Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Minimizing the Information Leakage Regarding High-Level Task Specifications

Michael Hibbard[2] * **Yagiz Savas**[2] * **Zhe Xu** † **Ufuk Topcu** *

* *Department of Aerospace Engineering, University of Texas at Austin, TX, USA (email:* {mwhibbard, yagiz.savas, utopcu}@utexas.edu*)*
† *Oden Institute for Computational Engineering and Sciences, University of Texas at Austin, TX, USA. (email:* zhexu@utexas.edu*)*

Abstract: We consider a scenario in which an autonomous agent carries out a mission in a stochastic environment while passively observed by an adversary. For the agent, minimizing the information leaked to the adversary regarding its high-level specification is critical in creating an informational advantage. We express the specification of the agent as a parametric linear temporal logic formula, measure the information leakage by the adversary's confidence in the agent's mission specification, and propose algorithms to synthesize a policy for the agent which minimizes the information leakage to the adversary. In the scenario considered, the adversary aims to infer the specification of the agent from a set of candidate specifications, each of which has an associated likelihood probability. The agent's objective is to synthesize a policy that maximizes the entropy of the adversary's likelihood distribution while satisfying its specification. We propose two approaches to solve the resulting synthesis problem. The first approach computes the exact satisfaction probabilities for each candidate specification, whereas the second approach utilizes the Fréchet inequalities to approximate them. For each approach, we formulate a mixed-integer program with a quasiconcave objective function. We solve the problem using a bisection algorithm. Finally, we compare the performance of both approaches on numerical simulations.

Keywords: Mission planning and decision making, Trajectory and Path Planning, Autonomous Mobile Robots.

1. INTRODUCTION

In environments where privacy and security concerns are of paramount importance, the ability of an agent to deceive an adversary regarding its specification is critical in creating an informational advantage. Pertinent environments for deceptive policies include military operations (Lloyd (2003)), criminal justice (Skolnick (1982)), and cybersecurity (Carroll and Grosu (2011)). We explore the concept of deception through the lens of minimizing the information leaked to an adversarial observer regarding the agent's high-level specification.

Specifically, we consider an autonomous agent operating in an environment while being passively observed by an adversary. We assume that both the agent and the adversarial observer have knowledge of a set of *specifications*. From this set, the agent maintains a secret *ground-truth specification*, *i.e.*, the specification that the agent actually seeks to satisfy. The adversarial observer attempts to infer the ground-truth specification based on the trajectories of the agent. The agent must behave in such a way as to prevent the adversary from inferring its ground-truth specification. By doing so, the agent may inhibit the adversarial observer from optimally allocating its resources towards preventing the satisfaction of the ground-truth specification.

Consider an autonomous agent that must deliver supplies to one of three possible locations, denoted A, B, and C. The set of candidate specifications for the agent is $\{$*deliver to A*, *deliver to B*, *deliver to C*$\}$. The agent's ground-truth specification is $\{$*deliver to C*$\}$. Although the agent need only travel to C to complete this specification, doing so makes it apparent to an observer that $\{$*deliver to C*$\}$ is the agent's ground-truth specification. Instead, the agent should travel to each possible location with an equal probability. By doing so, the adversarial observer cannot leverage these probabilities towards inferring which candidate specification is the ground-truth specification.

We develop a method for an autonomous agent to synthesize a policy satisfying the agent's ground-truth specification with a desired probability while also leading an adversarial observer to infer that each candidate specification is equally likely to be the ground-truth specification. We model the agent's behavior as a Markov decision process (MDP) (Puterman (2014)). MDPs are commonly used to model planning and acting in stochastic environments with nondeterministic action selection. Numerous methods exist to synthesize policies for MDPs, which resolve the nondeterminism by prescribing probability distributions for action selection.

To model the agent's specifications, we use parametric linear temporal logic (pLTL) (Chakraborty and Katoen (2014); Alur et al. (2001)). Standard linear temporal logic (LTL) allows for the formal expression of specifications related to the occurrence of events, causality between events, and the ordering of successive events (Baier and Katoen (2008)). pLTL extends LTL by introducing parameterized temporal operators, which allows specifications to be expressed over particular time horizons. We focus on the class of syntactically co-safe pLTL specifications, which can be satisfied in finite time (Xu et al. (2019)).

[1] This work was funded in part by the grants DARPA D19AP00004 and AFRL FA9550-19-1-0169.
[2] M. Hibbard and Y. Savas contributed equally to this work.

We assume that the adversarial observer assigns a likelihood probability to each candidate according to a simple averaging rule, and set the objective of the agent as maximizing the entropy of the adversary's likelihood distribution. The information-theoretic concept of entropy (Cover and Thomas (2012)) measures the average uncertainty of a random variable, an ideal measure for the task at hand. We propose two methods to solve the resulting synthesis problem. The first method exactly computes the probabilities that each specification is completed, which we formulate as a quasiconcave mixed-integer program (MIP) and solve using a bisection method. The exact solution method faces an exponential blow-up in the state space as a function of the number of candidate specifications. The second method we propose avoids the state-space blowup by instead using lower bounds for the probabilities that each specification is satisfied. However, because these probabilities are a conservative estimate, the approximate solution method may indicate that fewer specifications are satisfiable compared to the the exact solution method. We formulate this method as an MIP and again solve using a bisection method.

The challenge in synthesizing an exact solution for our problem is analogous to the difficulty of synthesizing policies for multi-agent systems with high-level task specifications. To combat the state-space explosion of these multi-agent problems, Menghi et al. (2018), for example, decentralizes the synthesis problem amongst each individual agent, while Maliah et al. (2017) does so in a privacy-preserving manner. Alternatively, Ulusoy et al. (2013) utilizes a depth-first search to consider only the relevant states. For the types of temporal logic specifications we consider, our approximate solution method likewise provides a process to avoid such an explosion in the state-space.

Recently, the works of Savas et al. (2019) and Karabag et al. (2019) focused on synthesizing policies that are either unpredictable or difficult for an adversarial observer to infer. These studies focused on the low-level actions rather than on the high-level specifications as we do. Inferring temporal logic formulas has been extensively studied. For example, Neider and Gavran (2018) inferred LTL properties classifying a labeled set of trajectories. Similarly, Xu and Julius (2019) consider privacy-preserving temporal logic inference, while Xu and Topcu (2019) consider temporal logic inference in the context of reinforcement learning. As for the inference of pLTL specifications, Xu et al. (2019) inferred pLTL formulas from a set of trajectories that was *informative* with respect to prior knowledge. As opposed to these studies, we seek to make the inference problem as difficult as possible for the observer.

2. PRELIMINARIES

Notation. We denote the set $\{1, 2, \ldots\}$ of natural numbers and the set $(-\infty, \infty)$ of real numbers by $\mathbb{N}$ and $\mathbb{R}$, respectively. For a given logical formula, $\top$ and $\bot$ denote that the formula is true and false, respectively. For a set S, we denote its power set by 2^S. Finally, for $N{\in}\mathbb{N}$, we denote the set $\{1, 2, \ldots, N\}$ by $[N]$.

2.1 Markov Decision Processes

Definition 1. A *Markov decision process* (MDP) is defined by the tuple $\mathcal{M}{=}(S, s_0, \mathcal{A}, \mathcal{P}, \mathcal{AP}, \mathcal{L})$ where S is a finite set of states, $\mathcal{A}$ is a finite set of actions, s_0 is a unique initial state, $\mathcal{P}{:}S{\times}\mathcal{A}{\times}S{\to}[0,1]$ is a transition function such that $\sum_{s'\in S}\mathcal{P}(s,a,s'){=}1$ for all $a{\in}\mathcal{A}$ and $s{\in}S$, $\mathcal{AP}$ is a set of atomic propositions, and $\mathcal{L}{:}S{\to}2^{\mathcal{AP}}$ is a labeling function.

We denote the transition probability $\mathcal{P}(s,a,s')$ by $\mathcal{P}_{s,a,s'}$. The size of an MDP is the number of triples $(s,a,s'){\in}S{\times}\mathcal{A}{\times}S$ in which $\mathcal{P}_{s,a,s'}{>}0$.

Definition 2. A *policy* π for an MDP $\mathcal{M}$ is a sequence $\pi{=}(d_1, d_2, d_3, \ldots)$ where each $d_t{:}S{\times}\mathcal{A}{\to}[0,1]$ is a mapping such that $\sum_{a\in\mathcal{A}} d_t(s,a){=}1$ for all $s{\in}S$. For an MDP $\mathcal{M}$, we denote the set of all admissible policies by $\Pi(\mathcal{M})$.

A *stationary* policy satisfies $\pi{=}(d_1, d_1, d_1, \ldots)$. We denote the probability of choosing an action $a{\in}\mathcal{A}$ in a state $s{\in}S$ under a stationary policy π by $\pi(s,a)$.

For an arbitrary length $L{\in}\mathbb{N}$, we refer to a sequence of states $\varrho^\pi{\triangleq}s_0 s_1 \ldots s_L$ generated in $\mathcal{M}$ under a policy $\pi{\in}\Pi(\mathcal{M})$ as a *trajectory*, which starts from the initial state s_0 and satisfies $\sum_{a_t\in\mathcal{A}} d_t(s_t,a_t)\mathcal{P}_{s_t,a_t,s_{t+1}}{>}0$ for all $0{\le}t{<}L$.

2.2 Parametric Linear Temporal Logic

Following Chakraborty and Katoen (2014), the syntax of parametric linear temporal logic (pLTL) is defined recursively as

$$\phi \triangleq \top \mid p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \bigcirc\phi \mid \phi_1 \mathcal{U} \phi_2 \mid \Diamond_{\sim i}\phi,$$

where p is an atomic proposition, $\neg$ and $\wedge$ stand for negation and conjunction, respectively, $\bigcirc$ and $\mathcal{U}$ are temporal operators representing "next" and "until", respectively, $\Diamond_{\sim i}$ is a parameterized temporal operator representing " *parameterized eventually*", where $\sim{\in}\{\ge,\le\}$, and $i{\in}\mathbb{N}$ is a temporal parameter. We recursively define the logical connective $\vee$ (disjunction), and temporal operators $\Diamond$ (eventually), $\Box$ (always), $\Box_{\sim i}$ (parameterized always) and $\mathcal{U}_{\sim i}$ (parameterized until) from the aforementioned operators (Chakraborty and Katoen (2014)). Furthermore, for $i_1{<}i_2$, we define the parameterized temporal operators $\Diamond_{[i_1,i_2]}$ and $\Box_{[i_1,i_2]}$ such that, for a formula ϕ, $\Diamond_{[i_1,i_2]}\phi = \Diamond_{\ge i_1}\phi \wedge \Diamond_{\le i_2}\phi$ and $\Box_{[i_1,i_2]}\phi = \Box_{\ge i_1}\phi \wedge \Box_{\le i_2}\phi$.

For an MDP $\mathcal{M}$ under a policy $\pi{\in}\Pi(\mathcal{M})$, a trajectory $\varrho^\pi{=}s_0 s_1 \ldots s_L$ generates a word $w^\pi{\triangleq}w_0 w_1 \ldots w_L$, where $w_k{=}\mathcal{L}(s_k)$ for all $0 \le k{\le}L$. For a pLTL formula ϕ and a trajectory ϱ^π at time index $k{\le}L$, the satisfaction relation $(w^\pi,k) \models \phi$ is defined recursively as

$$\begin{aligned}
(w^\pi,k) &\models p \quad \text{iff} \quad p \in \mathcal{L}(s_k),\\
(w^\pi,k) &\models \neg\phi \quad \text{iff} \quad (w^\pi,k) \not\models \phi,\\
(w^\pi,k) &\models \phi_1 \wedge \phi_2 \quad \text{iff} \quad (w^\pi,k) \models \phi_1\\
&\qquad\qquad \text{and} \quad (w^\pi,k) \models \phi_2,\\
(w^\pi,k) &\models \bigcirc\phi \quad \text{iff} \quad (w^\pi,k+1) \models \phi,\\
(w^\pi,k) &\models \phi_1 \mathcal{U} \phi_2 \quad \text{iff} \quad \exists k' \ge k,\ (w^\pi,k') \models \phi_2,\\
&\qquad\qquad \text{and } \forall k'' \in [k,k'), (w^\pi,k'') \models \phi_1,\\
(w^\pi,k) &\models \Diamond_{\sim i}\phi \quad \text{iff} \quad \exists k' \sim k+i,\ (w^\pi,k') \models \phi.
\end{aligned}$$

If the satisfaction relations are evaluated at time index $k = 0$, then we simply write $w^\pi \models \phi$. For an scpLTL formula ϕ, the set $\{\varrho^\pi{:}w^\pi{\models}\phi\}$ is measurable (Baier and Katoen (2008)). We denote $\Pr^\pi_{\mathcal{M}}(w{\models}\phi)$ as the probability that a word w, generated by an MDP $\mathcal{M}$ under a policy $\pi{\in}\Pi(\mathcal{M})$, satisfies a pLTL formula ϕ; *i.e.*, $w{\in}\{\varrho^\pi : w^\pi \models \phi\}$.

As discussed in Xu et al. (2019), *syntactically co-safe pLTL* (scpLTL) formulas are a special class of pLTL formulas that can be satisfied by words of finite length. The syntax of scpLTL is defined recursively as

$$\begin{aligned}
\phi \triangleq \top \mid \pi \mid \neg\pi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \bigcirc\phi \mid \Diamond\phi \mid \phi_1 \mathcal{U} \phi_2\\
\mid \Diamond_{\sim i}\phi \mid \Box_{\le i}\phi \mid \phi_1 \mathcal{U}_{\sim i} \phi_2.
\end{aligned}$$

Because scpLTL is a restriction of pLTL, the satisfaction relation of scpLTL formulas can be derived from the satisfaction relation of general pLTL formulas.

3. PROBLEM FORMULATION

We consider an agent operating in a stochastic environment whose behavior is modeled by an MDP. The agent aims to complete a task, expressed as a *ground-truth* scpLTL specification $\phi^\star$, with desired probability $\Gamma\in(0,1)$, while in the presence of an adversarial observer. The adversary aims to infer the task of the agent through observations of its trajectory. Aware of the adversary's objective, the agent aims to complete its task with the desired probability while simultaneously minimizing the information leaked to the adversary about the task.

Let $\phi \triangleq \{\phi_1, \phi_2, \ldots, \phi_N\}$ be a set of scpLTL specifications such that $\phi^\star\in\phi$. The adversary has a finite set $\phi_{can}\subseteq\phi$ of *candidate scpLTL specifications*, which it uses to describe the task of the agent. In particular, let $\beta\in(0,1)$ be a constant candidacy threshold, and $\pi\in\Pi(\mathcal{M})$ be the agent's policy. An scpLTL specification $\phi_i\in\phi$ is a candidate, i.e., $\phi_i\in\phi_{can}$, if and only if $\Pr^\pi_\mathcal{M}(w\models\phi_i)\geq\beta$. In other words, a specification is a candidate if the trajectories followed by the agent satisfy the specification with at least probability β.

We *assume* that, to each candidate specification $\phi_i\in\phi_{can}$, the adversary assigns a likelihood probability

$$\Pr(\phi_i = \phi^\star | \phi_i \in \phi_{can}) \triangleq \frac{\Pr^\pi_\mathcal{M}(s_0 \models \phi_i)\mathbb{I}\{\phi_i \in \phi_{can}\}}{\sum_{\phi_i\in\phi}\Pr^\pi_\mathcal{M}(s_0 \models \phi_i)\mathbb{I}\{\phi_i \in \phi_{can}\}} \quad (1)$$

where $\mathbb{I}\{s\in S\}$ is an indicator function such that $\mathbb{I}\{s\in S\}\triangleq 1$ if $s\in S$ and $\mathbb{I}\{s\in S\}\triangleq 0$ otherwise. The probability assignment (1) is a simple averaging rule representing the adversary's confidence in the candidate being the ground-truth specification. The adversary may also measure its confidence level using a distribution different from (1); *e.g.*, Boltzmann distribution. In that case, the solution techniques introduced in this paper can still be utilized to synthesize a policy minimizing the adversary's information about the task. However, depending on the distribution, the synthesis of such a policy may require one to employ different computational methods.

We use the adversary's certainty on the ground-truth specification as the measure of the information leakage. For a given policy $\pi\in\Pi(\mathcal{M})$, let $\Pr_{\mathcal{M},\pi,\phi_i}\triangleq\Pr(\phi_i=\phi^\star|\phi_i\in\phi_{can})$. We measure the adversary's *uncertainty* on the specification $\phi^\star$ by the entropy

$$H^\pi(\phi_{can}) \triangleq -\sum_{\phi_i\in\phi_{can}} \Pr_{\mathcal{M},\pi,\phi_i}\log\Pr_{\mathcal{M},\pi,\phi_i} \quad (2)$$

of the distribution $\Pr(\phi_i=\phi^\star|\phi_i\in\phi_{can})$. The rationale behind this choice can be better understood by recalling that the entropy of a random event is the lower bound on the average number of bits required to describe the outcomes of the event (Cover and Thomas (2012)). Moreover, this lower bound is maximized when the probability distribution associated with the event is uniform. By following a policy maximizing $H^\pi(\phi_{can})$, the agent satisfies all candidate specifications $\phi_i\in\phi_{can}$ with *similar* probabilities, making it more difficult for the adversary to guess the ground-truth specification $\phi^\star$ with high confidence.

Problem 1. Given an MDP $\mathcal{M}$, a set of candidate scpLTL formulas $\phi=\{\phi_1,\phi_2,\ldots,\phi_N\}$, a ground-truth formula $\phi^*\in\phi$, and constants $\Gamma,\beta\in(0,1)$ such that $\Gamma\geq\beta$, synthesize a policy $\pi\in\Pi(\mathcal{M})$ that solves the following problem:

$$\underset{\pi\in\Pi(\mathcal{M})}{\text{maximize}}\ H^\pi(\phi_{can}) \quad (3a)$$

$$\text{subject to: } \Pr^\pi_\mathcal{M}(w\models\phi^\star)\geq\Gamma \quad (3b)$$

$$\phi_i\in\phi_{can} \iff \Pr^\pi_\mathcal{M}(w\models\phi_i)\geq\beta \quad (3c)$$

Intuitively, a policy solving (3a)-(3c) maximizes the uncertainty of the adversary about the agent's task while ensuring that the agent completes the task with desired probability.

4. AN EXACT SOLUTION METHOD

We now present an exact solution method for the problem defined in (3a)-(3c). First, we construct a product MDP on which the satisfaction probability of each specification $\phi_i\in\phi$ can be verified. We then formulate a nonlinear optimization problem on this product MDP, whose solution provides a policy solving the problem defined in (3a)-(3c).

4.1 Product MDP

We construct a product MDP in three steps. First, we construct a deterministic finite automaton for each specification ϕ_i. Second, we form an expanded MDP whose state labels track the stage number of the underlying process. Finally, we take the product of the expanded MDP with each of the automata constructed in the first step.

For any scpLTL specification ϕ_i with fixed parameters, one can construct a deterministic finite automaton with the input alphabet $2^{\mathcal{AP}}$ which accepts a word w^π if and only if (iff) w^π satisfies the specification ϕ_i, *i.e.*, $w^\pi\models\phi_i$ (Kupferman and Vardi (2001)).

Definition 3. A *deterministic finite automaton* (DFA) is a tuple $A=(Q,q_0,2^{\mathcal{AP}},\delta,\mathcal{F})$, where Q is a finite set of states, q_0 is a unique initial state, $2^{\mathcal{AP}}$ is an alphabet, $\delta:Q\times 2^{\mathcal{AP}}\to Q$ is a transition function, and $\mathcal{F}\subseteq Q$ is a finite set of accepting states.

For a given scpLTL formula $\phi_i\in\phi$, we denote its corresponding DFA by A_{ϕ_i}. Without loss of generality (w.l.o.g.), we assume that the accepting states $\mathcal{F}$ of A_{ϕ_i} are absorbing, *i.e.*, $\delta(q,p)=q$ for all $q\in\mathcal{F}$ and $p\in 2^{\mathcal{AP}}$. We do not lose generality since an input word is accepted by a DFA A_{ϕ_i} iff it has a finite prefix that reaches an accepting state on A_{ϕ_i}. The continuation of the word after that prefix has no effect on its acceptance by A_{ϕ_i}.

We modify a given DFA A_{ϕ_i} by augmenting Q with a terminal state q_i^t which is absorbing and reachable only from the states in $\mathcal{F}$. Specifically, the modified DFA $\overline{A}_{\phi_i}$ has the finite set of states $\overline{Q}\triangleq Q\cup\{q_i^t\}$, with a transition function $\overline{\delta}:Q\times 2^{\mathcal{AP}}\to Q$ defined by $\overline{\delta}(q,p)\triangleq q_i^t$ if $q\in\mathcal{F}\cup\{q_i^t\}$ and $\delta(q,p)$ otherwise. In Fig. 1, we provide an example construction of the modified DFA $\overline{A}_{\phi_i}$ for the scpLTL formula $\phi_i=\square_{\leq 2}\neg a$ where $\{a\}\in\Sigma$.

We now form the expanded MDP whose state labels tracks the stage number of the underlying process so that the satisfaction of a given scpLTL specification can be verified.

Definition 4. Let $\mathcal{M}=(S,s_0,\mathcal{A},\mathcal{P},\mathcal{AP},\mathcal{L})$ be an MDP and $[\mathcal{T}]\triangleq\{1,2,\ldots,\mathcal{T}\}$ be an index set. The *expanded MDP* $\mathcal{M}\times[T]=(S^{[\mathcal{T}]},s_0^{[\mathcal{T}]},\mathcal{A},\mathcal{P}^{[\mathcal{T}]},\mathcal{L}^{[\mathcal{T}]},\mathcal{AP}^{[\mathcal{T}]})$ is a tuple where $S^{[\mathcal{T}]}=S\times[\mathcal{T}]$, $s_0^{[\mathcal{T}]}=(s_0,1)$, $\mathcal{P}^{[\mathcal{T}]}((s,t),a,(s',t'))=\mathcal{P}_{s,a,s'}$ if $t<\mathcal{T}$ and $t'=t+1$, $\mathcal{P}_{s,a,s'}$ if $t=\mathcal{T}$ and $t'=t$, and 0 otherwise, $\mathcal{L}^{[\mathcal{T}]}((s,t))=\mathcal{L}(s)\cup\{t\}$, and $\mathcal{AP}^{[\mathcal{T}]}=\mathcal{AP}\cup[\mathcal{T}]$.

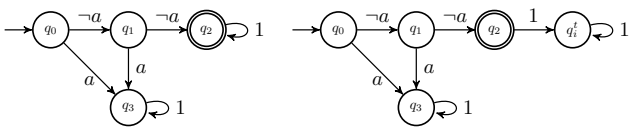

Fig. 1. An example construction of the modified DFA $\overline{A}_{\phi_i}$ for the scpLTL specification $\phi_i{=}\Box_{\leq 2}\neg a$. (Left) The nominal DFA A_ϕ where $q_2{\in}\mathcal{F}$ is the only accepting state. (Right) The modified DFA $\overline{A}_\phi$.

We note that by choosing $2^{\mathcal{AP}^{[\mathcal{T}_i]}}$ instead of $2^{\mathcal{AP}}$ as the input alphabet Σ, one can dramatically decrease the number of states in the DFA A_{ϕ_i}. In Fig. 2, we demonstrate the significance of the input alphabet on the size of a DFA corresponding to the formula $\phi_i{=}\Box_{\leq m}\neg a$. To reduce the size of the state-space in the solution of the problem (3a)-(3c), we verify the satisfaction of a given formula ϕ_i over $\mathcal{M}{\times}[\mathcal{T}_i]$ instead of $\mathcal{M}$.

To verify if the probability that the trajectories followed by an agent on $\mathcal{M}{\times}[\mathcal{T}_i]$ satisfies a specification ϕ_i exceeds a desired threshold, one can construct a product MDP and verify whether the agent's trajectories reach the accepting states of the product MDP with desired probability. Note in the following definition that we abuse the notation for the expanded MDP $\mathcal{M}{\times}[\mathcal{T}_i]$.

Definition 5. Let $\mathcal{M}{\times}[\mathcal{T}_i]{=}(S, s_0, \mathcal{A}, \mathcal{P}, \mathcal{AP}, \mathcal{L})$ be an expanded MDP and $\overline{A}_{\phi_i}{=}(\overline{Q}_i, q_0^i, 2^{\mathcal{AP}}, \overline{\delta}_i, \mathcal{F}_i)$ be a modified DFA. The *product MDP* $\mathcal{M}{\times}[\mathcal{T}_i]{\times}\overline{A}_{\phi_i}{=}(S_p, s_{0_p}, \mathcal{A}, \mathbb{P}, \mathcal{AP}, \mathcal{L}_p, \mathcal{F}_p)$ is a tuple where $S_p{=}S{\times}\overline{Q}_i$, $s_{0_p} = (s_0, q)$ such that $q = \overline{\delta}_i(q_0^i, \mathcal{L}(s_0))$, $\mathbb{P}((s,q),a,(s',q')){=}\mathcal{P}_{s,a,s'}$ if $q'{=}\overline{\delta}_i(q, \mathcal{L}(s'))$ and 0 otherwise, $\mathcal{L}_p((s,q)) = \{q\}$, and $\mathcal{F}_p{=}S \times \mathcal{F}_i$.

A product MDP $\mathcal{M} \times [\mathcal{T}_i] \times \overline{A}_{\phi_i}$ may contain states that are not reachable from the initial state. Unreachable states have no effect in the analysis of the MDP. These states can be found in time polynomial in the size of $\mathcal{M} \times [\mathcal{T}_i] \times \overline{A}_{\phi_i}$ by graph search algorithms, *e.g.*, breadth-first search, and can subsequently be removed from the product MDP w.l.o.g. We hereafter assume that there is no unreachable state in $\mathcal{M} \times [\mathcal{T}_i] \times \overline{A}_{\phi_i}$.

For a given specification ϕ_i with a fixed parameter set $\mathbf{p}_i$, let $\mathcal{T}_i{\triangleq}\max \mathbf{p}_i$ be the maximum element of $\mathbf{p}_i$, *e.g.*, $\mathbf{p}_i{=}\{4, 8\}$ and $\mathcal{T}_i{=}8$ for $\phi_i{=}\Box_{\leq 4}a \wedge \Diamond_{\geq 8}b$. We note that for nested formulas, the parameter set can be defined recursively. As an example, for $\phi_i{=}\Diamond_{[a,b]}\Box_{[c,d]}p$, letting $\phi_j{\triangleq}\Box_{[c,d]}p$, we have $\mathbf{p}_j{=}\{c, d\}$, and $\mathbf{p}_i{=}\{a+c, a+d, b+c, b+d\}$. For an MDP $\mathcal{M}$ and a set $\phi{=}\{\phi_1, \phi_2, \ldots, \phi_N\}$ of specifications, we form the product MDP $\mathcal{M}_p{\triangleq}\mathcal{M}{\times}[\mathcal{T}]{\times}\overline{A}_{\phi_1}{\times}\overline{A}_{\phi_2}{\times}\ldots{\times}\overline{A}_{\phi_N}$ by recursively applying Definition 5, where $\mathcal{T}{\triangleq}\max_{i\in[N]}\mathcal{T}_i$. In this construction, the input alphabet to each DFA $\overline{A}_{\phi_i}$ is $2^{\mathcal{AP}^{[\mathcal{T}]}}$.

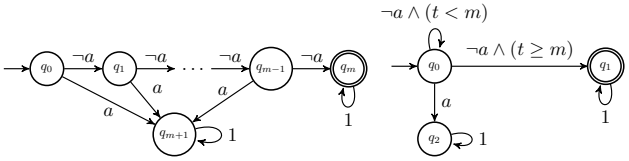

Fig. 2. The effect of the input alphabet on the size of the automaton for the scpLTL specification $\phi_i{=}\Box_{\leq m}\neg a$ where $m{\in}\mathbb{N}$ is a constant. (Left) The DFA has the input alphabet $2^{\{a\}}$. (Right) The DFA has the input alphabet $2^{\{a\}\cup[m]}$.

4.2 Policy Synthesis: An Optimization Problem

After constructing the product MDP $\mathcal{M}_p$ on which the satisfaction probability of each specification ϕ_i can be verified, we now provide a nonlinear optimization problem whose solution provides a policy solving the problem defined in (3a)-(3c).

Let the tuple $\mathbf{s}{\triangleq}(s, t, q_1, q_2, \ldots, q_N)$ denote a state in $\mathcal{M}_p$ such that $s{\in}S$, $t{\in}[\mathcal{T}]$, and $q_i{\in}Q_i$ for all $i{\in}\{1, 2, \ldots, N\}$. We denote the k^{th} element of the tuple $\mathbf{s}$ by $\mathbf{s}[k]$, *e.g.*, $\mathbf{s}[1]{=}s$, $\mathbf{s}[2]{=}t$, $\mathbf{s}[3]{=}q_1$ and $\mathbf{s}[N{+}2]{=}q_N$. Moreover, with an abuse of notation, we denote the transition function of $\mathcal{M}_p$ by $\mathbb{P}$.

We partition the states of $\mathcal{M}_p$ into the disjoint sets B and $S_p\backslash B$. Let B be the set of states $\mathbf{s}{\in}S_p$ such that

$$\sum_{\mathbf{s}'\in\mathbf{S_p}:\mathbf{s}'[\mathbf{i+2}]=\mathbf{s}[\mathbf{i+2}]\forall\mathbf{i}\in[\mathbf{N}]} \mathbb{P}_{\mathbf{s},a,\mathbf{s}'} = 1 \tag{4}$$

for all $a{\in}\mathcal{A}$. The set B is a collection of states $\mathbf{s}$ whose elements $\mathbf{s}[k{+}2]$ correspond to the automata states q_k that are absorbing. Once a state $\mathbf{s}{\in}B$ is reached by the agent, we know that each specification ϕ_i is either satisfied or violated by the agent. Note that the set B can be computed in time polynomial in the size of $\mathcal{M}_p$, as condition (4) can be verified by simply checking whether the automata elements of a state are absorbing or not.

We assume w.l.o.g. that $\phi_1{=}\phi^\star$, *i.e.*, ϕ_1 is the ground-truth specification $\phi^\star$. Let $\alpha{:}S_p{\to}[0,1]$ be a function such that $\alpha(s_{0_p}){=}1$ and $\alpha(s_{0_p}){=}0$ otherwise, *i.e.*, α is the initial state distribution of $\mathcal{M}_p$. The optimization problem is given as:

$$\underset{\lambda(\mathbf{s},a),x(i)}{\text{maximize}} -\sum_{i\in[N]} \frac{\nu(i)}{\sum_{j\in[N]}\nu(j)} \log\left(\frac{\nu(i)}{\sum_{j\in[N]}\nu(j)}\right) \tag{5a}$$

subject to:

$$\forall \mathbf{s} \in S_p\backslash B,\ \sum_{a\in\mathcal{A}}\lambda(\mathbf{s},a) - \sum_{\mathbf{s}'\in S_p}\sum_{a\in\mathcal{A}}\mathbb{P}_{\mathbf{s}',a,\mathbf{s}}\lambda(\mathbf{s}',a) = \alpha(\mathbf{s}) \tag{5b}$$

$$\forall i \in [N],\ \mu(i) = \sum_{\mathbf{s}\in S_p:\ \mathbf{s}[i+2]\in\mathcal{F}_i}\sum_{a\in\mathcal{A}}\lambda(\mathbf{s},a) \tag{5c}$$

$$\mu(1) \geq \Gamma \tag{5d}$$

$$\forall i \in [N], \quad \mu(i) \geq \beta x(i) \tag{5e}$$

$$\forall i \in [N], \quad \nu(i) = \mu(i)x(i) \tag{5f}$$

$$\forall \mathbf{s} \in S_p, \forall a \in \mathcal{A}, \lambda(\mathbf{s},a) \geq 0 \tag{5g}$$

$$\forall i \in [N], \quad x(i) \in \{0,1\} \tag{5h}$$

The decision variables in the above optimization problem are $\lambda(\mathbf{s},a)$ for each $\mathbf{s}{\in}S_p$ and $a{\in}\mathcal{A}$, and $x(i)$ for each $i{\in}[N]$. The variables $\mu(i)$ and $\nu(i)$ are functions of $\lambda(\mathbf{s},a)$ and $x(i)$, defined in (5c) and (5f), respectively, to simplify the notation. The variable $\lambda(\mathbf{s},a)$ corresponds to the expected number of times the state-action pair $(\mathbf{s},a)$ is visited (Etessami et al. (2007)). In particular, we have the relation

$$\lambda(\mathbf{s},a) = \sum_{t=1}^{\infty}\Pr^{\pi}_{\mathcal{M}_p}(S_t = \mathbf{s}, A_t = a | S_1 = s_{0_p}) \tag{6}$$

where the policy $\pi{\in}\Pi(\mathcal{M}_p)$ is defined as

$$\pi(\mathbf{s},a) \triangleq \begin{cases} \frac{\lambda(\mathbf{s},a)}{\sum_{a'\in\mathcal{A}}\lambda(\mathbf{s},a')} & \text{if } \sum_{a'\in\mathcal{A}}\lambda(\mathbf{s},a') > 0 \\ 1/|\mathcal{A}| & \text{otherwise.} \end{cases} \tag{7}$$

For more details on the variable $\lambda(\mathbf{s},a)$, we refer the reader to (Puterman, 2014, Chapter 6), (Altman, 1999, Chapter 2), and Etessami et al. (2007). Finally, the binary variable $x(i)$, under

the constraints (5b)-(5h), satisfies the relation that $x(i)=1$ if $\Pr^{\pi}_{\mathcal{M}}(w \models \phi_i) \geq \beta$ and 0 otherwise.

Constraint (5b) is traditionally referred to as the "flow constraint" (Etessami et al. (2007)), which ensures that the number of times the agent leaves a state is equal to the number of times it enters that state. Constraint (5c) defines the variable $\mu(i)$, the probability of reaching an accepting state of the automaton $\overline{A}_{\phi_i}$. Although the variable $\lambda(\mathbf{s}, a)$ is the expected number of visits to $(\mathbf{s}, a)$, because we use the modified automaton $\overline{A}_{\phi_i}$ in the product MDP $\mathcal{M}_p$, $\lambda(\mathbf{s}, a)$ corresponds to the reachability probability for states $\mathbf{s}$ satisfying $\mathbf{s}[i+2] \in \mathcal{F}_i$. Constraints (5d) and (5e) ensure, respectively, that the ground-truth specification ϕ_1 is satisfied by at least probability Γ, and that if $x(i)=1$, we have $\phi_i \in \phi_{can}$. Constraint (5f) defines the variable $\nu(i)$, which is equal to the satisfaction probability of the specification ϕ_i if $x(i)=1$ and zero otherwise. Finally, constraints (5g) and (5h) define the feasible domains of the decision variables.

The objective function (5a) is the entropy of the probability distribution $\nu(i)/\sum_{j\in[N]}\nu(j)$, which, under the constraints (5b)-(5h), is equal to the right hand side of (1). Specifically, it can be seen from the constraints (5c)-(5f) that we have $\nu(i) = \Pr^{\pi}_{\mathcal{M}}(w \models \phi_i)\mathbb{I}\{\phi_i \in \phi_{can}\}$.

We note that, once an optimal solution $\lambda^\star(\mathbf{s}, a)$ to the problem (5a)-(5h) is computed, one can obtain an optimal policy $\pi^\star \in \Pi(\mathcal{M}_p)$ on the product MDP $\mathcal{M}_p$ using the construction given in (7). Then, using the one-to-one correspondence between the policies on $\mathcal{M}$ and $\mathcal{M}_p$ (see, *e.g.*, Baier and Katoen (2008),Wolff et al. (2012)), we can finally construct a policy on $\mathcal{M}$, which solves the problem (3a)-(3c).

4.3 Policy Synthesis: A Solution Approach

The nonlinear optimization problem (5a)-(5h) has a structure which we can exploit to utilize off-the-shelf optimization toolboxes to obtain a global optimal solution. We now provide an algorithm, based on a bisection method (Boyd and Vandenberghe (2004)), that allows the utilization of such toolboxes.

We begin with the exact relaxation of the constraint (5f). Note that (5f) is a bilinear constraint since both $\mu(i)$ and $x(i)$ are variables in the optimization problem. Such constraints are not handled by most off-the-shelf toolboxes. However, recalling that $\mu(i)$ represents the probability of reaching an accepting state of the automaton $\overline{A}_{\phi_i}$, we know that $0 \leq \mu(i) \leq 1$. Using this additional information, we can replace each constraint (5f), with its corresponding McCormick envelope (McCormick (1976)), given by the following inequalities

$$\nu(i) \geq 0, \qquad \nu(i) \leq x(i), \tag{8}$$

$$\nu(i) \leq \mu(i), \qquad \nu(i) \geq x(i) + \mu(i) - 1. \tag{9}$$

Note that, using these inequalities, we have $\nu(i)=0$ if $x(i)=0$, and $\nu(i)=\mu(i)$ if $x(i)=1$. Therefore, the relaxation of the constraint (5f) with the above inequalities is exact. Moreover, since the above constraints are affine in the variables $\mu(i)$ and $x(i)$, they can now be handled by off-the-shelf toolboxes.

Next, we utilize the quasiconcavity of the objective function in (5a) in the variables $\nu(i)$. A quasiconcave function is formally defined below. For additional details on convex sets and functions, we refer the reader to Boyd and Vandenberghe (2004).

Definition 6. (Boyd and Vandenberghe (2004)) A function $g: \mathcal{D} \to \mathbb{R}$ is called *quasiconcave* if its domain $\mathcal{D}$ and all its superlevel sets $\Omega_\theta \triangleq \{x \in \mathcal{D} : g(x) \geq \theta\}$ for $\theta \in \mathbb{R}$ are convex.

Let $\boldsymbol{\nu} \triangleq [\nu(1), \nu(2), \ldots, \nu(N)]$ be a vector of variables $\nu(i)$ for $i \in [N]$, and $f: \mathbb{R}^N_+ \to \mathbb{R}$ be a function such that

$$f(\boldsymbol{\nu}) \triangleq -\sum_{i\in[N]} \frac{\nu(i)}{\sum_{j\in[N]}\nu(j)} \log\left(\frac{\nu(i)}{\sum_{j\in[N]}\nu(j)}\right).$$

The function $f(\boldsymbol{\nu})$ is not concave, as shown in Fig. 3 for $\boldsymbol{\nu} \in [0,1]^2$. By defining functions $f_1: \mathbb{R}^N_+ \to \mathbb{R}$ and $f_2: \mathbb{R}^N_+ \to \mathbb{R}$ such that

$$f_1(\boldsymbol{\nu}) \triangleq -\sum_{i\in[N]} \nu(i) \log\left(\frac{\nu(i)}{\sum_{j\in[N]}\nu(j)}\right), f_2(\boldsymbol{\nu}) \triangleq \sum_{j\in[N]} \nu(j),$$

we obtain the relation $f(\boldsymbol{\nu}) = f_1(\boldsymbol{\nu})/f_2(\boldsymbol{\nu})$. Convexity of the sublevel set $\Omega_\theta \triangleq \{\boldsymbol{\nu} \in \mathbb{R}^N_+ : f(\boldsymbol{\nu}) \geq \theta\}$ for any $\theta \in \mathbb{R}$ follows from the fact that $f_1(\boldsymbol{\nu}) \geq \theta f_2(\boldsymbol{\nu})$ defines a convex region since the functions $f_1(\boldsymbol{\nu})$ and $f_2(\boldsymbol{\nu})$ are, respectively, concave and affine over their domains (Boyd and Vandenberghe (2004)). We thus conclude the quasiconcavity of $f(\boldsymbol{\nu})$ from Definition 6.

We are now ready to introduce an iterative algorithm for the solution of (5a)-(5h), which is a variant of the bisection method for quasiconcave optimization (Algorithm 4.1 in Boyd and Vandenberghe (2004)). Let $v^\star$ be the optimal value of the problem in (5a)-(5h), and $u \in \mathbb{R}$ be an arbitrarily large constant which satisfies $u \geq v^\star$. Moreover, let $l \in \mathbb{R}$ be a constant such that $l \leq v^\star$, *e.g.*, $l=0$. At each iteration of the algorithm, we set $\theta \triangleq (u+l)/2$ and solve the feasibility problem given in (10). If the problem has a feasible solution, in the next iteration of the algorithm, we set $u \triangleq \theta$, otherwise, we set $l \triangleq \theta$. The algorithm terminates when the stop condition $u - l \leq \epsilon$ is satisfied, where $\epsilon > 0$ is a constant tolerance parameter.

$$\begin{aligned} \text{find} \quad & \theta \\ \text{subject to:} \quad & f_1(\boldsymbol{\nu}) \geq \theta f_2(\boldsymbol{\nu}) \\ & \boldsymbol{\nu} = [\nu(1), \nu(2), \ldots, \nu(N)] \\ & \text{(5b), (5c), (5d), (5e), (5g), (5h), (8), (9)} \end{aligned} \tag{10}$$

As mentioned in Section 4.2, an optimal policy $\pi^\star \in \Pi(\mathcal{M}_p)$ on the product MDP $\mathcal{M}_p$ can be obtained using the construction given in (7), once an optimal solution $\lambda^\star(\mathbf{s}, a)$ to the problem (5a)-(5h) is computed using the algorithm defined in (10).

5. AN APPROXIMATE SOLUTION METHOD

Although the method presented in Section 4 provides an exact solution to the Problem 1, it requires one to form the product MDP, which is the product of the expanded MDP with the automata corresponding to each specification ϕ_i. The construction

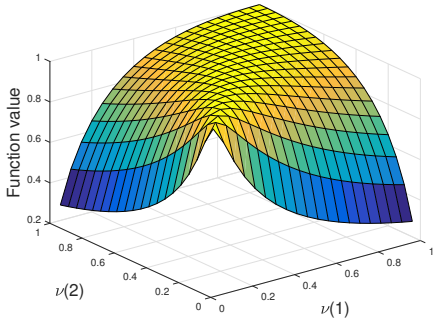

Fig. 3. The function $f(\boldsymbol{\nu})$ is not concave. However, $f(\boldsymbol{\nu})$ has a different useful property, which is quasiconcavity.

of the product MDP is, in general, a computationally expensive operation; therefore, for practical purposes, it is desirable to develop algorithms that synthesize policies directly on the expanded MDP. Focusing on a subset of scpLTL specifications, we now present a method that conservatively approximates the satisfaction probabilities of the specifications ϕ_i and allows one to synthesize policies on the expanded MDP. Throughout this section, we restrict our attention to a subset of scpLTL specifications with the following assumption.

Assumption 1. Each scpLTL specification $\phi_i \in \phi$ we consider has one of four possible forms: $\Diamond_{[a,b]} p$, $\Box_{[a,b]} p$, $\Diamond_{[a,b]}\Box_{[c,d]} p$, or $\Box_{[a,b]}\Diamond_{[c,d]} p$, where $a, b, c, d \in \mathbb{N}$ and $p \in \mathcal{AP}$.

5.1 The Fréchet inequalities

We now present the Fréchet inequalities (Fréchet (1935); Hailperin (1965)), which allow us to conservatively approximate the satisfaction probability of a specification ϕ_i. For each $i \in [K]$ where $K \in \mathbb{N}$, let E_i be a logical proposition, and e_i be the probability that the proposition E_i is true. Then,

$$\Pr\Big(\bigwedge_{i=1}^{K} E_i\Big) \geq \max\Big\{0, \sum_{i=1}^{K} e_i - (n-1)\Big\}, \tag{11}$$

$$\Pr\Big(\bigvee_{i=1}^{K} E_i\Big) \geq \max\{e_1, e_2, \ldots, e_K\}. \tag{12}$$

These lower bounds are the best possible bounds if nothing is known about the events $E_1, E_2, \ldots, E_K$ except that their probabilities are $e_1, e_2, \ldots, e_K$, respectively (Hailperin (1965)). A remarkable property of these lower bounds is that they are in terms of the satisfaction probabilities e_i of the subformulas E_i only. If one can represent a logical formula E as a conjunction or disjunction of the subformulas E_i whose satisfaction probability can be computed easily, then by ensuring that the lower bound exceeds a desired threshold β, one can guarantee that the formula E is satisfied with at least probability β.

In what follows, we form an expanded MDP on which, instead of measuring the satisfaction probability of a specification ϕ_i, we measure the satisfaction probabilities of subformulas of ϕ_i corresponding to *each time step*. As an example, instead of measuring the satisfaction probability of $\phi_i = \Box_{[1,3]} p$, we measure the probability that the predicate p holds true at a given time step $1 \leq t \leq 3$. Then, using the syntax of scpLTL specifications, we utilize these measurements to derive the lower bound on the satisfaction probability of ϕ_i.

Recall from Section 4.1 that for a given specification ϕ_i with a fixed parameter set $\mathbf{p}_i$, we have $\mathcal{T}_i = \max \mathbf{p}_i$. For an MDP $\mathcal{M}$ and a set $\phi = \{\phi_1, \phi_2, \ldots, \phi_N\}$ of specifications, using Definition 4, we form the expanded MDP $\overline{\mathcal{M}} \triangleq \mathcal{M} \times [\mathcal{T}+1]$ where $\mathcal{T} \triangleq \max_{i \in [N]} \mathcal{T}_i$. On $\overline{\mathcal{M}}$, we can measure the satisfaction probability of a predicate $p \in \mathcal{AP}$ at time $t \in [\mathcal{T}]$ by the expected number of visits to states $\mathbf{s} \in S^{[\mathcal{T}+1]}$ such that $\mathbf{s}[2] = t$ and $p \in \mathcal{L}(\mathbf{s})$.

To synthesize a policy $\pi \in \Pi(\overline{\mathcal{M}})$ on the expanded MDP $\overline{\mathcal{M}}$, we solve a modified version of the problem (5a)-(5h) on the state-space $S^{[\mathcal{T}+1]}$ of $\overline{\mathcal{M}}$. In particular, for each specification ϕ_i, we replace the corresponding constraint (5c) with a series of other constraints. Recall that the variable $\mu(i)$ in (5c) is equal to the probability of satisfying the specification ϕ_i. Instead of using the exact satisfaction probability, for each specification form in Assumption 1, we introduce a set of constraints which ensure that $\mu(i)$ is a lower bound on the actual satisfaction probability.

5.2 Additional constraints for each approximation

For each scpLTL formula considered, we now present the sets of additional constraints that allow for the estimation of the true satisfaction probabilities using only the expanded MDP.

• $\phi_i = \Diamond_{[k_1,k_2]} p$. We first introduce the variables $\eta(t) \in \mathbb{R}$ for each $t \in \mathbb{N}$ such that $k_1 \leq t \leq k_2$. Using the syntax of pLTL specifications, we can show that the lower bound in (12) is equal to $\mu(i)$, defined by the following constraints:

$$\sum_{\substack{\mathbf{s} \in \mathbf{S}^{[\mathcal{T}+1]}: \\ \mathbf{s}[2]=t,\ p \in \mathcal{L}(\mathbf{s})}} \sum_{a \in A} \lambda(\mathbf{s}, a) = \eta(t), \tag{13a}$$

$$\mu(i) = \max\Big\{\eta(k_1), \eta(k_1+1), \ldots, \eta(k_2)\Big\}. \tag{13b}$$

In the above constraints, each variable $\eta(t)$ captures the probability that the formula ϕ_i holds at the particular time step t. To utilize the off-the-shelf toolboxes for encoding the above constraints, we need to relax the constraint (13b). We do so by replacing (13b) with the following set of constraints:

$$\forall k_1 \leq t \leq k_2,\ \mu(i) \geq \eta(t); \quad \mu(i) = \sum_{t=k_1}^{k_2} y(t)\eta(t), \tag{14a}$$

$$\forall k_1 \leq t \leq k_2,\ y(t) \in \{0,1\}; \quad \sum_{t=k_1}^{k_2} y(t) = 1. \tag{14b}$$

The above relaxation is exact. In (14a), the term $y(t)\eta(t)$ is bilinear as both $y(t)$ and $\eta(t)$ are variables. However, since we know that $0 \leq \eta(t) \leq 1$ from (13a), by defining an extra variable $\gamma(t) \triangleq y(t)\eta(t)$, we can represent each term $\gamma(t)$ exactly with its corresponding McCormick envelope given in (8)-(9).

• $\phi_i = \Box_{[k_1,k_2]} p$. We first introduce the variables $\eta(t) \in \mathbb{R}$ for each $t \in \mathbb{N}$ such that $k_1 \leq t \leq k_2$. Using the syntax of pLTL specifications, we can show that the lower bound in (11) is equal to $\mu(i)$, defined by the following constraints:

$$\sum_{\substack{\mathbf{s} \in \mathbf{S}^{[\mathcal{T}+1]}: \\ \mathbf{s}[2]=t,\ p \in \mathcal{L}(\mathbf{s})}} \sum_{a \in A} \lambda(\mathbf{s}, a) = \eta(t), \tag{15a}$$

$$\mu(i) = \max\left\{0, \sum_{t=k_1}^{k_2} \eta(t) - (k_2 - k_1)\right\}. \tag{15b}$$

To utilize the off-the-shelf toolboxes for encoding the above constraints, we need to relax the constraint (15b). We do so by replacing (15b) with the following set of constraints

$$\mu(i) \geq 0; \qquad \mu(i) \geq \left(\sum_{t=k_1}^{k_2} \eta(t) - (k_2 - k_1)\right), \tag{16a}$$

$$y \in \{0,1\}; \quad \mu(i) = y\left(\sum_{t=k_1}^{k_2} \eta(t) - (k_2 - k_1)\right). \tag{16b}$$

The above relaxation is exact, but it involves the bilinear terms $y\eta(t)$. However, once again, by introducing new variables $\gamma(t) \triangleq y\eta(t)$, we can represent each term $\gamma(t)$ exactly with its corresponding McCormick envelope given in (8)-(9).

• $\phi_i = \Diamond_{[k_1,k_2]}\Box_{[k_3,k_4]} p$. We first introduce the variables $\eta(t) \in \mathbb{R}$ for each $t \in \mathbb{N}$ such that $k_1 + k_3 \leq t \leq k_2 + k_4$, and $\zeta(m)$ for each $m \in \mathbb{N}$ such that $k_1 \leq m \leq k_2$. Using the syntax of pLTL specifications and both of the bounds in (11)-(12), we can obtain a lower bound $\mu(i)$ on the satisfaction probability of ϕ_i using:

$$\sum_{\substack{\mathbf{s}\in\mathbf{S}^{[\mathcal{T}+1]}:\\ \mathbf{s}[2]=t,\ p\in\mathcal{L}(\mathbf{s})}} \sum_{a\in A} \lambda(\mathbf{s},a) = \eta(t), \tag{17a}$$

$$\zeta(m) = \max\left\{0, \sum_{t=m+k_3}^{m+k_4} \eta(t) - (k_4-k_3)\right\}, \tag{17b}$$

$$\mu(i) = \max\Big\{\zeta(k_1), \zeta(k_1+1), \ldots, \zeta(k_2)\Big\}. \tag{17c}$$

We can perform the relaxation of the constraints in (17b)-(17c) by introducing new binary variables and subsequently using the corresponding McCormick envelopes as previously explained in the relaxation of the specifications $\Diamond_{[k_1,k_2]}p$ and $\Box_{[k_1,k_2]}p$.

• $\phi_i{=}\Box_{[k_1,k_2]}\Diamond_{[k_3,k_4]}p$. We first introduce the variables $\eta(t){\in}\mathbb{R}$ for each $t{\in}\mathbb{N}$ such that $k_1{+}k_3{\leq}t{\leq}k_2{+}k_4$, and $\zeta(m)$ for each $m{\in}\mathbb{N}$ such that $k_1{\leq}m{\leq}k_2$. Using the syntax of pLTL specifications and both of the bounds in (11)-(12), we can obtain a lower bound $\mu(i)$ on the satisfaction probability of ϕ_i using:

$$\sum_{\substack{\mathbf{s}\in\mathbf{S}^{[\mathcal{T}+1]}:\\ \mathbf{s}[2]=t,\ p\in\mathcal{L}(\mathbf{s})}} \sum_{a\in A} \lambda(\mathbf{s},a) = \eta(t), \tag{18a}$$

$$\zeta(m) = \max\Big\{\eta(m+k_3), \ldots, \eta(m+k_4)\Big\}, \tag{18b}$$

$$\mu(i) = \max\left\{0, \sum_{m=k_1}^{k_2} \zeta(m) - (k_2-k_1)\right\}. \tag{18c}$$

Again, we perform the relaxation of the constraints in (18b)-(18c) by introducing new binary variables and using the corresponding McCormick envelopes as explained in the relaxation of the specifications $\Diamond_{[k_1,k_2]}p$ and $\Box_{[k_1,k_2]}p$.

Finally, after replacing each constraint (5c) in (5a) with its corresponding set of constraints introduced in this section, we solve the resulting nonlinear optimization problem using the bisection method presented in Section 4.3.

6. NUMERICAL EXAMPLES

We now provide several examples to demonstrate the efficacy of the proposed solution methods. For each example, we use a tolerance of $\epsilon{=}1{\times}10^{-4}$ for the bisection method. We use the GUROBI solver with the CVX (Grant and Boyd (2014)) interface to solve the exact and approximate optimization problems.

6.1 A Resupply Mission

We first consider an autonomous agent operating on the gridworld shown in Fig. 4. The colored states represent different bases that the agent can travel to. The agent's mission is to resupply the blue base, which we encode as the pLTL formula "$\Box_{[a,b]}blue$"; i.e., the agent must reach the blue base at a specified time a and remain there until its supplies are unloaded after

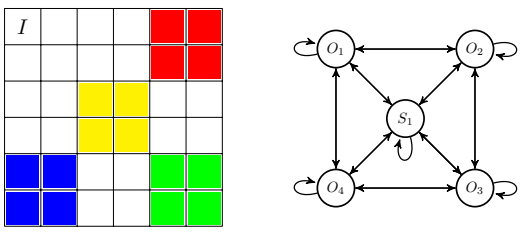

Fig. 4. (Left) Gridworld considered in the resupply mission. (Right) MDP considered in the surveillance mission.

Table 1. Sets of specifications used for the examples. ϕ^* indicates the ground-truth specification.

Example	Specifications
Resupply-1	$\phi^*{:}\Box_{[9,10]}blue$, $\phi_2{:}\Box_{[29,30]}red$
Resupply-2	$\phi^*{:}\Box_{[9,10]}blue$, $\phi_2{:}\Box_{[16,18]}yellow$ $\phi_3{:}\Box_{[23,25]}green$, $\phi_4{:}\Box_{[29,30]}red$
Surveillance	$\phi^*{:}\Box_{[1,10]}\Diamond_{[0,5]}outpost_1$, $\phi_2{:}\Box_{[1,10]}\Diamond_{[0,5]}outpost_2$ $\phi_3{:}\Box_{[1,10]}\Diamond_{[0,5]}outpost_3$, $\phi_4{:}\Box_{[1,10]}\Diamond_{[0,5]}outpost_4$

$b{-}a$ time steps. Due to the presence of an adversarial observer, the agent must additionally attempt to obfuscate which base it actually delivers the supplies to. By doing so, the adversary is least able to infer which base actually received the supplies.

The agent is assumed to start in the upper left corner of the gridworld in state I. In each state, the agent can select one of four possible actions: move left, move right, move up, or move down. Once the agent has selected an action, it transitions to its desired state with probability 0.99, while slipping to the left, to the right, or backwards each with probability $.01/3$.

We study two cases for the specifications of the agent. We first consider that the agent only seeks to prevent information leakage by additionally visiting the red base. We then consider that the agent seeks to prevent information leakage by visiting the green and yellow bases as well. For each set of specifications, we let $\Gamma{=}0.95$ and $\beta{=}0.8$. The sets of specifications are provided in Table 1, under "Resupply-1" and "Resupply-2", respectively. For each set, we run the exact and approximate methods to solve the optimization problem (5a)-(5h).

Table 2 lists the relevant output information for each solution method and specification set. We note that the number of variables in the optimization problem is after GUROBI completed presolving the problem. Because of how the approximation for specifications of the form $\Box_{[a,b]}$ was constructed, the two solution methods have the same numbers of binary variables. However, as the approximate solution method does not require taking the product with each specification automaton, the number of continuous variables in its corresponding optimization problem scales better than that of the exact solution method, requiring less time to solve. The approximate solution method also performs nearly as well as the exact solution method at minimizing the information leakage about the ground-truth specification. Both methods obtain the maximum cardinality of the set ϕ_{can} and nearly obtain the maximum-entropy upper bounds of 1 and 2 bits for each specification set, respectively.

6.2 A Surveillance Mission

We now consider an agent that must repeatedly surveil an outpost containing sensitive information on the boundary of its base. Specifically, the agent operates on the MDP shown in Fig. 4, where four outposts surround the central base. We assume that the adversarial observer does not know which of the outposts contains the sensitive information. For this reason, the agent must additionally surveil the three non-sensitive outposts. By doing so, the adversary cannot use the fact that the agent visits an outpost towards inferring which outpost contains the sensitive information. Thus, the adversary cannot optimally allocate its resources towards infiltrating the correct outpost.

We assume that the agent's initial state is in the central state S_1. In each state, the agent can either remain in its current state or transition to a neighboring state, where it transitions with

Table 2. Number of continuous and binary variables, solution times, probabilities of satisfaction, resulting entropy, and the size of the set ϕ_{can} for exact and approximate solution methods.

Example	Num. of Var. exact	Num. of Var. approx.	Time exact	Time approx.	$\Pr^{\pi}_{\mathcal{M}}(w \models \phi^*)$ exact	$\Pr^{\pi}_{\mathcal{M}}(w \models \phi^*)$ approx., comp.	$\Pr^{\pi}_{\mathcal{M}}(w \models \phi^*)$ approx., actual	$H^{\pi}(\phi_{can})$ exact	$H^{\pi}(\phi_{can})$ approx.	$\lvert\phi_{can}\rvert$ exact	$\lvert\phi_{can}\rvert$ approx.
Resupply-1	4870 con. 1 binary	3079 con. 1 binary	9.25s	6.75s	0.950	0.950	0.971	1.000	0.999	2	2
Resupply-2	6980 con. 3 binary	3125 con. 3 binary	53.71s	25.11s	0.950	0.950	0.971	1.999	1.999	4	4
Surveillance	22209 con. 3 binary	617 con. 239 binary	148.37s	26.59s	0.950	0.951	0.991	1.999	1.999	4	4

probability 1. We again set Γ=0.95 and β=0.8, respectively. We use the pLTL structure "$\square_{[a,b]}\lozenge_{[c,d]} outpost_i$" to encode the surveillance specification; i.e., at each time step over the time horizon, the agent must eventually visit an outpost within $[c, d]$ time steps. The specifications for the agent are listed in Table 1 under "Surveillance". We again run the exact and approximate methods to solve the optimization problem (5a)-(5h).

Table 2 shows the comparison of the output between the exact and approximate solution methods for the set of surveillance specifications. Although the approximate solution method uses a large number of binary variables compared to the exact solution method, it still solves the optimization problem (5a)-(5h) more quickly. Both solution methods perform similarly well in minimizing the information leakage about which outpost contained the sensitive information. Likewise, each method obtains the maximum cardinality of the set ϕ_{can} and nearly achieve the upper bound on the maximum entropy of 2 bits.

7. CONCLUSIONS

We study the problem of synthesizing a policy for an autonomous agent that leaks the minimum amount of information regarding its high-level task specification to an adversarial observer. We measure the information leakage as the adversary's confidence that a candidate mission specification is the ground-truth mission specification. Modelling the inference problem of the adversary as an averaging rule, we formulate the problem of the agent as a mixed-integer program with a quasiconcave objective function, and develop two methods for its solution. The first method exactly computes the probability that the agent satisfies a specification, whereas the second method approximates these probabilities using the Fréchet inequalities. We provide two numerical examples to demonstrate the efficacy of the proposed methods in minimizing the information leakage.

REFERENCES

Altman, E. (1999). *Constrained Markov decision processes*. CRC Press.

Alur, R., Etessami, K., La Torre, S., and Peled, D. (2001). Parametric temporal logic for model measuring. *ACM Transactions on Computational Logic*, 2(3), 388–407.

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

Boyd, S. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Carroll, T.E. and Grosu, D. (2011). A game-theoretic investigation of deception in network security. *Security and Communication Networks*, 4(10), 1162–1172.

Chakraborty, S. and Katoen, J.P. (2014). Parametric LTL on Markov chains. In *Theoretical Computer Science*, 207–221.

Cover, T.M. and Thomas, J.A. (2012). *Elements of information theory*. John Wiley & Sons.

Etessami, K., Kwiatkowska, M., Vardi, M.Y., and Yannakakis, M. (2007). Multi-objective model checking of Markov decision processes. In *Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 50–65.

Fréchet, M. (1935). Généralisation du théoreme des probabilités totales. *Fundamenta mathematicae*, 1(25), 379–387.

Grant, M. and Boyd, S. (2014). CVX: Matlab software for disciplined convex programming, version 2.1.

Hailperin, T. (1965). Best possible inequalities for the probability of a logical function of events. *The American Mathematical Monthly*, 72(4), 343–359.

Karabag, M.O., Ornik, M., and Topcu, U. (2019). Least inferable policies for Markov decision processes. In *American Control Conference*, 1224–1231.

Kupferman, O. and Vardi, M.Y. (2001). Model checking of safety properties. *Formal Methods in System Design*.

Lloyd, M. (2003). *The Art of Military Deception*. Pen and Sword.

Maliah, S., Shani, G., and Stern, R. (2017). Collaborative privacy preserving multi-agent planning. *Autonomous agents and multi-agent systems*, 31(3), 493–530.

McCormick, G.P. (1976). Computability of global solutions to factorable nonconvex programs: Part I — convex underestimating problems. *Mathematical Programming*.

Menghi, C., García, S., Pelliccione, P., and Tumova, J. (2018). Multi-robot ltl planning under uncertainty. In *International Symposium on Formal Methods*, 399–417. Springer.

Neider, D. and Gavran, I. (2018). Learning linear temporal properties. *Formal Methods in Computer Aided Design*.

Puterman, M.L. (2014). *Markov Decision Processes.: Discrete Stochastic Dynamic Programming*. John Wiley & Sons.

Savas, Y., Ornik, M., Cubuktepe, M., Karabag, M.O., and Topcu, U. (2019). Entropy maximization for Markov decision processes under temporal logic constraints. *IEEE Transactions on Automatic Control*.

Skolnick, J.H. (1982). Deception by police. *Criminal Justice Ethics*, 1(2), 40–54.

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

Wolff, E.M., Topcu, U., and Murray, R.M. (2012). Robust control of uncertain Markov decision processes with temporal logic specifications. In *Conference on Decision and Control*.

Xu, Z., Ornik, M., Julius, A.A., and Topcu, U. (2019). Information-guided temporal logic inference with prior knowledge. In *American Control Conference*.

Xu, Z. and Julius, A.A. (2019). Robust temporal logic inference for provably correct fault detection and privacy preservation of switched systems. *IEEE Systems Journal*.

Xu, Z. and Topcu, U. (2019). Transfer of temporal logic formulas in reinforcement learning. In *International Joint Conferences on Artificial Intelligence*, 4010–4018.