Decoy Allocation Games on Graphs with Temporal Logic Objectives

Abhishek N. Kulkarni[1][0000−0002−1083−8507], Jie Fu[1][0000−0002−4470−2827], Huan Luo[1][0000−0002−1578−9409], Charles A. Kamhoua[2][0000−0003−2169−5975], and Nandi O. Leslie[2][0000−0001−5888−8784]

[1] Worceter Polytechnic Institute, Worcester MA 01609, USA
{ankulkarni,jfu2}@wpi.edu, hluo12@126.com

[2] U.S. Army Research Laboratory, Adelphi, MD 20783, USA
{charles.a.kamhoua.civ,nandi.o.leslie.ctr}@mail.mil

Abstract. We study a class of games, in which the adversary (attacker) is to satisfy a complex mission specified in linear temporal logic, and the defender is to prevent the adversary from achieving its goal. A deceptive defender can allocate decoys, in addition to defense actions, to create disinformation for the attacker. Thus, we focus on the problem of jointly synthesizing a decoy placement strategy and a deceptive defense strategy that maximally exploits the incomplete information the attacker about the decoy locations. We introduce a model of hypergames on graphs with temporal logic objectives to capture such adversarial interactions with asymmetric information. Using the hypergame model, we analyze the effectiveness of a given decoy placement, quantified by the set of deceptive winning states where the defender can prevent the attacker from satisfying the attack objective given its incomplete information about decoy locations. Then, we investigate how to place decoys to maximize the defender's deceptive winning region. Considering the large search space for all possible decoy allocation strategies, we incorporate the idea of compositional synthesis from formal methods and show that the objective function in the class of decoy allocation problem is monotone and non-decreasing. We derive the sufficient conditions under which the objective function for the decoy allocation problem is submodular, or supermodular, respectively. We show a sub-optimal allocation can be efficiently computed by iteratively composing the solutions of hypergames with a subset of decoys and the solution of a hypergame given a single decoy. We use a running example to illustrate the proposed method.

Keywords: Games on Graphs · Hypergames · Deception · Temporal Logic

1 Introduction

In security and defense applications, deception plays a key role to mitigate the information and strategic disadvantages of the defender against adversaries. In this paper, we investigate the design of active defense with deception for a class of games on graphs, also known as ω-regular games [11,7,8]. A game in this class captures the attack-defend sequential interaction in which the attacker is to complete an attack mission specified in temporal logic [19] and the defender is to mitigate attacks by selecting counter-actions and allocating decoys to create a disinformation to the attacker. We are interested in the following question: How to design the decoy allocation strategy so that the defender can influence the attacker into taking (or not taking) certain actions that minimize the set of attacker's winning region? The winning region is defined as the set of game states from which the attacker has a strategy to successfully complete its attack mission irrespective of the defender's counter-strategy.

Games on graphs with temporal logic objectives have been studied extensively in the synthesis of reactive programs [7]. In a reactive program, the system (player 1) is to synthesize a program (a finite-memory strategy) to provably satisfy a desired behavior specification, no matter which actions are taken by the uncontrollable environment (player 2). In these games, players' payoffs are temporal goals and constraints, described using linear temporal logic formulas and a labeling function. A player receives a payoff equal to one if the *labeling* over the outcome (state-sequence) of the game satisfies its temporal logic formula. In our recent work [17], we have shown that a class of decoy-based deception can be captured by assuming that the defender has the true labels of game states but the attacker has incorrect labels. For example, a state labeled "unsafe" by the defender may be mislabeled as "safe" for the attacker. By modeling the interactions between the defender and the attacker as a hypergame, we developed the solutions of subjective rationalizable strategies for both players in this class of hypergames. The defender's subjective rationalizable strategy is by nature deceptive, as it ensures the security temporal logic specification to be satisfied by exploiting the attacker's misperception and mistakes in the attacker's subjective rationalizable strategy. We introduced deceptive winning region as the set of states (or finite game histories) from which the defender can ensure to satisfy a security specification in this hypergame.

However, an important problem remains: How to *control the attacker's misinformation in the labeling function* so as to maximize the deceptive winning region? To restrict the freedom in crafting the disinformation, we formulate a class of *decoy-based deception game*: In this game, the defender can allocate a subset of states as hidden decoys or "traps", unknown to the attacker. During these interactions, the defender is to strategically select actions to lure the attacker into the traps, whereas the attacker plays rationally to satisfy her temporal logic objective given her subjective view of the interaction. In addition, the defender strategy should be *stealthy*, in the sense that the attacker cannot realize a misperception exists before getting caught by one of the traps. To determine the decoy allocation, we employ the aforementioned solutions of hypergames [17] to calculate the defender's deceptive sure-winning region given each individual decoys. The selection of decoy locations is based on compositional synthesis [10,18], which answers, given the two deceptive sure-winning regions for decoys allocated at two different states s and s', what is the deceptive sure-winning region when both states are allocated as decoys simultaneously? We derive the sufficient conditions when the objective function for the decoy allocation problem is submodular, or supermodular, respectively. Based on this, we can construct an under-approximation of the deceptive sure-winning regions incrementally (in polynomial time), instead of having to solve a combinatorially large number of hypergames for all possible decoy configurations.

Related Work Decoy allocation, also called honeypot allocation and camouflage, has been studied in recent years with applications to cyber- and physical security problems. In [22,16], the authors propose a game-theoretic method to place honeypots in a network so as to maximize the probability that the attacker attacks a honeypot and not a real system. In their game formulation, the defender decides where to insert honeypots in a network, and the attacker chooses one server to attack and receives different payoffs when attacking a real system (positive reward) or a honeypot (zero reward). The game is imperfect information as the real systems and honeypots are indistinguishable for the attacker. By the solution of imperfect information games, the defender's honeypot placement strategy is solved to minimize the attacker's rewards.

Security games [24,15] are another class of important models for resource allocation in adversarial environments. In [25], the authors formulate a security game (Stackelberg game) to allocate limited decoy resources in a cybernetwork to mask network configurations from the attacker. This class of deception manipulates the adversary's perception of the payoffs and thus causes the adversary to take (or not to take) certain actions that aid the objective of the defender. In [9], the authors formulate an Markov decision process to assess the effectiveness of a fixed honeypot allocation in an attack graph, which captures multi-stage lateral movement attacks in a cybernetwork and dependencies between vulnerabilities [13,21]. In [2], the authors analyze the honeypot allocation problem for attack graphs using normal-form games, where the defender allocates honeypots that changes the payoffs matrix of players. The optimal allocation strategy is determined using the minimax theorem. The attack graph is closely related to our game on graph model, which generalizes the attack graph to *attack-defend game graphs* [14,3] by incorporating the defender counter-actions in active defense.

There are several key distinctions between our work and the prior work. First, our work focuses on a qualitative approach to decoy allocation instead of a quantitative one, which often requires solving an optimization problem over a well-defined reward/cost function. In the qualitative approach, we represent the attacker's goal using a linear temporal logic formula, which captures rich, qualitative behavioral objectives such as reachability, safety, recurrence, persistence or a combination of these. Second, we show how to incorporate the attacker's misinformation about decoy locations into a ω-regular hypergame model by representing it as labeling misperception. Hypergames [6,23,27] are a class of games with asymmetric (one-sided incomplete) information in which different players might play according to different perceptual games that capture the information and higher-order information known to that player. While the underlying idea behind our game model is similar to "indistinguishable honeypots" discussed in [22], we are able to leverage the solution approaches for hypergames to address decoy allocation problem. Third, we solve for a stealthy strategy for the defender, which ensures that defender's actions will not inform the attacker that deceptive tactics are being used. Lastly, we borrow the idea of compositional reasoning from formal methods to find approximately optimal solutions for the decoy allocation problem for this class of hypergames.

The paper is structured as follows. In Sec. 2, we discuss the preliminaries of attack-defend game on graph model and define the problem statement. In Sec. 3, we present the main results of this paper including an algorithm for the decoy allocation based on the ideas of deceptive synthesis and compositional synthesis. We employ a running example to provide intuition and illustrate the correctness as well as (near-)optimality of the proposed algorithm. Sec. 4 concludes the paper and discusses the future directions.

2 Problem Formulation

2.1 Attack-Defend Games on Graph

In a zero-sum two-player *game on graph*, player 1 (P1, pronoun 'he') plays against player 2 (P2, pronoun 'she') to satisfy a given temporal logic formula. Formally, a game on graph consists of a tuple $\mathcal{G} = \langle G, \varphi \rangle$, where G is a *game arena* modeling the dynamics of the interaction between P1 and P2, and φ is the temporal logic specification of P1. As the game is zero-sum, the temporal logic specification of P2 is $\neg\varphi$, that is, the negation of P1's specification.

Definition 1 (Game Arena). *A two-player turn-based, deterministic game arena between P1 and P2 is a tuple*

$$G = \langle S, Act, T, AP, L \rangle,$$

where

- $S = S_1 \cup S_2$ *is a finite set of states partitioned into two sets* S_1 *and* S_2*. At a state in* S_1*, P1 chooses an action. At a state in* S_2*, P2 selects an action;*
- $Act = Act_1 \cup Act_2$ *is the set of actions.* Act_1 *(resp.,* Act_2*) is the set of actions for P1 (resp., P2);*
- $T : (S_1 \times Act_1) \cup (S_2 \times Act_2) \to S$ *is a* deterministic *transition function that maps a state-action pair to a next state;*
- AP *is a set of atomic propositions;*
- $L : S \to 2^{AP}$ *is the labeling function that maps each state* $s \in S$ *to a set* $L(s) \subseteq AP$ *of atomic propositions that evaluate to true at that state.*

A *run* in G is a (finite/infinite) ordered sequence of states $\rho = (s_0, s_1, \ldots)$ such that for any $i > 0$, $s_i = T(s_{i-1}, a)$ for some $a \in Act$. Given the labeling function L, every run ρ in G can be mapped to a word over an alphabet $\Sigma = 2^{AP}$ as $w = L(\rho) = L(s_0)L(s_1)\ldots$.

In this paper, we use Linear Temporal Logic (LTL) [19] to define the objectives of P1 and P2. Formally, an LTL formula is defined as

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \bigcirc\varphi \mid \varphi \mathsf{U}\, \varphi \mid \varphi \mathsf{W}\, \varphi$$

where $p \in AP$ is an atomic proposition, $\neg$ (negation), $\wedge$ (and), and $\vee$ (or) are Boolean operators, and $\bigcirc$ (next), U (strong until) and W (weak until) are temporal operators. Formula $\bigcirc\varphi$ means that the formula φ will be true in the next state. Formula $\varphi_1 \mathsf{U}\, \varphi_2$ means that φ_2 will be true in some future time step, and before that φ_1 holds true for every time step. Formula $\varphi_1 \mathsf{W}\, \varphi_2$ means that φ_1 holds true until φ_2 is true, but does not require that φ_2 becomes true. We define two additional temporal operators: $\Diamond$ (eventually) and $\Box$ (always) as follows: $\Diamond\, \varphi = \top \mathsf{U}\, \varphi$ and $\Box\, \varphi = \neg \Diamond\, \neg\varphi$.

Given a word $w \in \Sigma^\omega$, let $w[i]$ be the i-th element in the word and $w[i \ldots]$ be the subsequence of w starting from the i-th element. For example, for a word $w = abc$, $w[0] = a$ and $w[1 \ldots] = bc$. We write $w \models \varphi$ if the word w satisfies the temporal logic formula φ. The semantics of LTL are defined as follows.

- $w \models p$ if $p \in w[0]$;
- $w \models \neg\varphi$ if $w \not\models \varphi$;
- $w \models \varphi_1 \wedge \varphi_2$ if $w \models \varphi_1$ and $w \models \varphi_2$;
- $w \models \bigcirc\varphi$ if $w[1 \ldots] \models \varphi$;
- $w \models \varphi \mathsf{U}\, \psi$ if $\exists i \geq 0$, $w[i \ldots] \models \psi$ and $\forall 0 \leq j < i$, $w[j \ldots] \models \varphi$.
- $w \models \varphi \mathsf{W}\, \psi$ if either $w \models \varphi \mathsf{U}\, \psi$ or $\forall 0 \leq j$, $w[j \ldots] \models \varphi$.

A subclass of LTL formula, called syntactically cosafe LTL (scLTL), does not include the weak until operator W and allows the negation operator $\neg$ to only occur before an atomic proposition. An scLTL formula can be equivalently represented by a finite-state deterministic automaton with regular acceptance conditions, defined as follows.

Definition 2 (Specification DFA). *Given an scLTL formula φ, its corresponding specification Deterministic Finite Automaton (DFA) is a tuple*

$$\mathcal{A} = \langle Q, \Sigma, \delta, \iota, Q_F \rangle,$$

which includes a finite set Q of states, a finite set $\Sigma = 2^{AP}$ of symbols, a deterministic transition function $\delta : Q \times \Sigma \to Q$, a unique initial state $\iota \in Q$, and a set $Q_F \subseteq Q$ of final states.

The transition function is recursively extended as $\delta(q, aw) = \delta(\delta(q, a), w)$ for given $a \in \Sigma$ and $w \in \Sigma^*$, where Σ^* is the set of all finite words (also known as the Kleene closure of Σ). A word w is *accepted* by the DFA if and only if $\delta(q, u) \in Q_F$ and u is a prefix of w, i.e., $w = uv$ for $u \in \Sigma^*$ and $v \in \Sigma^\omega$, where Σ^ω is the set of all infinite words defined over Σ. A word is accepted by the specification DFA $\mathcal{A}$ if and only if it satisfies the LTL formula φ.

Putting together the game arena G and the scLTL objective φ of P1, we can formally define a graphical model for the zero-sum game $\mathcal{G}$.

Definition 3 (Product game). *Let $G = \langle S, Act, T, AP, L\rangle$ be a game arena and let $\mathcal{A} = \langle Q, \Sigma, \delta, \iota, Q_F\rangle$ be the specification DFA given the LTL formula φ. Then, the product game $\mathcal{G} = G \otimes \mathcal{A}$ is the tuple,*

$$\mathcal{G} = \langle S \times Q, Act, \Delta, F\rangle,$$

where

- *$S \times Q$ is a set of states partitioned into P1's states $S_1 \times Q$ and P2's states $S_2 \times Q$.*
- *$\Delta : (S_1 \times Q \times Act_1) \cup (S_2 \times Q \times Act_2) \to S \times Q$ is a deterministic transition function that maps a game state $(s, q) \in S \times Q$ and an action $a \in Act$ to a next state $(s', q') \in S \times Q$ such that $s' = T(s, a)$ and $q' = \delta(q, L(s'))$;*
- *$F = S \times Q_F$ is the set of final states in $\mathcal{G}$.*

It is noted that we did not include an initial state in the definition of the game arena. This is because any state in S can be selected to be the initial state. Let $s_0 \in S$ be the initial state of the game arena, the corresponding initial state in the product game is $q_0 = \delta(\iota, L(s_0))$. By construction, for each run $\rho = (s_0, s_1, \ldots)$ in G, there is a unique run $\hat{\rho} = (s_0, q_0), (s_1, q_1), \ldots$ in the product game, where $q_0 = \delta(\iota, L(s_0))$ for $i = 0$ and $q_i = \delta(q_{i-1}, L(s_i))$ for $i \geq 1$. The run ρ satisfies the scLTL formula φ if and only if $L(\rho) \models \varphi$ and as a result of construction, there exists $(s_i, q_i) \in \hat{\rho}$ for some $i \geq 0$ such that $(s_i, q_i) \in F$. Thus, P1's objective of satisfying an scLTL specification over the game arena G is reduced to that of reaching one of the final states F in product game $\mathcal{G}$. In the zero-sum game, P2's objective of satisfying $\neg\varphi$ is reduced to preventing P1 from reaching any final states in F.

A memoryless, randomized *strategy* for i-th player, for $i \in \{1, 2\}$, is a function $\pi_i : S_i \times Q \to \mathcal{D}(Act_i)$, where $\mathcal{D}(Act_i)$ is the set of discrete probability distributions over Act_i. It is noted that a memoryless strategy in a product game is a finite-memory strategy in game arena. A strategy is deterministic if $\pi_i(\rho)$ is a Dirac delta function. We say that player i commits to (or follows) a strategy π_i if and only if for a given state (s, q), if $\pi_i(s, q)$ is defined, then an action is sampled from the distribution $\pi_i(s, q)$, otherwise, player i selects an action at random. Let Π_i be the set of memoryless strategies of player i in the product game.

A strategy $\pi_1 \in \Pi_1$ is said to be sure-winning for P1 if, for every P2's strategy $\pi_2 \in \Pi_2$, P1 can ensure to reach F in finitely many steps. A strategy $\pi_2 \in \Pi_2$ is sure-winning for P2 if for every P1's strategy $\pi_1 \in \Pi_1$, P2 can ensure the game to stay in $(S \times Q) \setminus F$ for infinitely many steps. The product game is known to be determined [11,20]. That is, at any state (s, q), only one of the players has a winning strategy and the winning strategy is memoryless.

The set of states in the product game $\mathcal{G}$ from which P1 (resp. P2) has a sure-winning strategy are called the *sure-winning region* for P1 (resp. P2), denoted as Win_1 (resp. Win_2). Players' sure-winning regions can be computed by using the Alg. 1 by letting $S_i \times Q$ to be V_i, Act_i to be A_i, the transition function Δ and

Algorithm 1: SURE-WIN: Compute Player's Sure-Winning Regions of Zero-Sum Product Games with Reachability Objective [20,11].

```
Input: A reachability game $\langle V = V_1 \cup V_2, A_1 \cup A_2, \Delta, F\rangle$ where $V_i$ are states
       where player $i$ takes an action, $A_i$ are player $i$'s actions,
       $\Delta : V \times A \to V$ and P1's goal is to reach the set $F$ and P2's goal is to
       stay within $V \setminus F$.
Output: The winning regions $\mathsf{Win}_1$ and $\mathsf{Win}_2$ for P1 and P2.
$Z_0 \leftarrow F$, $Z_1 \leftarrow \emptyset$, $k \leftarrow 0$;
while $Z_{k+1} \neq Z_k$ do
    $\mathsf{Pre}_1(Z_k) \leftarrow \{v \in V_1 \mid \exists a \in A_1 \text{ s.t. } \Delta(v,a) \in Z_k\}$;
    $\mathsf{Pre}_2(Z_k) \leftarrow \{v \in V_2 \mid \forall b \in A_2 \text{ s.t. } \Delta(v,b) \in Z_k\}$;
    $Z_{k+1} \leftarrow Z_k \cup \mathsf{Pre}_1(Z_k) \cup \mathsf{Pre}_2(Z_k)$;
    $k \leftarrow k+1$;
end
$\mathsf{Win}_1 \leftarrow Z_k$, $\mathsf{Win}_2 \leftarrow (V_1 \cup V_2) \setminus \mathsf{Win}_1$;
return $\mathsf{Win}_1, \mathsf{Win}_2$.
```

F are the same components in $\mathcal{G}$. The interested readers are referred to Chap 2 of [11] for more details.

The sure-winning strategy is defined for P1 as follows: Let $Z_1, Z_2, \ldots Z_k$ be the sequence of sets generated by Alg. 1, for a state $v \in (Z_i \setminus Z_{i-1}) \cap V_1$, let a be the action that $\Delta(v,a) \in Z_{i-1}$, then $\pi_1(v) = a$ (by construction, such an action a exists). P2's sure-winning strategy is constructed as: For each $v \in \mathsf{Win}_2$, $\pi_2(v) = a$ such that $\Delta(v,a) \in \mathsf{Win}_2$. Clearly, there may exist more than one sure-winning strategies for each player.

2.2 Formulating the Decoy Allocation Problem

We consider an interaction between the defender (P1, pronoun 'he') and the attacker (P2, pronoun 'she') in which the defender can use decoys to introduce incorrect information to the attacker about the game. Our goal is to investigate *how to create the attacker's misinformation by allocating the decoys so as to minimize the size of the sure-winning region of the attacker.*

We now formalize the problem of decoy allocation using the game arena (Def.1). Let `decoy` be an atomic proposition that evaluates to true at a state if the state is equipped with a decoy.

Assumption 1. *In P2's knowledge of the game arena, no state is labeled as decoy, i.e.,* $\texttt{decoy} \notin L(s)$ *for all* $s \in S$.

Assumption 1 captures one important function of decoys—*concealing fictions* [12]. The idea behind *concealing fictions* is that P1 simulates the decoy states to function like a real system. As a result, P1 and P2 play with different subjective views of their interaction. With this in mind, we formalize the notion of *perceptual game arena* of the players to characterize these subjective views.

Perceptual Game Arena. Given that P2 does not know about the decoys, we distinguish between her view of the game arena from P1's view by introducing a different labeling function for P2. Let P1's perceptual game arena be $G^1 = G = \langle S, Act, T, AP, L\rangle$. That is, P1 knows the ground truth. And, let P2's perceptual game arena be $G^2 = \langle S, Act, T, AP, L^2\rangle$ such that for any $s \in S$, we have $L_2(s) = L(s) \setminus \{\texttt{decoy}\}$. In other words, if a state is not a decoy, then P1 and P2 share the same label for that state. If it is a decoy, then P1 knows that the proposition `decoy` evaluates to true at that state, but P2 does not.

The Attacker and Defender Temporal Logic Objectives. Over the perceptual game arenas G and G^2, P1 and P2 aim to satisfy their LTL objectives. We consider that P2's objective is specified by an scLTL formula φ_2, whose specification DFA is $\mathcal{A}_2 = \langle Q, \Sigma, \delta_2, \iota, Q_F\rangle$.

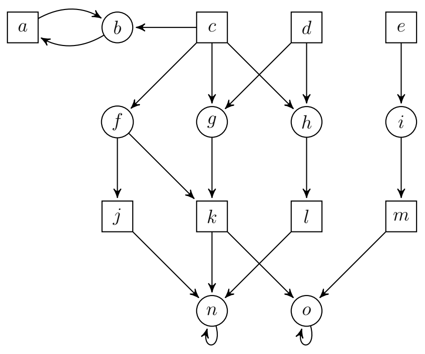
Given P2's perceptual game arena G^2 and the specfication DFA $\mathcal{A}_2$, we can construct a perceptual product game of P2 as $\mathcal{G}_2 = G^2 \otimes \mathcal{A}_2$. P1's objective is an LTL formula $\neg\varphi_2 \mathsf{W}\, \texttt{decoy}$. That is, P1 satisfies the goal by preventing P2 from satisfying φ_2 before reaching a decoy. However, reaching a decoy is not necessary due to the semantics of the "weak until" operator.

Example 2 (Part 1). Consider a game arena as shown in Fig. 1a consisting of 15 states. At a circle state, P1 takes an action, and at a square state, P2 takes an action. As the actions are deterministic, we use edges to indicate players' actions. For example, $(c, f), (c, g), (c, h)$ are possible actions for P2 at the state c. Over this game arena, P2 wants to satisfy an scLTL specification $\varphi_2 = \Diamond\, (n \vee o) \wedge (f \implies \Diamond\, n) \wedge (g \implies \Diamond\, o)$, which, in words, means that P2 must reach either the state n or o with the condition that whenever she visits the state f, she must visit n and whenever she visits g, she must visit o. If she does not visit either f or g, then she can visit either n or o to successfully complete her objective. The DFA equivalent to φ_2 is shown in Fig. 2a.

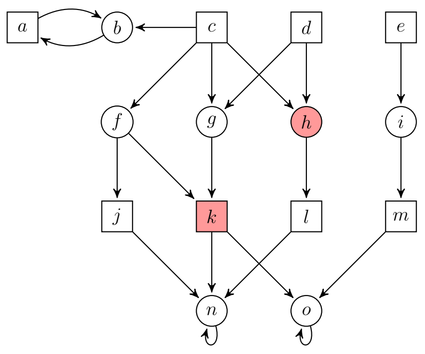
Suppose that P1 allocates the states $D = \{h, k\}$ as decoys. The perceptual game arenas of P1 and P2 under decoy allocation D are now different. P1's perceptual game arena in Fig. 1b has the same underlying graph as the perceptual game arena of P2 shown in Fig. 1a but P1 has the knowledge of where the decoys are placed. We have $\texttt{decoy} \in L(h)$ and $\texttt{decoy} \in L(k)$ but $\texttt{decoy} \notin L(s)$ for any state s except $s = h, k$. Figure 2b shows the perceptual product games of P2. A transition $(c, 0) \to (f, 1)$ is based on the transition $c \to f$ and $\delta_2(0, L(f)) = 1$ in the DFA $\mathcal{A}_2$ (shown in Fig. 2a). We omit all nodes that do not have a path leading to $(n, 3)$ or $(o, 3)$.

We now formalize our problem statement.

Problem 1. Given a set of k decoys and a set $\mathcal{D} \subseteq S$ of states at which decoys can be placed, identify the decoy locations $D \subseteq \mathcal{D}$ with $|D| \leq k$ such that by letting $\texttt{decoy} \in L(s)$ for each $s \in D$, the number of states in the product game $\mathcal{G}_1$ from which P1 has a strategy to satisfy the security specification φ_1 is maximized, given that P2 may choose any counter-strategy that she considers rational in her perceptual game, $\mathcal{G}_2$.



(a) Perceptual Game Arena of P2



(b) Perceptual Game Arena of P2 (Ground-Truth)

Fig. 1: Perceptual Game Arenas of P1 and P2 in Ex. 2.

The objective of P1 is intuitively understood as to maximize the set of system states protected by the defense strategy.

3 Main Result

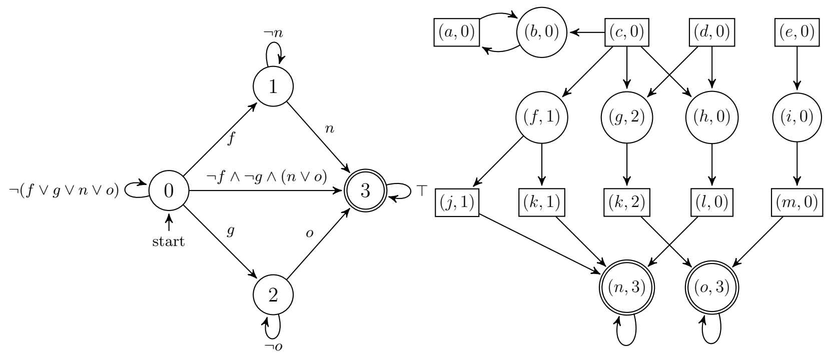
Our proposed solution to Problem 1 is based upon two key ideas from formal methods and hypergame theory, namely (a) deceptive synthesis, and (b) compositional synthesis. In Sec. 3.1, we introduce deceptive synthesis to construct a strategy for P1 to deceive P2 into reaching a pre-defined decoy set in finitely many steps by exploiting the incomplete information of P2. The strategy is called *deceptive sure-winning strategy* and depends on the chosen set of decoys. Then, in Sec. 3.2, we introduce a compositional synthesis approach to identify an approximately optimal allocation of decoys.

3.1 Deceptive Synthesis: Hypergames on Graphs

Consider a set $D \subseteq S$ of states are allocated with decoys, unknown to P2. In such an interaction, as seen in Sec. 2.2, the players have different perceptual game arenas that share the same set of states, actions, and transitions but different labeling functions. We introduce a model of hypergame on graph to integrate the games $\mathcal{G}_1$ of P1 and $\mathcal{G}_2$ of P2 into a single graphical model.

Definition 4 (Hypergame on Graph (modified from [17][3])). *Given the perceptual game arenas $G = \langle S, Act, T, AP, L\rangle$ and $G^2 = \langle S, Act, T, AP, L^2\rangle$,*

[3] Def. 4 is a simplified version of [17, Def. 6], which considers the general case when P1 and P2's objectives are both general scLTL formulas, not necessarily in the current form.



(a) Specification DFA for φ_2 (b) Perceptual Product Game of P2

Fig. 2: P2's specification DFA and the perceptual product game in Ex. 2.

and P2's specification DFA $\mathcal{A}_2 = \langle Q, 2^{AP}, \delta_2, \iota, Q_F \rangle$*, let* $D \subsetneq S$ *be a set of states such that* $\texttt{decoy} \in L(s)$*. The* hypergame on graph *given the players' objectives* $\neg\varphi_2 \mathsf{W}\, \texttt{decoy}$ *for P1 and* φ_2 *for P2 is a transition system*

$$\mathcal{H}_D = \langle S \times Q, Act, \Delta, F_D, F_2 \rangle,$$

where

- $S \times Q$ *is the set of states;*
- $\Delta : (S_1 \times Q \times Act_1) \cup (S_2 \times Q \times Act_2) \to S \times Q$ *is a* deterministic *transition function such that* $\Delta((s,q),a) = (s',q')$ *if and only if* $s' = T(s,a)$ *and* $q' = \delta_2(q, L_2(s'))$*;*
- $F_D = \{(s,q) \mid \texttt{decoy} \in L(s)\}$ *is the set of states which P1 must reach in order to satisfy* $\neg\varphi_2 \mathsf{W}\, \texttt{decoy}$*;*
- $F_2 = \{(s,q) \mid q \in Q_F\}$ *is the set of final states which P2 must reach in order to satisfy* φ_2*.*

It is noted that the sets of states, actions, transitions, and P2's final states F_2 in $\mathcal{H}_D$ are defined exactly as these components in P2's perceptual product game $\mathcal{G}_2$ (see Def. 3). The additional set F_D is introduced to represent P1's objective.

Let us denote the sure-winning region of player i in player j's perceptual game $\mathcal{G}_j$ by Win_i^j. The attacker's perceptual winning regions can be solved with the attacker's reachability game using Alg. 1 by letting $V_1 \coloneqq S_2 \times Q$, $V_2 \coloneqq S_1 \times Q$, $A_1 \coloneqq Act_2$, $A_2 \coloneqq Act_1$, Δ is the same as in $\mathcal{H}_D$, and $F \coloneqq F_2$. The following observations are noted:

- For every state $(s,q) \in \mathsf{Win}_1^2$ (P1's sure-winning region perceived by P2), P1 can ensure to satisfy $\neg\varphi_2$ no matter which strategy P2 uses. Decoys are not needed for states within Win_1^2.

– For every state $(s, q) \in \mathsf{Win}_2^2$ (P2's sure-winning region perceived by P2), P2 can ensure satisfying φ_2 when no decoy is used. However, when decoys are introduced, P1 can exploit P2's lack of knowledge about the decoys and lure P2 into reaching decoys before P2 is able to satisfy φ_2.

It is known [5] that we can rewrite $\neg\varphi_2 \mathsf{W}\, \texttt{decoy}$ using the temporal operators: U (until) and $\Box$ (always), as $(\neg\varphi_2 \mathsf{U}\, \texttt{decoy}) \vee \Box\neg\varphi_2$, where $\Box\varphi = \neg\Diamond\neg\varphi$. When the game state is within P2's perceptual winning region Win_2^2, then P1 does not have a strategy to ensure $\Box\neg\varphi_2$ (reads "always φ_2 is false") and can only satisfy his specification by enforcing P2 to visit a decoy. The following Lemma formalizes this statement.

Lemma 1. *For any state $(s, q) \in \mathsf{Win}_2^2$, any strategy π_1 of P1 that satisfies $\neg\varphi_2 \mathsf{W}\, \texttt{decoy}$ also satisfies $\neg\varphi_2 \mathsf{U}\, \texttt{decoy}$.*

We omit the proof noting that it follows from the definition of weak until and the property of winning region.

Thus, when we focus our attention on the region Win_2^2, P1's objective is equivalently $\neg\varphi_2 \mathsf{U}\, \texttt{decoy}$. Before addressing the decoy allocation problem, we must answer: From which states in Win_2^2, P1 can ensure to satisfy $\neg\varphi_2 \mathsf{U}\, \texttt{decoy}$ by exploiting P2's lack of knowledge about the decoy states, i.e., F_D?

To answer this question, we formulate a deceptive game for P1. We first restrict P1's actions to those considered rational for P2 in her perceptual game. At the same time, P2's irrational actions are removed as P1 knows a rational P2 will not use these actions. As the rational actions are based on P2's subjective view of the game, we formalize this notion of rationality using the concept of *subjective rationalizability* from game theory (we refer the interested readers to [17] for rigorous treatment).

Definition 5 (Subjectively Rationalizable Actions in $\mathcal{G}_2$). *Given P2's perceptual product game $\mathcal{G}_2 = \langle S \times Q, Act, \Delta, F_2 \rangle$, a player i's action $a \in Act_i$ is said to be* subjectively rationalizable *at his/her winning state $(s, q) \in \mathsf{Win}_i^2$ in $\mathcal{G}_2$ if and only if $\Delta((s, q), a) \in \mathsf{Win}_i^2$. At player i's losing state $(s, q) \notin \mathsf{Win}_i^2$, any action of player i is assumed to be subjectively rationalizable for player i.*

Based on Def. 5, we define the set of subjectively rationalizable actions of player i at a state $(s, q) \in S \times Q$ as follows:

$$\begin{aligned}\mathsf{SRActs}_i^2(s, q) = &\{a \in Act_i \mid (s, q) \in \mathsf{Win}_i^2 \text{ and } \Delta((s, q), a) \in \mathsf{Win}_i^2\} \cup \\ &\{a \in Act_i \mid (s, q) \notin \mathsf{Win}_i^2 \text{ and } \Delta((s, q), a) \text{ is defined}\} \qquad (1)\end{aligned}$$

Assumption 2. *Subjective rationalizability is a common knowledge between P1 and P2.*

Assumption 2 means that both players know that their opponent is subjectively rational and that the opponent is aware of this fact. Thus, P2 would become aware of her misperception in the game arena, when P1 uses an action which is not subjectively rationalizable in P2's perceptual game, $\mathcal{G}_2$. We can refine the hypergame on graph $\mathcal{H}_D$ to eliminate: 1) states that do not require decoys: This is the set Win_1^2 from which P1 has a sure-winning strategy for $\neg\varphi_2$; 2) actions that contradict P2's perception. After this elimination, we obtain a deceptive reachability game for P1, for synthesizing P1's deceptive strategy.

Definition 6 (P1's deceptive reachability game). *Given the hypergame on graph $\mathcal{H}_D = \langle S \times Q, Act, \Delta, F_D, F_2 \rangle$, P1's deceptive reachability game is*

$$\widehat{\mathcal{H}}_D = \langle \mathsf{Win}_2^2, Act, \widehat{\Delta}, F_D \rangle,$$

where

- Win_2^2 *is a set of P2's perceptual winning states, and game state space for P1's deceptive reachability game.*
- $\widehat{\Delta} : S \times Q \times Act \to S \times Q$ *is a* deterministic *transition function such that*
  - *if $(s, q) \notin F_2$ then $\widehat{\Delta}((s, q), a) = \Delta((s, q), a)$ whenever $s \in S_i$ and $a \in \mathsf{SRActs}_i^2(s, q)$ for $i = 1, 2$. Otherwise, $\widehat{\Delta}((s, q), a)$ is undefined.*
  - *if $(s, q) \in F_2$, then for any action $a \in Act$, $\widehat{\Delta}((s, q), a) = (s, q)$. That is, the set F_2 are modified into sink states.*
- F_D *is the set of states that P1 aims to reach.*

Lemma 2. *For a given state (s, q), if P1 has a sure-winning strategy in $\widehat{\mathcal{H}}_D$ starting from (s, q), then P1 can ensure to satisfy $\neg\varphi_2 \mathsf{U}\, \mathtt{decoy}$ by following this sure-winning strategy in $\widehat{\mathcal{H}}_D$.*

Proof. A path satisfies $\neg\varphi_2 \mathsf{U}\, \mathtt{decoy}$ if it reaches F_D and before reaching F_D, it does not visit any state in F_2. By construction of $\widehat{\mathcal{H}}_D$, if any path reaches F_D, it must not have visited F_2 because if F_2 is reached prior to F_D, then the game stays in the sink state and will never reach F_D. Thus, P1's sure-winning strategy that ensures a path to reach F_D alone satisfies $\neg\varphi_2 \mathsf{U}\, \mathtt{decoy}$. □

Formally, P1's sure-winning strategy π_1 in the deceptive reachability game is said to be *deceptively sure winning*. A state from which P1 has a deceptive sure-winning strategy is called a *deceptively sure-winning state*. The set of all deceptively sure-winning states of P1 in $\widehat{\mathcal{H}}_D$ is called P1's *deceptive sure-winning region*. The deceptive sure-winning region for P1 can be computed by using Alg. 1 with $\widehat{\mathcal{H}}_D$ by letting $V_1 \coloneqq (S_1 \times Q) \cap \mathsf{Win}_2^2$, $V_2 \coloneqq (S_2 \times Q) \cap \mathsf{Win}_2^2$, $\Delta \coloneqq \widehat{\Delta}$, and $F \coloneqq F_D$ (see the description of terms in Alg. 1). We denote the deceptive sure-winning region for P1 as DSWin_D.

It is noted that the deception is induced by the set F_D which is hidden from P2, and the fact that during the interaction, P1 does not choose any action that contradicts P2's misperception. Additionally, we note that deceptive sure-winning region is not defined for P2, as she is unaware of her lack of information until a decoy is reached.

We now continue with the running example to illustrate the hypergame and P1's deceptive reachability game.

Example 2 (Part 3). From Def. 4, we note that the hypergame on graph $\mathcal{H}_D$ shares the same underlying graph as P2's perceptual game, $\mathcal{G}_2$. That is, in our example, $\mathcal{H}_D$ would have the same graph as Fig. 2b but has the states $(h, 0), (k, 1)$ and $(k, 2)$ labeled as the sink states (shown in red). Now, let us understand the construction of $\widehat{\mathcal{H}}_D$ from $\mathcal{H}_D$. We start by computing Win_2^2 using Alg. 1 over the model $\mathcal{G}_2$ by letting $V_1 \coloneqq S_2 \times Q, V_2 \coloneqq S_1 \times Q, \Delta \coloneqq \Delta$ and $F \coloneqq F_2$. This results in Win_2^2 to include all states except $(a, 0), (b, 0)$. Intuitively, at the state $(b, 0)$, P1 can always choose the transition $b \to a$ to reach $(a, 0)$ and keep the game state within $\{(a, 0), (b, 0)\}$. Consequently, any action that leads to $(a, 0), (b, 0)$ is **not** subjectively rationalizable for P2 and thereby removed. Additionally, the states $(a, 0)$ and $(b, 0)$ are also removed from $\mathcal{H}_D$ to get $\widehat{\mathcal{H}}_D$, which is shown in Fig. 3.

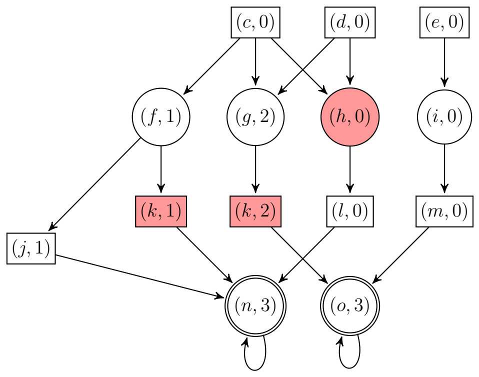


Fig. 3: P1's deceptive reachability game.

3.2 Compositional Synthesis for Decoy Allocation

Given a subset $\mathcal{D} \subseteq S$ of states that can be allocated as decoys, for every different choice of decoy allocation $D \subseteq \mathcal{D}$ we have a different hypergame, $\widehat{\mathcal{H}}_D$. In this context, solving Problem 1 is equivalent to identifying one hypergame that has the largest deceptive sure-winning region $|\mathsf{DSWin}_D|$ for P1. A naïve approach to solve this problem would be to compute DSWin_D for each $D \subseteq \mathcal{D}$ and then select a set D for which $|\mathsf{DSWin}_D|$ is the largest. However, this approach is not scalable because the number of subsets increases combinatorially with the size of game. To address this issue, we introduce a compositional approach to decoy allocation in which we show that when certain conditions hold, the decoy allocation problem can be formulated as a sub or supermodular optimization problem. We propose an algorithm to approximate the optimal decoy allocation.

Proposition 1. *Let* $\mathsf{DSWin}_{\{s_1\}}$ *and* $\mathsf{DSWin}_{\{s_2\}}$ *be P1's deceptive sure-winning regions in the hypergames* $\widehat{\mathcal{H}}_{\{s_1\}}$ *and* $\widehat{\mathcal{H}}_{\{s_2\}}$ *respectively. Then, P1's deceptive* *sure-winning region* $\mathsf{DSWin}_{\{s_1,s_2\}}$ *in the reachability game* $\widehat{\mathcal{H}}_{\{s_1,s_2\}}$ *is equal to the sure-winning region for P1 in the following zero-sum, reachability game:*

$$\widehat{\mathcal{H}}_{\{s_1,s_2\}} = \langle \mathsf{Win}_2^2, Act, \widehat{\Delta}, \mathsf{DSWin}_{\{s_1\}} \cup \mathsf{DSWin}_{\{s_2\}} \rangle,$$

where P1's goal is to reach the target set $\mathsf{DSWin}_{\{s_1\}} \cup \mathsf{DSWin}_{\{s_2\}}$ *and P2's goal is to prevent P1 from reaching the target set.*

Proof. First, it is noted that all the three deceptive reachability games: $\widehat{\mathcal{H}}_{\{s_1\}}$, $\widehat{\mathcal{H}}_{\{s_2\}}$ and $\widehat{\mathcal{H}}_{\{s_1,s_2\}}$, share the same underlying graphs but different reachability objectives for P1: $F_{\{s_1\}}, F_{\{s_2\}}$, and $F_{\{s_1,s_2\}}$. In addition, $F_{\{s_1\}} \cup F_{\{s_2\}} = F_{\{s_1,s_2\}}$. By definition of sure-winning regions, from every state $(s,q) \in \mathsf{DSWin}_{\{s_i\}}$ for $i = 1,2$, there exists a deceptive sure-winning strategy $\pi^*_{\{s_i\}}$ for P1 to ensure $F_{\{s_i\}}$ is reached in finitely many steps, for any subjectively rationalizable counter-strategy of P2.

In $\widehat{\mathcal{H}}_{\{s_1,s_2\}}$, let $W^* \subseteq \mathsf{Win}_2^2$ be the sure-winning region for P1 and π^* be the sure-winning strategy of P1. From a state (s,q) in W^*, P1 can ensure to reach a state, say $(s',q') \in \mathsf{DSWin}_{\{s_1\}} \cup \mathsf{DSWin}_{\{s_2\}}$ by following π^*. Upon reaching a state (s',q'), P1 can ensure to reach a state in either $F_{\{s_1\}}$ or $F_{\{s_2\}}$—that is, P1 can ensure to reach a state in $F_{\{s_1,s_2\}}$. Hence, a sure-winning state (s,q) in the above reachability game is deceptive sure-winning in $\widehat{\mathcal{H}}_{\{s_1,s_2\}}$ in which $F_{\{s_1,s_2\}}$ is P1's reachability objective. The deceptive sure-winning strategy is sequentially composed of strategies π^*, $\pi^*_{\{s_1\}}$, and $\pi^*_{\{s_1\}}$ as follows: From a state $(s,q) \in W^*$, P1 uses π^* until a state in $\mathsf{DSWin}_{\{s_1\}} \cup \mathsf{DSWin}_{\{s_2\}}$ is reached. If $\mathsf{DSWin}_{\{s_1\}} \setminus \mathsf{DSWin}_{\{s_2\}}$ is reached, P1 uses the sure-winning strategy $\pi^*_{\{s_1\}}$; If $\mathsf{DSWin}_{\{s_2\}} \setminus \mathsf{DSWin}_{\{s_1\}}$ is reached, P1 uses the sure-winning stratgy $\pi^*_{\{s_2\}}$; if $\mathsf{DSWin}_{\{s_1\}} \cap \mathsf{DSWin}_{\{s_2\}}$, P1 selects one of $\pi^*_{\{s_1\}}$ and $\pi^*_{\{s_2\}}$ arbitrarily. □

Prop. 1 provides us a way for composing the deceptive sure-winning regions of two deceptive reachability games $\widehat{\mathcal{H}}_{s_1}$ and $\widehat{\mathcal{H}}_{s_2}$ to compute the deceptive sure-winning region in the deceptive reachability game $\widehat{\mathcal{H}}_{\{s_1,s_2\}}$ where both s_1 and s_2 are allocated as decoys. A more general result can be obtained by applying Prop. 1 repeatedly.

Corollary 1. *Given* DSWin_D *and* $\mathsf{DSWin}_{\{s\}}$ *as P1's deceptive sure-winning regions in hypergames* $\widehat{\mathcal{H}}_D$ *and* $\widehat{\mathcal{H}}_{\{s\}}$ *respectively, P1's deceptive sure-winning region* $\mathsf{DSWin}_{D\cup\{s\}}$ *in the deceptive reachability game* $\widehat{\mathcal{H}}_{D\cup\{s\}}$ *equals the sure-winning region for P1 in the following zero-sum, reachability game:*

$$\langle \mathsf{Win}_2^2, Act, \widehat{\Delta}, \mathsf{DSWin}_D \cup \mathsf{DSWin}_{\{s\}} \rangle$$

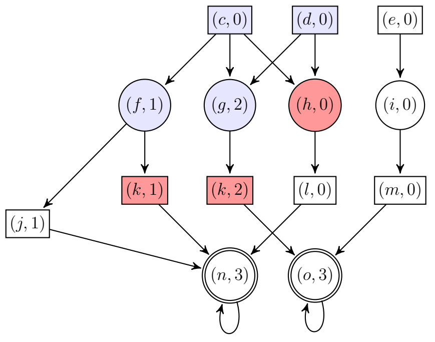
where P1's goal is to reach the target set $\mathsf{DSWin}_D \cup \mathsf{DSWin}_{\{s\}}$ *and P2's goal is to prevent P1 from reaching the target set.*

Corollary 2. *Given a set* $D \subseteq \mathcal{D}$ *and a state* $s \in \mathcal{D}$*, we have*

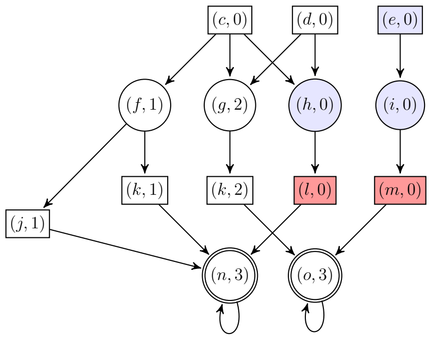
$$\mathsf{DSWin}_D \cup \mathsf{DSWin}_{\{s\}} \subseteq \mathsf{DSWin}_{D\cup\{s\}}$$

Corollary 2 follows immediately from Proposition 1 and Alg. 1. To see this, consider a P1 state $s \in \mathcal{D}$ which is neither in DSWin_D nor in $\mathsf{DSWin}_{\{s\}}$ but has exactly two transitions: one leading to s and another leading to a state in DSWin_D. Clearly, the new state will be added to $\mathsf{DSWin}_{D \cup \{s\}}$. Thus, if we consider the size of DSWin_D to be a measure of effectiveness of allocating the states in $D \subseteq \mathcal{D}$ as decoys, then Corollary 2 states that the effectiveness of adding a new state to a set of decoys is greater than or equal to the sum of their individual effectiveness.

Example 2 (Part 4). Given the underlying graph of P1's reachability game $\widehat{\mathcal{H}}_D$ from Fig. 3, let us observe the effect of choosing different D on P1's deceptive sure-winning region, DSWin_D. Letting $k = 2$, Fig. 4 shows the DSWin_D for $D = \{h, k\}$ (Fig. 4a) and $D = \{l, m\}$ (Fig. 4b). In the figure, the colored states represent P1's deceptive sure-winning region, DSWin_D. The states in F_D are colored red and the states from which P1 has deceptive sure-winning strategy to reach a state in F_D are colored blue. For instance, for $D = \{h, k\}$, a P1 state $(f, 1)$ is included in $F_{\{h,k\}}$ because there exists an action for P1 that leads to $(k, 1)$, which is in $F_{\{h,k\}}$. Similarly, a P2 state $(d, 0)$ is included in $\mathsf{DSWin}_{\{h,k\}}$ because both the outgoing transitions from $(d, 0)$ lead to a deceptively sure-winning state. We also notice that the states $(c, 0)$ and $(d, 0)$ from $\mathsf{DSWin}_{\{h,k\}}$ are *not* included in either $\mathsf{DSWin}_{\{h\}} = \{(h, 0)\}$ or $\mathsf{DSWin}_{\{k\}} = \{(k, 1), (k, 2), (f, 1), (g, 2)\}$ because both the states have at least one transition that does not lead to deceptive sure-winning state. For instance, the transition $(d, 0) \to (g, 2)$ prevents the state $(d, 0)$ to be added to DSWin_h.



(a) Deceptive sure-winning region of P1 when $D = \{h, k\}$



(b) Deceptive sure-winning region of P1 when $D = \{l, m\}$

Fig. 4: Deceptive sure-winning region of P1 under different choice of D.

We now define a composition operator $\uplus$ over deceptive sure-winning regions which represent the true effect of adding a new state to a given set of decoys.

That is, given $D \subseteq \mathcal{D}$ and $s \in \mathcal{D}$, let $\biguplus$ be an operator such that

$$\mathsf{DSWin}_D \biguplus \mathsf{DSWin}_{\{s\}} = \mathsf{DSWin}_{D \cup \{s\}}.$$

That is, the composition operator returns the deceptive sure-winning region in the reachability game $\langle \mathsf{Win}_2^2, Act, \widehat{\Delta}, \mathsf{DSWin}_D \cup \mathsf{DSWin}_{\{s\}} \rangle$, which equals P1's deceptive sure-winning region when the set $D \cup \{s\}$ are selected to be decoys.

With this notation, Problem 1 becomes equivalent to identifying a set $D^* \subseteq \mathcal{D}$ such that

$$D^* = \arg\max_{D \subseteq \mathcal{D}} \left| \biguplus_{s \in D} \mathsf{DSWin}_{\{s\}} \right| \quad \text{subject to: } |D| \leq k. \tag{2}$$

It is noted that if we replace the composition operator $\biguplus$ with the union operator $\cup$ in (2), then the problem becomes

$$\max_{D \subseteq \mathcal{D}} \left| \bigcup_{s \in D} \mathsf{DSWin}_{\{s\}} \right| \quad \text{subject to: } |D| \leq k. \tag{3}$$

which is a maximum set-cover problem. The maximum set-cover problem is well-known submodular optimization problem and can be solved using a greedy algorithm: Given the current choice D_i of decoys at iteration i, the greedy algorithm selects a new decoy $s \in \mathcal{D} \setminus D_i$ that covers the greatest number of uncovered states in Win_2^2. This selection iterates until k decoys are selected. It is also known that the greedy algorithm is $(1 - 1/e)$-approximate. The reader is referred to [26] for more details.

Let $f^{\cup}(D) = \left| \bigcup_{s \in D} \mathsf{DSWin}_{\{s\}} \right|$ and $f^{\uplus}(D) = \left| \biguplus_{s \in D} \mathsf{DSWin}_{\{s\}} \right|$. It follows from Corollary 2 that $f^{\cup}(D) \leq f^{\uplus}(D)$ for all $D \subseteq \mathcal{D}$. In other words, $f^{\cup}(D)$ under-approximates the effectiveness of allocating the states in D as decoys, which is captured by $f^{\uplus}(D)$.

While the function $f^{\cup}$ is submodular, a similar sub/supermodularity condition does not necessarily hold for the function $f^{\uplus}$. In the sequel, we provide sufficient conditions on when $f^{\uplus}$ is submodular and when it is supermodular.

Theorem 1. *The following statements about* $f^{\uplus}(D) = \left| \biguplus_{s \in D} \mathsf{DSWin}_{\{s\}} \right|$ *are true.*

(a) $f^{\uplus}$ *is monotone and non-decreasing.*
(b) $f^{\uplus}$ *is submodular if* $\mathsf{DSWin}_{D \cup \{s\}} = \mathsf{DSWin}_D \cup \mathsf{DSWin}_{\{s\}}$ *for all* $D \subseteq \mathcal{D}$ *and* $s \in \mathcal{D}$.
(c) $f^{\uplus}$ *is supermodular if* $\mathsf{DSWin}_D = \mathsf{DSWin}_{D \cup \{s_1\}} \cap \mathsf{DSWin}_{D \cup \{s_2\}}$ *for all* $D \subseteq \mathcal{D}$ *and all* $s_1, s_2, \in \mathcal{D}$.

Proof. (**a**). Based on Corollary 2, for any set $D \subseteq \mathcal{D}$ and a state $s \in \mathcal{D} \setminus D$, $f^{\uplus}(D) = |\mathsf{DSWin}_D|$ and $f^{\uplus}(D \cup \{s\}) = |\mathsf{DSWin}_{D\cup\{s\}}|$, because $\mathsf{DSWin}_D \subseteq \mathsf{DSWin}_{D\cup\{s\}}$, $f^{\uplus}(D) \leq f^{\uplus}(D \cup \{s\})$.

(**b**). When $\mathsf{DSWin}_{D\cup\{s\}} = \mathsf{DSWin}_D \cup \mathsf{DSWin}_{\{s\}}$, we can write $f^{\uplus}(D) = \left|\biguplus_{s\in D} \mathsf{DSWin}_{\{s\}}\right| = \left|\bigcup_{s\in D} \mathsf{DSWin}_{\{s\}}\right| = f^{\cup}(D)$, which is submodular.

(**c**). We will show that

$$LHS \coloneqq f^{\uplus}(D \cup \{s_1\}) + f^{\uplus}(D \cup \{s_2\}) - f^{\uplus}(D) \leq f^{\uplus}(D \cup \{s_1, s_2\}) \coloneqq RHS$$

for all $D \subseteq \mathcal{D}$ and all $s_1, s_2 \in \mathcal{D}$. Given that $\mathsf{DSWin}_D = \mathsf{DSWin}_{D\cup\{s_1\}} \cap \mathsf{DSWin}_{D\cup\{s_2\}}$ holds for any $D \subseteq \mathcal{D}$ and any $s_1, s_2 \in \mathcal{D}$, we have that $f^{\uplus}(D \cup \{s_1\}) + f^{\uplus}(D \cup \{s_2\}) - f^{\uplus}(D)$ counts every state in $\mathsf{DSWin}_{D\cup\{s_1\}} \cup \mathsf{DSWin}_{D\cup\{s_2\}}$ exactly once. On the other hand, we have $f^{\uplus}(D \cup \{s_1, s_2\}) = |\mathsf{DSWin}_{D\cup\{s_1,s_2\}}|$ and $\mathsf{DSWin}_{D\cup\{s_1,s_2\}} \supseteq \mathsf{DSWin}_{D\cup\{s_1\}} \cup \mathsf{DSWin}_{D\cup\{s_2\}}$, by Corollary 2. Thus, there may exist a state in $\mathsf{DSWin}_{D\cup\{s_1,s_2\}}$ which is not included in either $\mathsf{DSWin}_{D\cup\{s_1\}}$ or $\mathsf{DSWin}_{D\cup\{s_2\}}$. In other words, RHS may be greater than or equal to LHS and the statement follows. □

Based on Thm. 1, we now propose a greedy algorithm described in Alg. 2. This greedy algorithm is an extension of the GreedyMax algorithm for maximizing monotone submodular-supermodular functions in [4]. It starts with an empty set of states labeled with `decoy` and incrementally adds new decoys in the the game arena. At each step, given the deceptive winning region of the chosen decoys, a new decoy is selected such that by adding the new decoy into the chosen decoys, P1's deceptive sure-winning region covers the largest number of states in Win_2^2. The algorithm iterates until k decoys are added, where k is the upper bound on the number of decoys.

Algorithm 2: GreedyMax Algorithm for Decoy Allocation

```
Input: P1's deceptive reachability game $\langle \mathsf{Win}_2^2, Act, \widehat{\Delta}, F_D = \emptyset \rangle$, the set
       $\mathcal{D} \subseteq S$, the bound $k$ on the number of decoys.
Output: An approximate solution $\overline{D}$ for the optimization problem in Eq. 2.
$\overline{D} \leftarrow \emptyset$;
$\mathsf{DSWin}_{\overline{D}} \leftarrow \emptyset$;
while $|\overline{D}| < k$ do
    for $s \in \mathcal{D} \setminus \overline{D}$ do
        $\mathcal{G}_s \leftarrow \langle \mathsf{Win}_2^2, Act, \widehat{\Delta}, \mathsf{DSWin}_{\{s\}} \cup \mathsf{DSWin}_{\overline{D}} \rangle$;
        $\mathsf{DSWin}_{\{s\}\cup\overline{D}} \leftarrow$ Sure-Win($\mathcal{G}_s$);          ... by Alg.1;
    end
    $s^* \leftarrow \arg\max_{s\in\mathcal{D}\setminus\overline{D}} |\mathsf{DSWin}_{\{s\}\cup\overline{D}}|$;
    $\overline{D} \leftarrow s^* \cup \overline{D}$;
end
return $\overline{D}$
```

Complexity Analysis The complexity of Alg. 2 is $\mathcal{O}(k|\mathcal{D}|N)$ where N is the number of state-action pairs in P1's deceptive reachability game. This is because to add $(i+1)$-th state to $\overline{D}$, we update deceptive sure-winning regions of $|\mathcal{D}|-i$ states. The complexity of solving a reachability game is linear in the size N of the game, measured by the number of state-action pairs.

Example 2 (Part 5). We maximize $|\mathsf{DSWin}_D|$, under the constraint that a maximal two decoys to be placed within the set $\mathcal{D} = \{j, k, l, m\}$. Following the compositional approach, we compute the following deceptive sure-winning regions: $\mathsf{DSWin}_{\{j\}} = \{(j, 1), (f, 1)\}$, $\mathsf{DSWin}_{\{k\}} = \{(k, 1), (k, 2), (f, 1), (g, 2)\}$, $\mathsf{DSWin}_{\{l\}} = \{(l, 0), (h, 0)\}$ and $\mathsf{DSWin}_{\{m\}} = \{(m, 0), (i, 0), (e, 0)\}$.

First, we use the greedy algorithm for maximum set-cover to solve for $D \subseteq \mathcal{D}$ that maximizes $f^{\cup}(D)$ under the constraint $|D| \leq 2$. In the first iteration, the greedy algorithm selects the largest the state corresponding to $|\mathsf{DSWin}_{\{s\}}|$, which is $s = k$. In the second iteration, it selects the set that has the largest number of states not already included in $\mathsf{DSWin}_{\{k\}}$. Thus, it selects m as the second state to place the decoy. In conclusion, it selects $D = \{k, m\}$ as solution to decoy allocation problem, for which $|\mathsf{DSWin}_{\{k,m\}}| = 7$.

Second, we use Alg. 2 to solve for $D \subseteq \mathcal{D}$ that maximizes $f^{\uplus}(D)$ under the constraint $|D| \leq 2$. In the first iteration, s^* is selected to be k because $|\mathsf{DSWin}_{\{k\}}|$ is the largest. In the second iteration, s^* is selected to be l because $\mathsf{DSWin}_{\{l\}\cup\overline{D}} = \{(l, 0), (h, 0), (k, 1), (k, 2), (f, 1), (g, 2), (c, 0), (d, 0)\}$. In conclusion, it selects $D = \{k, l\}$ as solution to decoy allocation problem, for which $|\mathsf{DSWin}_{\{k,l\}}| = 8$, which coincidentally in this example is also the globally optimal solution for the problem. We note the improvement in the solution is attributed to incremental computation of $\mathsf{DSWin}_{D\cup\{s\}}$ in Alg. 2.

Due to space limitation, we omit other examples with larger game arena. But the interested readers can find more examples in which the decoy allocation problems are solved with both the greedy algorithm for submodular optimization and Alg. 2 in https://github.com/abhibp1993/decoy-allocation-problem.

4 Conclusion

In this paper, we investigated the optimal decoy allocation problems in a class of games where players' objectives are specified in temporal logic and players have asymmetric information. The contributions of the paper are twofold: First, we develop a hypergame on graph model to capture the deceivee (the adversary)'s incomplete and incorrect information due to the decoys and the deceiver (the defender)'s information about the deceivee's information. Using decoy-based deception, we designed algorithms to compute a deceptive sure-winning strategy with which the defender can take actions deceptively and lure the adversary into decoys, from a state where the adversary perceives herself a winner (i.e., has a strategy to achieve the attack objective). Second, to compute the optimal choice of decoy locations, we employed compositional synthesis from formal methods and proved that the optimal decoy allocation problem is monotone, and non-decreasing. However, the problem can be submodular or supermodular or neither in different games. We design two greedy algorithms, one is based on maximizing an under-approximation of the deceptive winning regions given the effectiveness of individual decoys using maximum set cover, another is to use submodular-supermodular optimization to find approximate solutions of the optimal decoy placement.

Future work include the study of decoy allocation with other types of decoy-induced misperception. In this scope, the decoys are set up as "traps" for the adversary. But it is possible to use decoys as "fake targets" for distracting the adversary. We intend to explore a mixture of types of decoys given their functionalities in cyber-physical defense and the respective deceptive synthesis problems and decoy-allocation problems. Also, we are interested in deceptive planning for other class of games, for example, concurrent(i.e., simultaneous-move) reachability games [1]. We intend to implement a toolbox for the proposed algorithm and apply the methods to practical network security problems.

References

1. de Alfaro, L., Henzinger, T.A., Kupferman, O.: Concurrent Reachability Games. Theoretical Computer Science **386**(3), 188–217 (Nov 2007)
2. Anwar, A.H., Kamhoua, C., Leslie, N.: Honeypot Allocation over Attack Graphs in Cyber Deception Games. In: 2020 International Conference on Computing, Networking and Communications (ICNC). pp. 502–506 (2020)
3. Aslanyan, Z., Nielson, F., Parker, D.: Quantitative Verification and Synthesis of Attack-Defence Scenarios. In: 2016 IEEE 29th Computer Security Foundations Symposium (CSF). pp. 105–119. IEEE (Jun 2016)
4. Bai, W., Bilmes, J.A.: Greed is still good: Maximizing monotone submodular+supermodular functions. arXiv preprint arXiv:1801.07413 (2018)
5. Baier, C., Katoen, J.P.: Principles of model checking (2008)
6. Bennett, P.G., Bussel, R.R.: Hypergame Theory and Methodology: The Current "State of the Art". In: Wilkin, L. (ed.) The Management of Uncertainty: Approaches, Methods and Applications, pp. 158–181. Springer Netherlands, Dordrecht (1986)
7. Bloem, R., Chatterjee, K., Jobstmann, B.: Graph Games and Reactive Synthesis. In: Clarke, E.M., Henzinger, T.A., Veith, H., Bloem, R. (eds.) Handbook of Model Checking, pp. 921–962. Springer International Publishing (2018)
8. Chatterjee, K., Henzinger, T.A.: A survey of stochastic omega-regular games. Journal of Computer and System Sciences **78**(2), 394–413 (2012)
9. Durkota, K., Lisy, V., Bosansky, B., Kiekintveld, C.: Optimal Network Security Hardening Using Attack Graph Games. In: Twenty-Fourth International Joint Conference on Artificial Intelligence (2015)
10. Filiot, E., Jin, N., Raskin, J.F.: Antichains and Compositional Algorithms for LTL Synthesis. Formal Methods in System Design pp. 261–296 (Dec 2011)
11. Gradel, E., Thomas, W.: Automata, Logics, and Infinite Games: A Guide to Current Research, vol. 2500. Springer Science & Business Media (2002)
12. Heckman, K.E., Stech, F.J., Thomas, R.K., Schmoker, B., Tsow, A.W.: Bridging the Classical D&D and Cyber Security Domains. In: Cyber Denial, Deception and Counter Deception: A Framework for Supporting Active Cyber Defense, pp. 5–29. Springer International Publishing (2015)

13. Jha, S., Sheyner, O., Wing, J.: Two Formal Analyses of Attack Graphs. In: Proceedings 15th IEEE Computer Security Foundations Workshop. CSFW-15. pp. 49–63 (2002)
14. Jiang, W., Fang, B.x., Zhang, H.l., Tian, Z.h., Song, X.f.: Optimal Network Security Strengthening Using Attack-Defense Game Model. In: 2009 Sixth International Conference on Information Technology: New Generations. pp. 475–480 (2009)
15. Kiekintveld, C., Jain, M., Tsai, J., Pita, J., Ordóñez, F., Tambe, M.: Computing optimal randomized resource allocations for massive security games. In: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1. pp. 689–696 (2009)
16. Kiekintveld, C., Lisỳ, V., Píbil, R.: Game-theoretic foundations for the strategic use of honeypots in network security. In: Cyber Warfare, pp. 81–101. Springer (2015)
17. Kulkarni, A.N., Luo, H., Leslie, N.O., Kamhoua, C.A., Fu, J.: Deceptive Labeling: Hypergames on Graphs for Stealthy Deception. IEEE Control Systems Letters **5**(3), 977–982 (2021)
18. Kulkarni, A.N., Fu, J.: A compositional approach to reactive games under temporal logic specifications. In: 2018 Annual American Control Conference (ACC). pp. 2356–2362. IEEE (2018)
19. Manna, Z., Pnueli, A.: The Temporal Logic of Reactive and Concurrent Systems: Specification. Springer-Verlag (1992)
20. McNaughton, R.: Infinite games played on finite graphs. Annals of Pure and Applied Logic **65**(2), 149–184 (1993)
21. Ou, X., Boyer, W.F., McQueen, M.A.: A scalable approach to attack graph generation. In: Proceedings of the 13th ACM Conference on Computer and Communications Security - CCS '06. pp. 336–345. ACM Press, Alexandria, Virginia, USA (2006)
22. Píbil, R., Lisỳ, V., Kiekintveld, C., Bošanskỳ, B., Pěchouček, M.: Game theoretic model of strategic honeypot selection in computer networks. In: International Conference on Decision and Game Theory for Security. pp. 201–220. Springer (2012)
23. Sasaki, Y., Kijima, K.: Hierarchical Hypergames and Bayesian Games: A Generalization of the Theoretical Comparison of Hypergames and Bayesian Games Considering Hierarchy of Perceptions. Journal of Systems Science and Complexity **29**(1), 187–201 (Feb 2016)
24. Sinha, A., Fang, F., An, B., Kiekintveld, C., Tambe, M.: Stackelberg Security Games: Looking Beyond a Decade of Success. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence. pp. 5494–5501. International Joint Conferences on Artificial Intelligence Organization (2018)
25. Thakoor, O., Tambe, M., Vayanos, P., Xu, H., Kiekintveld, C., Fang, F.: Cyber Camouflage Games for Strategic Deception. In: Alpcan, T., Vorobeychik, Y., Baras, J.S., Dán, G. (eds.) Decision and Game Theory for Security. pp. 525–541. Lecture Notes in Computer Science, Springer International Publishing (2019)
26. Vazirani, V.V.: Approximation Algorithms. Springer-Verlag (2003)
27. Wang, M., Hipel, K.W., Fraser, N.M.: Solution Concepts in Hypergames. Applied Mathematics and Computation **34**(3), 147–171 (Dec 1989)