Opacity-enforcing active perception and control against eavesdropping attacks *

Sumukha Udupa, Hazhar Rahmani, and Jie Fu

University of Florida, Gainesville FL 32611, USA
{sudupa, h.rahmani, fujie}@ufl.edu

Abstract. In this paper, we consider *opacity-enforcing* planning with temporally-extended goals in partially observable stochastic environment. We consider a probabilistic environment modeled as partially observable Markov decision process (POMDP) in which the observation function is actively controlled as the agent decides which sensors to query at each decision step. The agent's objective is to achieve a temporal objective, expressed using linear temporal logic for finite traces (LTL_f), while making achieving its goal *opaque* to a passive observer who can access the sensor readings of a subset of sensors that are unsecured. Opacity, as a security property, means that when from the observer's perspective, the execution that satisfied the temporal goal is observational-equivalent to an execution that does not satisfy the temporal goal. When both secured and unsecured sensors are available upon query, the agent must be selective in its sensor query to prevent information leaking to the observer and to ensure task completion. We propose an algorithm that synthesizes a strategy that decides jointly the control actions and sensor queries to guarantee that the temporal goal is achieved and made opaque with probability 1. Our approach is based on planning with augmented belief state space. Further, we show how to employ properties in the temporal logic formula to reduce the size of the planning state space and improve scalability. We show the applicability of our algorithm on a case study with robotic planning in a stochastic gridworld with partial observations.

Keywords: Game theory · Opacity · Markov decision processes · Eavesdropping Attacks.

1 Introduction

Opacity is a security property, specified for a system interacting with a passive observer (also called an attacker). By enforcing opacity, the goal is to make the attacker, who knows the system model and observes the system execution, confused about if a secret state has been reached or a secret behavior has been realized. The notion of opacity was first introduced by Mazaré [13] for cryptographic protocols. Since then, variants of opacity have been studied for different classes

* This work was supported in part by ARL W911NF-22-2-0233, ARO W911NF-22-1-0034, and NSF #2144113.

of systems including Petri nets[4], labeled transitions systems [3], finite state automata [15], probabilistic finite automata [18], hidden Markov models [10], and Markov decision processes (MDPs) [1]. Depending on the nature of the secret to be made opaque, there are different variants of opacity, including *state-based*, which requires the secret behavior of the system (i.e., the membership of its current state to the set) to remain opaque (uncertain) until the system enters a non-secret state; *language-based*, which aims to hide a set of secret executions; and "*model-based*", which wants to prevents the observer to find out the true model of the system among several potential models known to the observer.

In this work, we investigate language-based opacity in stochastic systems with asymmetric, incomplete information to the planning agent and the eavesdropping attacker. To motivate the problem formulation, consider a scenario where an autonomous robot, denoted Player 1 (P1), is entrusted with the confidential task of delivering vital medical supplies to remote camp sites. The robot must retrieve these supplies from a base station and navigate through a probabilistic environment characterized by slippery conditions and the absence of GPS signals. P1 employs a sensor network to monitor its own location. However, this sensor network includes certain vulnerable sensors that leak information to an adversarial observer. The planning question is how can the robot ensure the success of its mission, while ensuring its opacity to the observer? That is, the observer cannot tell if the task is achieved or not from the observations. This problem is interesting because instead of considering a passive observer/attacker has a stationary observation function, the robot can strategically select sensor queries and thus control the amount of information leaked to the attacker with access to only unsecured sensors.

Our approach and contributions: We propose to investigate a *joint active perception and control* framework for the planning agent (P1) to decide if the opacity and mission success can be both achieved given different initial states of the stochastic system and initial observations for both P1 and the passive observer (P2). Our contributions involve incorporating 2-beliefs system to capture P1's knowledge about the current state and P1's knowledge about P2's knowledge about the current state. Reasoning with this 2-beliefs system, we develop an algorithm that computes a strategy for P1 that actively selects which sensor to query and which control action to exercise to ensure satisfaction of the mission objective and opacity constraint simultaneously. However, maintaining and updating the 2-belief system requires exponential memory due to the subset construction. To address the issue of scalability, we introduce a method to determine when to stop tracking P2's belief, while ensuring the correctness in the computed strategy. Finally, we demonstrate the correctness of our method on robot motion planning in a stochastic gridworld environment with partial observations provided by range sensors and location sensors.

Related Work: Opacity-enforcing control has been extensively studied in the context of discrete event systems (DESs). Previous work in the supervisory control framework focuses on enforcing opacity in deterministic systems by modeling the control system as a deterministic finite state automaton in the

presence of a passive observer [14,15,23]. Different approaches have been proposed to synthesize opacity [5,21,22] or to verify if a system is opaque [16,11,20]. To synthesize opacity in deterministic systems, Saboori *et al.* [17] synthesizes supervisor that restricts certain system behaviors to ensure opacity. Cassez *et al.*[5] propose the use of dynamic masks that filter out unobservable events and verify the opaqueness of a secret by reducing the opacity enforcement problem to a 2-player safety game. Xie et al. [22] design non-deterministic supervisors (controller) so that the observer, knowing the nondeterministic supervisor, cannot determine if a secret is satisfied or not. Besides masking and nondeterministic control design, [9,21] investigate how to edit the observations to ensure opacity using a non-deterministic edit function.

Game-theoretic approach for opacity enforcement has also been proposed. Maubert et al. [12] introduce the game with opacity condition, in which one player with perfect observation aims to enforce current-state opacity against another player with imperfect observation during their two-player interactions. They reduce the opacity-guarantee game to a safety game and the opacity-violating game to a reachability game, for which existing solution can be used to solve players' strategies. Hélouët et al.[8] present a framework for enforcing opacity against different types of attackers with different information about the input and the observations. The existing approaches for opacity-enforcement in deterministic systems or games are not applicable for stochastic systems with partial observations to both the system and the observer.

For opacity in stochastic DESs, Saboori and Hadjicostis [18] propose three probabilistic variants of current state opacity and develop opacity verification algorithms for systems modeled as probabilistic finite automata. Keroglou and Hadjicostis [10] investigate model-based opacity where the user wants to conceal a true system modeled by a hidden Markov model among several potential hidden Markov models from an intruder. Bérard *et al.* [1] extend language-based opacity for ω-regular properties on MDPs. They assume a static observation function and present several decidability results. In contrast, we focus on synthesizing an opacity-enforcing strategy for an agent operating in a stochastic environment and utilizing active sensor queries of secured and unsecured sensors.

2 Preliminaries and Problem Formulation

Notations. Given a finite set X, $\mathcal{D}(X)$ denotes the set of all probability distributions over X. For a probability distribution $d \in \mathcal{D}(X)$, $\mathsf{Supp}(d)$, the support of d, is the set of elements in X with non-zero probabilities under d.

We model the interaction between an agent, Player 1/P1, and the environment as a Partially Observable Markov Decision Process (POMDP). The agent actively queries sensors in the environment to obtain task-relevant information.

Definition 1 (POMDP with active perception). *The* stochastic system with partial observations *is a tuple*

$$M = (S, A, \mathbf{P}, \Omega, \Gamma = \Gamma_1 \cup \Gamma_2, O, s_0, \omega_0, \omega_0^+, AP, L)$$

in which (1) S is a finite set of states; (2) A is the set of P1's control actions *that change the state in S; (3) $\mathbf{P} : S \times A \to \mathcal{D}(S)$ is a* probability transition function *such that for each $s, s' \in S$ and $a \in A$, $\mathbf{P}(s, a, s')$ is the probability of reaching s' given action a taken at state s; (4) $\Omega \subseteq 2^S$ is the set of all observations; (5) $\Gamma = \{\gamma_0, \gamma_1, \cdots, \gamma_N\}$ is a set of* indexed sensors*, partitioned into secured sensors Γ_1 and unsecured sensors Γ_2; (6) $O : S \times 2^\Gamma \to \Omega$ is the* observation function *such that for a state $s \in S$ and a sensor subset $X \subseteq \Gamma$, $O(s, X) \in \Omega$ is the set of states whose sensor readings for sensor set X are the same as the readings given state s (we call $O(s, X)$ the set of observation-equivalent states to s based on sensor information in X); (7) $s_0 \in S$ is the initial state; (8) ω_0 and ω_0^+ are respectively the initial observations of P1 and P2, where $s_0 \in \omega_0 \cap \omega_0^+$ and $\omega_0 \subseteq \omega_0^+$; (9) AP is a set of atomic propositions; and (10) $L : S \to 2^{AP}$ is the labeling function that maps a state $s \in S$ to a set $L(s)$ of atomic propositions evaluated to true at s.*

The following regularity assumption is made on the observation function.

Assumption 1 *Let $X_1, X_2 \subseteq \Gamma$ be two sets of sensors. If $X_1 \subseteq X_2$, then for any state $s \in S$, $O(s, X_2) \subseteq O(s, X_1)$.*

That is, the more sensor readings the less uncertainty about the current state. A *perception action* is a subset of sensors being queried. A *joint control and perception action* of P1, or a *control-perception action* for short, is a tuple (a, X) including a control action $a \in A$ and a perception action $X \subseteq \Gamma$. In the following, by a P1's action, we mean a control-perception action, from $\mathcal{A}_1 = A \times 2^\Gamma$.

We consider an eavesdropping attacker, Player 2/P2, who accesses the information from the *unsecured* sensors queried by P1.

Definition 2 (Observations of an eavesdropping attacker P2). *For any state $s \in S$, for any perception action $X \subseteq \Gamma$ performed by P1 when the system entered s, P2's observation is $O(s, X \cap \Gamma_2)$ where $X \cap \Gamma_2$ is the subset of unsecured sensors in the queried sensor set X. In addition, P2 does not observe the control actions taken by P1.*

Remark 1. We assume that a sensor will emit signal only if it is queried by P1 and P2 can only obtain information from the *unsecured* sensors queried by P1. However, this assumption can be easily lifted for the case where there is a subset $\mathcal{Z}$ of sensors that always emit signals and P2 can access the subset of unsecured sensors in $\mathcal{Z}$ at all times. In this case, we only need to include these sensors $\mathcal{Z}$ into each perception action of P1.

Game Play. The game play in M is constructed as follows. The game starts from the initial state s_0, P1 gets the observation ω_0, and P2 receives observation ω_0^+. Based on ω_0, P1 takes an action $(a_0, X_0) \in \mathcal{A}_1$. The system then moves to a state s_1, drawn randomly based on $\mathbf{P}(s_0, a_0, \cdot)$. P1 receives observation $O(s_1, X_0)$ and P2 receives observation $O(s_1, X_0 \cap \Gamma_2)$. At each step, P1 can either decide to terminate the game play or take a control-perception action and move to the next state. Assuming that the game play will be eventually terminated, the resulting play is a finite sequence of state-action pairs $\rho = s_0(a_0, X_0)s_1(a_1, X_1)s_2 \ldots s_n$

such that $\mathbf{P}(s_i, a_i, s_{i+1}) > 0$ for all integers $0 \leq i < n$. The labeling of this play is $L(\rho) = L(s_0)L(s_1)\dots L(s_n)$. The set of finite plays in M is denoted by $\mathsf{Plays}(M)$.

The finite play generates a sequence of observations for both players. For P1, given that P1 knows his own action decisions, we have

$$O_1(\rho) = \omega_0(a_0, X_0)\omega_1(a_1, X_1)\dots\omega_n,$$

where ω_0 is the initial observation and for $i \geq 1$, ω_i is the observation of state s_i given the sensor query X_{i-1}, i.e., $\omega_i = O(s_i, X_{i-1})$. For P2, as P1's control actions are invisible,

$$O_2(\rho) = \omega_0^+(X_0 \cap \Gamma_2)\omega_1^+(X_1 \cap \Gamma_2)\dots(X_{n-1} \cap \Gamma_2)\omega_n^+,$$

where ω_0^+ is the initial P2's observation and for $i \geq 1$, ω_i^+ is P2's observation of state s_i given the sensor query $X_{i-1} \cap \Gamma_2$, i.e., $\omega_i^+ = O(s_i, X_{i-1} \cap \Gamma_2)$. Two plays $\rho_1, \rho_2 \in \mathsf{Plays}(M)$ are observation-equivalent to player i iff $O_i(\rho_1) = O_i(\rho_2)$. Given a play $\rho \in \mathsf{Plays}(M)$, we denote by $[\rho]_1$ (resp. $[\rho]_2$) the set of plays that are observation-equivalent to ρ from P1's perception (resp. P2's perception). The set of all sequences of observations P1 (resp. P2) can observe for the plays of M is denoted $\mathsf{Obs}_1(M)$ ($\mathsf{Obs}_2(M)$). The inverse of observation function for player i is the function $O_i^{-1} : \mathsf{Obs}_i(M) \to 2^{\mathsf{Plays}(M)}$ such that for each $\eta \in \mathsf{Obs}_i(M)$, $O_i^{-1}(\eta) = \{\rho \in \mathsf{Plays}(M) \mid O_i(\rho) = \eta\}$. The following property is easy to prove:

Proposition 1. *For every play $\rho \in \mathsf{Plays}(M)$ and for each player i, $\rho \in O_i^{-1}(O_i(\rho))$.*

Objective in temporal logic: P1 has a temporal objective φ specified in Linear Temporal Logic over Finite Traces (LTL_f). The syntax of LTL_f formulas is given as follows.

Definition 3 ($\mathbf{LTL}_f$ [7]). *Let AP be a set of atomic propositions. An (LTL_f) formula over AP is defined inductively as follows:*

$$\varphi := p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \bigcirc\varphi \mid \varphi_1 \,\mathsf{U}\, \varphi_2 \mid \Diamond\varphi \mid \Box\varphi,$$

*where $p \in AP$; $\neg$, $\wedge$ and $\vee$ are the Boolean operators negation, conjunction and disjunction, respectively; and $\bigcirc$, U, $\Diamond$ and $\Box$ denote the temporal modal operators for **`next`**, **`until`**, **`eventually`** and **`always`** respectively.*

The operator $\bigcirc\varphi$ specifies that formula φ holds at the next time instant, while the operator $\varphi_1 \,\mathsf{U}\, \varphi_2$ denotes that there exists a future time instant at which φ_2 holds, and that φ_1 holds at all time instants up to and including that future instant. That is, the system must satisfy φ_1 continuously until a time instant in future at which φ_2 holds. The temporal operator $\Diamond\varphi$ specifies that φ holds at some instant in the future and the operator $\Box\varphi$ specifies that φ holds at all time instants from the current instant.

For any LTL_f formula φ over AP, a set of words $\mathsf{Words}(\varphi) \subseteq (2^{AP})^*$ that satisfy the formula is associated. A finite word $w \in (2^{AP})^*$ satisfies φ, denoted by $w \vDash \varphi$, iff w belongs to $\mathsf{Words}(\varphi)$. See [7] for detailed semantics of LTL_f.

To illustrate our definitions, we introduce a running example.

Example 1. (Part I) Consider the POMDP in Fig.1. To reduce visual clutter, only the transitions with non-zero probabilities are drawn and the exact probabilities of those transitions are omitted. The POMDP has 5 states, s_1 through s_5, and the sensor set Γ consists of sensors, A,B,C and D, which cover $\{s_2, s_3\}$, $\{s_3\}$, $\{s_4, s_5\}$ and $\{s_2, s_3, s_4\}$, respectively (shown by the dotted shapes in the figure). All the sensors are Boolean sensors. A Boolean sensor returns True when the current state is covered by the sensor and False otherwise. This sensor set is divided into the secured sensors $\Gamma_1 = \{B\}$ and the unsecured sensors $\Gamma_2 = \{A, C, D\}$. P1's task is to eventually reach state s_5, which is expressed using the LTL_f formula $\varphi = \Diamond\, s_5$. An example of the observations obtained by P1 and P2 is as follows. P1 starts from state s_1 and takes a control-perception action $(a, \{A, B\})$, and reaches state s_2 probabilistically. P1 obtains the observations based on both the sensors and thus $O_1(s_2, \{A, B\}) = \{s_2\}$, while P2 obtains the observations only based on the unsecured sensors $O_2(s_2, \{A, B\} \cap \Gamma_2) = O_2(s_2, \{A\}) = \{s_2, s_3\}$.

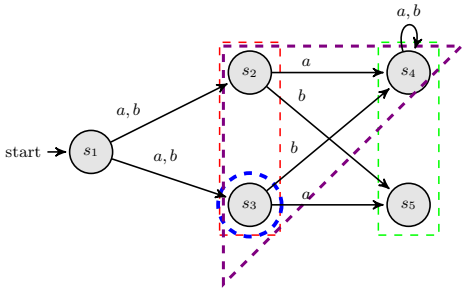


Fig. 1: An illustrative running example, POMDP with active perception M. The dashed region represents the sensors: red(A), blue(B), green(C) and violet(D).

P1's Strategy. In the POMDP with active perception M, P1 has to simultaneously either determine a control-perception action or terminate the game. A finite-memory, randomised strategy for P1 is a function $\pi : \mathsf{Plays}(M) \to \mathcal{D}(\mathcal{A}_1 \cup \{\ltimes\})$ where $\ltimes$ means P1 terminates the play. Because P1 has partial observations, P1 can only use observation-based strategy $\pi : \mathsf{Obs}_1(M) \to \mathcal{D}(\mathcal{A}_1 \cup \{\ltimes\})$ that maps an observation of a play to an action. P1 maintains for each time step k, the observation sequence $\eta_k = w_0(a_0, X_0)w_1 \dots w_k$ it has perceived up to time step k, and then, it feeds η_k to π to choose an action. A policy π induces a probability distribution $\Pr^\pi$ over $\mathsf{Plays}(M)$. Let Π be the set of all finite-memory, randomized, observation-based strategies for P1.

Definition 4 (Qualitative opacity). *An LTL_f formula φ is* opaque *to P2 with respect to a play $\rho \in \mathsf{Plays}(M)$ if and only if 1. $L(\rho) \vDash \varphi$; and 2. there exists at least one observation-equivalent play $\rho' \in [\rho]_2$ such that $L(\rho') \nvDash \varphi$.*

In words, P2 cannot tell from its observation if the formula is satisfied or not.

Definition 5 (Opacity-enforcing winning play). *Given a secret φ, a play $\rho \in \mathsf{Plays}(M)$ is* winning *if $L(\rho) \vDash \varphi$. The set of winning plays is denoted* WPlays.

A winning play ρ is opaque-enforcing winning *if φ is opaque to P2 with respect to ρ. The set of opacity-enforcing winning plays is denoted* $\mathsf{OWPlays}$.

Definition 6 (Opacity-enforcing winning strategy). *Given a prefix $\rho \in \mathsf{Plays}(M)$, a strategy $\pi \in \Pi$ is said to be a* winning *strategy for P1 if P1 can ensure to satisfy φ with probability one, that is, $\Pr^{\pi}(\{\rho' | \rho \cdot \rho' \in \mathsf{WPlays}\}) = 1$. A strategy π is said to be opacity-enforcing winning strategy for P1 if P1 can enforce an opacity-enforcing winning play starting from ρ with probability one, that is, $\Pr^{\pi}(\{\rho' | \rho \cdot \rho' \in \mathsf{OWPlays}\}) = 1$.*

Problem 1. Given a POMDP with active perception M in Def. 1 and P1's specification φ, given that P2 receives all sensor readings from insecured sensors, compute, if exists, an opacity-enforcing winning strategy for P1.

3 Main Result: Opacity-Enforcing Winning with 2-beliefs

In this section, we present a solution to Problem 1. We first make use of the fact that an LTL_f formula can be represented as a deterministic finite automaton to construct a product POMDP that augments the original POMDP states with task-relevant information. We use this product POMDP to formulate the opacity-enforcing planning problem and present solutions to compute P1's opacity-enforcing winning strategy using joint active perception and control.

As a first step, we encode the LTL_f formula into a finite-state automaton.

Definition 7 (Deterministic Finite Automaton (DFA)). *A DFA is a tuple $\mathbf{A} = (Q, \Sigma, \delta, \iota, F)$ with a finite set of states Q, a finite alphabet Σ, a transition function $\delta : Q \times \Sigma \to Q$, an initial state ι, and a set of accepting states $F \subseteq Q$.*

We assume the transition function is complete. That is, for any $(q, \sigma) \in Q \times \Sigma$, $\delta(q, \sigma)$ is defined. [1] The extended transition function $\delta : Q \times \Sigma^* \to Q$ is defined in the usual manner, i.e. , for each state $q \in Q$ and word $w_0 w_1 \dots w_n \in \Sigma^*$, $\delta(q, w_0 w_1 \dots w_n) = \delta(\delta(q, w_0), w_1 \dots w_n)$. Word $w = w_0 w_1 \dots w_n \in \Sigma^*$ is accepted by $\mathbf{A}$ if and only if $\delta(\iota, w) \in F$. The language of $\mathbf{A}$, denoted $L(\mathbf{A})$, consists of all those words accepted by $\mathbf{A}$, i.e. , $L(\mathbf{A}) = \{w \in \Sigma^* \mid \delta(\iota, w) \in F\}$.

The algorithm uses the idea of De Giacomo and Vardi [7] to convert the LTL_f formula φ into a DFA $\mathbf{A}$ with $\Sigma = 2^{AP}$ such that $\mathsf{Words}(\varphi) = L(\mathbf{A})$. From now on, we assume $\Sigma = 2^{AP}$. In the next step, we construct a product POMDP from the POMDP M and the DFA $\mathbf{A}$ to determine whether P1 can enforce an opacity-enforcing winning play from the initial state s_0.

Definition 8 (Product POMDP). *The product POMDP between the POMDP with active perception $M = (S, A, \mathbf{P}, \Omega, \Gamma = \Gamma_1 \cup \Gamma_2, O, s_0, \omega_1^0, AP, L)$ and the DFA $\mathbf{A} = (Q, \Sigma, \delta, \iota, F)$ is a tuple*

$$\mathcal{M} = (S \times Q, A, \Gamma, T, (s_0, q_0), Z, \mathcal{O}, B_1^0, B_2^0, S \times F)$$

[1] Any incomplete transition function can be made complete by adding a sink state and redirecting all the missing transitions to it.

in which (1) $S \times Q$ is the set of states; (2) A is the set of control actions; (3) Γ is the set of sensors; (4) $T : (S \times Q) \times A \to \mathcal{D}(S \times Q)$ is the transition function s.t. for states (s,q) and (s',q') and action a, $T((s,q),a,(s',q')) = \boldsymbol{P}(s,a,s')$ if $\delta(q, L(s')) = q'$. Otherwise, $T((s,q),a,(s',q')) = 0$; (5) (s_0,q_0) is the initial state where $q_0 = \delta(\iota, L(s_0))$; (6) $Z \subseteq 2^{(S\times Q)}$ is the set of observations; (7) $\mathcal{O} : (S\times Q)\times 2^{\Gamma} \to Z$ is the observation function s.t. for a state $(s,q) \in S\times Q$ and sensor subset $X \subseteq \Gamma$, $\mathcal{O}((s,q),X) = O(s,X) \times Q$; (8) $B_1^0 = \{(s,q) \mid s \in \omega_0, q = \delta(q_0, L(s))\}$ is the initial observation for P1; (9) $B_2^0 = \{(s,q) \mid s \in \omega_0^+, q = \delta(q_0, L(s))\}$ is the initial observation for P2; and (10) $S \times F$ is the set of goal states.

A finite play $\gamma = (s_0,q_0)(a_0,X_0)(s_1,q_1)(a_1,X_1)\dots(s_n,q_n)$ in $\mathcal{M}$, by the construction of the product POMDP, can be projected into a single finite play $\rho = s_0(a_0,X_0)s_1(a_1,X_1)\dots s_n$ in M. The projection of plays in $\mathcal{M}$ to plays in M is a bijection due to the deterministic transitions in the DFA.

By construction, the play ρ satisfies the specification φ iff there exists an integer $0 \le i \le n$ such that $(s_i,q_i) \in S \times F$.

The observation function in this product game is used to update both P1's belief and P2's belief. Let X be a sensor query performed by P1 and $(s,q) \in S\times Q$ be the state the product game $\mathcal{M}$ enters. P1's belief about the current state is updated using $\mathcal{O}((s,q),X)$ and P2's belief is updated using $\mathcal{O}((s,q),X \cap \Gamma_2)$. For this product game, function $\mathsf{Post}_{\mathcal{M}} : (S \times Q) \times A \to 2^{S\times Q}$ maps a state $(s,q) \in S \times Q$ and an action $a \in A$ to the set of possible reachable states as $\mathsf{Post}_{\mathcal{M}}((s,q),a) = \{(s',q') \in S \times Q \mid T((s,q),a,(s',q')) > 0\}$. We extend this function for domains $2^{(S\times Q)} \times A$ and $2^{(S\times Q)} \times 2^A$ such that for each $B \subseteq S \times Q$, $a \in A$, and $Y \subseteq A$, $\mathsf{Post}_{\mathcal{M}}(B,a) = \bigcup_{(s,q)\in B} \mathsf{Post}_{\mathcal{M}}((s,q),a)$ and $\mathsf{Post}_{\mathcal{M}}(B,Y) = \bigcup_{a\in Y} \mathsf{Post}_{\mathcal{M}}(B,a)$. Given a state $(s,q) \in S \times Q$, we use $\mathcal{M}[(s,q)]$ to denote a product POMDP obtained from $\mathcal{M}$ by letting (s,q) to be the initial state.

3.1 Computing an opacity-enforcing strategy

From the product POMDP $\mathcal{M}$, we formulate a one-player stochastic game to model the interaction of P1 and the environment, along with P2's observation.

Definition 9 (POMDP augmented with 2-beliefs). *Given the product POMDP $\mathcal{M} = (S \times Q, A, \Gamma, T, (s_0,q_0), Z, \mathcal{O}, B_1^0, B_2^0, S \times F)$, the POMDP augmented with 2-beliefs is a tuple*

$$\mathcal{G} = \langle V, \mathcal{A}_1, v_0, V_F, \Delta \rangle,$$

where (1) $V = \{((s,q),B_1,B_2) \mid s \in S, q \in Q, B_1 \subseteq (S\times Q), B_2 \subseteq (S\times Q)\}$ is the set of states, where B_1 and B_2 are beliefs of P1 and P2, respectively; (2) $\mathcal{A}_1 = A\times 2^{\Gamma}$ is the set of control-perception actions that can be taken by P1, as given in $\mathcal{M}$; (3) $v_0 = ((s_0,q_0),B_1^0,B_2^0)$ is the initial state; (4) $V_F = \{((s_F,q_F),B_1^F,B_2^F) \in V \mid (s_F,q_F) \in (S \times F), B_1^F \subseteq (S \times F), B_2^F \cap (S \times F) \neq \varnothing, B_2^F \cap S \times (Q \setminus F) \neq \varnothing\}$ are the set of goal *states (P1 aims to reach one of such a goal state); and (5) $\Delta : V \times \mathcal{A}_1 \to \mathcal{D}(V)$ is the probabilistic transition function such that all states in V_F*

are sink states, and for each state $v = ((s,q),B_1,B_2) \in V_F$*, action* $(a,X) \in \mathcal{A}_1$*, and state* $v' = ((s',q'),B_1^{'},B_2^{'})) \in V$*,* $\Delta(v,(a,X),v') = T((s,q),a,(s',q'))$ *if* $B_1^{'} = \mathsf{Post}_{\mathcal{M}}(B_1,a) \cap \mathcal{O}((s',q'),X)$*,* $B_2^{'} = \mathsf{Post}_{\mathcal{M}}(B_2,A) \cap \mathcal{O}((s',q'),X \cap \Gamma_2)$*, and* $q' = \delta(q,L(s'))$*, and otherwise,* $\Delta(v,(a,X),v') = 0$*.*

Each state $((s,q),B_1,B_2)$ of this product game indicates a situation where the true state of the system is (s,q), P1's belief about the current state is B_1, and P2's belief about the current state is B_2. Each transition $((s,q),B_1,B_2) \xrightarrow{(a,X)} ((s',q'),B_1^{'},B_2^{'})$ corresponds to a situation where P1 selects an action $(a,X) \in \mathcal{A}_1$, after which, the game stochastically transitions from the true state (s,q) to the new true state (s',q'). Then, with the sensor query, P1 observes $\mathcal{O}((s',q'),X)$ and thus updates its belief from B_1 to $B_1^{'}$ by considering the possible states in which it can be given the taken action a and eliminating the states that are inconsistent with the observation. Likewise, P2 observes $\mathcal{O}((s',q'),X \cap \Gamma_2)$ and updates P2's belief from B_2 to $B_2^{'}$ based on the information from unsecured sensors. If a state in V_F is reached, P1 chooses to terminate the play.

The following example shows the construction as described in Def. 9.

Example 2. (Part II) In Example 1, the secret task for P1 is $\varphi = \Diamond s_5$. The DFA corresponding to φ is shown in the Fig. 2a. Fig. 2b shows the product POMDP of this DFA and the POMDP in Example 1. From this product POMDP, the POMDP augmented with 2-beliefs $\mathcal{G}$ is constructed. Fig. 2c shows a partial construction of $\mathcal{G}$. In this figure, $B_1^0 = \{(s_1,0)\}$, $B_2^0 = \{(s_1,0),(s_2,0),(s_3,0),(s_4,0),(s_5,1)\}$, $B_1^4 = \{(s_5,1)\}$ and $B_2^4 = \{(s_4,0),(s_5,1)\}$. To see how the belief is updated, consider state $((s_3,0),\{(s_3,0)\},\{(s_2,0),(s_3,0)\})$, at which P1 knows the exact current state and P2 is uncertain if the current state is s_2 or s_3. If P1 takes action a and query sensor set $\{C,D\}$, state $(s_5,1)$ is reached with probability one. The sensor C returns 1 and sensor D returns 0. In this case, P2, who has access to both C and D, will know that $(s_5,1)$ is reached. On the other hand, if P1 takes action a and queries B and C, at state s_5, B outputs 0 and C outputs 1. P1 will know that s_5 is reached. P2, with only sensor C's information, cannot distinguish if state $(s_5,1)$ or $(s_4,0)$ is reached. The opacity is enforced and the play is winning for P1. The goal state in this figure has a self-loop for all actions. This is because all states in V_F are absorbing.

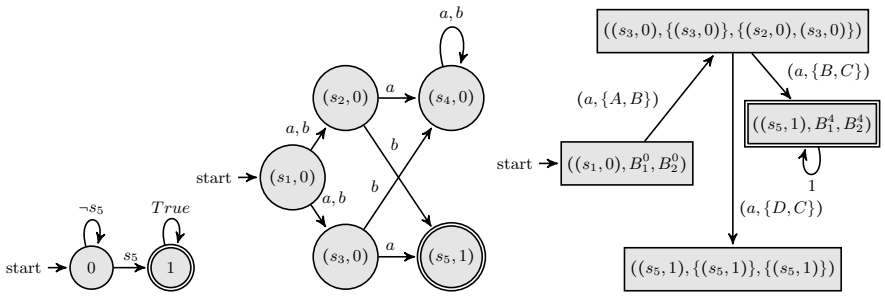
The opacity-enforcing winning strategy computation relies on the proof that in the presence of the eavesdropping attacker and partial observations, either player is sure that one of the states in its belief is the true state.

We now also show that the belief of P2 always includes the belief of P1.

Lemma 1. *For any state* $((s,q),B_1,B_2) \in V$ *reachable from the initial state* v_0*,* $B_1 \subseteq B_2$*.*

Proof. By induction on the lengths of the plays in the product game. For the initial state $v_0 = ((s_0,q_0),B_1^0,B_2^0)$, by the construction of B_1^0 and B_2^0 in Def. 8 and the assumption that $w_0 \subseteq w_0^+$ in Def. 1, we have $B_1^0 \subseteq B_2^0$.

Consider a play with length k in $\mathcal{G}$ such that $v_k = ((s_k,q_k),B_1^k,B_2^k)$ is the last state of the play. By induction hypothesis, $B_1^k \subseteq B_2^k$. For any state



(a) (b) (c)

Fig. 2: **(a)** DFA for the temporal goal $\varphi = \Diamond s_5$. **(b)** Product POMDP $\mathcal{M}$. **(c)** A fragment of POMDP with 2-beliefs $\mathcal{G}$, constructed from $\mathcal{M}$.

$v_{k+1} = ((s_{k+1}, q_{k+1}), B_1^{k+1}, B_2^{k+1})$ reached by an action $(a, X) \in \mathcal{A}_1$ from v_k, it holds that $B_1^{k+1} = \mathsf{Post}_{\mathcal{M}}(B_1^k, a) \cap \mathcal{O}((s_{k+1}, q_{k+1}), X)$ and $B_2^{k+1} = \mathsf{Post}_{\mathcal{M}}(B_2^k, A) \cap \mathcal{O}((s_{k+1}, q_{k+1}), X \cap \Gamma_s)$. Since $B_1^k \subseteq B_2^k$ and $a \subseteq A$, $\mathsf{Post}_{\mathcal{M}}(B_1^k, a) \subseteq \mathsf{Post}_{\mathcal{M}}(B_2^k, A)$. Also, given $(X \cap \Gamma_2) \subseteq X$, it holds that $\mathcal{O}((s_{k+1}, q_{k+1}), X) \subseteq \mathcal{O}((s_{k+1}, q_{k+1}), X \cap \Gamma_2)$. Hence, $B_1^{k+1} \subseteq B_2^{k+1}$. □

Lemma 2. *For any state $((s, q), B_1, B_2) \in V$ that is reachable from the initial state v_0, it holds that $(s, q) \in B_1$ and $(s, q) \in B_2$.*

Proof. We first show that $(s, q) \in B_1$ using induction on the lengths of the plays of $\mathcal{G}$. For the initial state $v_0 = ((s_0, q_0), B_1^0, B_2^0)$, by construction of belief from the initial observation, $(s_0, q_0) \in B_1^0$. Consider a play with length k in $\mathcal{G}$ such that $v_k = ((s_k, q_k), B_1^k, B_2^k)$ is the k-th state reached by a sequence of P1's actions. Assume $(s_k, q_k) \in B_1^k$. For any state $v_{k+1} = ((s_{k+1}, q_{k+1}), B_1^{k+1}, B_2^{k+1})$ reached from v_k by an action $(a, X) \in \mathcal{A}_1$, taken by P1, it holds that $T((s_k, q_k), a, (s_{k+1}, q_{k+1})) > 0$, $B_1^{k+1} = \mathsf{Post}_{\mathcal{M}}(B_1^k, a) \cap \mathcal{O}((s_{k+1}, q_{k+1}), X)$. Because $(s_k, q_k) \in B_1^k$, by construction we have $(s_{k+1}, q_{k+1}) \in \mathsf{Post}_{\mathcal{M}}(B_1^k, a)$. Also, because $s_{k+1} \in O(s_{k+1}, X)$, $(s_{k+1}, q_{k+1}) \in \mathcal{O}((s_{k+1}, q_{k+1}), X)$. Thus, $(s_{k+1}, q_{k+1}) \in B_1^{k+1}$.

This proof combined with the result of Lemma 1 proves $(s, q) \in B_2$. □

Lemma 3. *Let $\rho_{\mathcal{G}} = ((s_0, q_0), B_1^0, B_2^0)(a_0, X_0)((s_1, q_1), B_1^1, B_2^1) \dots ((s_n, q_n), B_1^n, B_2^n)$ be a play of $\mathcal{G}$. For each $(s'_n, q'_n) \in B_2^n$, there exists a play $\rho_{\mathcal{M}} = (s'_0, q'_0)(a'_0, X'_0)(s'_1, q'_1)(a'_1, X'_1) \dots (s'_n, q'_n)$ of $\mathcal{M}$ such that for each $0 \leq i < n$, $(s'_i, q'_i) \in B_2^i$ and $X'_i = X_i \cap \Gamma_2$.*

Proof. Proof by induction on $k = n, n-1, \dots, 0$. For $k = n$, $\rho_{\mathcal{G}}^n = ((s_n, q_n), B_1^n, B_2^n)$. By the statement assumption, $(s'_n, q'_n) \in B_2^n$, and clearly $\rho_{\mathcal{M}}^n = (s'_n, q'_n)$ is a play of $\mathcal{M}[(s'_n, q'_n)]$. Therefore, the statement holds for the induction's base case.

For the induction hypothesis, assume given the play

$$\rho_{\mathcal{G}}^k = ((s_k, q_k), B_1^k, B_2^k)(a_k, X_k)((s_{k+1}, q_{k+1}), B_1^{k+1}, B_2^{k+1}) \dots ((s_n, q_n), B_1^n, B_2^n)$$

of $\mathcal{G}$, there exists a sequence

$$\rho^k_{\mathcal{M}} = (s'_k, q'_k)(a'_k, X_k \cap \Gamma_2)(s'_{k+1}, q'_{k+1})(a'_{k+1}, X_{k+1} \cap \Gamma_2) \dots ((s'_n, q'_n))$$

where $(s'_i, q'_i) \in B^i_2$ for each $k \leq i \leq n$, is a play of $\mathcal{M}$. Now, consider the play

$$\rho^{k-1}_{\mathcal{G}} = ((s_{k-1}, q_{k-1}), B^{k-1}_1, B^{k-1}_2)(a_{k-1}, X_{k-1}) \dots ((s_n, q_n), B^n_1, B^n_2)$$

of $\mathcal{G}$. Given $\rho^{k-1}_{\mathcal{G}}$ is a play in $\mathcal{G}$, $\mathcal{G}$ has a transition from $((s_{k-1}, q_{k-1}), B^{k-1}_1, B^{k-1}_2)$ to $((s_k, q_k), B^k_1, B^k_2)$ with action (a_{k-1}, X_{k-1}). By the construction of $\mathcal{G}$'s transition function, this implies that $B^k_2 = \mathsf{Post}_{\mathcal{M}}(B^{k-1}_2, A) \cap \mathcal{O}((s_k, q_k), X_{k-1} \cap \Gamma_2)$, which means (1) $B^k_2 \subseteq \mathsf{Post}_{\mathcal{M}}(B^{k-1}_2, A)$, and (2) $B^k_2 \subseteq \mathcal{O}((s_k, q_k), X_{k-1} \cap \Gamma_2)$. Given (1) and that $(s'_k, q'_k) \in B^k_2$, it holds that there exists $(s'_{k-1}, q'_{k-1}) \in B^{k-1}_2$ and $a'_{k-1} \in A$ such that $(s'_k, q'_k) \in \mathsf{Post}_{\mathcal{M}}((s'_{k-1}, q'_{k-1}), a'_{k-1})$, which implies that in $\mathcal{M}$, there exists a transition with action a'_{k-1} from state (s'_{k-1}, q'_{k-1}) to state (s'_k, q'_k), and by (2), it holds that $(s'_k, q'_k) \in \mathcal{O}((s_k, q_k), X_{k-1} \cap \Gamma_2)$. These two combined imply that $\rho^{k-1}_{\mathcal{M}} = (s'_{k-1}, q'_{k-1})(a'_{k-1}, X_{k-1} \cap \Gamma_2)(s'_k, q'_k)(a'_k, X_k \cap \Gamma_2) \dots ((s'_n, q'_n))$ is a play of $\mathcal{M}$ and for each $k \leq i \leq n$, $(s'_k, q'_k) \in B^k_2$. □

The above properties are important to construct an opacity-enforcing strategy to satisfy the given specification. Even in situations where P1 may only have a belief such that it is a subset of $(S \times F)$ and P1 does not perfectly know the current true state, P1 knows that the specification has been satisfied from Lemma 2. Also, with Lemma 1, we know that for P1 to enforce opacity, it is not sufficient to reach a state such that only P1's belief is a subset of $(S \times F)$ as it means that P2's belief always encompasses P1's belief and hence, P1 must ensure that P2's belief has at least one additional state that is not in $(S \times F)$.

In the above constructed *POMDP augmented with 2-beliefs*, P1 is tasked with reaching the goal states in V_F with probability one. We show that reaching these goal states with probability one would ensure that P1 would satisfy the given specification while ensuring opacity in the POMDP M.

Definition 10 (Belief-Based Winning Strategy/Region).
A strategy $\pi : V \to \mathcal{D}(\mathcal{A}_1)$ in the POMDP with 2-beliefs $\mathcal{G}$ is winning *at state v_0 for P1 if by starting from v_0 and following π, P1 ensures to reach a state $((s, q), B_1, B_2)$ where $B_1 \subseteq S \times F$ with probability 1. Strategy π is* opacity-enforcing winning *at a state v_0 if by starting from v_0 and following π, P1 guarantees to reach a goal state V_F with probability 1. It is well known that belief-based strategies are sufficient to win almost-surely the reachability game for P1 [2] and thus, a strategy $\pi : V \to \mathcal{D}(\mathcal{A}_1)$ is* belief-based *if for every pair of states $((s, q), B_1, B_2), ((s', q'), B_1, B_2) \in V$, it holds $\pi((s, q), B_1, B_2) = \pi((s', q'), B_1, B_2)$. A set of states from which P1 has a belief-based winning strategy is called* P1's winning region*, denoted as $\mathsf{Win}(\mathcal{G})$. P1's* opacity-enforcing winning region*, denoted $\mathsf{OWin}(\mathcal{G})$, consists of those states from which P1 has an opacity-enforcing winning strategy.*

Theorem 1. *A belief-based winning strategy π in the POMDP augmented with 2-beliefs $\mathcal{G}$ is also winning in the POMDP with active perception M and enforces opacity and winning with respect to its temporal objective φ.*

Proof. Let $\rho_o = ((s_0,q_0),B_1^0,B_2^0)(a_0,X_0)((s_1,q_1),B_1^1,B_2^1)\dots((s_n,q_n),B_1^n,B_2^n)$ be a play generated by π over $\mathcal{G}$. This play is projected onto play $\rho_M = s_0(a_0,X_0)$ $s_1\dots s_n$ on M. Because π is a winning strategy, $((s_n,q_n),B_1^n,B_2^n) \in V_F$, implying $q_n \in F$ and $B_1^n \subseteq (S\times F)$. This means that $L(s_0s_1\cdots s_n) \vDash \varphi$, which means that $\rho_M \vDash \varphi$ and the play ρ_M is winning for P1. Because this ρ_o is selected arbitrarily, then this means π is a winning strategy for M. Also, given that $B_1^n \subseteq (S\times F)$, P1 knows that being in any one of the states in its belief, it satisfies the specification.

For P2's belief, it holds that $B_2^n \cap (S\times(Q\setminus F)) \neq \varnothing$. That is, there exists a state $(s',q') \in B_2^n$ such that $q' \neq F$. For each such a state (s',q'), by Lemma 3, there exists a play $\rho_{\mathcal{M}}^- = (s_0',q_0')(a_0',X_0')(s_1',q_1')\dots(s_n',q_n')$ of $\mathcal{M}$ where $X_i' = X_i \cap \Gamma_2$ for all $0 \le i < n$. This implies that $\sigma_M^- = s_0's_1'\dots s_n'$ is a run of M and $L(\sigma_M^-) \nvDash \varphi$. Given Proposition 1, P2 believes that run $\sigma_M^+ = s_0s_1\dots s_n$ has a non-zero probability to have been executed. Therefore, because P2 believes any of the two runs $\sigma_M^+ = s_0s_1\dots s_n$ and σ_M^- of M where $L(\sigma_M^+) \vDash \varphi$ and $L(\sigma_M^-) \nvDash \varphi$ could have been executed, π is opacity-enforcing. □

We introduce Alg. 1 to compute a belief-based winning strategy for P1. The algorithm initializes a set $Y_0 = V$ and iteratively computes Y_{k+1} from Y_k for $k \ge 0$. At each iteration k, Alg. 1 computes the set of states from which it can reach a state in V_F with a positive probability while ensuring to stay within the set Y_k with probability 1. The algorithm uses the following function

$$\mathsf{Allow}(v,Y) = \{(a,X) \in \mathcal{A}_1 \mid \mathsf{Post}_{\mathcal{G}}(v,(a,X)) \subseteq Y\}, \forall v \in V, Y \subseteq V$$

where $\mathsf{Post}_{\mathcal{G}}(v,(a,X)) = \{v' \in V \mid \Delta(v,(a,X),v') > 0\}$ is the set of states reachable from state v by playing action (a,X). By definition, by playing an action from the allowed set $\mathsf{Allow}(v,Y)$, P1 can be sure to stay within state set Y. Let for a state $v = ((s,q),B_1,B_2) \in V$, $[v]_\sim$ denote the set of belief-equivalent states with v such that $[v]_\sim = \{((s'q'),B_1',B_2') \in V \mid B_1' = B_1, B_2' = B_2\}$. Then, let

$$\mathsf{Allow}([v]_\sim,Y) = \bigcap_{v'\in[v]_\sim} \mathsf{Allow}(v',Y).$$

Thus, we have that an action is allowed for P1 to play at a state v if and only if that action is allowed for P1 at all of its belief-equivalent states v'.

Next, we define the following progress function given a set of states $Y \subseteq V$ and a set $R \subseteq Y$,

$$\mathsf{Prog}(R,Y) = \{v \in Y \mid \exists (a,X) \in \mathsf{Allow}([v]_\sim,Y), \mathsf{Post}_{\mathcal{G}}(v,(a,X)) \cap R \neq \varnothing\}.$$

The progress function yields a set of states from which P1 has at least one allowed action to reach R in the next state.

In the inner loop of Alg. 1 (Lines 4-7), we fix the Y_k and iteratively update R_i until a fixed point is reached. Intuitively, the fixed point is a set of states from which P1 can ensure to reach V_F with probability ≥ 0 and stay within Y_k with probability 1. Then, we set this fixed point in the inner loop as Y_{k+1} and continue with the computation until the fixed point is reached at the outer loop $Y_{n+1} = Y_n$ for some $n \ge 0$. The belief-based winning region for P1 is $\mathsf{OWin}(\mathcal{G}) = Y_n$.

We next show that $\mathsf{OWin}(\mathcal{G})$ obtained using Alg. 1 is, in fact, the opacity-enforcing almost-sure winning region for P1 and at every state in $\mathsf{OWin}(\mathcal{G})$, P1 has an opacity-enforcing strategy to reach R with probability one.

Algorithm 1 Belief-Based Opacity-Enforcing ASW Region

Inputs: POMDP augmented with 2-belief $\mathcal{G}$.
Outputs: P1's opacity-enforcing ASW region, $\mathsf{OWin}(\mathcal{G})$.

$k \leftarrow 0$; $Y_k \leftarrow V$
repeat
  $i \leftarrow 0$; $R_i \leftarrow V_F$
  repeat
    $R_{i+1} \leftarrow R_i \cup \mathsf{Prog}(R_i, Y_k)$
    $i \leftarrow i + 1$
  until $R_{i+1} = R_i$
  $Y_{k+1} \leftarrow R_i$; $k \leftarrow k + 1$
until $Y_{k+1} = Y_k$
return $\mathsf{OWin}(\mathcal{G}) \leftarrow Y_k$.

Lemma 4. $\mathsf{OWin}(\mathcal{G})$ *computed by Alg. 1 is opacity-enforcing winning for P1.*

Proof is in the appendix. From the obtained $\mathsf{OWin}(\mathcal{G})$, P1's opacity-enforcing belief-based strategy can be defined as function $\pi_1^* : \mathsf{OWin}(\mathcal{G}) \to 2^{\mathcal{A}_1}$ such that

$$\pi_1^*(v) = \{(a, X) \in \mathcal{A}_1 \mid (a, X) \in \mathsf{Allow}([v]_\sim, \mathsf{OWin}(\mathcal{G}))\}. \tag{1}$$

Thus, at each state $v \in \mathsf{OWin}(\mathcal{G})$, P1 has to play an action in $\pi_1^*(v)$. Also, by the construction of $\mathsf{Allow}(\cdot)$, for every states $v = ((s, q), B_1, B_2)$ and $v' = ((s', q'), B_1', B_2')$, if $B_1 = B_1'$ and $B_2 = B_2'$, then $\pi_1^*(v) = \pi_1^*(v')$.

Theorem 2. *By playing the strategy π_1^* defined in Eq. 1, P1 ensures that the game eventually reaches V_F.*

The proof for the theorem is provided in the appendix.

Complexity Analysis: The algorithm first encodes the secret goal φ into a DFA. This step takes a doubly-exponential time in the size of φ in the worst case [19,6]. However, for commonly seen LTL$_f$ formulas in robotic planning and AI applications, this translation is tractable. The POMDP augmented with 2-beliefs has $O(|S||Q|2^{|S||Q|}2^{|S||Q|})$ reachable states in the worst case. Each non-goal state $v = ((s, q), B_1, B_2)$ has $O(|S|)$ transitions with non-zero probabilities for each action $(a, X) \in A \times 2^\Gamma$ in the worst case. Accordingly, using appropriate data structures, mainly hash tables, it takes $O(|A||S||Q|2^{2|S||Q|+|S|})$ to construct the POMDP augmented with 2-beliefs. Computing an opacity-enforcing winning strategy for $\mathcal{G}$ takes a quadratic time to the size of $\mathcal{G}$ in the worst case. Therefore, the final running time of our algorithm is $O(|A|^2|S|^2|Q|^2 2^{(4|S||Q|+2|S|)})$.

Example 3. (Part III) We show how to use Alg. 1 to obtain the opacity-enforcing winning region for the fragment of the opacity-enforcing game in Fig. 2c. From

Fig. 2c, we have, $v_1 = \{((s_1, 0), B_1^0, B_2^0)\}$, $v_2 = \{((s_3, 0), \{(s_3, 0)\}, \{(s_2, 0), (s_3, 0)\})\}$, $v_3 = \{((s_5, 1), \{(s_5, 1)\}, \{(s_5, 1)\})\}$ and $v_4 = \{((s_5, 1), \{(s_5, 1)\}, \{(s_4, 0), (s_5, 1)\})\}$.

We start with $Y_0 = \{v_1, v_2, v_3, v_4\}$ where v_1 is the initial state in Fig. 2c, and v_4 is the final state.

Moving on to the inner loop, we initialize the set R_0 as $\{v_4\}$ since it is the only final state in the fragment under consideration. Next, we compute $\mathsf{Prog}(R_0, Y_0)$, which yields $\{v_2, v_4\}$. Subsequently, R_1 is updated as $\{v_2, v_4\}$, and we compute $\mathsf{Prog}(R_1, Y_0)$, resulting in $\{v_1, v_2, v_4\}$. This set represents the fixed point for this iteration. Consequently, we update Y_1 as $\{v_1, v_2, v_4\}$, which is the fixed point for the outer loop and the opacity-enforcing winning region.

3.2 When To Stop Tracking P2's Beliefs?

In the construction of the POMDP augmented with 2-beliefs presented in Def. 9, we face the issue of exponential growth of the state space. To mitigate the issue, we use the minimal DFA accepting φ. This reduces the state space of $\mathcal{M}$ and $\mathcal{G}$. Additionally, in certain situations, we can stop tracking P2's belief by leveraging certain properties in the specification automaton and P1's winning region computed from the product POMDP without opacity constraints.

First, we introduce some notions.

Given a DFA $\mathcal{I}$, we define the subautomata for each state $q \in Q$ as follows:

Definition 11. *Given a DFA $\mathcal{I} = (Q, \Sigma, \delta, \iota, F)$, for any $q \in Q$,*

- good suffixes $\mathcal{L}_q$ *is the language of the DFA* $\mathcal{I}_q = (Q, \Sigma, \delta, q, F)$.
- bad suffixes $\overline{\mathcal{L}_q}$ *is the language of the DFA* $\overline{\mathcal{I}}_q = (Q, \Sigma, \delta, q, Q \setminus F)$.

Given two languages $\mathcal{L}_1$ and $\mathcal{L}_2$, represented by DFAs $\mathcal{I}_1 = (Q_1, \Sigma, \delta_1, \iota_1, F_1)$ and $\mathcal{I}_2 = (Q_2, \Sigma, \delta_2, \iota_2, F_2)$, checking if $\mathcal{L}_1 \subseteq \mathcal{L}_2$ is equivalent to see whether $\mathcal{L}_1 \cap \overline{\mathcal{L}_2} = \varnothing$, which can be achieved by checking whether the language of the product DFA $\mathcal{I}_1 \times \mathcal{I}_2 = (Q_1 \times Q_2, \Sigma, \delta, (\iota_1, \iota_2), F_1 \times (Q \setminus F_2))$ where $\delta((q_1, q_2), \sigma) = (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma))$ for each $(q_1, q_2) \in Q_1 \times Q_2$ and $\sigma \in \Sigma$, is empty or not.

Next, we solve P1's belief-based winning strategy without enforcing opacity to the observer. In this case, we need not to keep track of P2's belief.

Definition 12 (Product POMDP augmented with P1's belief). *Given the product POMDP $\mathcal{M} = (S \times Q, A, \Gamma, T, (s_0, q_0), Z, \mathcal{O}, B_1^0, B_2^0, S \times F)$, the product POMDP augmented with P1's belief is a tuple*

$$\mathcal{H} = \langle H, \mathcal{A}_1, H_F, \mathcal{T}, h_0 \rangle$$

in which 1. $H = \{((s, q), B_1) \mid s \in S, q \in Q, B_1 \subseteq (S \times Q)\}$ is the set of states, where B_1 is P1's beleif; 2. $\mathcal{A}_1$ is the set of control-perception actions that can be taken by P1, as given in $\mathcal{M}$; 3. $h_0 = ((s_0, q_0), B_1^0)$ is the initial state, where B_1^0 is the initial belief as in $\mathcal{M}$; 4. $H_F = \{((s_F, q_F), B_1^F) \mid (s_F, q_F) \in (S \times F), B_1^F \subseteq (S \times F)\}$ is the set of final states, which ensure P1 satisfies the objective φ; and 5. $\mathcal{T} : H \times \mathcal{A}_1 \to \mathcal{D}(H)$ is the probabilistic transition function. First, all states

in H_F are sink states. For a state $h \in H \setminus H_F$, for each action $(a, X) \in \mathcal{A}_1$ and state $h' = ((s', q'), B_1^{'})) \in H$, $\mathcal{T}(h, (a, X), h') = T((s, q), a, (s', q'))$ if $B_1^{'} = \mathsf{Post}_{\mathcal{M}}(B_1, a) \cap \mathcal{O}((s', q'), X)$, and otherwise, $\mathcal{T}(h, (a, X), h') = 0$.

P1's belief-based winning strategy in $\mathcal{H}$ can be solved using a slight modification of Alg. 1: Let

$$\mathsf{Allow}(h, Y) = \{(a, X) \in \mathcal{A}_1 \mid \mathsf{Post}_{\mathcal{H}}(h, (a, X)) \subseteq Y\}, \forall h \in H, Y \subseteq H.$$

and $\mathsf{Allow}([h]_{\sim}, Y) = \bigcap_{h' \in [h]_{\sim}} \mathsf{Allow}(h', Y)$. where $[((s, q), B_1))]_{\sim} = \{((s'q'), B_1^{'}) \in H \mid B_1^{'} = B_1\}$.

The progress function is defined as

$$\mathsf{Prog}(R, Y) = \{h \in Y \mid \exists (a, X) \in \mathsf{Allow}([h]_{\sim}, Y), \mathsf{Post}_{\mathcal{H}}(h, (a, X)) \cap R \neq \varnothing\}.$$

We have that Y_k is initialized to H and R_i is initialized to H_F. Intuitively, these modification allows P1 to compute belief-based winning strategy only considering his own belief.

Thus, without constructing the POMDP with two-beliefs $\mathcal{G}$, we can obtain $\mathsf{Win}(\mathcal{G}) = \{((s, q), B_1, B_2) \in V \mid ((s, q), B_1) \in \mathsf{Win}(\mathcal{H})\}$ that includes a set of states P1 can enforce a winning play (with/without opacity to P2).

Remark 2. $\mathsf{OWin}(\mathcal{G}) \subseteq \mathsf{Win}(\mathcal{G})$.

The following Lemma is crucial: It enables us to determine if an opacity-enforcing winning strategy exists without enumerating all beliefs that can be reached in the POMDP augmented with 2-beliefs $\mathcal{G}$.

Lemma 5. *For any state $v = ((s, q), B_1, B_2) \in V$ where $v \in \mathsf{Win}(\mathcal{G})$, if there exists $p \in Q$ s.t. $(s, p) \in B_2$ and $\mathcal{L}_q \subseteq \overline{\mathcal{L}_p}$, then $v \in \mathsf{OWin}(\mathcal{G})$.*

Proof. Given that $v \in \mathsf{Win}(\mathcal{G})$, P1 can enforce a play $\rho = s_0(a_0, X_0)s_1(a_1, X_1)\ldots s_n$ where $s_0 = s$ such that $\delta(q, L(\rho)) \in F$. As a result, $L(\rho) \in \mathcal{L}_q$ is a good suffix for the language $\mathcal{L}(\mathbf{A})$ given the state q.

Given that $\mathcal{L}_q \subseteq \overline{\mathcal{L}_p}$, $L(\rho) \in \overline{\mathcal{L}_p}$. Let $\mathsf{p} = ((s_0, q_0), B_1^0, B_2^0)(a_0, X_0)\ldots ((s_n, q_n), B_1^n, B_2^n)$ be in $\mathcal{G}$, the play corresponding to the play ρ. That is, the projection of p onto S is ρ. Let $p_n = \delta(p, L(\rho))$. It holds that $(s_n, p_n) \in B_2^n$ by the construction of P2's belief. Because $L(\rho) \subseteq \overline{\mathcal{L}_p}$, then $p_n \in Q \setminus F$. Further since $B_1^n \subseteq B_2^n$ and $B_1^n \subseteq S \times F$, then $B_2^n \cap (S \times (Q \setminus F)) \neq \varnothing$ and $B_2^n \cap S \times F \neq \varnothing$. The play ρ is opaque and winning for P1. □

Definition 13 (Augmented POMDPs with Trimmed 2-Beliefs). *Given the product POMDP $\mathcal{M}$ and the winning region of P1 without opacity constraint $\mathsf{Win}(\mathcal{G})$, the Augmented POMDPs with Trimmed 2-Beliefs is a tuple*

$$\mathcal{G}' = \langle W := \mathsf{Win}(\mathcal{G}) \cup \mathsf{Win}(\mathcal{H}), \mathcal{A}_1, W_F := V_F \cup \mathsf{Win}(\mathcal{H}), \Delta', v_0 \rangle$$

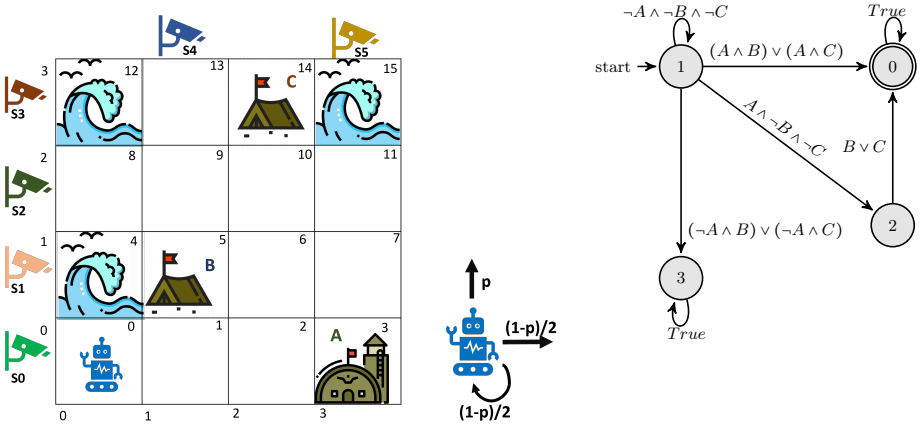
where W is the set of states; $\mathcal{A}_1$ and v_0 are from Def. 9; W_F is the set of goals states and is the union of V_F (the goal states of $\mathcal{G}$, as in Def. 9) and $\mathsf{Win}(\mathcal{H})$; and Δ' is the probabilistic transition function s.t. all the goal states in W_F are sink states, and for each non-goal state $w = ((s, q), B_1, B_2) \in \mathsf{Win}(\mathcal{G}) \setminus W_F$,

– *if there exists* $(s,p) \in B_2$ *such that* $\mathcal{L}_q \subseteq \overline{\mathcal{L}_p}$, $\Delta'(w,((s,q),B_1)) = 1$. *That is, with probability one, P1 reaches a state in* $\mathsf{Win}(\mathcal{H})$ *and stop tracking P2's belief.*
– *otherwise, for each action* $(a,X) \in \mathcal{A}_1$, $\Delta'(w,(a,X),((s',q'),B_1',B_2')) = T((s,q),a,(s',q'))$, *where* $B_1' = \mathsf{Post}_{\mathcal{M}}(B_1,a) \cap \mathcal{O}((s',q'),X)$ *and* $B_2' = \mathsf{Post}_{\mathcal{M}}(B_2,A) \cap \mathcal{O}((s',q'),X \cap \Gamma_2)$.

We now solve the above augmented POMDPs with trimmed 2-beliefs using Alg. 1. P1 follows the policy computed from $\mathcal{G}'$ until a state $((s,q),B_1) \in \mathsf{Win}(\mathcal{H})$ or a goal state V_F is reached. If a state $((s,q),B_1) \in \mathsf{Win}(\mathcal{H})$ is reached, then P1 transitions to the winning policy of the game $\mathcal{H}$ and adhere to it.

4 Experimental validation

In this section, we demonstrate the use of the developed opacity-enforcing planning for P1, an autonomous robot tasked with delivering medicine in a GPS denied environment. Fig. 3a shows the environment setup. The robot is tasked with first reaching the base station, A in the gridworld, to pick up essential supplies and then delivering the critical supplies to one of the critical zones B and C. This task is specified by the LTL$_f$ formula $\varphi = \neg(\mathsf{B} \vee \mathsf{C})\,\mathsf{U}\,(\mathsf{A} \wedge \Diamond(\mathsf{B} \vee \mathsf{C}))$. The robot is expected to keep this task opaque from the adversarial observer.

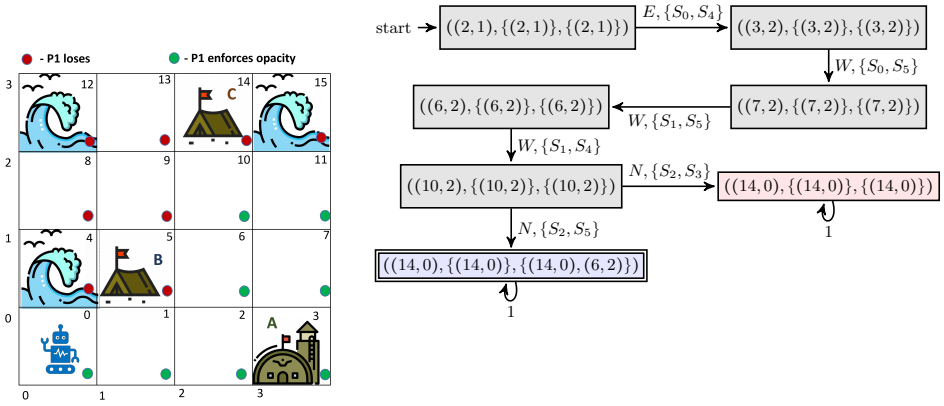


(a) (b) (c)

Fig. 3: **(a)** Autonomous robot in an adversarial environment. **(b)** The stochasticity in the environment when robot chooses to traverse N. **(c)** Specification DFA for $\varphi = \neg(\mathsf{B} \vee \mathsf{C})\,\mathsf{U}\,(\mathsf{A} \wedge (\mathsf{B} \vee \mathsf{C}))$.

The environment is monitored by the sensors $\{S_0, S_1, S_2, S_3, S_2, S_5\}$ as shown in the Fig. 3a. The sensors S_0,S_1,S_2 and S_3 respectively cover the rows 0, 1, 2 and 3, and the sensors S_4 and S_5 cover the columns 1 and 3 respectively. Only S_4 is

a secured sensor. Sensors S_0 and S_2 are precision range sensors and each returns the the exact position of the robot if the robot is in a cell in the range of the sensor and returns FALSE otherwise. The other sensors are Boolean range sensors and each returns TRUE if the robot is in a cell that is covered by the sensor and returns FALSE if not. The robot has partial observations of its position, obtained from the deployed sensor network. The eavesdropping adversary has partial observation of the robot's position from unsecured sensors. Due to limited communication constraint, for any belief B and any control action in A, the set of perception actions for the robot to select along with the control action includes querying any two sensors that cover at least one of the possible next states, i.e. , states in $\mathsf{Post}_{\mathcal{M}}(B, A)$. The robot traverses through the environment with four possible actions $\{\mathsf{N}, \mathsf{S}, \mathsf{E}, \mathsf{W}\}$. When the robot performs an action, its actuators guarantee that it reaches the *intended* cell with a probability p, make the robot to *stay* in the same cell with a probability $(1-p)/2$, and transit the robot to the cell that is 90 degrees (clockwise) apart from the intended cell with a probability $(1-p)/2$ as shown in the Fig. 3b. The gridworld environment is surrounded by bouncy walls. If the agent takes an action and hits a wall, it remains in the original cell. The cells 4, 12, and 15 are the unsafe zones for the robot. In a unsafe zone is reached, the robot gets stuck.



(a) (b)

Fig. 4: **(a)** The results of experimentation for every initial state in the gridworld. **(b)** Fragment of opacity-enforcing and not enforcing run starting from cell 2.

Fig. 3c shows the DFA encoding the temporal formula φ. This DFA has a non-accepting sink state state 3, and an accepting goal state 0. Fig. 4 shows results obtained from solving the opacity-enforcing winning strategy for the robot. Each green dot represents an initial state from which the robot has a opacity-enforcing winning strategy. Each red dot represent a state that has no winning strategy.

We discuss our results for the case where P1's initial position is cell 0 and both P1 and P2 are aware of this. Fig. 4b shows a fragment of a run in the mod-

ified opacity-enforcing game for the above example. In the shown fragment, we have an opacity-enforcing winning run and a run that does not enforce opacity when the robot starts from the cell 0 and has reached the cell 2. Here, we encode a state as $((s,q),B_1,B_2)$ where (s,q) represents the robot's true state with s being the cell number and q the automata state, B_1 is the robot's belief of its position and B_2 is the observer's belief of the robots position. From Fig. 4b, we observe that the sensor query of the robot affects the enforcement of opacity at state $((10,2),\{(10,2)\},\{(10,2)\})$. From state $((10,2),\{(10,2)\},\{(10,2)\})$, for the robot to reach the critical zone C it can take action N along with one perception action, i.e. , a query of a pair of sensors selected from $\{S_2,S_5\}$, $\{S_2,S_3\}$, $\{S_2,S_4\}$ or $\{S_2,S_1\}$ since any of these possible perception actions will enable the robot know it's true next state. Querying $\{S_2,S_3\}$ or $\{S_2,S_1\}$ does not enforce opacity as they provide the observer with enough information to ensure that the robot is precisely in the critical zone C.

To assess the effectiveness of the opacity-enforcing policy, we conducted empirical evaluations through statistical analysis. We performed $50,000$ iterations of the simulation in the gridworld setup, starting from the initial state $((0,1),\{(0,1)\},\{(0,1)\})$, for which the robot has an opacity-enforcing winning strategy. We let the robot to randomly select actions from the winning strategy computed from $\mathsf{Win}(\mathcal{H})$. We observed the robot achieved task satisfaction in all iterations while enforcing opacity in 47.07% of the iterations. In the remaining 52.93% of iterations, the robot successfully satisfied the task specification but did not enforce opacity. If the robot uses the strategy computed from $\mathsf{Win}(\mathcal{G})$, with probability 1, the robot not only satisfies the task specification but also enforces opacity.

Next, we see how the trimming technique reduces computation. The construction of $\mathcal{G}$ resulted in a total of $135,334$ states, and with the use of the modified construction of the game, i.e. on constructing the game $\mathcal{G}'$ with the trimming of the beliefs results in a total of $51,763$. This significant reduction of the state space aids in faster computation of the almost-sure winning region/strategy.

The experiments were executed on an Intel (R) Core (TM) i7 CPU @ 3.2GHz.

5 Conclusion and Future Work

In this work, we formulated and solved the problem of synthesizing a joint control and active perception for an agent in a stochastic environment to satisfy a secret temporal goal while enforcing opacity against a passive observer with partial observations. Building on the modeling and solution approaches, several future directions can be considered: First, a quantitative variant of opacity-enforcing planning remains to be investigated. For example, from the set of states at which P1 does not have a winning and opacity-enforcing strategy, is it possible to compute a strategy that ensures winning with probability $\geq p$ and opacity with probability $\geq \epsilon$? This would enable a more nuanced understanding of opacity enforcement and the development of strategies that account for varying degrees of opacity and task performance. Second, opacity-enforcement can be considered for the agent and the observer with different capabilities in control and percep-

tion. In this work, the information to the observer is a subset of the agent's information. This assumption is no longer valid if the observer has a different information channel that is uncontrollable and inaccessible by the agent. It would be interesting to know which subsets of opacity-enforcing games are decidable.

A Proof for Lemma 4

Proof. Let N be the index where $Y_N = Y_{N+1}$. To prove $Y_N = \mathsf{OWin}(\mathcal{G})$, we prove the following: 1. $\mathsf{OWin}(\mathcal{G}) \subseteq Y_j$ for all $0 < j \leq N$, using induction. Since $Y_0 = V$, we have that $\mathsf{OWin}(\mathcal{G}) \subseteq Y_0$. Assume that $\mathsf{OWin}(\mathcal{G}) \subseteq Y_i$ for some $i > 0$. Then, Y_{i+1} includes any state that has a strategy to reach V_F with positive probability while staying within Y_i. Thus, for any state in $Y_i \setminus Y_{i+1}$, P1 cannot stay within Y_i with probability one. However, P1 has a strategy to ensure that the game stays within $\mathsf{OWin}(\mathcal{G})$ and thereby Y_i, given $\mathsf{OWin}(\mathcal{G}) \subseteq Y_i$. As Y_{i+1} only removes the states that cannot ensure to stay within $\mathsf{OWin}(\mathcal{G})$, we have that $\mathsf{OWin}(\mathcal{G}) \subseteq Y_{i+1}$.

2. $Y_N \setminus \mathsf{OWin}(\mathcal{G}) = \varnothing$, by contradiction. Assume that there exists a state $v \in Y_N \setminus \mathsf{OWin}(\mathcal{G})$. Then, by construction, for any $v \in R_k \cup \mathsf{Prog}(R_k, Y_N)$, P1 has a strategy to reach V_F with positive probability in finitely many steps. Let E_T be the event that "starting from a state in Y_N, a run reaches a state in V_F within T steps" and let $\gamma > 0$ be the minimal probability for an event E_T to occur for any state $v \in Y_N$. Then, the probability of not reaching a state in V_F in infinitely many steps can be upper bounded by $\lim_{k\to\infty} (1-\gamma)^k = 0$. Therefore, for any $v \in Y_N$, P1 has a strategy to ensure a state in V_F is reached with probability one. This contradicts the assumption $v \notin \mathsf{OWin}(\mathcal{G})$. Thus, $Y_N = \mathsf{OWin}(\mathcal{G})$. □

B Proof for Theorem 2

Proof. Consider the level sets $R_0, \ldots, R_N$ obtained using Alg. 1 with input $\mathsf{OWin}(\mathcal{G})$. Let $0 < k \leq K$ be a level and $v \in R_k$ be a state. Suppose there exists an action $(a, X) \in \pi_1^*(v)$ s.t. $\mathsf{Post}_{\mathcal{G}}(v, (a, X)) \in R_{k-1}$. Since the probability of taking action (a, X) is non-zero, the level strictly decreases with a positive probability. Moreover, for any action in $\pi_1^*(v)$ and its probabilistic outcomes, the game remains within $\mathsf{OWin}(\mathcal{G})$ with probability 1. Let E_n denote the event of "Reaching R_{k-1} from a state in R_k in n steps". It follows $\lim_{n\to\infty} P(E_n) = 1$. By repeating for levels $k = K, \ldots, 1$, we conclude $R_0 = V_F$ is reached with probability 1. □

References

1. B. Bérard, K. Chatterjee, and N. Sznajder. Probabilistic opacity for markov decision processes. *Information Processing Letters*, 115(1):52–59, 2015.
2. N. Bertrand, B. Genest, and H. Gimbert. Qualitative determinacy and decidability of stochastic games with signals. *Journal of the ACM (JACM)*, 64(5):1–48, 2017.
3. J. W. Bryans, M. Koutny, L. Mazaré, and P. Y. Ryan. Opacity generalised to transition systems. *International Journal of Information Security*, 7:421–435, 2008.

4. J. W. Bryans, M. Koutny, and P. Y. Ryan. Modelling opacity using petri nets. *Electronic Notes in Theoretical Computer Science*, 121:101–115, 2005.
5. F. Cassez, J. Dubreil, and H. Marchand. Synthesis of opaque systems with static and dynamic masks. *Formal Methods in System Design*, 40:88–115, 2012.
6. G. De Giacomo and M. Favorito. Compositional approach to translate ltlf/ldlf into deterministic finite automata. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 31, pages 122–130, 2021.
7. G. De Giacomo and M. Y. Vardi. Linear temporal logic and linear dynamic logic on finite traces. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 854–860. ACM, 2013.
8. L. Hélouët, H. Marchand, and L. Ricker. Opacity with powerful attackers. *IFAC-PapersOnLine*, 51(7):464–471, 2018.
9. Y. Ji, X. Yin, and S. Lafortune. Opacity enforcement using nondeterministic publicly known edit functions. *IEEE Transactions on Automatic Control*, 64(10):4369–4376, 2019.
10. C. Keroglou and C. N. Hadjicostis. Probabilistic system opacity in discrete event systems. *Discrete Event Dynamic Systems*, 28:289–314, 2018.
11. F. Lin. Opacity of discrete event systems and its applications. *Automatica*, 47(3):496–503, 2011.
12. B. Maubert, S. Pinchinat, and L. Bozzelli. Opacity issues in games with imperfect information. *arXiv preprint arXiv:1106.1233*, 2011.
13. L. Mazaré. Using unification for opacity properties. *Proceedings of the 4th IFIP WG1*, 7:165–176, 2004.
14. A. Saboori. *Verification and enforcement of state-based notions of opacity in discrete event systems*. University of Illinois at Urbana-Champaign, 2010.
15. A. Saboori and C. N. Hadjicostis. Notions of security and opacity in discrete event systems. In *IEEE Conference on Decision and Control*, pages 5056–5061, 2007.
16. A. Saboori and C. N. Hadjicostis. Verification of initial-state opacity in security applications of des. In *9th International Workshop on Discrete Event Systems*, pages 328–333. IEEE, 2008.
17. A. Saboori and C. N. Hadjicostis. Opacity-enforcing supervisory strategies via state estimator constructions. *IEEE Transactions on Automatic Control*, 57(5):1155–1165, 2012.
18. A. Saboori and C. N. Hadjicostis. Current-state opacity formulations in probabilistic finite automata. *IEEE Transactions on Automatic Control*, 59(1):120–133, 2014.
19. P. Wolper. Constructing Automata from Temporal Logic Formulas: A Tutorial. In G. Goos, J. Hartmanis, J. van Leeuwen, E. Brinksma, H. Hermanns, and J.-P. Katoen, editors, *Lectures on Formal Methods and PerformanceAnalysis*, volume 2090, pages 261–277. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.
20. Y.-C. Wu and S. Lafortune. Comparative analysis of related notions of opacity in centralized and coordinated architectures. *Discrete Event Dynamic Systems*, 23(3):307–339, 2013.
21. Y.-C. Wu and S. Lafortune. Synthesis of insertion functions for enforcement of opacity security properties. *Automatica*, 50(5):1336–1348, 2014.
22. Y. Xie, X. Yin, and S. Li. Opacity enforcing supervisory control using nondeterministic supervisors. *IEEE Transactions on Automatic Control*, 67(12):6567–6582, 2021.
23. Y. Zhou, Z. Chen, and Z. Liu. Verification and enforcement of current-state opacity based on a state space approach. *European Journal of Control*, 71:100795, 2023.