tau,\gamma)$ is non-negative. If $\chi^\varphi(\mu,\tau,\gamma) > 0$, 593
then $I(\chi_s^\varphi(\mu,\tau,\gamma) \geq \epsilon_s) = 1$, and hence $\chi_s^\varphi(\mu,\tau,\gamma) \geq \epsilon_s > 0$. This implies $\mathcal{B}_{\mathcal{G}}^{\mu\tau\gamma} \subseteq \mathcal{L}$, i.e., 594
all runs obtained under policies μ and (τ,γ) are accepting. This gives $Pr^{\mu\tau\gamma}(\varphi) = 1$, or 595
almost-sure satisfaction of φ under the respective agent policies. □ 596

Proof of Proposition 2. The statement that $Pr((s',q',\mathbf{v}')|(s,q,\mathbf{v}),u_C,u_A) \in [0,1]$ for all transitions in $\mathcal{P}$ follows from the fact that $T_{\mathcal{G}}(\delta|s,u_C,u_A,s') \in [0,1]$ and $Pr_{\mathcal{G}}(s'|s,u_C,u_A) \in [0,1]$. We have that $Pr((s',q',\mathbf{v}')|(s,q,\mathbf{v}),u_C,u_A) = 0$ iff $T_{\mathcal{G}}(\delta|s,u_C,u_A,s') = 0$, or $Pr_{\mathcal{G}}(s'|s,u_C,u_A) = 0$, or both. Moreover, we have that $Pr((s',q',\mathbf{v}')|(s,q,\mathbf{v}),u_C,u_A) = 1$ iff $T_{\mathcal{G}}(\delta|s,u_C,u_A,s') = 1$ and $Pr_{\mathcal{G}}(s'|s,u_C,u_A) = 1$. Let $I^\delta_{(q,\mathbf{v}),(q',\mathbf{v}')} := \mathbb{1}((q,\mathbf{v}) \xrightarrow{L(s'),\delta} (q',\mathbf{v}'))$, which is an indicator function that takes value 1 if its argument is true, and 0 otherwise. Then, Equation (7) can be rewritten as:

$$\sum_{(s',q',\mathbf{v}')} T_{\mathcal{G}}(\delta|s,u_C,u_A,s')Pr_{\mathcal{G}}(s'|s,u_C,u_A)$$
$$= \sum_{s'}\sum_{\delta} T_{\mathcal{G}}(\delta|s,u_C,u_A,s')I^\delta_{(q,\mathbf{v}),(q',\mathbf{v}')}Pr_{\mathcal{G}}(s'|s,u_C,u_A) \quad \text{(A1)}$$

This follows from substitution from Equation (6) and product DSG in Definition 8. The 597
result follows by $\sum_{s'\in S_{\mathcal{G}}} Pr_{\mathcal{G}}(s'|s,u_C,u_A) = 1$ and $\sum_{\delta\in\Delta} T_{\mathcal{G}}(\delta|s,u_C,u_A,s') = 1$. □ 598

Proof of Proposition 3. We proceed by showing that each loop in Algorithm 1 is executed 599
a finite number of times. The PDSG $\mathcal{P}$ has a finite number of states and actions since DSG 600
$\mathcal{G}$ has a finite number of states and actions, the DTBA $\mathcal{A}$ has a finite number of states, and 601
the clock valuation set V bounded due to the boundedness of time interval I. Therefore, 602
the *for-loops* in *Line 7, 10, 11, and 27* are executed for finite number of times. The *while-loop* 603
in *Line 18* is executed a finite number of times since $R \subseteq S$ is a finite set. Moreover, there 604
are finite number of states that will be added to R (*Line 14*), and this will be carried out 605
finitely many times. The overall complexity is $O(|V|(|V|+|E|))$, where $|V|$ and $|E|$ are the 606
number of vertices and edges in $\mathcal{P}$. □ 607

Proof of Theorem 1. We leverage the recursive robust MITL semantics to prove the theo- 608
rem and consider the following cases. 609

Case 1— $\varphi = \pi \in \Pi$: In this case, the temporal robustness is computed by *Lines 4–7* of Algorithm 2: $\textsc{Temporal}(\varphi, \mathfrak{s}_0, \delta, M(\mathfrak{s}')) = \min\{left\_temp, right\_temp\} = \epsilon > 0$,. This means there must exist a state $\mathfrak{s}''$ that is reachable from $\mathfrak{s}$ under policies μ and τ such that $\mathfrak{s}'' \models \pi$. Without loss of generality, we assume $\textsc{Temporal}(\varphi, \mathfrak{s}_0, \delta, M(\mathfrak{s}')) = Time(\mathfrak{s}'') - Time(\mathfrak{s}_0) = \epsilon$. Since $Time(\mathfrak{s}_0) = 0$, we have $Time(\mathfrak{s}'') = \epsilon$, i.e., ϵ is the time index of state $\mathfrak{s}''$. Therefore, a shift to the left by $\hat{\epsilon} \in [0, \epsilon]$ will not affect the satisfaction of π since $\pi \in \mathcal{L}(\mathfrak{s}'')$ holds true independent of time. If the accepting run is temporally perturbed by more than ϵ time units, the clock valuation becomes negative. This contradicts our assumption that clock valuations take positive values.

Case 2— $\varphi = \phi_1 \wedge \phi_2$: Consider *Lines 8–11* of Algorithm 2. Suppose $\phi_1, \phi_2 \in \Pi$. Let $\textsc{Temporal}(\varphi, \mathfrak{s}_0, \delta, M(\mathfrak{s}')) = \textsc{Temporal}(\phi_1, \mathfrak{s}_0, \delta, M(\mathfrak{s}')) = \epsilon > 0$. From *Line 11*, it follows that $\textsc{Temporal}(\phi_2, \mathfrak{s}_0, \delta, M(\mathfrak{s}')) := \epsilon' > \epsilon$. Since $\phi_1, \phi_2 \in \Pi$, we can apply Case 1 to $\textsc{Temporal}(\phi_1, \mathfrak{s}_0, \delta, M(\mathfrak{s}'))$ and $\textsc{Temporal}(\phi_2, \mathfrak{s}_0, \delta, M(\mathfrak{s}'))$. Therefore, if we shift the run synthesized under policies μ and τ by $\hat{\epsilon} \in [0, \epsilon]$ time units to the left, ϕ_1 will still be satisfied. Moreover, since $\epsilon < \epsilon'$, ϕ_2 will also be satisfied. Hence, $\phi = \phi_1 \wedge \phi_2$ will still be satisfied if we shift the run synthesized under policies μ and τ by at most $\hat{\epsilon} < \epsilon$ time units.

Case 3 — $\varphi = \phi_1 \vee \phi_2$: Consider *Lines 12–15*. Suppose $\phi_1, \phi_2 \in \Pi$. Let $\textsc{Temporal}(\phi_1, \mathfrak{s}_0, \delta, M(\mathfrak{s}')) = \epsilon > 0$. From Case 1, we can shift any accepting run starting from $\mathfrak{s}_0$ by at most ϵ time units without violating ϕ_1. Then by semantics of the disjunction operator, $\varphi = \phi_1 \vee \phi_2$ is also satisfied when the accepting run is shifted by at most ϵ time units.

Case 4 — $\varphi = \phi_1 \mathcal{U}_I \phi_2$: In this case, the temporal robustness is computed by *Lines 16–27*. We consider the case that $\phi_1, \phi_2 \in \Pi$. Let $t := \inf\{t' | \phi_2 \text{ is satisfied at } t'\}$.

If $t = 0$, ϕ_2 is satisfied at state $\mathfrak{s}_0$, and hence φ is satisfied at $\mathfrak{s}_0$. Therefore $\mathfrak{s}_0$ is in GAMEC. From the definition of GAMEC, we have that $\mathfrak{s}_0$ and its neighboring states are in GAMEC (the defender does not take any action that steers the PDSG outside GAMEC), and hence $M(\mathfrak{s}') = 1$. Thus Algorithm 2 will execute *Lines 19–20*. We have that $r_1 \leftarrow \min \bigcup_{\mathfrak{s}'} \{(Time(\mathfrak{s}') - \underline{I}), (\overline{I} - Time(\mathfrak{s}'))\}$, where $\overline{I} = \sup\{t' | t' \in I\}$ and $\underline{I} = \inf\{t' | t' \in I\}$ are upper and lower bounds of I. Since $t = 0$, *Line 23* will be executed. $\phi_2 \in \Pi$ indicates that $r_2 = \textsc{Temporal}(\phi_2, \mathfrak{s}_0, \delta, M(\mathfrak{s}'))$ can be obtained from *Lines 4–7*. This gives $r_1 = r_2 = 0$, and hence $\epsilon = 0$. We remark that this only indicates that we cannot shift the accepting run to the left temporally without violating φ. Shifting the run to the right might not lead to violation of φ. However, since the temporal robustness is defined as the minimum of left and right temporal robustness, the algorithm returns $\epsilon = 0$.

If $t > 0$, from the semantics of time constrained until operator $\mathcal{U}_I$, ϕ_1 is satisfied up to time $t \in I$ and ϕ_2 is satisfied immediately after time t, and thus φ is satisfied. Therefore, we will eventually reach some accepting state so that $M(\mathfrak{s}') = 1$ for some $\mathfrak{s}'$. In this case, $\epsilon = \max\{r_1, r_2\}$, where r_1 is given in *Line 20* and r_2 is given in *Lines 22–26* of Algorithm 2. Suppose $\epsilon = r_1$. From *Line 27*, we must have $r_1 \geq r_2$. From *Line 20*, $r_1 = \min\{t - \underline{I}, \overline{I} - t\} = \epsilon$. Thus, we can shift any accepting run by at most $t - \underline{I}$ time units to the left without violating φ if $\epsilon = t - \underline{I}$. After the perturbation, ϕ_1 is satisfied at time $\underline{I}$ and ϕ_2 is satisfied immediately after $\underline{I}$. The case where $\epsilon = \overline{I} - t$ can be obtained analogously. Suppose $\epsilon = r_2$. From *Lines 22–26*, $r_2 = \textsc{Temporal}(\phi_2, \mathfrak{s}_0, \delta, M(\mathfrak{s}'))$. Since $\phi_2 \in \Pi$, r_2 can be obtained from *Lines 4–7*. Recall that we consider a bounded clock valuation set. Let $\overline{V} := \sup\{t | t \in I\} = \overline{I}$. Then r_2 models the maximum distance between the time index at which ϕ_2 is satisfied and the upper bound of I. From Case 1, we have that perturbing an accepting run by at most ϵ time units will not violate φ since the run obtained after perturbation satisfies ϕ_2 at the boundary of I.

Case 5 — ϕ_1 and ϕ_2 in Cases 2-4 are MITL formulae: In this case, we can apply the previous analyses using the recursive definition of MITL formula. □

Proof of Theorem 2. We prove the theorem in the following way. At each iteration within the while loop (starting at *Line 5*), Algorithm 4 executes one of the three cases of the if-else statement (*Lines 7, 9, or 17*), with each case corresponding to the satisfaction of spatio-temporal robustness constraint, violation of temporal robustness constraint, or violation of spatial robustness constraint. We denote the execution of *Line 7* as Scenario I, that of *Line 9* as Scenario II, and of *Line 17* as Scenario III. We will show that Algorithm 4 reaches Scenario I at most once, and reaches Scenarios II and III finitely many times. If Algorithm 4 reaches Scenario I, it terminates (*Line 8*). For Scenarios II and III, we will show that there exists an index k such that if Algorithm 4 reaches Scenario II or III at iteration k, then Scenario I will be executed at iteration $k+1$ and hence terminates or *Lines 26-29* will be executed and the process will terminate at iteration k.

Scenario I — executing Line 7: Suppose Algorithm 4 reaches Scenario I at iteration k. In this case, the control policy μ^k satisfies the spatio-temporal robustness constraints. By *Line 8* we have that Scenario I is reached exactly once and hence Algorithm 4 terminates.

Scenario II — executing Line 9: Suppose Algorithm 4 reaches Scenario II at iteration k. In this case, the policy μ^k satisfies the spatial robustness constraint but violates the temporal robustness constraint. Let $\mathfrak{s}$ be the state that results in temporal robustness constraint violation, and let $\mathfrak{s}'$ be a neighboring state of $\mathfrak{s}$. We decompose our discussion into the following cases:

1. Suppose $U_C(\mathfrak{s}') = \emptyset$. In this case, state $\mathfrak{s}'$ is included in set $\mathcal{E}_t$. If adding $\mathfrak{s}'$ to $\mathcal{E}_t$ makes states in GAMEC not reachable from $\mathfrak{s}_0$, then Algorithm 4 executes *Lines 26-29*, and terminates by reporting failure.
2. Suppose $U_C(\mathfrak{s}') \neq \emptyset$. However, the remaining control actions $u_C \in U_C(\mathfrak{s}')$ cannot make GAMEC reachable from the initial state $\mathfrak{s}_0$. In this case, Algorithm 4 will execute *Lines 26-29*, and terminates.
3. Suppose $U_C(\mathfrak{s}') \neq \emptyset$, and GAMEC is reachable from $\mathfrak{s}_0$. We further assume that all actions $u_C \in U_C(\mathfrak{s}')$ that are admissible by the policy generated at *Line 25* result in robustness greater than or equal to ϵ_t. As a consequence, the remaining control actions in $U_C(\mathfrak{s}')$ must steer the system into some neighboring state $\mathfrak{s}''$ of $\mathfrak{s}'$ such that $\chi^\varphi(\mu, \tau, \gamma, \mathfrak{s}'') > \epsilon_t$. Therefore, Algorithm 4 will execute Scenario I at iteration $k+1$, and thus terminates.
4. Suppose $U_C(\mathfrak{s}') \neq \emptyset$, and GAMEC is reachable from the initial state $\mathfrak{s}_0$. Now assume that there exists some action $u_C \in U_C(\mathfrak{s}')$ such that it is admissible by the policy generated at *Line 25* and results in robustness below ϵ_t for some neighboring state $\mathfrak{s}''$ of $\mathfrak{s}$. In this case, this u_C will be removed according *Line 12* at iteration $k+1$. Since there are only finitely many states and control actions, this case will converge to one of the cases discussed in (1), (2) or (3) in a finite number of iterations.

Scenario III — executing Line 17: Suppose Algorithm 4 reaches Scenario III at iteration k. In this case, the control policy μ^k violates the spatial robustness constraint. We use $\mathfrak{s}$ to denote the state that violates spatial robustness constraint, and use $\mathfrak{s}'$ to denote the neighboring state of $\mathfrak{s}$. We analyze Scenario III by dividing our discussion into the following cases.

1. Suppose $U_C(\mathfrak{s}') = \emptyset$. From *Line 18*, $\mathfrak{s}'$ is included in set $\mathcal{E}_s$. If adding $\mathfrak{s}'$ to $\mathcal{E}_s$ makes states in GAMEC not reachable from $\mathfrak{s}_0$, then Algorithm 4 executes *Lines 26-29*, and terminates by reporting failure.
2. Suppose $U_C(\mathfrak{s}') \neq \emptyset$ and GAMEC is not reachable from the $\mathfrak{s}_0$ for all $u_C \in U_C(\mathfrak{s}')$. In this case, Algorithm 4 will execute *Lines 26-29*, and terminate.
3. Suppose $U_C(\mathfrak{s}') \neq \emptyset$, and GAMEC is reachable from $\mathfrak{s}_0$. Assume that all actions $u_C \in U_C(\mathfrak{s}')$ that are admissible by the policy generated at *Line 25* result in robustness $\geq \epsilon_t$. In this case, the game must be steered to a neighboring state $\mathfrak{s}''$ of $\mathfrak{s}'$ such that $\chi^\varphi(\mu, \tau, \gamma, \mathfrak{s}'') > \epsilon_t$. Then, Algorithm 4 will execute Scenario I at iteration $k+1$, and terminate.
4. Suppose $U_C(\mathfrak{s}') \neq \emptyset$, and GAMEC is reachable from $\mathfrak{s}_0$. Now assume that the policy generated at *Line 25* results in robustness below ϵ_t for some neighboring state $\mathfrak{s}''$ of $\mathfrak{s}$. In this case, the control action u_C will be removed according *Lines 12 and 20* at iteration $k+1$. Since there are only finitely many states and control actions, this case will converge to one of the cases discussed in (1), (2) or (3) in a finite number of iterations.

From the preceding discussion, the control action set U_C will converge to a set $\hat{U}_C$ that will never lead Algorithm 4 to Scenarios II or III. In the worst case, $\hat{U}_C = \emptyset$ when there will be at most $|S| \times |U_C|$ actions being removed due to Scenarios II and III, leading Algorithm 4 to *Line 28* where it terminates by reporting failure. Therefore, Algorithm 4 converges to a set $\hat{U}_C$ that will never cause violations of the robustness constraints and the game can be driven to GAMEC in a finite number of iterations. If no such set exists, it terminates by reporting failure. If $\hat{U}_C \neq \emptyset$, then Algorithm 4 returns a policy over $\hat{U}_C$. $\square$

Proof of Theorem 3. Suppose Algorithm 4 returns a policy μ^*. From Theorem 2, μ^* is defined over $\hat{U}_C \neq \emptyset$ (otherwise μ^* should not be returned by Algorithm 4 since no admissible defender action is available). From *Lines 10-16* in Algorithm 4, the defender's policy μ^* will not result in temporal robustness below ϵ_t. From *Lines 17-23*, μ^* guarantees positive spatio-temporal robustness. Therefore, if μ^* is returned by Algorithm 4, we must have spatio-temporal robustness $\chi^{\varphi}(\mu^*, \tau^*, \gamma^*, \mathfrak{s}) \geq \epsilon_t$, where (τ^*, γ^*) are best responses of the adversary. Thus, μ^* is a feasible solution to robust policy synthesis for defender in Problem 1. From Proposition 1, the probability of satisfying the MITL formula φ equals 1, which is the maximum value that can be achieved for any control policy. Therefore, μ^* is an optimal policy. $\square$

References

1. Baheti, R.; Gill, H. Cyber-physical systems. *The Impact of Control Technology* **2011**, *12*, 161–166. https://doi.org/10.1109/ICMECH.2019.8722929.
2. Baier, C.; Katoen, J.P.; Larsen, K.G. *Principles of Model Checking*; MIT Press, 2008.
3. Alur, R.; Dill, D.L. A theory of timed automata. *Theoretical Computer Science* **1994**, *126*, 183–235. https://doi.org/10.1016/0304-3975(94)90010-8.
4. Kress-Gazit, H.; Fainekos, G.E.; Pappas, G.J. Temporal-logic-based reactive mission and motion planning. *IEEE Transactions on Robotics* **2009**, *25*, 1370–1381. https://doi.org/10.1109/TRO.2009.2030225.
5. Ding, X.; Smith, S.L.; Belta, C.; Rus, D. Optimal control of Markov decision processes with linear temporal logic constraints. *IEEE Transactions on Automatic Control* **2014**, *59*, 1244–1257. https://doi.org/10.1109/TAC.2014.2298143.
6. Zhou, Y.; Maity, D.; Baras, J.S. Timed automata approach for motion planning using metric interval temporal logic. In Proceedings of the European Control Conference. IEEE, 2016, pp. 690–695. https://doi.org/10.1109/ECC.2016.7810369.
7. Fu, J.; Topcu, U. Computational methods for stochastic control with metric interval temporal logic specifications. In Proceedings of the Conference on Decision and Control. IEEE, 2015, pp. 7440–7447. https://doi.org/10.1109/CDC.2015.7403395.
8. Fainekos, G.E.; Pappas, G.J. Robustness of temporal logic specifications for continuous-time signals. *Theoretical Computer Science* **2009**, *410*, 4262–4291. https://doi.org/10.1016/j.tcs.2009.06.021.
9. Donzé, A.; Maler, O. Robust satisfaction of temporal logic over real-valued signals. In Proceedings of the International Conference on Formal Modeling and Analysis of Timed Systems. Springer, 2010, pp. 92–106. https://doi.org/https://doi.org/10.1007/978-3-642-15297-9_9.
10. Niu, L.; Clark, A. Optimal Secure Control with Linear Temporal Logic Constraints. *IEEE Transactions on Automatic Control* **2020**, *65*. https://doi.org/10.1109/TAC.2019.2930039.
11. Zhu, M.; Martinez, S. Stackelberg-game analysis of correlated attacks in cyber-physical systems. In Proceedings of the American Control Conference. IEEE, 2011, pp. 4063–4068. https://doi.org/10.1109/ACC.2011.5991463.
12. Wang, J.; Tu, W.; Hui, L.C.; Yiu, S.M.; Wang, E.K. Detecting time synchronization attacks in cyber-physical systems with machine learning techniques. In Proceedings of the International Conference on Distributed Computing Systems. IEEE, 2017, pp. 2246–2251. https://doi.org/10.1109/ICDCS.2017.25.
13. Jewell, W.S. Markov-renewal programming: Formulation, finite return models. *Operations Research* **1963**, *11*, 938. https://doi.org/10.1287/opre.11.6.938.
14. Ross, S.M. *Introduction to Stochastic Dynamic Programming*; Academic Press, 2014.
15. Stidham, S.; Weber, R. A survey of Markov decision models for control of networks of queues. *Queueing Systems* **1993**, *13*, 291–314. https://doi.org/10.1007/BF01158935.
16. Leitmann, G. On generalized Stackelberg strategies. *Journal of Optimization Theory and Applications* **1978**, *26*, 637–643. https://doi.org/10.1142/S0129054103002114.
17. Wei, L.; Sarwat, A.I.; Saad, W.; Biswas, S. Stochastic games for power grid protection against coordinated cyber-physical attacks. *IEEE Transactions on Smart Grid* **2016**, *9*, 684–694. https://doi.org/10.1109/TSG.2016.2561266.
18. Garnaev, A.; Baykal-Gursoy, M.; Poor, H.V. A game theoretic analysis of secret and reliable communication with active and passive adversarial modes. *IEEE Transactions on Wireless Communications* **2015**, *15*, 2155–2163. https://doi.org/10.1109/TWC.2015.2498934.

19. Bouyer, P.; Laroussinie, F.; Markey, N.; Ouaknine, J.; Worrell, J. Timed temporal logics. In *Models, Algorithms, Logics and Tools*; Springer, 2017; pp. 211–230. https://doi.org/https://doi.org/10.1007/978-3-319-63121-9_11.
20. Alur, R.; Feder, T.; Henzinger, T.A. The benefits of relaxing punctuality. *Journal of the ACM* **1996**, *43*, 116–146. https://doi.org/10.1145/227595.227602.
21. Maler, O.; Nickovic, D.; Pnueli, A. From MITL to timed automata. In Proceedings of the International Conference on Formal Modeling and Analysis of Timed Systems. Springer, 2006, pp. 274–289. https://doi.org/10.1007/11867340_20.
22. Karaman, S.; Frazzoli, E. Vehicle routing problem with metric temporal logic specifications. In Proceedings of the Conference on Decision and Control. IEEE, 2008, pp. 3953–3958. https://doi.org/10.1109/CDC.2008.4739366.
23. Liu, J.; Prabhakar, P. Switching control of dynamical systems from metric temporal logic specifications. In Proceedings of the International Conference on Robotics and Automation. IEEE, 2014, pp. 5333–5338. https://doi.org/10.1109/ICRA.2014.6907643.
24. Nikou, A.; Tumova, J.; Dimarogonas, D.V. Cooperative task planning of multi-agent systems under timed temporal specifications. In Proceedings of the American Control Conference. IEEE, 2016, pp. 7104–7109. https://doi.org/10.1109/ACC.2016.7526793.
25. Hansen, E.A. Solving POMDPs by searching in policy space. In Proceedings of the Conference on Uncertainty in Artificial Intelligence, 1998, pp. 211–219. https://doi.org/10.5555/2074094.2074119.
26. Sharan, R.; Burdick, J. Finite state control of POMDPs with LTL specifications. In Proceedings of the American Control Conference. IEEE, 2014, p. 501. https://doi.org/10.1109/ACC.2014.6858909.
27. Ramasubramanian, B.; Clark, A.; Bushnell, L.; Poovendran, R. Secure control under partial observability with temporal logic constraints. In Proceedings of the American Control Conference. IEEE, 2019, pp. 1181–1188. https://doi.org/10.23919/ACC.2019.8814630.
28. Ramasubramanian, B.; Niu, L.; Clark, A.; Bushnell, L.; Poovendran, R. Secure control in partially observable environments to satisfy LTL specifications. *IEEE Transactions on Automatic Control* **2021**, *66*, 5665–5679. https://doi.org/doi:10.1109/TAC.2020.3039484.
29. Zhao, G.; Li, H.; Hou, T. Input–output dynamical stability analysis for cyber-physical systems via logical networks. *IET Control Theory & Applications* **2020**, *14*, 2566–2572. https://doi.org/10.1049/iet-cta.2020.0197.
30. Zhao, G.; Li, H. Robustness analysis of logical networks and its application in infinite systems. *Journal of the Franklin Institute* **2020**, *357*, 2882–2891. https://doi.org/https://doi.org/10.1016/j.jfranklin.2019.12.002.
31. Simon, D. *Optimal State Estimation: Kalman, H infinity, and Nonlinear Approaches*; John Wiley & Sons, 2006.
32. Angeli, D. A Lyapunov approach to incremental stability properties. *IEEE Transactions on Automatic Control* **2002**, *47*, 410–421. https://doi.org/10.1109/9.989067.
33. Rizk, A.; Batt, G.; Fages, F.; Soliman, S. A general computational method for robustness analysis with applications to synthetic gene networks. *Bioinformatics* **2009**, *25*, i169–i178. https://doi.org/10.1093/bioinformatics/btp200.
34. Jakšić, S.; Bartocci, E.; Grosu, R.; Nguyen, T.; Ničković, D. Quantitative monitoring of STL with edit distance. *Formal Methods in System Design* **2018**, *53*, 83–112. https://doi.org/10.1007/s10703-018-0319-x.
35. Aksaray, D.; Jones, A.; Kong, Z.; Schwager, M.; Belta, C. Q-learning for robust satisfaction of signal temporal logic specifications. In Proceedings of the Conference on Decision and Control. IEEE, 2016, pp. 6565–6570. https://doi.org/10.1109/CDC.2016.7799279.
36. Lindemann, L.; Dimarogonas, D.V. Robust control for signal temporal logic specifications using discrete average space robustness. *Automatica* **2019**, *101*, 377–387. https://doi.org/10.1016/j.automatica.2018.12.022.
37. Rodionova, A.; Lindemann, L.; Morari, M.; Pappas, G. Temporal robustness of temporal logic specifications: Analysis and control design. *ACM Transactions on Embedded Computing Systems* **2022**, *22*, 1–44. https://doi.org/10.1145/3550072.
38. Rodionova, A.; Lindemann, L.; Morari, M.; Pappas, G.J. Combined left and right temporal robustness for control under STL specifications. *IEEE Control Systems Letters* **2022**, *7*, 619–624. https://doi.org/10.1109/LCSYS.2022.3209928.
39. Niu, L.; Ramasubramanian, B.; Clark, A.; Bushnell, L.; Poovendran, R. Control Synthesis for Cyber-Physical Systems to Satisfy Metric Interval Temporal Logic Objectives under Timing and Actuator Attacks. In Proceedings of the International Conference on Cyber-Physical Systems. ACM/ IEEE, 2020, pp. 162–173. https://doi.org/10.1109/ICCPS48487.2020.00023.
40. Ouaknine, J.; Worrell, J. Some recent results in metric temporal logic. In Proceedings of the International Conference on Formal Modeling and Analysis of Timed Systems. Springer, 2008, pp. 1–13. https://doi.org/10.1007/978-3-540-85778-5_1.
41. Levenshtein, V.I. Binary codes capable of correcting deletions, insertions, and reversals. In Proceedings of the Soviet Physics Doklady, 1966, Vol. 10, pp. 707–710.
42. Mohri, M. Edit-distance of weighted automata: General definitions and algorithms. *International Journal of Foundations of Computer Science* **2003**, *14*, 957–982. https://doi.org/10.1142/S0129054103002114.
43. Coogan, S.; Gol, E.A.; Arcak, M.; Belta, C. Traffic network control from temporal logic specifications. *IEEE Transactions on Control of Network Systems* **2015**, *3*, 162–172. https://doi.org/10.1109/TCNS.2015.2428471.