Secure Control in Partially Observable Environments to Satisfy LTL Specifications*,+

Bhaskar Ramasubramanian[1], Luyao Niu[2], Andrew Clark[2], Linda Bushnell[1], *Fellow, IEEE*, and Radha Poovendran[1], *Fellow, IEEE*

***Abstract*—This paper studies the synthesis of control policies for an agent that has to satisfy a temporal logic specification in a partially observable environment, in the presence of an adversary. The interaction of the agent (defender) with the adversary is modeled as a partially observable stochastic game. The goal is to generate a defender policy to maximize satisfaction of a given temporal logic specification under any adversary policy. The search for policies is limited to the space of finite state controllers, which leads to a tractable approach to determine policies. We relate the satisfaction of the specification to reaching (a subset of) recurrent states of a Markov chain. We present an algorithm to determine a set of defender and adversary finite state controllers of fixed sizes that will satisfy the temporal logic specification, and prove that it is sound. We then propose a value-iteration algorithm to maximize the probability of satisfying the temporal logic specification under finite state controllers of fixed sizes. Lastly, we extend this setting to the scenario where the size of the finite state controller of the defender can be increased to improve the satisfaction probability. We illustrate our approach with an example.**

***Index Terms*—linear temporal logic (LTL), partially observable stochastic games (POSGs), finite state controllers (FSCs), Stackelberg equilibrium, value iteration, policy iteration, global Markov chain (GMC)**

I. Introduction

Cyber-physical systems (CPSs) are entities in which the working of a physical system is governed by its interactions with computing devices and algorithms. These systems are ubiquitous [2], and vary in scale from energy systems to medical devices and robots. In applications like autonomous cars and robotics, CPSs are expected to operate in dynamic and potentially dangerous environments with a large degree of autonomy. In such a setting, the system might be the target of malicious attacks that aim to prevent it from accomplishing a goal. An attack can be carried out on the physical system, on the computers that control the physical system, or on communication channels between components of the system. Such attacks by an intelligent attacker have been reported across multiple domains, including power systems [3], automobiles [4], water networks [5], and nuclear reactors [6]. Adversaries are often stealthy, and tailor their attacks to cause maximum damage. Therefore, strategies designed to only address modeling and sensing errors may not satisfy performance requirements in the presence of an intelligent adversary who can manipulate system operation.

The preceding discussion makes it imperative to develop methods to specify and verify CPSs and the environments they operate in. Formal methods [7] enable the verification of the behavior of CPS models against a rich set of specifications [8]. Properties like safety, liveness, stability, and priority can be expressed as formulas in linear temporal logic (LTL) [9], [10], and can be verified using off-the-shelf model solvers [11], [12] that take these formulas as inputs. Markov decision processes (MDPs) [13], [14] have been used to model environments where outcomes depend on both, an inherent randomness in the model (transition probabilities) and an action taken by an agent. These models have been extensively used in applications like robotics [15] and unmanned aircrafts [16].

Current literature on the satisfaction of an LTL formula over an MDP assumes that states are fully observable [10], [15], [17]. In many practical scenarios, states may not be observable. For example, as seen in [18], a robot might only have an estimate of its location based on the output of a vision sensor. The inability to observe all states necessitates the use of a framework that accounts for partial observability. For LTL formula satisfaction in partially observable environments with a single agent, partially-observable Markov decision processes (POMDPs) can be used to model and solve the problem [19], [20]. However, determining an 'optimal policy' for an agent in a partially observable environment is NP-hard for the infinite horizon case, which was shown in [21]. This demonstrates the need for techniques to determine approximate solutions.

Heuristics to approximately solve POMDPs include belief replanning [22], most likely belief state policy and entropy weighting [23], grid-based methods [24], and point-based methods [25]. The difficulty in computing exactly optimal policies and the lack of complete observability may be exploited by an adversary to launch new attacks on the system. The synthesis of parameterized finite state controllers (FSCs) for a POMDP to maximize the probability of satisfying of an LTL formula (in the absence of an adversary) was proposed in [19] and [20]. This

*This work was supported by the U.S. Army Research Office, the National Science Foundation, and the Office of Naval Research via Grants W911NF-16-1-0485, CNS-1941670, and N00014-17-S-B001 respectively.

+A preliminary version of this work appears in [1].

[1]Network Security Lab, Department of Electrical and Computer Engineering, University of Washington, Seattle, WA 98195, USA.
{bhaskarr, lb2, rp3}@uw.edu

[2]Department of Electrical and Computer Engineering, Worcester Polytechnic Institute, Worcester, MA 01609, USA.
{lniu, aclark}@wpi.edu

is an approximate strategy since it does not use the observation and action histories; it uses only the most recent observation in order to determine an action. This restricts the class of policies that are searched over, but the finite cardinality of states in an FSC makes the problem computationally tractable. The authors of [26] showed the existence of $\epsilon-$optimal FSCs for the average cost POMDP. In comparison, for the setting in this paper where we have two competing agents, we present guarantees on the convergence of a value-iteration based procedure in terms of the number of states in the environment and the FSCs.

In this paper, we study the problem of determining strategies for an agent that has to satisfy an LTL formula in the presence of an adversary in a partially observable environment. The agent and the adversary take actions simultaneously, and these jointly influence transitions between states.

A. Contributions

The setting that we consider in this paper assumes two players or agents– a defender and an adversary– who are each limited in that they do not exactly observe the state. The policies of the agents are represented as FSCs. The goal for the defender will be to synthesize a policy that will maximize the probability of satisfying an LTL formula for any adversary policy. We make the following contributions.

- We show that maximizing the satisfaction probability of the LTL formula under any adversary policy is equivalent to maximizing the probability of reaching a recurrent set of a Markov chain constructed by composing representations of the environment, the LTL objective, and the respective agents' controllers.
- We develop a heuristic algorithm to determine defender and adversary FSCs of fixed sizes that will satisfy the LTL formula with nonzero probability, and show that it is sound. The search for a defender policy that will maximize the probability of satisfaction of the LTL formula for any adversary policy can then be reduced to a search among these FSCs.
- We propose a procedure based on value-iteration that maximizes the probability of satisfying the LTL formula under fixed defender and adversary FSCs. This satisfaction probability is related to a Stackelberg equilibrium of a partially observable stochastic game involving the defender and adversary. We also give guarantees on the convergence of this procedure.
- We study the case when the size of the defender FSC can be changed to improve the satisfaction probability.
- We present an example to illustrate our approach.

The value-iteration procedure and the varying defender FSC size described above is new to this work, along with more detailed examples. This differentiates the present paper from a preliminary version that appears in [1].

B. Outline

An overview of LTL and partially observable stochastic games (POSGs) is given in Section II. We define FSCs for the two agents, and show how they can be composed with a POSG to yield a Markov chain in Section III. Section IV relates LTL satisfaction on a POSG to reaching specific subsets of recurrent sets of an associated Markov chain. Section V gives a procedure to determine defender and adversary FSCs of fixed sizes that will ensure that the LTL formula will be satisfied with non-zero probability. A value-iteration procedure to maximize the probability of satisfying the LTL formula under fixed defender and adversary FSCs is detailed in Section VI. Section VII addresses the scenario when states may be added to the defender FSC in order to improve the probability of satisfying the LTL formula under an adversary FSC of fixed size. Illustrative examples are presented in Section VIII. Section IX summarizes related work in POMDPs and TL satisfaction on MDPs. Section X concludes the paper.

II. Preliminaries

In this section, we give a concise introduction to linear temporal logic and partially observable stochastic games. We then detail the construction of an entity which will ensure that runs on a POSG will satisfy an LTL formula.

A. Linear Temporal Logic

Temporal logic frameworks enable the representation and reasoning about temporal information on propositional statements. *Linear temporal logic (LTL)* is one such framework, where the progress of time is 'linear'. An *LTL formula* [7] is defined over a set of atomic propositions $\mathscr{AP}$, and can be written as: $\phi := \mathtt{T} | \sigma | \neg\phi | \phi \wedge \phi | \mathbf{X}\phi | \phi \mathbf{U} \phi$, where $\sigma \in \mathscr{AP}$, and $\mathbf{X}$ and $\mathbf{U}$ are temporal operators denoting the *next* and *until* operations respectively.

The semantics of LTL are defined over (infinite) words in $2^{\mathscr{AP}}$. We write $\eta_0\eta_1\cdots := \eta \models \phi$ when a trace $\eta \in (2^{\mathscr{AP}})^\omega$ satisfies an LTL formula ϕ. Here, the superscript ω serves to indicate the potential infinite length of the word[1]

Definition 2.1 (LTL Semantics): Let $\eta^i = \eta_i\eta_{i+1}\ldots$. Then, the semantics of LTL can be recursively defined as:

1) $\eta \models \mathtt{T}$ if and only if (iff) η_0 is true;
2) $\eta \models \sigma$ iff $\sigma \in \eta_0$;
3) $\eta \models \neg\phi$ iff $\eta \not\models \phi$;
4) $\eta \models \phi_1 \wedge \phi_2$ iff $\eta \models \phi_1$ and $\eta \models \phi_2$;
5) $\eta \models \mathbf{X}\phi$ iff $\eta^1 \models \phi$;
6) $\eta \models \phi_1 \mathbf{U} \phi_2$ iff $\exists j \geq 0$ such that $\eta^j \models \phi_2$ and for all $k < j, \eta^k \models \phi_1$.

Moreover, the logic admits derived formulas of the form: *i)* $\phi_1 \vee \phi_2 := \neg(\neg\phi_1 \wedge \neg\phi_2)$; *ii)* $\phi_1 \Rightarrow \phi_2 := \neg\phi_1 \vee \phi_2$; *iii)* $\mathbf{F}\phi := \mathtt{T}\mathbf{U}\phi$ (eventually); *iv)* $\mathbf{G}\phi := \neg\mathbf{F}\neg\phi$ (always).

Definition 2.2 (Deterministic Rabin Automaton): A *deterministic Rabin automaton (DRA)* is a quintuple $\mathscr{RA} = (Q, \Sigma, \delta, q_0, F)$ where Q is a nonempty finite set of states, Σ is a finite alphabet, $\delta : Q \times \Sigma \to Q$ is a transition function, $q_0 \in Q$ is the initial state, and $F := \{(L(i), K(i)\}_{i=1}^M$ is such that $L(i), K(i) \subseteq Q$ for all i, and M is a positive integer.

[1]To be more precise, η is a word in an *$\omega-$regular language*, which is a generalization of regular languages to words of infinite length [7].

A *run* of $\mathscr{RA}$ is an infinite sequence of states $q_0q_1\ldots$ such that $q_i \in \delta(q_{i-1},\alpha)$ for all i and for some $\alpha \in \Sigma$. The run is *accepting* if there exists $(L,K) \in F$ such that the run intersects with L finitely many times, and with K infinitely often. An LTL formula ϕ over $\mathscr{AP}$ can be represented by a DRA with alphabet $2^{\mathscr{AP}}$ that accepts all and only those runs that satisfy ϕ.

B. Stochastic Games and Markov Chains

A stochastic game involves one or more players, and starts with the system in a particular state. Transitions to a subsequent state are probabilistically determined by the current state and the actions chosen by each player, and this process is repeated. Our focus will be on two-player stochastic games, and we omit the quantification on the number of players for the remainder of this paper.

Definition 2.3 (Stochastic Game): A *stochastic game* [17] is a tuple $\mathscr{G} := (S,s_0,U_d,U_a,\mathbb{T},\mathscr{AP},\mathscr{L})$. S is a finite set of states, s_0 is the initial state, U_d and U_a are finite sets of actions of the defender and adversary, respectively. $\mathbb{T} : S\times U_d\times U_a\times S \to [0,1]$ encodes $\mathbb{T}(s'|s,u_d,u_a)$, the transition probability from state s to s' when defender and adversary actions are u_d and u_a, such that $\sum_{s'}\mathbb{T}(s'|s,u_d,u_a) = 1$ for all s,u_d,u_a. $\mathscr{AP}$ is a set of atomic propositions. $\mathscr{L} : S \to 2^{\mathscr{AP}}$ is a labeling function that maps a state to a subset of atomic propositions that are satisfied in that state.

Stochastic games can be viewed as an extension of Markov Decision Processes when there is more than one player taking an action. For a player, a *policy* is a mapping from sequences of states to actions, if it is deterministic, or from sequences of states to a probability distribution over actions, if it is randomized. A policy is *Markov* if it is dependent only on the most recent state.

In this paper, we focus our attention on the Stackelberg setting [27]. In this framework, the first player (leader) commits to a policy. The second player (follower) observes the leader's policy and chooses its policy as the best response to the leader's policy, defined as the policy that maximizes the follower's utility. We also assume that the players take their actions concurrently at each time step.

We now define the notion of a Stackelberg equilibrium, which indicates that a solution to a Stackelberg game has been found. Let $Q_L(l,f)$ ($Q_F(l,f)$) be the utility gained by the leader (follower) by adopting a policy l (f).

Definition 2.4 (Stackelberg Equilibrium): A pair (l,f) is a *Stackelberg equilibrium* if $l = \arg\max_{l'} Q_L(l',BR(l'))$, where $BR(l') = \{f : f = \arg\max Q_F(l',f)\}$. That is, the leader's policy is optimal given that the follower observes the leader's policy and plays its best response.

When $|U_a| = |U_d| = 1$, $\mathscr{G}$ is a *Markov chain* [28]. For $s,s' \in S$, s' is *accessible* from s, written $s \to s'$, if $\mathbb{P}(s_a|s)\mathbb{P}(s_b|s_a)\ldots\mathbb{P}(s_i|s_j)\mathbb{P}(s'|s_i) > 0$ for some (finite subset of) states $s_a,s_b,\ldots,s_i,s_j$. Two states *communicate* if $s \to s'$ and $s' \to s$. *Communicating classes* of states cover the state space of the Markov chain. A state is *transient* if there is a nonzero probability of not returning to it when we start from that state, and is *positive recurrent* otherwise. In a finite state Markov chain, every state is either transient or positive recurrent.

C. Partially Observable Stochastic Games

Partially observable stochastic games (POSGs) extend Definition 2.3 to the case when instead of observing a state directly, each player receives an observation that is derived from the state. This can be viewed as an extension of POMDPs to the case when there is more than one player.

Definition 2.5 (Partially Observable Stochastic Game): A *partially observable stochastic game* is defined by the tuple $\mathscr{SG} := (S,s_0,U_d,U_a,\mathbb{T},\mathscr{O}_d,\mathscr{O}_a,O_d,O_a,\mathscr{AP},\mathscr{L})$, where $S,s_0,U_d,U_a,\mathbb{T},\mathscr{AP},\mathscr{L}$ are as in Definition 2.3. $\mathscr{O}_d$ and $\mathscr{O}_a$ denote the (finite) sets of observations available to the defender and adversary. $O_* : S\times\mathscr{O}_* \to [0,1]$ encodes $\mathbb{P}(o_*|s)$, where $* \in \{d,a\}$.

The functions O_* model imperfect sensing. In order for O_* to satisfy the conditions of a probability distribution, we need $O_*(o|s) \geq 0 \forall o \in \mathscr{O}_*$ and $\sum_{o\in\mathscr{O}_*} O_*(o|s) = 1$.

D. Adversary and Defender Models

The initial state of the system is s_0. A transition from a state s_t to the next state s_{t+1} is determined jointly by the actions of the defender and adversary according to the transition probability function $\mathbb{T}$.

At a state s_t, the adversary makes an observation, O_a^t of the state according to O_a. The adversary is also assumed to be aware of the policy (sequence of actions), μ_d, committed to by the defender. Therefore, the overall information available to the adversary is $\mathfrak{I}_a^t := \bigcup_{i=0:t} O_a^i \cup \{\mu_d\}$.

Different from the information available to the adversary, at state s_t, the defender makes an observation O_d^t of the state according to O_d. Therefore, the overall information for the defender is $\mathfrak{I}_d^t := \bigcup_{i=0:t} O_d^i$.

Definition 2.6 (POSG Policy): A *(defender or adversary) policy* for the POSG is a map from the respective overall information to a probability distribution over the corresponding action space, i.e. $\mu_*^t : \mathfrak{I}_*^t \times U_* \to [0,1]$, $* \in \{d,a\}$.

Policies of the form above are called *randomized policies*. If $\mu_*^t : \mathfrak{I}_*^t \to U_*$, it is called a *deterministic policy*. In the sequel, we will use finite state controllers as a representation of policies that consider only the most recent observation.

E. The Product-POSG

In order to find runs on $\mathscr{SG}$ that would be accepted by a DRA $\mathscr{RA}$ built from an LTL formula ϕ, we construct a product-POSG. This construction is motivated by the product-stochastic game construction in [17] and the product-POMDP construction in [19].

Definition 2.7 (Product-POSG): Given $\mathscr{SG}$ and $\mathscr{RA}$ (built from LTL formula ϕ), a *product-POSG* is $\mathscr{SG}^\phi = (S^\phi,s_0^\phi,U_d,U_a,\mathbb{T}^\phi,\mathscr{O}_d,\mathscr{O}_a,O_d^\phi,O_a^\phi,F^\phi,\mathscr{AP},\mathscr{L}^\phi)$, where $S^\phi = S\times Q$, $s_0^\phi = (s_0,q_0)$, $O_*^\phi(o|(s,q)) = O_*(o|s)$, $\mathbb{T}^\phi((s',q')|(s,q),u_d,u_a) = \mathbb{T}(s'|s,u_d,u_a)$ iff $\delta(q,\mathscr{L}(s')) = q'$, and 0 otherwise, $F^\phi = \{(L^\phi(i),K^\phi(i))\}_{i=1}^M$, $L^\phi(i),K^\phi(i) \subset S^\phi$,

$$\begin{aligned}\bar{\mathbb{T}} &:= \mathbb{T}^{\phi,\mathscr{C}_d,\mathscr{C}_a}((s',q'),g_d',g_a'|(s,q),g_d,g_a) \\ &= \sum_{o\in\mathscr{O}_d}\sum_{o'\in\mathscr{O}_a}\sum_{u_d}\sum_{u_a} O_d(o|s)O_a(o'|s)\mu_d(g_d',u_d|g_d,o)\mu_a(g_a',u_a|g_a,o')\mathbb{T}^{\phi}((s',q')|(s,q),u_d,u_a)\end{aligned} \tag{1}$$

and $(s,q)\in L^{\phi}(i)$ iff $q\in L(i)$, $(s,q)\in K^{\phi}(i)$ iff $q\in K(i)$, $\mathscr{L}^{\phi}((s,q))=\mathscr{L}(s)$.

From the above definition, it is clear that acceptance conditions in the product-POSG depend on the DRA while the transition probabilities of the product-POSG are determined by transition probabilities of the original POSG. Therefore, a run on the product-POSG can be used to generate a path on the POSG and a run on the DRA. Then, if the run on the DRA is accepting, we say that the product-POSG satisfies the LTL specification ϕ.

III. Problem Setup

This section details the construction of FSCs for the two agents. An FSC for an agent can be interpreted as a policy for that agent. The defender and adversary policies will be determined by probability distributions over transitions in finite state controllers that are composed with the POSG. When the FSCs are composed with the product-POSG, the resulting entity is a Markov chain. We then establish a way to determine satisfaction of an LTL specification on the product-POSG in terms of runs on the composed MC.

A. Finite State Controllers

Finite state controllers consist of a finite number of internal states. Transitions between states is governed by the current observation of the agent. In our setting, we will have two FSCs, one for the defender and another for the adversary. We will then limit the search for defender and adversary policies to one over FSCs of fixed cardinality.

Definition 3.1 (Finite State Controller): A *finite state controller for the defender (adversary)*, denoted $\mathscr{C}_d$ ($\mathscr{C}_a$), is a tuple $\mathscr{C}_* = (G_*, g_{0_*}, \mu_*)$, where G_* is a finite set of (internal) states of the controller, g_{0_*} is the initial state of the FSC, and $\mu_* : G_* \times \mathscr{O}_* \times G_* \times U_* \to [0,1]$, written $\mu_*(g_*', u_*|g_*, o_*)$, is a probability distribution of the next internal state and action, given a current internal state and observation. Here, $*\in\{d,a\}$.

An FSC is a finite-state probabilistic automaton that takes the current observation of the agent as its input, and produces a distribution over the actions as its output. The FSC-based control policy is defined as follows: initial states of the FSCs are determined by the initial state of the POSG. The defender commits to a policy at the start. At each time step, the policy returns a distribution over the actions and the next state of $\mathscr{C}_d$, given the current state of the FSC $\mathscr{C}_d$ and the state of $\mathscr{SG}^{\phi}$ observed according to O_d. The adversary observes this and the state according to O_a and responds with $\mu_a(\cdot)$ generated by $\mathscr{C}_a$. Actions at each step are taken concurrently.

Definition 3.2 (Proper FSCs): An FSC is *proper* with respect to an LTL formula ϕ if there is a positive probability of satisfying ϕ under this policy in an environment represented as a partially observable stochastic game.

This is similar to the definition in [29], with the distinction that the terminal state of an FSC in that context will be related to Rabin acceptance pairs of an MC formed by composing $\mathscr{C}_d$ and $\mathscr{C}_a$ with $\mathscr{SG}^{\phi}$ (Sec III-B).

B. The Global Markov Chain

The FSCs $\mathscr{C}_d$ and $\mathscr{C}_a$, when composed with $\mathscr{SG}^{\phi}$, will result in a finite-state, (fully observable) Markov chain. To maintain consistency with the literature, we will refer to this as the *global Markov chain (GMC)* [19].

Definition 3.3 (Global Markov Chain (GMC)): The *GMC* resulting from a product-POSG $\mathscr{SG}^{\phi}$ controlled by FSCs $\mathscr{C}_d$ and $\mathscr{C}_a$ is $\mathscr{M} := \mathscr{M}^{\phi,\mathscr{C}_d,\mathscr{C}_a} = (\bar{S}, \bar{s}_0, \bar{\mathbb{T}}, \mathscr{AP}, \bar{\mathscr{L}})$, where $\bar{S} = S^{\phi}\times G_d \times G_a$, $\bar{s}_0 = (s_0, q_0, g_{0_d}, g_{0_a})$, $\bar{\mathbb{T}}$ is given by Equation (1), and $\bar{\mathscr{L}} = \mathscr{L}^{\phi}((s,q))$.

Similar to $\mathscr{SG}^{\phi}$, the Rabin acceptance condition for $\bar{\mathscr{M}}$ is: $\bar{F} = \{(\bar{L}(i), \bar{K}(i))\}_{i=1}^{M}$, with $(s,q,g_d,g_a)\in\bar{L}(i)$ iff $(s,q)\in L^{\phi}(i)$ and $(s,q,g_d,g_a)\in\bar{K}(i)$ iff $(s,q)\in K^{\phi}(i)$.

A state of $\mathscr{M}$ is $\mathfrak{s} := (s,q,g_d,g_a)$. A path on $\mathscr{M}$ is a sequence $\pi := \mathfrak{s}_0\mathfrak{s}_1\ldots$ such that $\mathbb{T}(\mathfrak{s}_{k+1}|\mathfrak{s}_k) > 0$, where $\mathbb{T}(\cdot)$ is the transition probability in $\mathscr{M}$. The path is accepting if it satisfies the Rabin acceptance condition. This corresponds to an execution in $\mathscr{SG}^{\phi}$ controlled by $\mathscr{C}_d$ and $\mathscr{C}_a$.

To quantitatively reason about $\mathscr{M}$, we define a probability space following the treatment in [7]. The set of paths in $\mathscr{M}$, denoted $\texttt{Paths}(\mathscr{M})$ forms the sample space. The set of events $\mathscr{F}$ is the smallest $\sigma-$algebra generated by *cylinder sets* spanned by path fragments of finite length in $\mathscr{M}$. The cylinder set spanned by $\hat{\pi} := \mathfrak{s}_0\mathfrak{s}_1\ldots\mathfrak{s}_n$ is given by paths $\pi\in\texttt{Paths}(\mathscr{M})$ that start with $\hat{\pi}$. This is denoted $Cyl(\mathfrak{s}_0\mathfrak{s}_1\ldots\mathfrak{s}_n)$. Then, the (unique) probability measure on $\mathscr{F}$ for the events is given by $Pr^{\mathscr{M}}(Cyl(\mathfrak{s}_0\mathfrak{s}_1\ldots\mathfrak{s}_n)) := \mathbb{P}(\mathfrak{s}_0)\bar{\mathbb{T}}(\mathfrak{s}_1|\mathfrak{s}_0)\bar{\mathbb{T}}(\mathfrak{s}_2|\mathfrak{s}_1)\ldots\bar{\mathbb{T}}(\mathfrak{s}_n|\mathfrak{s}_{n-1})$. The probability of the LTL objective ϕ being satisfied in state $\mathfrak{s}$ is $Pr^{\mathscr{M}}(\mathfrak{s}\models\phi) := Pr^{\mathscr{M}}\{\pi\in\texttt{Paths}(\mathfrak{s})|\pi\models\phi\}$. In the sequel, we write $\mathbb{P}(\mathscr{M}\models\phi) := \mathbb{P}(\mathscr{M}^{\phi,\mathscr{C}_d,\mathscr{C}_a}\models\phi)$ to denote $Pr^{\mathscr{M}}(s_0\models\phi)$. We direct the reader to Section 10.1 in [7] for a characterization of the probability spaces for different LTL objectives.

C. Problem Statement

The goal is to synthesize a defender policy that maximizes the probability of satisfaction of an LTL specification under any adversary policy. Clearly, this will depend on the FSCs, $\mathscr{C}_d$ and $\mathscr{C}_a$. In this paper, we will assume that the size of the adversary FSC is fixed, and known to the defender. This can be interpreted as one way for the defender to have knowledge of the capabilities of an

adversary. Future work will consider the problem for FSCs of arbitrary sizes. The problem is stated below.

Problem 3.4: Given a partially observable environment and an LTL formula, determine a defender policy specified by an FSC that maximizes the probability of satisfying the LTL formula under any adversary policy that is represented as an FSC of fixed size.

IV. LTL Satisfaction and Recurrent Sets

The first result in this section relates the probability of the LTL specification being satisfied by the product-POSG, denoted $\mathscr{SG}^{\phi} \models \phi$, in terms of recurrent sets of the GMC. We then present a procedure to generate recurrent sets of the GMC that additionally satisfy the LTL formula. The main result of this section relates Problem 3.4 to determining FSCs that maximize the probability of reaching certain types of recurrent sets of the GMC.

Let $\mathscr{R} = \mathscr{R}^{\phi,\mathscr{C}_d,\mathscr{C}_a}$ denote the recurrent states of $\mathscr{M}$ under FSCs $\mathscr{C}_d$ and $\mathscr{C}_a$. Let $\mathscr{R}^S := (s,q)$ be the restriction of a recurrent state to a state of $\mathscr{SG}^{\phi}$.

Proposition 4.1: $\mathbb{P}(\mathscr{M} \models \phi) = \mathbb{P}(\mathscr{SG}^{\phi} \models \phi | \mathscr{C}_d, \mathscr{C}_a) > 0$ if and only if there exists $\mathscr{C}_d$ such that for any $\mathscr{C}_a$, there exists a Rabin acceptance pair $(L^{\phi}(i), K^{\phi}(i))$ and an initial state of $\mathscr{M}$, $\bar{s}_0$, where the following conditions hold:

$$\begin{aligned} & K^{\phi}(i) \cap \mathscr{R}^S \neq \emptyset \\ & \bar{s}_0 \rightarrow (K^{\phi}(i) \times G_d \times G_a) \cap \mathscr{R} \\ & \bar{s}_0 \not\rightarrow (L^{\phi}(i) \times G_d \times G_a) \cap \mathscr{R} \end{aligned} \tag{2}$$

Proof: If for every $(L^{\phi}(i), K^{\phi}(i))$, at least one of the conditions in Equation (2) does not hold, then at least one of the following statements is true: *i)*: no state that has to be visited infinitely often is recurrent; *ii)*: there is no initial state from which a recurrent state that has to be visited infinitely often is accessible; *iii)*: some state that has to be visited only finitely often in steady state is recurrent. This means $\mathscr{SG}^{\phi} \not\models \phi$ for all $\mathscr{C}_d$.

Conversely, if all the conditions in Equation (2) hold for some $(L^{\phi}(i), K^{\phi}(i))$, then $\mathscr{SG}^{\phi} \models \phi$ by construction. ■

To quantify the satisfaction probability for a defender policy under any adversary policy, assume that the recurrent states of $\mathscr{M}$ are partitioned into recurrence classes $\{R_1, \ldots, R_p\}$. This partition is maximal, in the sense that two recurrent classes cannot be combined to form a larger recurrent class, and all states within a given recurrent class communicate with each other [20].

Definition 4.2 (ϕ–feasible Recurrent Set): A recurrent set R_k is *ϕ–feasible* under FSCs $\mathscr{C}_d$ and $\mathscr{C}_a$ if there exists $(L^{\phi}(i), K^{\phi}(i))$ such that $K^{\phi}(i) \cap R_k^S \neq \emptyset$ and $L^{\phi}(i) \cap R_k^S = \emptyset$. Let $\phi\text{–}RecSets^{\mathscr{C}_d,\mathscr{C}_a}$ denote the set of ϕ–feasible recurrent sets under the respective FSCs.

Let $\pi \rightarrow R$ be the event that a path of $\mathscr{M}$ will reach a recurrent set. Algorithm 1 returns ϕ–feasible recurrent sets of $\mathscr{SG}^{\phi}$ under fixed FSCs $\mathscr{C}_d, \mathscr{C}_a$.

We have the following result:

Theorem 4.3: The probability of satisfying an LTL formula ϕ in a POSG with policies $\mathscr{C}_d$ and $\mathscr{C}_a$ is equal to the probability of paths in the GMC (under the same FSCs) reaching ϕ–feasible recurrent sets. That is,

$$\mathbb{P}(\mathscr{SG}^{\phi} \models \phi | \mathscr{C}_d, \mathscr{C}_a) = \sum_{R \in \phi\text{–}RecSets^{\mathscr{C}_d,\mathscr{C}_a}} \mathbb{P}(\pi \rightarrow R) \tag{3}$$

Proof: Since the recurrence classes are maximal, $\mathbb{P}(\pi \rightarrow (R_1 \cup \cdots \cup R_p)) = \sum_{k=1}^{p} \mathbb{P}(\pi \rightarrow R_k)$. From Definition 4.2, a ϕ–feasible recurrent set will necessarily contain a Rabin acceptance pair. Therefore, the probability of $\mathscr{SG}^{\phi}$ satisfying the LTL formula under $\mathscr{C}_d$ and $\mathscr{C}_a$ is equivalent to the probability of paths on $\mathscr{M}$ leading to ϕ–feasible recurrent sets, which is given by Equation (3). ■

Corollary 4.4: From Theorem 4.3, it follows that:

$$\begin{aligned} \max_{\mathscr{C}_d} \min_{\mathscr{C}_a} \mathbb{P}(\mathscr{M} \models \phi) &= \max_{\mathscr{C}_d} \min_{\mathscr{C}_a} \mathbb{P}(\mathscr{SG}^{\phi} \models \phi | \mathscr{C}_d, \mathscr{C}_a) \\ &= \max_{\mathscr{C}_d} \min_{\mathscr{C}_a} \sum_{R \in \phi\text{–}RecSets^{\mathscr{C}_d,\mathscr{C}_a}} \mathbb{P}(\pi \rightarrow R) \end{aligned} \tag{4}$$

We note that Proposition 4.1, Theorem 4.3, and Corollary 4.4 address a broader class of problems than in Problem 3.4 since they do not assume that the size of the adversary FSC is fixed. Corollary 4.4 also indicates that the objective of Problem 3.4 can be formally expressed as:

$$\max_{\mathscr{C}_d} \min_{\mathscr{C}_a} \mathbb{P}(\mathscr{M} \models \phi || G_a | = G_A) \tag{5}$$

Algorithm 1 Generate ϕ–feasible Recurrent Sets for $\mathscr{SG}^{\phi}$ under FSCs $\mathscr{C}_d, \mathscr{C}_a$

Input: $\mathscr{M} := \mathscr{M}^{\phi,\mathscr{C}_d,\mathscr{C}_a}, \{L^{\phi}(i), K^{\phi}(i)\}_{i=1}^{M}$
Output: $\{R_k\}$, that is recurrent and ϕ–feasible

```
Induce digraph 𝒢 of ℳ of 𝒮𝒢^φ under 𝒞_d, 𝒞_a as (𝔖, ℰ),
    s.t. ∀𝔰_1, 𝔰_2 ∈ 𝔖 : 𝔰_1 → 𝔰_2 ∈ ℰ ⇔ 𝕋(𝔰_2|𝔰_1) > 0.
𝒞 = SCCs(𝒢) = {C_1, ..., C_N} // strongly connected components of digraph
RecSets := {R_1, ..., R_p} such that R_i ∈ 𝒞 and R_i is a sink SCC
φ−RecSets^{𝒞_d,𝒞_a} = ∅
for j = 1 to p do
    for i = 1 to M do
        if (L^φ(i) ∩ R_j^S = ∅) ∧ (K^φ(i) ∩ R_j^S ≠ ∅) then
            φ−RecSets^{𝒞_d,𝒞_a} = φ−RecSets^{𝒞_d,𝒞_a} ∪ R_j
        end if
    end for
end for
```

V. Determining Candidate FSCs of Fixed Sizes

If the sizes of $\mathscr{C}_d$ and $\mathscr{C}_a$ are fixed, then their design is equivalent to determining the transition probabilities between their internal states. In this section, we present a heuristic procedure that uses only the most recent observations of the defender and adversary to generate a set of admissible FSC structures such that the resulting GMC will have a ϕ–feasible recurrent set. We show that the procedure has a computational complexity that is polynomial in the number of states of the GMC and additionally establish that this algorithm is sound.

$$O_d(o_d|s)O_a(o_a|s)\mu_d(g'_d,u_d|g_d,o_d)\mu_a(g'_a,u_a|g_a,o_a)\mathbb{T}^{\phi}((s',q')|(s,q),u_d,u_a)>0 \quad (6)$$

$$O_d(o_d|s)O_a(o_a|s)\mu_d(g''_d,u_d|g_d,o_d)\mu_a(g''_a,u_a|g_a,o_a)\mathbb{T}^{\phi}((s'',q'')|(s,q),u_d,u_a)>0 \quad (7)$$

Definition 5.1: An algorithm is *sound* if any solution returned by it is the Boolean constant *true* when evaluated on the output of the algorithm (i.e., every output is a correct output). It is *complete* if it returns a result for any input, and reports 'failure' if no solution exists.

Let $\mathscr{I}_* : G_* \times \mathscr{O}_* \times G_* \times U_* \to \{0,1\}$, where $\mathscr{I}_*(g',u|g,o) = 1 \Leftrightarrow \mu_*(g',u|g,o) > 0$. $\mathscr{I}_*(\cdot)$ shows if an observation o can enable the transition from an FSC state g to g' while issuing action u. We also assume that $\forall (g,o) \in G_* \times \mathscr{O}_*, \exists (g',u) \in G_* \times U_*$ such that $\mathscr{I}_*(g',u|g,o) = 1$ [20].

In Algorithm 2, for defender and adversary FSCs with fixed number of states, we determine candidate $\mathscr{C}_d$ and $\mathscr{C}_a$ such that the resulting $\mathscr{M}$ will have a $\phi-$feasible recurrent set. We start with initial candidate structures $\mathscr{I}^o_*$ and induce the digraph of the resulting GMC (*Line 1*). In our case, $\mathscr{I}^o_*$ is such that $\mathscr{I}^o_*(g'_*,u_*|g_*,o_*) = 1$ for all g'_*,g_*,u_*,o_*. We first determine the set of communicating classes of the GMC, which is equivalent to determining the strongly connected components (SCCs) of the induced digraph (*Line 3*). A communicating class will be recurrent if it is a *sink* SCC of the corresponding digraph. The states in Bad_i are those in C that are part of the Rabin accepting pair that has to be visited only finitely many times (and therefore, to be visited with very low probability in steady state) (*Line 6*). Bad_i further contains states that can be transitioned to from some state in C. This is because once the system transitions out of C, it will not be able to return to it in order to satisfy the Rabin acceptance condition (*Line 5*) (and hence, C will not be recurrent). $Good_i$ contains those states in C that need to be visited infinitely often according to the Rabin acceptance condition (*Line 7*).

The agents have access to a state only via their observations. A defender action is forbidden if there exists an adversary action that will allow a transition to a state in Bad_i under observations o_d and o_a. This is achieved by setting corresponding entries in $\mathscr{I}_d$ to zero (*Lines 12-17*). An adversary action is not useful if for every defender action, the probability of transitioning to a state in $Good_i$ is nonzero under o_d and o_a. This is achieved by setting the corresponding entry in $\mathscr{I}_a$ to zero (*Lines 18-23*).

Proposition 5.2: Define $|\mathscr{O}| = |\mathscr{O}_d| + |\mathscr{O}_a|$ and $|U| = |U_d| + |U_a|$. Then, Algorithm 2 has an overall computational complexity of $\mathbf{O}(|S|^2|G_d|^2|G_a|^2|\mathscr{O}||U|)$.

Proof: The overall complexity depends on: *(i)* Determining strongly connected components (*Line 3*): This can be done in $\mathbf{O}(|\mathfrak{S}| + |\mathscr{E}|)$ [30]. Since $|\mathfrak{S}| = |S||G_d||G_a|$ and $|\mathscr{E}| \leq |\mathfrak{S}|^2$, this is $\mathbf{O}(|S|^2|G_d|^2|G_a|^2)$ in the worst case, and *(ii)* Determining the structures in *Lines 9-26*: This is $\mathbf{O}(|\mathfrak{S}|(|\mathscr{O}_d| + |\mathscr{O}_a|)(|\mathfrak{S}|(|U_d| + |U_a|))$. The result follows by combining the two terms. ∎

Proposition 5.3: Algorithm 2 is sound.

Algorithm 2 Generate candidate FSCs $\mathscr{C}_d, \mathscr{C}_a$

Input: G_d, G_a, $\mathscr{SG}^{\phi}$, $\mathscr{I}^o_d$, $\mathscr{I}^o_a$

Output: Set of admissible FSC structures $\mathbb{I} := (\mathbb{I}_d, \mathbb{I}_a)$, such that GMC has a $\phi-$feasible recurrent set

1: Induce digraph $\mathscr{G}$ of $\mathscr{M}$ of $\mathbb{SG}^{\phi}$ under $\mathscr{I}^o_d$ and $\mathscr{I}^o_a$ as $(\mathfrak{S},\mathscr{E})$, s.t. $\forall \mathfrak{s}_1,\mathfrak{s}_2 \in \mathfrak{S} : \mathfrak{s}_1 \to \mathfrak{s}_2 \in \mathscr{E} \Leftrightarrow \mathbb{T}(\mathfrak{s}_2|\mathfrak{s}_1) > 0$.
2: $\mathbb{I}_d = \mathbb{I}_a = \emptyset$
3: $\mathscr{C} = SCCs(\mathscr{G}) = \{C_1,\ldots,C_N\}$
4: **for** $C \in \mathscr{C}$ **and** $(L^{\phi}(i),K^{\phi}(i)) \in F^{\phi}$ **do**
5: $Bad_i = \{\mathfrak{s}' \notin C : \exists \mathfrak{s} \in C$ s.t. $\mathfrak{s} \to \mathfrak{s}'\}$
6: $Bad_i = Bad_i \cup (C \cap (L^{\phi}(i) \times G_d \times G_a))$
7: $Good_i = C \cap (K^{\phi}(i) \times G_d \times G_a)$
8: Set $\mathscr{I}_*(g'_*,u_*|g_*,o_*) = 1$ for all g'_*,g_*,u_*,o_*
9: **while** $\sum_{g'_*,u_*} \mathscr{I}_*(g'_*,u_*|g_*,o_*) > 0 \forall o_*,g_*$ **and** $Bad_i \neq \emptyset$ **do**
10: Choose $\mathfrak{s}' = (s',q',g'_d,g'_a) \in Bad_i$, $\mathfrak{s}'' = (s'',q'',g''_d,g''_a) \in Good_i$
11: **for** $\mathfrak{s} = (s,q,g_d,g_a) \in C \setminus Bad_i$ **do**
12: **for** $u_d \in U_d$ **do**
13: $\mu_*(g'_*,u_*|g_*,o_*) = \frac{\mathscr{I}_*(g'_*,u_*|g_*,o_*)}{\sum_{g'_*,u_*}\mathscr{I}_*(g'_*,u_*|g_*,o_*)}$
14: **if** $\exists u_a \in U_a$ Eqn (6) holds **then**
15: $\mathscr{I}_d(g'_d,u_d|g_d,o_d) \leftarrow 0$ $\forall g'_d,g_d \in G_d$
16: **end if**
17: **end for**
18: **for** $u_a \in U_a$ **do**
19: $\mu_*(g''_*,u_*|g_*,o_*) = \frac{\mathscr{I}_*(g''_*,u_*|g_*,o_*)}{\sum_{g''_*,u_*}\mathscr{I}_*(g''_*,u_*|g_*,o_*)}$
20: **if** $\forall u_d \in U_d$, Eqn (7) holds **then**
21: $\mathscr{I}_a(g''_a,u_a|g_a,o_a) \leftarrow 0$
22: **end if**
23: **end for**
24: **end for**
25: $Bad_i = Bad_i \setminus \{\mathfrak{s}'\}$
26: **end while**
27: Construct digraph $\mathscr{G}_{new}$ of GMC of $\mathscr{SG}^{\phi}$ under modified $\mathscr{I}_d$ and $\mathscr{I}_a$
28: $\mathscr{C}_{new} = SCCs(\mathscr{G}_{new})$
29: **if** $\exists \mathfrak{s} \in Good_i$ s.t. $\mathfrak{s}$ is *recurrent* in $\mathscr{G}_{new}$ **then**
30: $\mathbb{I} = (\mathbb{I}_d \cup \mathscr{I}_d, \mathbb{I}_a \cup \mathscr{I}_a)$
31: **end if**
32: **end for**

Proof: This is by construction. The output of the algorithm is a set $\{\mathscr{I}^i_d,\mathscr{I}^i_a\}_{i=1}^W$ such that the resulting GMC for each case has a state that is recurrent and has to be visited infinitely often. This state, by Definition 4.2, belongs to $\phi - RecSet^{\mathscr{C}^i_d,\mathscr{C}^i_a}$. Moreover, if the algorithm

returns a nonempty solution, a solution to Problem 3.4 will exist since the FSCs are proper. ∎

Algorithm 2 is *suboptimal* since we only consider the most recent observations of the defender and adversary. It is also not complete, since there might be a feasible solution that cannot be determined by the algorithm. If no FSC structures of a particular size is returned by Algorithm 2, a heuristic is to increase the number of states in the defender FSC by one, and run the Algorithm again. Once we obtain proper FSC structures of fixed sizes, we will show in Section VII that the satisfaction probability can be improved by adding states to the defender FSC in a principled manner (for adversary FSCs of fixed size). Algorithm 2 and Proposition 5.3 will allow us to restrict our treatment to proper FSCs for the rest of the paper.

VI. Value Iteration for POSGs

In this section, we present a value-iteration based procedure to maximize the probability of satisfying the LTL formula ϕ for FSCs $\mathscr{C}_d$ and $\mathscr{C}_a$ of fixed sizes. We prove that the procedure converges to a unique optimal value, corresponding to the Stackelberg equilibrium.

Notice that in Equation (1), the defender and adversary policies are specified as probability distributions over the next FSC internal state and the respective agent action, and conditioned on the current FSC internal state and the agent observation. With $* \in \{d,a\}$, we rewrite these in terms of a mapping $\hat{\mu}_* : G_* \times S \times G_* \times U_* \to [0,1]$:

$$\hat{\mu}_*(g'_*,u_*|g_*,s) := \sum_{o_* \in \mathscr{O}_*} O_*(o_*|s)\mu_*(g'_*,u_*|g_*,o_*) \quad (8)$$

This will allow us to express Equation (1) as:

$$\begin{aligned} &Pr^{\phi}((s',q'),g'_d,g'_a|(s,q),g_d,g_a) \\ &= \sum_{u_d}\sum_{u_a} \hat{\mu}_d(g'_d,u_d|g_d,s)\hat{\mu}_a(g'_a,u_a|g_a,s) \\ &\qquad \mathbb{T}^{\phi}((s',q'),g'_d,g'_a|(s,q),g_d,g_a) \end{aligned} \quad (9)$$

Define a value V over the state space of the GMC representing the probability of satisfying the LTL formula ϕ when starting from a state of the GMC. Additionally, define and characterize the following operators:

$$\begin{aligned} (T_{\hat{\mu}_d\hat{\mu}_a}V)(\mathfrak{s}) &= \sum_{\mathfrak{s}'} Pr(\mathfrak{s}'|\mathfrak{s})V(\mathfrak{s}'); \\ (T_{\hat{\mu}_d}V)(\mathfrak{s}) &= \min_{\hat{\mu}_a} \sum_{\mathfrak{s}'} Pr(\mathfrak{s}'|\mathfrak{s})V(\mathfrak{s}'); \\ (TV)(\mathfrak{s}) &= \max_{\hat{\mu}_d} \min_{\hat{\mu}_a} \sum_{\mathfrak{s}'} Pr(\mathfrak{s}'|\mathfrak{s})V(\mathfrak{s}') \end{aligned}$$

where $Pr(\mathfrak{s}'|\mathfrak{s})$ is the transition probability in the GMC induced by policies $\hat{\mu}_d$ and $\hat{\mu}_a$ (Equation (9)).

Proposition 6.1: Let

$$\begin{aligned} &V((s,q),g_d,g_a) \\ &\qquad = \max_{\hat{\mu}_d} \min_{\hat{\mu}_a} Pr(\phi|((s,q),g_d,g_a)). \end{aligned} \quad (10)$$

Then

$$\begin{aligned} &V((s,q),g_d,g_a) = \\ &\max_{\hat{\mu}_d} \min_{\hat{\mu}_a} \sum_{((s',q'),g'_d,g'_a)} \sum_{u_d}\sum_{u_a} \Big(\hat{\mu}_d(g'_d,u_d|g_d,s) \\ &\qquad \times \hat{\mu}_a(g'_a,u_a|g_a,s) \\ &\qquad \times T^{\phi}((s',q')|(s,q),u_d,u_a)V((s',q'),g'_d,g'_a)) \Big) \end{aligned} \quad (11)$$

Conversely, if the value vector V satisfies Equation (11), then Equation (10) holds true. Moreover, V is unique.

Before proving Proposition 6.1, we will need some intermediate results. Inequalities in the proofs of these statements are true element-wise.

Theorem 6.2: [Monotone Convergence Theorem][31] If a sequence is monotone increasing and bounded from above, then it is a convergent sequence.

Lemma 6.3: Let V be the satisfaction probability obtained under any pair of policies $\hat{\mu}_d$ and $\hat{\mu}_a$, where $\hat{\mu}_a$ is the best response to $\hat{\mu}_d$. Let T^k be the operation that composes the T operator k times, and V^k be the corresponding value obtained (i.e., $T^kV := V^k$). Then, there exists a value V^* such that $\lim_{k\to\infty} T^kV = V^*$.

Proof: We show Lemma 6.3 by showing that the sequence $V^k = T^kV$ is bounded and monotone.

We first show boundedness. By definition of the operator T, V^{k+1} is obtained as a convex combination of V^k. Since V is the satisfaction probability, it is in $[0,1]$. Thus, V^0 is bounded, and consequently, T^kV is bounded for all k.

We next show monotonicity by induction. We have that V^0 is the value function associated with a control policy $\hat{\mu}_d$. Denote the best response of the adversary to $\hat{\mu}_d$ as $\hat{\mu}_a$. Let $V^1 := TV^0$. From the definitions of T and $T_{\hat{\mu}_d}$, we have $TV^0 \geq T_{\hat{\mu}_d}V^0$. Furthermore, $V^0 = T_{\hat{\mu}_d}V^0$ since

$$\begin{aligned} T_{\hat{\mu}_d}V^0(\mathfrak{s}) = \min_{\hat{\mu}'_a} \sum_{\mathfrak{s}'} \sum_{u_d}\sum_{u_a} \Big(&V^k(\mathfrak{s}')\hat{\mu}_d(g'_d,u_d|g_d,s) \\ &\times \hat{\mu}_a(g'_a,u_a|g_a,s)\mathbb{T}^{\phi}((s',q')|(s,q),u_d,u_a) \Big) = V^0 \end{aligned}$$

by the definition that $\hat{\mu}'_a$ is the best response of $\hat{\mu}_d$. Therefore, we have that $V^1 = TV^0 \geq T_{\hat{\mu}_d}V^0 = V^0$. This gives us $V^1 \geq V^0$, which serves as the base case for the induction. Consider iteration k. Suppose $T^{k-1}V \leq T^kV$. We then show $T^kV \leq T^{k+1}V$. We have:

$$T^{k+1}V \geq \min_{\hat{\mu}_a} \sum_{\mathfrak{s}'} Pr(\mathfrak{s}'|\mathfrak{s})V^k(\mathfrak{s}') \geq \min_{\hat{\mu}_a} \sum_{\mathfrak{s}'} Pr(\mathfrak{s}'|\mathfrak{s})V^{k-1}(\mathfrak{s}') = V^k.$$

The first inequality holds by the definition of T, the second inequality holds by the induction hypothesis that $V^k \geq V^{k-1}$, and the last equality holds by construction of a control policy. The existence of V^* such that $\lim_{k\to\infty} T^kV = V^*$ follows from Theorem 6.2. ∎

We now prove Proposition 6.1.

Proof: We first prove the forward direction by contradiction, i.e., if Equation (10) holds then Equation (11) holds. Suppose $\hat{\mu}_d$ is a Stackelberg equilibrium policy with satisfaction probability being V, while Equation (11) does

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2020.3039484, IEEE Transactions on Automatic Control

not hold. Since $\hat{\mu}_d$ is the Stackelberg equilibrium policy, $V = T_{\hat{\mu}_d} V$. This is because, given $\hat{\mu}_d$, the stochastic game is an MDP, for which V is the optimal value [13]. From the definitions of T and $T_{\hat{\mu}_d}$, we must have $T_{\hat{\mu}_d} V \leq TV$. Composing the operators k times and letting $k \to \infty$,

$$V = \lim_{k\to\infty} T^k_{\hat{\mu}_d} V \leq \lim_{k\to\infty} T^k V = V^*,$$

where the first equality holds by the assumption that V is the satisfaction probability and $\hat{\mu}_d$ is the Stackelberg equilibrium policy, the last equality holds by Lemma 6.3. If $V = V^*$, Eqn. (11) is satisfied, which contradicts our assumption that Eqn. (10) holds while (11) does not hold. If $V \neq V^*$, then there must exist a state $\mathfrak{s}$ such that $V(\mathfrak{s}) < V^*(\mathfrak{s})$. This means that there is a policy (different from $\hat{\mu}_d$) corresponding to V^* for which we achieve a higher satisfaction probability starting at state $\mathfrak{s}$. This violates our assumption that $\hat{\mu}_d$ is the equilibrium policy. We must then have that Eqn. (11) holds given that Eqn. (10) holds.

We next prove uniqueness of the V. Let $\hat{V}$ and V be two solutions to Eqn. (11), and let $\hat{\mu}_d$ and μ_d denote the corresponding control policies. From the definitions of T and $T_{\hat{\mu}_d}$, we have that $V = TV \geq T_{\hat{\mu}_d} V$. Composing the operators on both sides k times and letting $k \to \infty$,

$$V = \lim_{k\to\infty} T^k V \geq \lim_{k\to\infty} T^k_{\hat{\mu}_d} V = \hat{V},$$

where the first equality holds by the assumption that $\hat{\mu}_d$ is the equilribrium policy, and the second equality holds by the fact that $\hat{V}$ is the unique fixed point of operator $T_{\hat{\mu}_d}$ (Proposition 2.2.1 in Volume 2 of [13]). Following a similar argument as before, we have the following inequality:

$$\hat{V} = \lim_{k\to\infty} T^k \hat{V} \geq \lim_{k\to\infty} T^k_{\hat{\mu}_d} \hat{V} = V.$$

We have that both $V \geq \hat{V}$ and $\hat{V} \geq V$ are true, which gives us $V = \hat{V}$. This implies that the value V is unique.

We finally show that if Eqn. (11) holds then Eqn. (10) holds. We observe that the value function at equilibrium satisfies (11), and the solution is unique. Therefore, any solution to (11) must be a Stackelberg equilibrium [32]. ■

Proposition 6.1 indicates that value-iteration algorithms can be used to determine optimal policies $\hat{\mu}_d$ and $\hat{\mu}_a$. Given a policy $\hat{\mu}_d$ and observation function O_d, we will be able to compute μ_d by solving a system of linear equations.

A value-iteration based procedure to solve the POSG under an LTL specification is proposed in Algorithm 3. The value $V(\mathfrak{s})$ is greedily updated at every iteration by computing the policy according to Proposition 6.1. The algorithm terminates when the difference in $V(\cdot)$ in consecutive iterations is below a pre-specified threshold.

Proposition 6.4: For any $\epsilon > 0$, there exist K, V such that $||V^k(\mathfrak{s}) - V||_\infty < \epsilon$ for all $k > K$. Further, V satisfies the value in Proposition 6.1 and is within the ϵ-neighborhood of the value function at Stackelberg equilibrium.

Algorithm 3 Maximizing probability of satisfying LTL formula ϕ under fixed FSCs $\mathscr{C}_d, \mathscr{C}_a$

Input: $\mathscr{M} := \mathscr{M}^{\phi,\mathscr{C}_d,\mathscr{C}_a}, \{L^\phi(i), K^\phi(i)\}_{i=1}^M, \epsilon$ (threshold)
Output: $V \in \mathbb{R}^{|S|\times|Q|\times|G_d|\times|G_a|}$

1: Determine $\phi - RecSets^{\mathscr{C}_d,\mathscr{C}_a}$ using Algorithm 1
2: $\hat{\mu}_*(g'_*, u_*|g_*, s) := \sum_{o_*} O_*(o_*|s)\mu(g'_*, u_*|g_*, o_*)$, $* \in \{d, a\}$
3: $V^0(\mathfrak{s}) \leftarrow 0$
4: $V^1(\mathfrak{s}) \leftarrow 1$ if $\mathfrak{s} \in \phi - RecSets^{\mathscr{C}_d,\mathscr{C}_a}$, and $V^1(\mathfrak{s}) \leftarrow 0$, else
5: $k \leftarrow 0$
6: **while** $\max_{\mathfrak{s}}\{V^{k+1}(\mathfrak{s}) - V^k(\mathfrak{s})\} > \epsilon$ **do**
7: $\quad k \leftarrow k + 1$
8: $\quad$ **for** $\mathfrak{s} \notin \phi - RecSets^{\mathscr{C}_d,\mathscr{C}_a}$ **do**
9: $\quad\quad V^{k+1}(\mathfrak{s}) \leftarrow \max_{\hat{\mu}_d} \min_{\hat{\mu}_a} \sum_{\mathfrak{s}'} \sum_{u_d} \sum_{u_a} \Big(V^k(\mathfrak{s}') \hat{\mu}_d(g'_d, u_d|g_d, s) \times \hat{\mu}_a(g'_a, u_a|g_a, s) \mathbb{T}^\phi((s', q')|(s, q), u_d, u_a) \Big)$
10: $\quad$ **end for**
11: **end while**
12: **return** $V (= V^k(\mathfrak{s}))$

Proof: Notice that $V^1(\mathfrak{s}) \geq V^0(\mathfrak{s})$. Since $V^1(\mathfrak{s}) = 0$ for $(s, q) \notin \phi - RecSets^{\mathscr{C}_d,\mathscr{C}_a}$, $V^2(\mathfrak{s}) \geq V^1(\mathfrak{s})$. We induct on k.Let $\hat{\mu}^k_d$ be the optimal defender policy at step k. Then,

$$\begin{aligned} V^{k+1}(\mathfrak{s}) &\geq \min_{\hat{\mu}_a} \sum_{\mathfrak{s}'} \sum_{u_d} \sum_{u_a} \Big(V^k(\mathfrak{s}') \hat{\mu}_d(g'_d, u_d|g_d, s) \\ &\qquad \times \hat{\mu}_a(g'_a, u_a|g_a, s) \mathbb{T}^\phi((s', q')|(s, q), u_d, u_a) \Big) \\ &\geq \min_{\hat{\mu}_a} \sum_{\mathfrak{s}'} \sum_{u_d} \sum_{u_a} \Big(V^{k-1}(\mathfrak{s}') \hat{\mu}_d(g'_d, u_d|g_d, s) \\ &\qquad \times \hat{\mu}_a(g'_a, u_a|g_a, s) \mathbb{T}^\phi((s', q')|(s, q), u_d, u_a) \Big) = V^k(\mathfrak{s}) \end{aligned}$$

The first inequality holds because $V^{k+1}(\mathfrak{s})$ is the value obtained by the maximizing policy, and dominates the value achieved by any other policy. The second and last inequalities follow from the induction hypothesis and definition of $V^k(\mathfrak{s})$ respectively. Further, for each state, $V^k(\mathfrak{s})$ is bounded since it is a convex combination of terms that are ≤ 1. Let V be the set of limit points so that K can be chosen such that $||V^k(\mathfrak{s}) - V||_\infty < \epsilon$ for $k > K$.

Since $V^k(\mathfrak{s})$ converges, it is a Cauchy sequence. Therefore, for every $\epsilon > 0$, there exists K sufficiently large, such that for all $k > K$, $|V^k(\mathfrak{s}) - V^{k+1}(\mathfrak{s})| < \epsilon$. From *Line 8* of Algorithm 3, this shows that V is within an $\epsilon-$neighborhood of a Stackelberg equilibrium for every $\epsilon > 0$. ■

A minor modification of the value update gives the following result on the termination of Algorithm 3 [33].

Proposition 6.5 ([33], Proposition 4): Suppose that at time k, the value update in *Line 8* of Algorithm 3 is performed as shown if the right hand side term is greater than $(1+\epsilon)V^k(\mathfrak{s})$, and $V^{k+1}(\mathfrak{s}) = V^k(\mathfrak{s})$ otherwise. Then, Algorithm 3 converges to a value V that satisfies $||V^{k+1}(\mathfrak{s}) - V^k(\mathfrak{s})||_\infty < \epsilon$ in at most N^* iterations, where $N^* = |S||Q||G_d||G_a| \max_{\mathfrak{s}} \Big\{ \log(\frac{1}{V_{min}(\mathfrak{s})}) / \log(1+\epsilon) \Big\}$. Here $V_{min}(\mathfrak{s})$ is the smallest non-zero value of $V^k(\mathfrak{s})$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2020.3039484, IEEE Transactions on Automatic Control

VII. Adding States to $\mathscr{C}_d$

Whenever Algorithm 2 returns a non-empty solution, the FSCs are proper (Definition 3.2). In this case, there is a nonzero probability of visiting a state in a ϕ−feasible recurrent set of $\mathscr{SG}^{\phi}$ under FSCs $\mathscr{C}_d, \mathscr{C}_a$.

It might be the case that the probability of satisfying the LTL formula can be increased by adding states to $\mathscr{C}_d$. In doing so, it is important to ensure that adding states to $\mathscr{C}_d$ does not decrease this satisfaction probability when compared to the satisfaction probability for $\mathscr{C}_d$ with fewer states. This lends itself to a policy iteration like approach. Policy iteration [13] is a procedure that alternates between policy evaluation and policy improvement until convergence to a Stackelberg equilibrium. The policy evaluation step involves solving a Bellman equation, while the policy improvement step then 'greedily' chooses a policy that maximizes the satisfaction probability.

In this section, we will assume that the size of $\mathscr{C}_a$ is fixed. Intuititively, this means that the defender is aware of the capabilities of the adversary. However, the defender does not know the transition probabilities in $\mathscr{C}_a$, which means it needs to determine the transition probabilities of its own FSC in order to be robust against the worst-case transition probabilities between states in $\mathscr{C}_a$. Future work will seek to address the situations when the defender knows the nature of the strongest possible adversary (this will correspond to $|G_a| \leq G_A$), and when the defender does not know anything about the abilities of an adversary.

We will work with the 'value' of a state $g_d \in G_d$ in $\mathscr{C}_d$. Denote this by $V_{g_d}(\mathfrak{s})$. From the defender's perspective, the transition probabilities in $\mathscr{C}_d$ are influenced by its *belief* of the state of $\mathscr{SG}^{\phi}$ under $\mathscr{C}_d$ and $\mathscr{C}_a$. The *belief* is a (prior) probability distribution over the states of the POSG. Then, the value of $g_d \in G_d$ under belief $b = \{b_1, \ldots, b_{|S|}\}$ can be written as $V_{g_d}(b) = \sum_i b_i V_{g_d}(s_i)$. The value function for a belief b is then given by

$$V(b) := \max_{g_d} V_{g_d}(b) \tag{12}$$

Figure 1 shows $V(b)$ for a two-state POSG with $|G_d| = 3$.

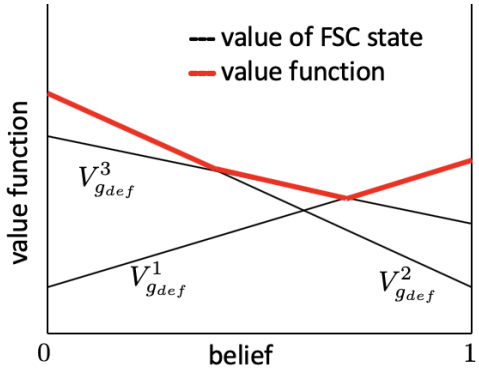


Figure 1: Value function of a two-state POSG with three states in $\mathscr{C}_d$. The value of each FSC state is linear in the belief (black lines). The value function is the point-wise maximum of the values of the FSC states (red curve).

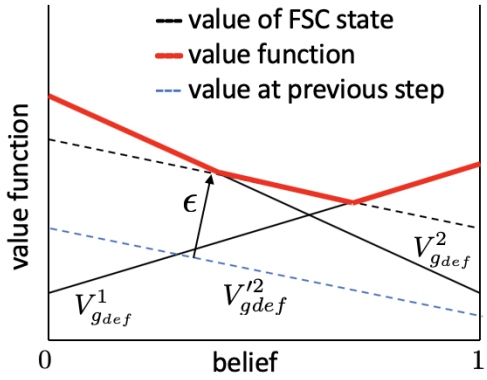


Figure 2: Robust linear program for state 2 of $\mathscr{C}_d$. Improved vector $V'^2_{g_d} + \epsilon$ is tangent to the one-step look-ahead value function.

When working with defender and adversary FSCs of fixed sizes, the value iteration in Algorithm 3 terminates when a (local) equilibrium is reached. This means there is no choice of transition probabilities in $\mathscr{C}_d$ that will improve the satisfaction probability for some belief state(s). This probability can be improved by adding states to $\mathscr{C}_d$ (since $|G_a|$ is fixed). The value of a belief b (Equation (12)) is the point-wise maximum of the value at each state of $\mathscr{C}_d$, which themselves are linear functions of the belief state. Therefore, at equilibrium, $V(b)$ will satisfy:

$$V(b) = \max_{u_d} \min_{u_a} \sum_{o \in \mathscr{O}_d} \mathbb{P}(o|b) V(b_o^{u_d u_a}) \tag{13}$$

$$\text{where } \mathbb{P}(o|b) := \sum_s O_d(o|s) b(s), \tag{14}$$

$$b_o^{u_d u_a}(s') := \sum_s \mathbb{T}(s'|s, u_d, u_a) \frac{O_d(o|s) b(s)}{\sum_{o \in \mathscr{O}_d} O_d(o|s) b(s)} \tag{15}$$

This set of equations is not easy to solve since the belief takes values in $[0,1]$. However, Equation (13) results in a point-wise improvement of the value function, until an optimum is reached. We will need the following definitions.

Definition 7.1 (Tangent FSC State): An FSC state g_d is *tangent* to the one-step look-ahead value function $V(b)$ in Equation (13) if $V(b) = V_{g_d}(b)$ at state b.

Definition 7.2 (Improved FSC State): A state $g_d \in G_d$ is *improved* if transition probabilities associated with that state are changed in a way that increases V_{g_d}.

For the setting where there are two agents with competing objectives, the problem of determining a policy μ_d that achieves an improvement in V_{g_d} under any adversary policy μ_a can be posed as a robust linear program [34], presented in Equations (16)-(19).

When $\epsilon > 0$, an improvement in the value of the FSC state (by ϵ) can be achieved. This is because there exists a convex combination of value vectors of the one-step look-ahead value function that dominates the present value of the FSC state [35]. The procedure is carried out for each $g_d \in G_d$, until no further improvement in the transition probabilities in $\mathscr{C}_d$ is possible. At this stage, the robust linear program yields $\epsilon = 0$ for every

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2020.3039484, IEEE Transactions on Automatic Control

$$\max_{\epsilon,\mu_d} \quad \epsilon \tag{16}$$

$$\text{subject to:} \quad V_{g_d}(\mathfrak{s}) + \epsilon \leq \sum_{\mathfrak{s}',o,o',g_d',g_a',u_d,u_a} \Big(V_{g_d'}(\mathfrak{s}') O_d(o|s) \mu_d(g_d', u_d | g_d, o) O_a(o'|s) \times \mu_a(g_a', u_a | g_a, o') \times \mathbb{T}^{\phi}((s',q')|(s,q),u_d,u_a) \Big) \quad \forall s, \forall \mu_a(g_a', u_a | g_a, o') \tag{17}$$

$$\sum_{g',u_*} \mu_*(g_*', u_* | g_*, o_*) = 1 \qquad \forall o_*; * \in \{d, a\} \tag{18}$$

$$\mu_*(g_*', u_* | g_*, o_*) \geq 0 \qquad \forall o_*, g_*', u_*; * \in \{d, a\} \tag{19}$$

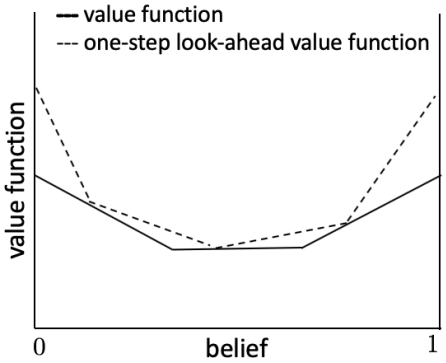


Figure 3: At a local equilibrium, all states are tangent to the one-step look-ahead value function.

$g_d \in G_d$. The following result generalizes Theorem 2 in [35] to a partially observable environment that includes an adversarial agent.

Proposition 7.3: The policy iteration procedure has reached a local equilibrium if and only if all the states $g_d \in G_d$ are tangent to $V(b)$.

Proof: The robust LP aims to maximize the improvement that can be achieved in the value of each state in $\mathscr{C}_d$. From the preceding discussion, and from Definition 7.1, a translation of the value vector of a state g_d by $\epsilon > 0$ will make it tangent to the one-step look-ahead value function. By a similar argument, $\epsilon = 0$ for each $g_d \in G_d$ indicates that improvement in the value of an FSC state will not be possible if it is already tangent to the one-step look-ahead value function. This is shown in Figures 2 and 3. ■

When $\epsilon = 0$ for each g_d, the satisfaction probability can be improved by adding states to $\mathscr{C}_d$ in a 'principled way'. Let $MaxNewStates$ denote the maximum number of states that can be added to $\mathscr{C}_d$, and let $\{b_k\} := B$ denote the set of belief states satisfying $V(b_k) = V_{g_d}(b)$ for each g_d from Equation (16).

Algorithm 4 presents a procedure to add states to the defender FSC to improve the satisfaction probability. Lines $6-11$ determine the one-step look-ahead beliefs. A new state is added to $\mathscr{C}_d$ if the defender 'believes' that the probability of satisfying the LTL formula from these states is higher than that from the current belief state (Lines $13-18$). Lines 13 and 14 respectively form the policy evaluation and policy improvement steps of the policy iteration. Edges from new states in the FSC are directed towards the action and FSC state that maximize the value function $V(b)$ in a deterministic manner (Line 16). Values of probabilities and transitions to and from the added state will be adjusted as other FSC states are improved.

Algorithm 4 Adding states to FSC $\mathscr{C}_d$

Input: Set of belief states $\{b_k\} := B$; $MaxNewStates$
Output: Set of improved states in FSC $\mathscr{C}_d$

1: $NewStateNumber \leftarrow 0$
2: **while** $B \neq \emptyset$ **do**
3: Choose $b \in B$
4: $B := B \setminus b$
5: $Ahead := \emptyset$
6: **for** $(u_d, u_a, o_d) \in U_d \times U_a \times \mathscr{O}_d$ **do**
7: **if** $\mathbb{P}(o|b) > 0$ in Equation (14) **then**
8: Determine $b_o^{u_d u_a}(s')$ from Equation (15)
9: $Ahead = Ahead \cup \{b_o^{u_d u_a}\}$
10: **end if**
11: **end for**
12: **for** $b_a \in Ahead$ **do**
13: Determine $V(b_a)$ from Equations (13, 14, 15)
14: Note maximizers u_d^*, g_d^* (Eqns. (13), (12))
15: **if** $V(b_a) > V(b)$ **AND** $NewStateNumber < MaxNewStates$ **then**
16: Add state, g_{new} to $\mathscr{C}_d$ with $\mu_d(g_d^*, u_d^* | g_{new}, o) = 1 \ \forall o \in \mathscr{O}_d$
17: $NewStateNumber \leftarrow NewStateNumber + 1$
18: **end if**
19: **end for**
20: **end while**

The procedure yields a new $\mathscr{C}_d$, from which candidate FSC structures can be found using Algorithm 2. The defender policy to maximize the probability of satisfying the LTL formula under any $\mathscr{C}_a$ of fixed size can be determined by Algorithm 3 or the robust linear program in Eqn. (16).

Proposition 7.4: Algorithm 4 terminates in finite time.

Proof: This follows from the facts the sets B, $\mathscr{O}_d$, U_d, and U_a have finite cardinality, and the one-step look-ahead values in Equation (13) are upper bounded. ■

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2020.3039484, IEEE Transactions on Automatic Control

VIII. Examples

This section presents an example and experiments that illustrate our approach.

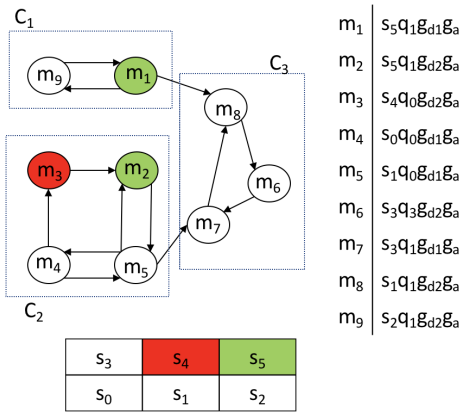


Figure 4: Clockwise, from *top-left*: Global Markov chain (GMC) for initial defender and adversary FSC structures-green states (m_1&m_2) must be visited infinitely often, and state in red (m_3) must be visited finitely often in steady-state; GMC state $m_i \in S \times Q \times G_d \times G_a$; State-space for $M = 3, N = 2$ showing unsafe (s_4) and target (s_5) states.

Example 8.1:

For this example, the state space is an $M \times N$ grid, $S := \{s_i : i = x + My, x \in \{0,\dots,M-1\}, y \in \{0,\dots,N-1\}\}$. The defender's actions are $U_d = \{R, L, U, D\}$ denoting right, left, up, and down, and the actions of the adversary are $U_a = \{A, NA\}$, denoting attack, and not attack respectively. The observations of both agents are $\mathcal{O}_d = \mathcal{O}_a = \{correct, wrong\}$, such that: $O_d(correct|s_i) = 0.8 = 1 - O_d(wrong|s_i)$, and $O_a(correct|s_i) = 0.6 = 1 - O_a(wrong|s_i)$. $\mathcal{O}_d$ and $\mathcal{O}_a$ are probabilities that the agents sense that their observation of the state is indeed the correct state or not. That is, $\mathbb{P}(o_i = s_i)$ or $\mathbb{P}(o_i \neq s_i)$.

Let $\mathcal{AP} = \{\texttt{obs}, \texttt{tar}\}$, denoting obstacle and target respectively. Then, if $\phi = \mathbf{GF}\texttt{tar} \wedge \mathbf{G}\neg\texttt{obs}$, the corresponding DRA will have two states q_0, q_1, with $F = (\{\emptyset\}, \{q_1\})$. Transition probabilities for $(u_d, u_a) = (R, NA)$ and (R, A) are defined below. Let N_{s_i} denote the neighbors of s_i.

$$\mathbb{T}(s_j|s_i, R, NA) = \begin{cases} 0.8 & j = i+1,\ (i+1) \not\equiv 0 \bmod M \\ \frac{0.2}{|N_{s_i}|} & (s_j \in \{s_i\} \cup N_{s_i} \setminus \{s_{i+1}\}), \\ & (i+1) \not\equiv 0 \bmod M \\ 1 & j = i \text{ and } (i+1) \equiv 0 \bmod M \end{cases}$$

$$\mathbb{T}(s_j|s_i, R, A) = \begin{cases} 0.6 & j = i+1,\ (i+1) \not\equiv 0 \bmod M \\ \frac{0.4}{|N_{s_i}|} & (s_j \in \{s_i\} \cup N_{s_i} \setminus \{s_{i+1}\}), \\ & (i+1) \not\equiv 0 \bmod M \\ 1 & j = i \text{ and } i+1 \equiv 0 \bmod M \end{cases}$$

Notice that in the above equations, the probability of the agent moving to the 'correct' next state for a particular defender action is larger for the adversary action NA than for the adversary action A. Further, in this case, if the defender is in a square along the right edge of the grid, then the action R does not result in a change of state. The probabilities for other action pairs can be defined similarly.

For this example, let $M = 3$ and $N = 2$. Then, $|S| = 6$. Let s_4 be an unsafe state, and s_5 be the goal state. This is indicated in Figure 4. Let $|G_d| = 2$, $|G_a| = 1$ for the FSCs. Assume that for some initial structures $\mathscr{I}_d^0, \mathscr{I}_a^0$ the GMC is given by Figure 4. The figure also indicates the states in terms of its individual components. Assume that the LTL formula ϕ is such that the states in green denote those that have to be visited infinitely often in steady state, while those in red must be avoided. Therefore $(L^\phi, K^\phi) = \{(\{\emptyset\}, \{m_1\}), (\{m_3\}, \{m_2\})\}$. The boxes C_1, C_2, C_3 indicate the communicating classes of the graph.

From Algorithm 2, for C_1, $Bad = \{m_8\}, Good = \{m_1\}$. For $m_1 \to m_8$, Eqn. (6) is true for all u_a and $u_d = \{D, L\}$. Thus, $\mathscr{I}_d(g', u_d|g, o) \leftarrow 0$ for $o = \{correct, wrong\}$. For $m_9 \to m_1$, since Eqn. (7) does not hold for $R, D \in U_d$, $\mathscr{I}_a(\cdot)$ is unchanged. Then, m_1 is recurrent in $\mathscr{G}_{new}$. For C_2, $Bad = \{m_3, m_7\}, Good = \{m_2\}$. Like for C_1, $\mathscr{I}_a(\cdot)$ remains unchanged, since (7) does not hold for $D \in U_d$. For $m_5 \to m_7$, $\mathscr{I}_d(g', u_d|g, o) \leftarrow 0 \forall u_d \in U_d \setminus D$. A similar conclusion is drawn for $m_4 \to m_3$. Then, m_2 will be recurrent in $\mathscr{G}_{new}$. For C_3, since $Bad = Good = \emptyset$, no structure is added to $\mathscr{I}$. Notice that these FSCs satisfy Proposition 4.1.

This example also shows the limitations of Algorithm 2. From the $M \times N$ grid, there is a policy that takes the defender from any $s \in S \setminus \{s_4\}$ to s_5 with probability 1. However, for FSCs of small size, the initial state of the defender might result in the Algorithm reporting that no solution was found, even if there exists a feasible solution.

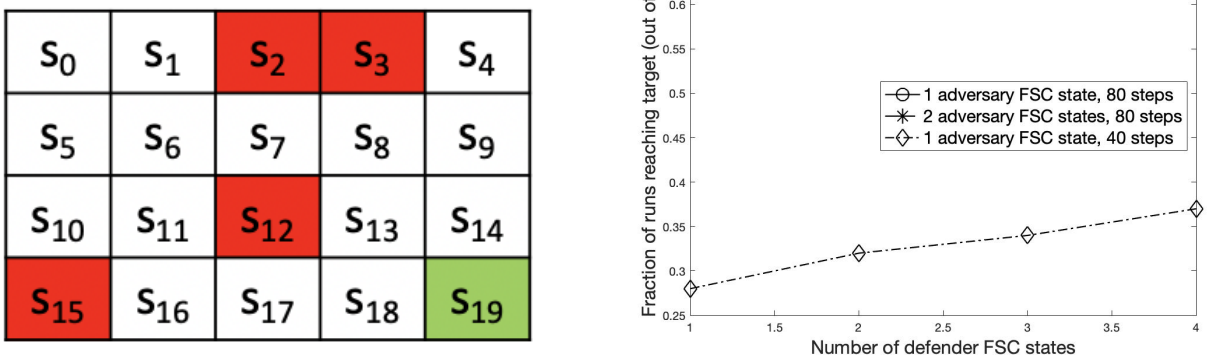
Example 8.2: Consider the model of Example 8.1, with $M = 5$, $N = 4$. A representation of the environment is shown in Figure 5a. Like in Example 8.1, the LTL formula to be satisfied is $\phi = \mathbf{GF}\texttt{tar} \wedge \mathbf{G}\neg\texttt{obs}$. The observation function of the defender is modified so that for a state s where $\mathscr{L}(s) = \texttt{obs}$ or $\mathscr{L}(s) = \texttt{tar}$, $O_d(correct|s) = 1$. That is, the defender recognizes an obstacle or the target correctly with probability one. Our experiments compute the probability of reaching the target under limited sensing capabilities of the agents with FSCs having different number of states. In each case, we assume that $|G_d| \geq |G_a|$. The number of states in the GMC varies from 60 to 480.

$\lvert\mathbf{G_d}\rvert$	$\lvert\mathbf{G_a}\rvert = \mathbf{1}$: $\mathbf{V(s_0)}$ [Std. Dev.]	$\lvert\mathbf{G_a}\rvert = \mathbf{2}$: $\mathbf{V(s_0)}$ [Std. Dev.]
1	0.53 [0.04]	0.45 [0.05]
2	0.55 [0.05]	0.45 [0.06]
3	0.56 [0.09]	0.47 [0.08]
4	0.57 [0.10]	0.48 [0.12]

Table I: Satisfaction probability and standard deviation (over 100 trials) of reaching the target state s_{19} from s_0 for LTL formula $\phi = \mathbf{GF}\texttt{tar} \wedge \mathbf{G}\neg\texttt{obs}$ starting from s_0 for varying number of defender FSC states $|G_d|$, when number of states in adversary FSC, $|G_a| = 1$ and $|G_a| = 2$.

Table I shows the average satisfaction probability and standard deviation of reaching the target state s_{19} starting from s_0 (expressed in terms of the value of the state from Equation (10)) when the adversary FSC has one and two states. Higher values of the standard deviation could be due to the fact that in some cases, Algorithm 3 may terminate before $V(s_0)$ is updated enough number of times.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2020.3039484, IEEE Transactions on Automatic Control



(a) (b)

Figure 5: The agent aims to satisfy the LTL formula $\phi = \mathbf{GF}\texttt{tar} \wedge \mathbf{G}\neg\texttt{obs}$ in the presence of an adversary, in a partially observable environment. The environment is the grid-world in Figure 5a. The states in red indicate the presence of an obstacle, the state in green is the target state, and the agent starts in state s_0. The agents' actions are determined by their observations of the state. Assume that the defender FSC has at least as many states as the adversary FSC, i.e. $|G_d| \geq |G_a|$. Figure 5b shows the fraction of runs (out of 100) when the agent reaches the target within 80 steps and 40 steps. This number is higher when $|G_d|$ is larger, for a fixed value of $|G_a|$ ($-\texttt{o}-$, $-*-$, and $-\Diamond-$ curves). For the same $|G_d|$, the fraction of successful runs is higher when $|G_a|$ is lower ($-\texttt{o}-$ and $-*-$ curves). The fraction of successful runs is also higher when the agent is allowed more steps to reach the target ($-\texttt{o}-$ and $-\Diamond-$ curves).

$\|G_d\|$	Benign Baseline	Adv. Baseline	Adv.-Aware (ours)
1	0.69	0.35	**0.53**
2	0.70	0.35	**0.55**
3	0.73	0.38	**0.56**
4	0.75	0.39	**0.57**

Table II: Comparison of probabilities of satisfying LTL formula $\phi = \mathbf{GF}\texttt{tar} \wedge \mathbf{G}\neg\texttt{obs}$ starting from s_0 in the presence and absence of an adversary. The first column lists the number of defender FSC states. Subsequent columns enumerate satisfaction probabilities in the following scenarios: i) absence of adversary (Benign Baseline [20]); ii) using a defender policy that was synthesized without an adversary, but realized in the presence of an adversary (Adversarial Baseline); iii) using a defender policy designed assuming the presence of an adversary (Adversary-Aware Design- **our approach**). Although the benign baseline gives the highest satisfaction probability, the same baseline when used in the presence of an adversary results in a much lower satisfaction probability. In comparison, our *Adversary-Aware Design* approach results in a higher satisfaction probability than the 'Adversarial Baseline' where we use a defender policy designed to account for adversarial behavior. We assume that the adversary FSC has one state, so that the GMC with and without the adversary FSC will have the same number of states.

Table II compares the probabilities of satisfying the LTL objective $\phi = \mathbf{GF}\texttt{tar} \wedge \mathbf{G}\neg\texttt{obs}$ starting from s_0 in the presence and absence of an adversary. We compare our approach with a baseline, which is a defender policy synthesized in the absence of an adversary. The GMC for the case without an adversary is constructed using the approach of [20]. This baseline defender policy in then realized in the presence of an adversary, and we compare it with our method of synthesizing a defender policy assuming the presence of an adversary. Although the highest satisfaction probability is got while using the baseline policy, when this baseline is used in the presence of an adversary, we obtain a much lower satisfaction probability. In comparison, our *adversary-aware defender policy* results in a higher satisfaction probability than when using the baseline in the presence of the adversary. We note that for this comparison, the adversary FSC has one state, i.e., $|G_a| = 1$, so that the GMCs with and without the adversary FSC have the same number of states.

Figure 5b shows the fraction of sample paths when the agent reaches the target for the first time. After this, since the agent is in a recurrent set, it will continue to visit states in this set with probability one. The following observations can be drawn from Figure 5b. First, for a fixed $|G_a|$, the fraction of runs when the agent successfully reaches the target increases as $|G_d|$ increases. Second, for a fixed $|G_d|$, the probability of satisfying ϕ is higher for a smaller $|G_a|$. Third, the fraction of successful runs improves with allowing the agent more steps to reach the target. One reason for the first two observations is that the number of states in an FSC models the 'memory' available to the agent. The defender can play better when it has more FSC states, or when the adversary has fewer FSC states. While our results agree with intuition, a caveat is that these numbers depend on the agents' observations,

$\mathcal{O}_d$ and $\mathcal{O}_a$. Here, the observations of the agents is an indication of whether the state is the actual state the defender is in. This could be a reason for fewer successful runs when the agent is allowed a maximum of 40 steps versus the case when it is allowed 80 steps.

IX. Related Work

A large body of work studies classes of problems that are relevant to this paper. These can be divided into three broad categories: *i)* synthesis of strategies for systems represented as an MDP that has to additionally satisfy a TL formula; *ii)* synthesis of strategies for POMDPs; *iii)* synthesis of defender and adversary strategies for an MDP under a TL constraint. While there has been recent work on the synthesis of controllers for POMDPs under TL specifications, these have largely been restricted to the single-agent case, and do not address the case when there might be an adversary with a competing objective.

Approaches that address the satisfaction of TL constraints for problems in motion-planning include hierarchical control [36], ensuring probabilistic satisfaction guarantees [15], and sensing-based strategies [9]. Controller synthesis for deterministic linear systems to ensure that the closed-loop system will satisfy an LTL formula is studied in [37]. The authors of [38] propose methods to synthesize a robust control policy that satisfies an LTL formula for a system represented as an MDP whose transitions are not exactly known, but are assumed to lie in a set. For MDPs under an LTL specification, a partial ordering on the states is leveraged to solve controller synthesis as a receding horizon problem in [39]. The synthesis of an optimal control policy that maximizes the probability of an MDP satisfying an LTL formula that additionally minimizes the cost between satisfying instances is studied in [10]. This is computed by determining *maximal end components* in an MDP. However, this approach will not work in the partially observable setting, where policies will depend on an observation of the state [40]. The synthesis of joint control and sensing strategies for discrete systems with incomplete information and sensing is presented in [41]. The setting of [10] in the presence of an adversary with competing objectives was presented in [33].

A policy in a POSG (or POMDP) at time t depends on actions and observations at all previous times. A memoryless policy, on the other hand, only depends on the current state. For fully observable stochastic games, it is possible to always find memoryless policies that are optimal. However, a policy with memory could perform much better than a memoryless policy for POSGs. One way of determining policies for a POSG is to keep track of the entire execution, observation, and action histories, which can be abstracted into determining a sufficient statistic for the POSG execution. One example is the *belief state*, which reflects the probability that the agent is in some state, based on receiving observations from the environment. Updating the *belief state* at every time step only requires knowledge of the previous belief state and the most recent action and observation. Thus, the belief states form the states of an MDP [42], which is more amenable to analysis [13] than a POMDP. However, the belief state is uncountable, and will hinder development of exact algorithms to determine optimal strategies.

Synthesis of memoryless strategies for POMDPs in order to satisfy a specification was shown to be NP-hard and in PSPACE in [21]. In [43], a discretization of the belief space was carried out *apriori*, resulting in a fully observable MDP. However, this approach is not practical if the state space is large [26]. The complexity of determining a strategy for maximizing the probability of satisfaction of parity objectives was shown to be undecidable in [44]. However, determining finite-memory strategies for the qualitative problem of parity objective satisfaction was shown to be EXPTIME-complete in [45].

Dynamic programming for POSGs for the finite horizon setting was studied in [46]. When agents cooperate to earn rewards, the framework is called a decentralized-POMDP (Dec-POMDP). The infinite horizon case for Dec-POMDPs was presented in [47], where the authors proposed a bounded policy iteration algorithm for policies represented as joint FSCs. A complete and optimal algorithm for deterministic FSC policies for DecPOMDPs was presented in [48]. Optimization techniques for 'fixed-size controllers' to solve Dec-POMDPs were investigated in [49]. A survey of recent research in Dec- POMDPs is presented in [50].

The satisfaction of an LTL formula in partially observable adversarial environments was presented for the first time in [1]. The authors used FSCs to represent the policies of the agents, and proposed an algorithm that yielded defender FSCs of fixed size satisfying the LTL formula under any adversary FSC of fixed size. The authors also broadened the scope of this problem to continuous state environments, settings when the adversary could potentially tamper with clocks that keep track of time in the environment, and where an additional privacy constraint had to be satisfied in [51], [52], [53].

X. Conclusion

This paper demonstrated the use of FSCs in order to satisfy an LTL formula in a partially observable environment, in the presence of an adversary. The FSCs represented agent policies, and these were composed with a POSG representing the environment to yield a fully observable MC. We showed that the probability of satisfaction of the LTL formula was equal to the probability of reaching recurrent classes of this MC. We subsequently presented a procedure to determine defender and adversary controllers of fixed sizes that result in nonzero satisfaction probability of the LTL formula, and proved its soundness. Maximizing the satisfaction probability was related to reaching a Stackelberg equilibrium of a stochastic game involving the agents through a value-iteration based procedure. Finally, we showed a means to add states to the defender FSC in a principled way in order to improve the satisfaction probability for adversary FSCs of fixed sizes.

In Section VII, when adding states to $\mathcal{C}_d$, we assumed that the size of $\mathcal{C}_a$ was fixed. Future work will seek to

relax this assumption, and study cases when the defender has limited knowledge of the abilities of an adversary. To address the challenge of *state explosion*, we will investigate state aggregation techniques [54], [55], [56] for the product-POSG and the GMC. This will enable solutions of more complex problems.

REFERENCES

[1] B. Ramasubramanian, A. Clark, L. Bushnell, and R. Poovendran, "Secure control under partial observability with temporal logic constraints," in *Proc. American Control Conf.*, 2019, pp. 1181–1188.

[2] R. Baheti and H. Gill, "Cyber-physical systems," *The Impact of Control Technology*, vol. 12, no. 1, pp. 161–166, 2011.

[3] J. E. Sullivan and D. Kamensky, "How cyber-attacks in Ukraine show the vulnerability of the US power grid," *The Electricity Journal*, vol. 30, no. 3, pp. 30–35, 2017.

[4] Y. Shoukry, P. Martin, P. Tabuada, and M. Srivastava, "Non-invasive spoofing attacks for anti-lock braking systems," in *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 2013, pp. 55–72.

[5] J. Slay and M. Miller, "Lessons learned from the Maroochy water breach," in *International Conference on Critical Infrastructure Protection*. Springer, 2007, pp. 73–82.

[6] J. P. Farwell and R. Rohozinski, "Stuxnet and the future of cyber war," *Survival*, vol. 53, no. 1, pp. 23–40, 2011.

[7] C. Baier and J.-P. Katoen, *Principles of Model Checking*. MIT Press, 2008.

[8] M. Lahijanian, S. B. Andersson, and C. Belta, "Formal verification and synthesis for discrete-time stochastic systems," *IEEE Transactions on Automatic Control*, vol. 60, no. 8, pp. 2031–2045, 2015.

[9] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, "Where's Waldo?: Sensor-based temporal logic motion planning," in *International Conference on Robotics and Automation*, 2007, pp. 3116–3121.

[10] X. Ding, S. L. Smith, C. Belta, and D. Rus, "Optimal control of MDPs with linear temporal logic constraints," *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1244–1257, 2014.

[11] A. Cimatti, E. Clarke, F. Giunchiglia, and M. Roveri, "Nusmv: A new symbolic model verifier," in *International Conference on Computer Aided Verification*. Springer, 1999, pp. 495–499.

[12] M. Kwiatkowska, G. Norman, and D. Parker, "Prism 4.0: Verification of probabilistic real-time systems," in *International Conference on Computer Aided Verification*. Springer, 2011, pp. 585–591.

[13] D. P. Bertsekas, *Dynamic Programming and Optimal Control 4th Edition, Volumes I and II*. Athena Scientific, 2015.

[14] M. L. Puterman, *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[15] M. Lahijanian, S. B. Andersson, and C. Belta, "Temporal logic motion planning and control with probabilistic satisfaction guarantees," *IEEE Transactions on Robotics*, vol. 28, pp. 396–409, 2012.

[16] S. Temizer, M. Kochenderfer, L. Kaelbling, T. Lozano-Pérez, and J. Kuchar, "Collision avoidance for unmanned aircraft using MDPs," in *AIAA Guidance, Navigation, and Control Conference*, 2010.

[17] L. Niu and A. Clark, "Secure control under LTL constraints," in *Proc. of the American Control Conference*, 2018, pp. 3544–3551.

[18] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT Press, 2005.

[19] R. Sharan and J. Burdick, "Finite state control of POMDPs with LTL specifications," in *Proceedings of the American Control Conference*, 2014, pp. 501–508.

[20] R. Sharan, "Formal methods for control synthesis in partially observed environments: Application to autonomous robotic manipulation," Ph.D. dissertation, California Institute of Technology, 2014.

[21] N. Vlassis, M. L. Littman, and D. Barber, "On the computational complexity of stochastic controller optimization in POMDPs," *ACM Transactions on Computation Theory*, vol. 4, no. 4, p. 12, 2012.

[22] A. R. Cassandra, L. P. Kaelbling, and J. A. Kurien, "Acting under uncertainty: Discrete bayesian models for mobile-robot navigation," in *International Conference on Intelligent Robots and Systems*, vol. 2, 1996, pp. 963–972.

[23] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[24] R. I. Brafman, "A heuristic variable grid solution method for POMDPs," in *AAAI/IAAI*, 1997, pp. 727–733.

[25] H. Kurniawati, D. Hsu, and W. S. Lee, "SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces." in *Robotics: Science and Systems*, 2008.

[26] H. Yu and D. P. Bertsekas, "On near optimality of the set of finite-state controllers for average cost POMDP," *Mathematics of Operations Research*, vol. 33, no. 1, pp. 1–11, 2008.

[27] D. Fudenberg and J. Tirole, *Game Theory*. MIT Press, 1991.

[28] S. P. Meyn and R. L. Tweedie, *Markov chains and stochastic stability*. Springer Science & Business Media, 2012.

[29] E. A. Hansen and R. Zhou, "Synthesis of hierarchical finite-state controllers for POMDPs." in *ICAPS*, 2003, pp. 113–122.

[30] R. Tarjan, "Depth-first search and linear graph algorithms," *SIAM Journal on Computing*, vol. 1, no. 2, pp. 146–160, 1972.

[31] H. Royden and P. Fitzpatrick, *Real Analysis*. Prentice Hall, 2010.

[32] V. Conitzer, "On Stackelberg mixed strategies," *Synthese*, vol. 193, pp. 689–703, 2016.

[33] L. Niu and A. Clark, "Optimal secure control with LTL constraints," *IEEE Transactions on Automatic Control*, 2019.

[34] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust optimization*. Princeton University Press, 2009.

[35] P. Poupart and C. Boutilier, "Bounded finite state controllers," in *Neural Information Processing Systems*, 2004, pp. 823–830.

[36] G. E. Fainekos, A. Girard, H. Kress-Gazit, and G. J. Pappas, "Temporal logic motion planning for dynamic robots," *Automatica*, vol. 45, no. 2, pp. 343–352, 2009.

[37] M. Kloetzer and C. Belta, "A fully automated framework for control of linear systems from temporal logic specifications," *IEEE Transactions on Automatic Control*, vol. 53, no. 1, pp. 287–297, 2008.

[38] E. M. Wolff, U. Topcu, and R. M. Murray, "Robust control of uncertain MDPs with LTL specifications," in *Proceedings of the IEEE Conference on Decision and Control*, 2012, pp. 3372–3379.

[39] T. Wongpiromsarn, U. Topcu, and R. M. Murray, "Receding horizon temporal logic planning," *IEEE Transactions on Automatic Control*, vol. 57, no. 11, pp. 2817–2830, 2012.

[40] D. Sadigh, E. S. Kim, S. Coogan, S. S. Sastry, and S. A. Seshia, "A learning based approach to control synthesis of MDPs for LTL specifications," in *Proceedings of the IEEE Conference on Decision and Control*, 2014, pp. 1091–1096.

[41] J. Fu and U. Topcu, "Synthesis of joint control and active sensing strategies under temporal logic constraints," *IEEE Transactions on Automatic Control*, vol. 61, no. 11, pp. 3464–3476, 2016.

[42] R. D. Smallwood and E. J. Sondik, "The optimal control of POMDPs over a finite horizon," *Operations Research*, vol. 21, no. 5, pp. 1071–1088, 1973.

[43] T. Wongpiromsarn and E. Frazzoli, "Control of probabilistic systems under dynamic, partially known environments with temporal logic specifications," in *Proceedings of the IEEE Conference on Decision and Control*, 2012, pp. 7644–7651.

[44] K. Chatterjee, L. Doyen, and T. A. Henzinger, "A survey of partial-observation stochastic parity games," *Formal Methods in System Design*, vol. 43, no. 2, pp. 268–284, 2013.

[45] K. Chatterjee, L. Doyen, S. Nain, and M. Y. Vardi, "The complexity of partial-observation stochastic parity games with finite-memory strategies," in *International Conference on Foundations of Software Science and Computation Structures*. Springer, 2014, pp. 242–257.

[46] E. A. Hansen, D. S. Bernstein, and S. Zilberstein, "Dynamic programming for partially observable stochastic games," in *AAAI*, vol. 4, 2004, pp. 709–715.

[47] D. S. Bernstein, E. A. Hansen, and S. Zilberstein, "Bounded policy iteration for decentralized POMDPs," in *International Joint Conference on Artificial Intelligence*, 2005, pp. 52–57.

[48] D. Szer and F. Charpillet, "An optimal best-first search algorithm for solving infinite horizon DEC-POMDPs," in *European Conference on Machine Learning*. Springer, 2005, pp. 389–399.

[49] C. Amato, D. S. Bernstein, and S. Zilberstein, "Optimizing fixed-size stochastic controllers for POMDPs and decentralized POMDPs," *Autonomous Agents and Multi-Agent Systems*, pp. 293–320, 2010.

[50] F. A. Oliehoek and C. Amato, *A Concise Introduction to Decentralized POMDPs*. Springer, 2016.

[51] B. Ramasubramanian, L. Niu, A. Clark, L. Bushnell, and R. Poovendran, "Linear temporal logic satisfaction in adversarial environments using secure control barrier certificates," in *Proc. Conf. on Decision and Game Theory for Security*, 2019, pp. 385–403.

[52] L. Niu, B. Ramasubramanian, A. Clark, L. Bushnell, and R. Poovendran, "Control synthesis for cyber-physical systems to satisfy metric interval temporal logic objectives under timing and actuator

attacks," in *Proc. of the ACM/ IEEE International Conference on Cyber-Physical Systems (ICCPS)*, 2020, pp. 162–173.
[53] B. Ramasubramanian, L. Niu, A. Clark, L. Bushnell, and R. Poovendran, "Privacy-preserving resilience of cyber-physical systems to adversaries," in *Proc. IEEE Conf. on Decision and Control*, 2020.
[54] Z. Ren and B. H. Krogh, "State aggregation in MDPs," in *Proc. IEEE Conference on Decision and Control*, 2002, pp. 3819–3824.
[55] L. Li, T. J. Walsh, and M. L. Littman, "Towards a unified theory of state abstraction for MDPs." in *International Symposium on Artificial Intelligence and Mathematics*, 2006.
[56] E. M. Clarke, W. Klieber, M. Nováček, and P. Zuliani, "Model checking and the state explosion problem," in *LASER Summer School on Software Engineering*. Springer, 2011, pp. 1–30.

Bhaskar Ramasubramanian is a Postdoctoral Researcher in the Network Security Lab in the Department of Electrical and Computer Engineering at the University of Washington, Seattle. He received a B.Tech. degree in Electrical and Electronics Engineering from Visvesvaraya National Institute of Technology, Nagpur in 2009, an M. Tech. in Systems and Control Engineering from the Indian Institute of Technology, Bombay, in 2011, and a Ph.D. in Electrical and Computer Engineering from the University of Maryland, College Park, in 2018. His research interests are in the areas of security and control of cyberphysical systems, formal methods, and feedback-driven machine learning. He was a finalist for the Best Paper Award at ICCPS 2020.

Luyao Niu received the B.Eng. degree from the School of Electro-Mechanical Engineering, Xidian University, Xian, China, in 2013, and the M.Sc. degree from the Department of Electrical and Computer Engineering, Worcester Polytechnic Institute (WPI) in 2015. He has been working towards his Ph.D. in Electrical and Computer Engineering at WPI since 2016. His current research interests include optimization, game theory, and control and security in CPSs. He was the winner of the Best Student Paper Award at GameSec 2018 and a finalist for the Best Paper Award at ICCPS 2020.

Andrew Clark is an Assistant Professor in the Department of Electrical and Computer Engineering at Worcester Polytechnic Institute. He received the B.S. degree in Electrical Engineering, the M.S. degree in Mathematics from the University of Michigan, Ann Arbor, and the Ph.D. degree in Electrical Engineering from the University of Washington, Seattle, in 2007, 2008, and 2014 respectively. He is a recipient of the NSF CAREER Award in 2020. His other accolades include the IEEE/ IFIP William C. Carter award-winning paper (2010), the WiOpt Best Paper (2012), the WiOpt Student Best Paper (2014), and the GameSec Student Best Paper (2018). He was a finalist for the IEEE CDC 2012 Best Student-Paper and ICCPS 2020 Best Paper Awards. He received the University of Washington Center for Information Assurance and Cybersecurity (CIAC) Distinguished Research Award (2012) and Distinguished Dissertation Award (2014). His research interests include control and security of complex networks, submodular optimization, and control-theoretic modeling of network security threats.

Linda Bushnell (F '17) is a Research Professor in the Department of Electrical and Computer Engineering at the University of Washington (UW) - Seattle. She received her Ph.D. in Electrical Engineering from University of California - Berkeley in 1994, her M.A. in Mathematics from University of California - Berkeley in 1989, her M.S. and B.S. in Electrical Engineering from University of Connecticut - Storrs in 1987 and 1985 respectively. She also received her MBA from the UW Foster School of Business in 2010. Her research interests include networked control systems and secure-control. She is a Fellow of the IEEE for contributions to networked control systems. She is a Fellow of IFAC for contributions to the analysis and design of networked control systems. She is a recipient of the US Army Superior Civilian Service Award, NSF ADVANCE Fellowship, and IEEE Control Systems Society Distinguished Member Award. She has been a member of the IEEE since 1985, and a member of the IEEE CSS since 1990. She is currently the Treasurer of the American Automatic Control Council, Member of the Technical Board for the International Federation on Automatic Control, Associate Editor for Automatica and for the IEEE Transactions on Control of Network Systems.

Radha Poovendran (F '15) is a Professor in the Department of Electrical and Computer Engineering at the University of Washington (UW) - Seattle. He served as the Chair of the Electrical and Computer Engineering Department at UW for five years starting January 2015. He is the Director of the Network Security Lab (NSL) at UW. He is the Associate Director of Research of the UW Center for Excellence in Information Assurance Research and Education. He received the B.S. degree in Electrical Engineering and the M.S. degree in Electrical and Computer Engineering from the Indian Institute of Technology- Bombay and University of Michigan - Ann Arbor in 1988 and 1992, respectively. He received the Ph.D. degree in Electrical and Computer Engineering from the University of Maryland - College Park in 1999. His research interests are in the areas of wireless and sensor network security, control and security of cyber-physical systems, adversarial modeling, smart connected communities, control-security, games-security, and information theoretic security in the context of wireless mobile networks. He is a Fellow of the IEEE for his contributions to security in cyber-physical systems. He is a recipient of the NSA LUCITE Rising Star Award (1999), National Science Foundation CAREER (2001), ARO YIP (2002), ONR YIP (2004), and PECASE (2005) for his research contributions to multi-user wireless security. He is also a recipient of the Outstanding Teaching Award and Outstanding Research Advisor Award from UW EE (2002), Graduate Mentor Award from Office of the Chancellor at University of California - San Diego (2006), and the University of Maryland ECE Distinguished Alumni Award (2016). He was co-author of award-winning papers including IEEE/IFIP William C. Carter Award Paper (2010) and WiOpt Best Paper Award (2012).