On Supervalid Inequalities for Binary Interdiction Games

Ningji Wei*
Industrial, Manufacturing and Systems Engineering Department
Texas Tech University
Lubbock, TX 79409
ningji.wei@ttu.edu

Jose L. Walteros
Department of Industrial and Systems Engineering
University at Buffalo, The State University of New York
Buffalo, NY 14228.
josewalt@buffalo.edu

Abstract

Supervalid inequalities are a specific type of constraints often used within the branch-and-cut framework to strengthen the linear relaxation of mixed-integer programs. These inequalities share the particular characteristic of potentially removing feasible integer solutions as long as they are already dominated by an incumbent solution. This paper focuses on supervalid inequalities for solving binary interdiction games. Specifically, we provide a general characterization of inequalities that are derived from bipartitions of the leader's strategy set and develop an algorithmic approach to use them. This includes the design of two verification subroutines that we apply for separation purposes. We provide three general examples in which we apply our results to solve binary interdiction games targeting shortest paths, spanning trees, and vertex covers. Finally, we explore a connection between the proposed supervalid inequalities and a specific type of set system called greedoids.

Keywords: Network interdiction, Cutting planes, Integer programming, Greedoids.

Aknowledgements: Parts of this material are based upon work supported by the Office of Naval Research under contract No. N00014-20-1-2242.

1 Introduction

This paper studies a broad class of Stackelberg competitions called binary interdiction games, where two players, denoted the *leader* and *follower*, sequentially solve interdependent optimization problems with conflicting objectives. Drawing from the definition and notation established in [53], binary interdiction games can be characterized as follows. Given a set of elements Δ representing the ground set of the game, let collections $\Pi \subseteq 2^{\Delta}$ and $\Omega \subseteq 2^{\Delta}$ denote the solution spaces of the leader and follower, respectively. Specifically, set Ω, called the *structure set*, comprises a collection of subsets of Δ satisfying some structural properties, from which the follower selects one that

*Corresponding author. Phone: 806-834-8287; Address: IMSE Building Room 214, 905 Canton, Lubbock, TX 79409; Email: ningji.wei@ttu.edu

minimizes a weight function $w : \Delta \to \mathbb{R}^+$. In turn, set Π, denoted the *strategy set*, contains all the interdiction actions the leader can take to block some of the follower's structure choices.

For any action pair $(U, T) \in \Pi \times \Omega$, we will say that strategy U blocks structure T if $|U \cap T| \geq 1$, and define $\Omega_U = \{T \in \Omega \mid |U \cap T| \geq 1\}$ to be the collection of structures that are blocked by such a strategy. Consequently, if the leader chooses as their interdiction strategy the set U, the follower is then blocked from selecting any structure from Ω_U. Given a cost function $c : \Delta \to \mathbb{R}^+$ and a predefined threshold value $r \in \mathbb{R}$, the objective of the leader is to select a strategy $U \in \Pi$ of minimum cost such that the optimal choice of the follower is a structure whose weight is no less than r. In other words, the leader aims to minimize the cost of adopting an interdiction strategy that ensures a desired disruption level is inflicted on the follower's objective. This type of binary interdiction game can be modeled as follows

$$\begin{aligned} \min_{U \in \Pi} \quad & c(U) \\ \text{s.t.} \quad & \min_{T \in \Omega \setminus \Omega_U} w(T) \geq r. \end{aligned} \tag{1}$$

It is important to note that a similar variation of this problem often considered in the literature corresponds to the case where the leader's objective is set instead to maximize the weight of the optimal structure selected by the follower while ensuring that the cost of the chosen interdiction strategy is kept below a predefined budget. In [53], the authors identified some commonalities between the two versions and showed that similar solution approaches apply to both. In what follows, we will focus on binary interdiction games that can be formulated as (1).

Interestingly, considering the abstract nature of the structure and strategy sets in this model, it is no surprise that binary interdiction encapsulates a wide variety of adversarial games extensively studied in recent literature. For example, when the ground set Δ is composed of the edges or vertices of a given network, the aforementioned characterization can be used to model problems where the leader aims to restrict the follower's ability to conduct some operation in the network represented by structures such as shortest paths [26], spanning trees [54], cliques [17, 18, 36, 37], connected components [2, 14], matchings [63], dominating sets [41], or vertex covers [7]. Indeed, applications of such types of problems have found their niche in a wide variety of areas, including homeland security [20, 24, 25, 39, 59], disaster management [38], immunization strategies [50], sex trafficking prevention [60], energy systems [46], supply chain management [40], communications [15, 55], and transportation and logistics [27], among others.

Furthermore, recent interest in interdiction has spurred new developments, including several variations and extensions for these problems. Some notable examples include three-player interdiction games in which the new player, often called the protector, aims to conduct some preemptive actions to mitigate the effects of the interdiction strategy on the follower's structures [33]; interdiction games where both players act simultaneously without knowing in advance the strategy chosen by the other [22, 52]; stochastic interdiction games where some data of the problem follows some random probability distribution [12]; dynamic interdiction where the interaction between the two players is repeated through multiple rounds [47]; interdiction games with incomplete or asymmetric information where the knowledge of the underlying data is incomplete for one of the players or perceived differently [10, 44, 62]. Interested readers can refer to [48] for a general discussion about interdiction games, including solution methods and other novel variations.

Among the different solution methods, most exact approaches can be classified as either *dualize-and-combine* or *sampling/enumerative* methods [48]. The first type is typically reserved for cases where the structure set Ω admits a characterization in the form of a convex set (often, a polyhedron). In such cases, the inner optimization problem of the bi-level model in (1) can be replaced by its dual representation, thereby producing a single-level reformulation that any off-the-shelf optimization solver can directly solve [12, 26, 31, 54]. Dualize-and-combine approaches like these mainly stem from developments to solve problems in the broader class of bi-level optimization of which binary interdiction is a part of [4, 13, 29].

As for the sampling/enumerative algorithms, the main idea is to start with a small *sample* of structures in Ω that the leader would like to block and iteratively identify new ones as the algorithm progresses. To this end, most approaches utilize mathematical models that are iteratively populated with new constraints that force the interdiction strategy to block the new structures that are progressively generated [11, 17, 26, 33, 37, 41, 54, 56].

For the particular case of binary interdiction games characterized by (1), the authors in [53] denote $\hat{\Omega} = \{t \in \Omega \mid w(T) < r\}$ as the set of "critical" structures that the leader should block; then, by letting $\mathbf{x} \in \{0,1\}^{|\Delta|}$ be the indicator vector of the leader's interdiction strategy and X_Π the characterization of the strategy set Π over the x space, they use the following model as a valid reformulation of (1)

$$\min \quad \sum_{a \in \Delta} c_a x_a \tag{2a}$$

$$\text{s.t.} \quad \sum_{a \in T} x_a \geq 1 \quad \forall T \in \hat{\Omega} \tag{2b}$$

$$\mathbf{x} \in X_\Pi. \tag{2c}$$

Here, for the sake of exposition, a linear cost function c is used, but note that the formulation can be directly adapted to accommodate other types of cost functions. Furthermore, in [53], the authors also proved that it is sufficient to use only the constraints in (2b) that are associated with the *minimal* critical structures in $\hat{\Omega}$ and hence refer to this formulation as the minimal critical structures (MCS) formulation. We will subsequently refer to (2) by such a name too.

In recent years, several papers have introduced different families of inequalities to strengthen the mathematical formulations used to solve these problems [17, 26, 31, 37, 41, 54]. While some efforts have been focused on general cuts that can be applied to multiple types of interdiction games [11, 33, 54], most of the literature focuses on ad hoc inequalities tailored for some particular kind of structures [17, 41, 53]. In this paper, we are interested in a specific kind of constraints generally called *supervalid inequalities* [6, 26, 28, 45, 54]. In essence, a supervalid inequality is a constraint that can be added to strengthen a mathematical program such as (2) that may cut off feasible or even optimal solutions as long as a given incumbent solution already dominates them. Their usage in the context of interdiction stems from the seminal work by Israeli and Wood [26], where those are specifically designed to solve shortest-path interdiction games.

In the current literature, despite some promising computational results, these types of inequalities have only been developed sparingly for interdiction games with specific follower's structures, such as paths [26], vehicle routing plans [28], or spanning trees [54]. To the best of our knowledge, this paper is the first to study a general class of supervalid inequalities that can be applied to the broad class of binary interdiction games. Specifically, the contributions of this paper follow.

1. We identify a relationship between any bipartition $\mathcal{P}$ of the leader's strategy space Π and a collection of subsets from ground set Δ, denoted $\mathcal{P}$*-structures*, each of which induces a supervalid inequality for the MCS formulation. We provide a general characterization of these inequalities and develop an algorithmic approach to use them in practice.

2. We study the separation process of the proposed supervalid inequalities. First, we develop an exact verification method used to identify whether any set $S \subseteq \Delta$ is a $\mathcal{P}$-structure. We

show, however, that using such a verification method can be computationally challenging in practice. To improve the overall efficacy of this separation method, we then identify two types set constructions, based on which a more efficient verification becomes achievable. As a side product, all these characterizations also lead to a hierarchy map of the $\mathcal{P}$-structures.

3. We provide three general examples where we apply our results to derive supervalid inequalities for binary interdiction games targeting shortest paths, spanning trees, and vertex covers. In doing so, we also identify some interesting properties of the particular $\mathcal{P}$-structures associated with each of these problems.

4. Finally, we derive a connection between the $\mathcal{P}$-structures and a special type of set system called *greedoid*. We prove that, under mild assumptions, the $\mathcal{P}$-structure separation procedure is guaranteed to be efficient for greedoid interdiction games.

The rest of the paper is organized as follows. In Section 2, we introduce some notation and three binary interdiction examples used throughout the paper for ilustration purposes. In Section 3, we derive a correspondence between classes of supervalid inequalities and bipartitions of the leader's strategy space. In Section 4, we develop several methods to separate the proposed supervalid inequalities. In Section 5, we explore the connection between the proposed supervalid inequalities and a particular type of set system called a greedoid. Finally, in Section 6, we provide some concluding remarks.

2 Preliminaries

2.1 Notation

We now proceed to introduce some notation that will be used throughout the paper. A set system $(\Delta, \mathcal{K})$ is defined as a pair composed of a ground set Δ and a family of subsets $\mathcal{K} \subseteq 2^{\Delta}$, such that $\mathcal{K}$ is a *partially ordered set* (*poset*) under the inclusion $(\subseteq)$. Some poset-related notions that will be used hereafter are listed below.

Definition 1. Given a poset $(\mathcal{K}, \subseteq)$,

- let $m(\mathcal{K})$ be the set of all minimal elements in $\mathcal{K}$.
- $\mathcal{S}$ is a subposet of $\mathcal{K}$ if $\mathcal{S}$ is a subset of $\mathcal{K}$ with the inherited ordering.
- $\mathcal{S}$ is a lower (or upper) set in $\mathcal{K}$ if $\mathcal{S}$ is a subposet of $\mathcal{K}$ and, for any $U_1, U_2 \in \mathcal{K}$ such that $U_1 \subseteq U_2$ (or $U_1 \supseteq U_2$), $U_2 \in \mathcal{S}$ implies that $U_1 \in \mathcal{S}$.
- Given a subset $\mathcal{U}$ of a poset $\mathcal{K}$,

$$\uparrow \mathcal{U} = \{U' \in \mathcal{K} \mid U' \supseteq U \text{ for some } U \in \mathcal{U}\} \text{ and } \downarrow \mathcal{U} = \{U' \in \mathcal{K} \mid U' \subseteq U \text{ for some } U \in \mathcal{U}\}$$

  are called the upper and lower closure of $\mathcal{U}$ in $\mathcal{K}$, respectively.

2.2 Examples of Binary Interdiction Games

In this subsection, we introduce three well-known interdiction games that fit into the binary characterization described before. They will be used for illustrative purposes throughout the paper.

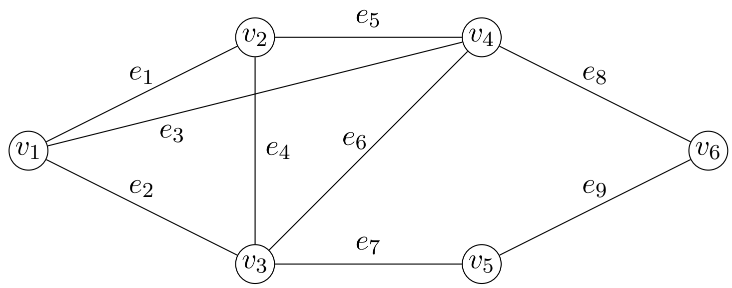


Figure 1: Example of a graph $G = (V, E)$ with $V = \{v_1, \ldots, v_6\}$ and $E = \{e_1, \ldots, e_9\}$.

Example 1 (Spanning Tree Interdiction [54])**.** Given an undirected graph $G = (V, E)$, with edge weights $\{w_e\}_{e\in E}$ and edge costs $\{c_e\}_{e\in E}$, a *spanning tree* is a connected acyclic subgraph that spans the vertex set V (e.g., $\{e_1, e_2, e_5, e_7, e_8\}$ in Figure 1). Let r be an interdiction target value; then, the spanning tree interdiction problem seeks to identify a subset of edges $U \subseteq E$ of minimum cost $\sum_{e\in U} c_e$, so that when removed from G, the minimum weight of any spanning tree left in the graph is at least r. Here, if the interdiction strategy U disconnects the graph, no spanning tree can be selected by the follower. Thus by convention, its objective is assumed to be infinity. △

Example 2 (Shortest Path Interdiction [26])**.** Given a directed graph $G = (V, E)$, with edge costs $\{c_e\}_{e\in E}$ and edge lengths $\{w_e\}_{e\in E}$, two terminal vertices $s, t \in V$, and an interdiction target r, the shortest path interdiction problem aims to identify a set of edges $U \subseteq E$ of minimum cost $\sum_{e\in U} c_e$ so that when removed from G, the length of the shortest s-t paths that are left is at least r. As before, if the interdiction strategy U disconnects terminals s and t, the follower's objective is assumed to be infinity. △

Example 3 (Vertex Cover Interdiction [8])**.** Given a graph $G = (V, E)$, with vertex costs $\{c_i\}_{i\in V}$ and vertex weights $\{w_i\}_{i\in V}$, a vertex cover is a vertex subset $D \subseteq V$ such that all edges in G are incident to at least one vertex in D (e.g., $D = \{v_1, v_2, v_4, v_5\}$ in Figure 1). Given an interdiction target r, the vertex cover interdiction problem is to identify a set of vertices $U \subseteq V$ of minimum costs to be blocked so that the weight of any vertex cover $D \subseteq V \setminus U$ of graph G is at least r. Similarly, if no vertex cover can be formed by selecting vertices from $D \subseteq V \setminus U$, the follower's objective is assumed to be infinity. △

3 Bipartition-Induced Supervalid Inequalities

3.1 Motivation

Many interdiction problems share an interesting feature: when the search for a solution is conducted over some particular restriction of the leader's solution space, the resulting problem often becomes significantly easier to solve. To illustrate this, consider the shortest path interdiction game depicted in Figure 2. Here, among the five possible s-t paths available for the follower, we will assume that the first three are critical—i.e., the lengths of such paths are less than the predefined interdiction target r. Under these settings, the leader's strategy set $\Pi = 2^E$ contains all the possible edge subsets in the graph, and among the feasible solutions therein, the minimal strategies that block all the critical paths are the following four edge sets

$$\{e_1, e_2, e_5\}, \{e_3, e_4, e_5\}, \{e_1, e_3, e_5\}, \{e_2, e_4, e_5\}.$$

Notice that the first two strategies are somewhat different from the other two as they are minimal *s-t* cuts. In fact, since any *s-t* cut is a feasible solution for the leader, one can use this *property* to produce the following partition of the leader's solution space Π:

$$\Pi_1 := \uparrow \{\{e_1, e_2, e_5\}, \{e_3, e_4, e_5\}\}, \quad \Pi_0 := \Pi \setminus \Pi_1.$$

Here, observe that set Π_1 is composed of the two strategies mentioned before and their supersets, as those are all the *s-t* cuts of the given graph. Furthermore, considering that finding an *s-t* cut of minimum cost can be done in polynomial time [23], solving the shortest path interdiction problem over the restricted space Π_1 is rather easy, whereas in general solving the problem over Π is not [3].

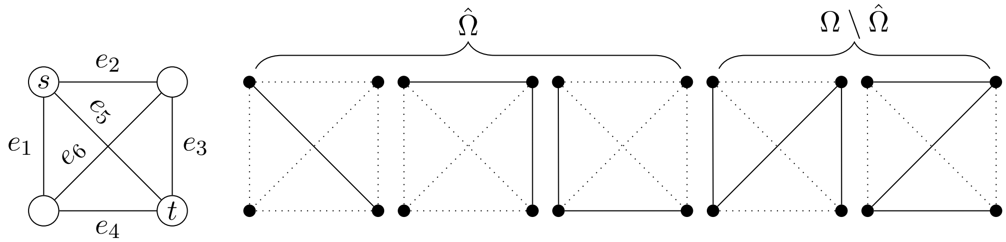


Figure 2: Illustration of bipartition in the shortest path interdiction.

This phenomenon seems to be a common occurrence in binary interdiction. Indeed, for the minimum spanning tree case, it is easy to see that any (global) edge-cut is a feasible strategy to block all spanning trees of a graph. Moreover, since finding a minimum cost edge-cut can be done in polynomial time [49], restricting the search over the set Π_1 composed of all edge cuts in the graph is a simple task while directly searching for an optimal solution over Π is, in general, NP-hard [9, 16].

In light of this discussion, a reasonable strategy when solving binary interdiction games is to first identify the best solution out of Π_1 and then focus all efforts on the more "difficult" task of searching for a potentially better solution over the remaining set Π_0. To speed up this search, one may wonder if it is possible to use relevant information about the given bipartition $\{\Pi_0, \Pi_1\}$ to produce a tighter mathematical formulation focused on searching for solutions in Π_0 rather than over the entire solution space Π. The goal is to systematically derive inequalities for (2) that are valid for the feasible solutions in Π_0 but may cut off strategies from Π_1, as well as some undesired fractional solutions from the formulation's original LP relaxation. The expected result is a tighter formulation defined by *supervalid* inequalities.

The use of exploratory searches over restricted solution spaces to speed up the running times of exact optimization solvers has been a common practice in integer programming since its conception. For example, exact approaches based on branch and bound often use local-search-based algorithms to explore some predefined neighborhood to identify good candidate solutions that may help prune unpromising branches [58]. While this type of approach can be directly applied here as well, our aim goes beyond using the search over Π_1 to heuristically generate candidate solutions. Instead, we focus on producing stronger mathematical formulations.

As will be discussed in the following sections, our approach relies on exploiting a bipartition of the solution space Π into a set Π_1 whose exploration is "easy" and a set Π_0 whose exploration is "difficult". An important remark is that here we are using the terms *easy/difficult* rather loosely, as our methodology works for any bipartition of the solution space Π—even for cases where finding the best solution in Π_1 is also NP-Hard. However, in practice, our method is better suited for bipartitions for which solving the problem over Π_1 can be done in polynomial time. Furthermore, considering that the characterization of set Π_1 is generally linked to some structural attributes of its members (e.g., s-t cuts in a graph), in what follows, we will assume that the bipartitions $\{\Pi_0, \Pi_1\}$ are induced by some given property.

3.2 Bipartition Property $\mathcal{P}$ and the $\mathcal{P}$-Structure Formulation

We begin with the following definitions.

Definition 2 (Bipartition Property)**.** Given a set Π, a *bipartition property* is a function $\mathcal{P} : \Pi \to \{0,1\}$. We set $\Pi_i^{\mathcal{P}} = \{U \in \Pi \mid \mathcal{P}(U) = i\}$ for $i \in \{0,1\}$. When $\mathcal{P}$ is clear from the context, we drop the superscript. Given two bipartition properties $\mathcal{P}_1$ and $\mathcal{P}_2$, we say $\mathcal{P}_2$ is stronger than $\mathcal{P}_1$, denoted $\mathcal{P}_1 \preceq \mathcal{P}_2$, if and only if $\Pi_1^{\mathcal{P}_1} \subseteq \Pi_1^{\mathcal{P}_2}$.

Definition 3 (Cut/Cut Operator)**.** Given a family of sets $\Pi \subseteq 2^\Delta$, $S \subseteq \Delta$ is a *cut* of Π if $|S \cap U| \geq 1$ for all $U \in \Pi$. The *cut operator* $\mathcal{C} : 2^\Delta \to 2^\Delta$ is defined as $\mathcal{C}(\Pi) = \{\text{all cuts of } \Pi\}$.

Notice that the cut operator also establishes a connection between the leader's feasible strategies, denoted by $\hat{\Pi}$, and the follower's critical structures $\hat{\Omega}$ through the following identity, which can be verified directly from the definitions

$$\hat{\Pi} = \mathcal{C}(\hat{\Omega}) \cap \Pi.$$

In fact, it is easy to see that each constraint (2b) of the MCS formulation is there to enforce that the feasible solutions of the leader represent cuts of $\hat{\Omega}$.

Some properties of the cut operator $\mathcal{C}$ are listed in the following proposition,

Proposition 1. *Given two posets $\mathcal{K}, \mathcal{K}' \subseteq 2^\Delta$ with the ordering inherited from $\subseteq$ on 2^Δ, the following statements about the cut operator $\mathcal{C}$ are true,*

1. $\mathcal{C}(\emptyset) = 2^\Delta$ *and* $\mathcal{C}(2^\Delta) = \emptyset$,
2. $\mathcal{C}(\mathcal{K}) = \emptyset$ *if* $\emptyset \in \mathcal{K}$,
3. $\mathcal{C}(\mathcal{K})$ *is an upper set,*
4. $\mathcal{C}(\mathcal{K}) = \mathcal{C}(m(\mathcal{K}))$,
5. $\mathcal{C}$ *is decreasing, i.e., if* $\mathcal{K} \subseteq \mathcal{K}'$*, then* $\mathcal{C}(\mathcal{K}) \supseteq \mathcal{C}(\mathcal{K}')$,
6. $\mathcal{C}(\mathcal{C}(\mathcal{K})) = \uparrow \mathcal{K}$.

Proof. For *1.*, the first part is vacuously true, and the second part is true because $\emptyset \in 2^\Delta$. Statements *2.-5.* can be verified directly from the definition of $\mathcal{C}$. For *6.*, if $\emptyset \in \mathcal{K}$, then by *1.* and *2.*, $\mathcal{C}(\mathcal{C}(\mathcal{K})) = 2^\Delta$, which is equal to $\uparrow \mathcal{K}$, since $\emptyset \in \mathcal{K}$. Suppose $\emptyset \notin \mathcal{K}$, then we have,

$$U \in \mathcal{C}(\mathcal{C}(\mathcal{K})) \iff \forall S \in \mathcal{C}(\mathcal{K}),\ |U \cap S| \geq 1 \iff U \in \uparrow \mathcal{K},$$

where the first equivalence follows from the definition of the operator $\mathcal{C}$. For the "$\Leftarrow$" direction of the second equivalence, $U \in \uparrow \mathcal{K}$ means $U \supseteq U'$ for some $U' \in \mathcal{K}$. By definition, for all $S \in \mathcal{C}(\mathcal{K})$, $|S \cap U'| \geq 1$, then this is also true for the superset U. For "$\Rightarrow$", we prove the contrapositive. Given $U \notin \uparrow \mathcal{K}$ and $\emptyset \notin \mathcal{K}$ by assumption, then for every $U' \in \mathcal{K}$, $U' \setminus U \neq \emptyset$. Then, $S = \bigcup_{U' \in K} U' \setminus U$ intersects all elements in $\mathcal{K}$, but $S \cap U = \emptyset$ by construction. This negates the second sentence in the above equivalence chain, completing the proof. □

Using the cut operator, we define the following structures that are induced by a given bipartition property $\mathcal{P}$.

Definition 4 ($\mathcal{P}$-Structure)**.** Given the set of follower's critical structures $\hat{\Omega} \subseteq 2^{\Delta}$, the set of leader's strategies Π, and a bipartition property $\mathcal{P} : \Pi \to \{0, 1\}$, the set of bipartition-induced structures with respect to $\mathcal{P}$, or simply $\mathcal{P}$-structures, is defined as

$$\hat{\Omega}_{\mathcal{P}} := \mathcal{C}(\mathcal{C}(\hat{\Omega}) \cap \Pi_0) = \mathcal{C}(\hat{\Pi} \cap \Pi_0).$$

Since $\hat{\Pi} = \mathcal{C}(\hat{\Omega})$ is the set of all the feasible strategies, we use $\hat{\Pi}_0$ to denote all the feasible strategies in Π_0. Hence, $\hat{\Omega}_{\mathcal{P}} = \mathcal{C}(\hat{\Pi}_0)$.

Intuitively, a $\mathcal{P}$-structure is a subset of the ground set Δ that intersects all the feasible strategies in $\hat{\Pi}_0$ that resulted from the bipartition property $\mathcal{P}$. These $\mathcal{P}$-structures are particularly important for our development, as we will soon show that they can directly induce supervalid inequalities for Formulation (2).

Since every bipartition property $\mathcal{P}$ induces a certain set of $\mathcal{P}$-structures, it is then reasonable to analyze first the following two extreme properties.

Proposition 2. *Define bipartition properties* $\mathbf{0}$ *and* $\mathbf{1}$ *as*

$$\mathbf{0}(U) = 0 \text{ and } \mathbf{1}(U) = 1$$

for all $U \in \Pi$*. Then, we have* $\hat{\Omega}_{\mathbf{0}} = \uparrow \hat{\Omega}$ *and* $\hat{\Omega}_{\mathbf{1}} = \uparrow \emptyset = 2^{\Delta}$*.*

Proof. For $\mathbf{0}$, we have $\Pi_0 = \Pi$, which leads to the following

$$\hat{\Omega}_{\mathbf{0}} = \mathcal{C}(\hat{\Pi}) = \mathcal{C}(\mathcal{C}(\hat{\Omega})) = \uparrow \hat{\Omega},$$

where the first two identities are by Definition 4 and the last is by the statement *6.* of Proposition 1. A similar verification can be done for property $\mathbf{1}$. □

Intuitively, these two extreme properties correspond to the following two cases. First, the $\mathbf{0}$ property induces the bipartition where set $\Pi_1 = \emptyset$, thus all leader strategies are contained in $\Pi_0 = \Pi$. Here, since the problem over Π_0 is the same as the problem over Π, the corresponding $\mathcal{P}$-structure set is simply the upper closure of the critical structures in $\hat{\Omega}$. Second, the $\mathbf{1}$ property induces the bipartition where $\Pi_1 = \Pi$ and thus $\Pi_0 = \emptyset$. Here, under the assumption that Π_1 is "easy to explore", property $\mathbf{1}$ represents the extreme case where the original interdiction problem is solvable in polynomial time. Any property $\mathcal{P}$ that is strictly sandwiched between $\mathbf{0}$ and $\mathbf{1}$ (i.e., $\mathbf{0} \preceq \mathcal{P} \preceq \mathbf{1}$) will produce nontrivial $\mathcal{P}$-structures. We are interested in these $\mathcal{P}$-structures due to the following theorem.

Theorem 1. *In a binary interdiction game with leader's strategy set* Π *and follower's critical structure set* $\hat{\Omega}$*, given any bipartition property* $\mathcal{P}$ *on* Π*, an optimal strategy* $U^{\star}$ *must satisfy exactly one of the following two assertions,*

- $U^{\star} \in \Pi_1$*;*
- $U^{\star} \in \mathcal{C}(\hat{\Omega}_{\mathcal{P}})$*.*

Proof. An optimal strategy $U^{\star}$ that is not in Π_1 must be in $\hat{\Pi}_0$ since Π_0 and Π_1 form a bipartition of Π and $U^{\star}$ is feasible. On the other hand, we have

$$\mathcal{C}(\hat{\Omega}_{\mathcal{P}}) = \mathcal{C}(\mathcal{C}(\hat{\Pi}_0)) = \uparrow \hat{\Pi}_0 \supseteq \hat{\Pi}_0,$$

which concludes the proof. □

This theorem allows us to utilize a bipartition on Π and the assumed "easy-to-obtain" solution in Π_1 to set up a stronger formulation for the problem defined over the "difficult-to-explore" space Π_0. In particular, we get the following minimal $\mathcal{P}$-structure (MPS) formulation,

$$\text{(MPS)} \quad \min \quad \sum_{a\in\Delta} c_a x_a \tag{3a}$$

$$\text{s.t.} \quad \sum_{a\in T} x_a \geq 1 \quad \forall T \in m(\hat{\Omega}_{\mathcal{P}}) \tag{3b}$$

$$\mathbf{x} \in X_\Pi. \tag{3c}$$

Note that this formulation only searches for an optimal strategy within $\uparrow \Pi_0$, thus it has a smaller solution space than the MCS formulation. In other words, the inequalities in (3c) are indeed supervalid for the general space Π, as they may remove feasible solutions to the original problem. The following corollary characterizes the solution space of (3) exactly.

Corollary 1. *A strategy $U \in \Pi$ is feasible to the MPS formulation if and only if it is a superset of some $U' \in \hat{\Pi}_0$.*

Proof. For sufficiency, note that any $U' \in \hat{\Pi}_0$ intersects all elements in $\hat{\Omega}_{\mathcal{P}}$ by definition, so as any superset of U'. For necessity, we assume $\emptyset \notin \hat{\Pi}_0$, otherwise, all strategies are supersets of $\emptyset$. Hence, to prove the contrapositive, given a strategy U that is not a superset of any $U' \in \hat{\Pi}_0$, we have $U' \setminus U \neq \emptyset$ for all $U' \in \hat{\Pi}_0$. Let $S = \bigcup_{U'\in\hat{\Pi}_0} U' \setminus U$, then S is a cut of $\hat{\Pi}_0$ but $S \cap U = \emptyset$ by construction. The former implies $S \in \hat{\Omega}_{\mathcal{P}}$ and the latter means U does not intersect the $\mathcal{P}$-structure S, that is, U is not feasible to the MPS formulation. □

In essence, this corollary states that the optimal face of (3) consists of solutions from Π_0 since all the strategies in Π_1 are either strictly dominated by the minimal strategies in Π_0 or removed by the constraint set (3c). The following corollary further studies the effect of choosing different bipartition properties.

Corollary 2. *Given two bipartition properties $\mathcal{P}$ and $\mathcal{P}'$, if $\mathcal{P} \preceq \mathcal{P}'$ then $\hat{\Omega}_{\mathcal{P}} \subseteq \hat{\Omega}_{\mathcal{P}'}$. In particular, $\hat{\Omega} \subseteq \hat{\Omega}_{\mathbf{0}} \subseteq \hat{\Omega}_{\mathcal{P}}$ for every bipartition property $\mathcal{P}$.*

Proof. By definition, $\mathcal{P} \preceq \mathcal{P}'$ if and only if $\Pi_0^{\mathcal{P}} \supseteq \Pi_0^{\mathcal{P}'}$, which implies $\hat{\Pi}_0^{\mathcal{P}} \supseteq \hat{\Pi}_0^{\mathcal{P}'}$. Then, the claim is true since the operator $\mathcal{C}$ is decreasing by the statement *5.* of Proposition 1. □

This result confirms the intuition that the constraint set induced by $\hat{\Omega}_{\mathcal{P}}$ becomes stronger when the $\mathcal{P}$ carves out a larger portion of the solution space Π into the "easy-to-explore" part Π_1.

Next, we will provide three concrete examples to illustrate the corresponding $\mathcal{P}$-structures.

3.3 Examples of $\mathcal{P}$-Structures

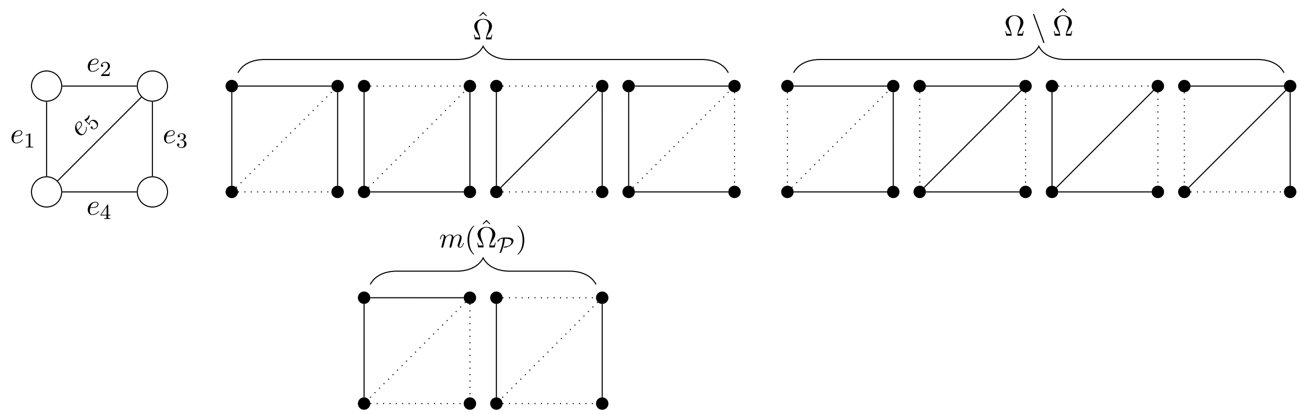
Example 1 (Spanning Tree Interdiction, Cont.)**.** As discussed previously, a natural bipartition property $\mathcal{P}$ for the minimum spanning tree interdiction is whether a solution is an edge-cut of graph G, since the edge cut of minimum cost can be found in polynomial time [49]. That is, for all $U \in \Pi$,

$$\mathcal{P}(U) = \begin{cases} 1, & U \text{ is an edge-cut,} \\ 0, & \text{otherwise.} \end{cases}$$

Figure 3 provides an example of the corresponding $\mathcal{P}$-structures for the minimum spanning tree interdiction problem over a 4-vertex graph. Here, among the eight spanning trees comprising the follower's structure set Ω, we assume the first four are the critical ones. The computation of $m(\hat{\Omega}_{\mathcal{P}})$ follows,

- $\hat{\Omega} = \{\{e_1, e_2, e_3\}, \{e_1, e_2, e_4\}, \{e_1, e_3, e_4\}, \{e_1, e_3, e_5\}\}$.
- $m(\hat{\Pi}) = \{\{e_1\}, \{e_2, e_3\}, \{e_3, e_4\}, \{e_2, e_4, e_5\}\}$.
- $m(\hat{\Pi}_0) = \{\{e_1\}, \{e_2, e_3\}\}$, since the other two are edge-cuts.
- $m(\hat{\Omega}_{\mathcal{P}}) = m(\mathcal{C}(m(\hat{\Pi}_0))) = \{\{e_1, e_2\}, \{e_1, e_3\}\}$.

This leads to the corresponding MPS formulation in Figure 3. Notice that the feasible edge cut solution $\{e_3, e_4\}$ becomes infeasible to this MPS formulation as it violates the first constraint. This confirms that these inequalities are supervalid for Π. We also note that, in this example, every $\mathcal{P}$-structure is a subset of some critical spanning tree.



$$\text{(MCS)} \quad \min \ \mathbf{cx} \quad \text{s.t.} \quad \begin{aligned} & x_1 + x_2 + x_3 \geq 1 \\ & x_1 + x_3 + x_4 \geq 1 \\ & x_1 + x_3 + x_5 \geq 1 \\ & x_1 + x_2 + x_4 \geq 1 \\ & \mathbf{x} \in \{0,1\}^E \end{aligned}$$

$$\text{(MPS)} \quad \min \ \mathbf{cx} \quad \text{s.t.} \quad \begin{aligned} & x_1 + x_2 \geq 1 \\ & x_1 + x_3 \geq 1 \\ & \mathbf{x} \in \{0,1\}^E \end{aligned}$$

Figure 3: $\mathcal{P}$-structures and supervalid inequalities in the minimum spanning tree interdiction.

In [54], these minimal $\mathcal{P}$-structures are called *critical edge sets* and are derived from a different perspective. △

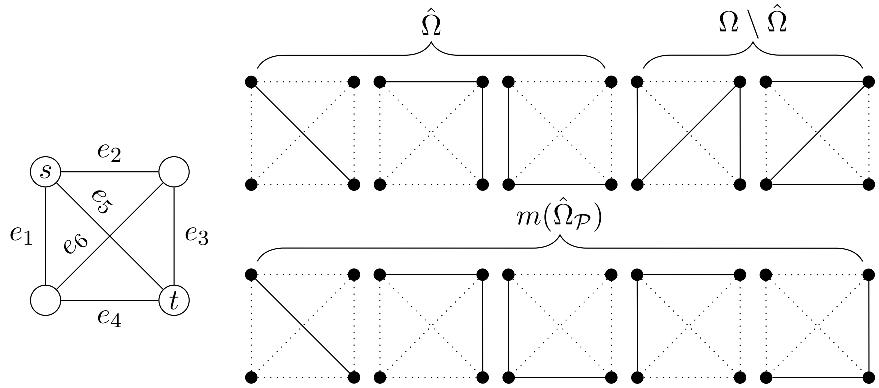
Example 2 (Shortest Path Interdiction, Cont.)**.** As mentioned before, a natural bipartition property for the shortest path interdiction is as follows. For each $u \in \Pi$,

$$\mathcal{P}(U) = \begin{cases} 1, & U \text{ is a } s\text{-}t \text{ cut}, \\ 0, & \text{otherwise}. \end{cases}$$

Figure 4 provides an example of the corresponding $\mathcal{P}$-structures for the shortest path interdiction problem over a 4-vertex graph. Here, among the five s-t paths comprising the follower's structure set Ω, we assume the first three are the critical ones. As before, we compute all the corresponding minimal $\mathcal{P}$-structures as follows.

- $\hat{\Omega} = \{\{e_1, e_4\}, \{e_2, e_3\}, \{e_5\}\}$.
- $m(\hat{\Pi}) = \{\{e_1, e_2, e_5\}, \{e_1, e_3, e_5\}, \{e_2, e_4, e_5\}, \{e_3, e_4, e_5\}\}$.
- $m(\hat{\Pi}_0) = \{\{e_1, e_3, e_5\}, \{e_2, e_4, e_5\}\}$, since the other two are s-t cuts.
- $m(\hat{\Omega}_{\mathcal{P}}) = m(\mathcal{C}(m(\hat{\Pi}_0))) = \{\{e_1, e_2\}, \{e_1, e_4\}, \{e_2, e_3\}, \{e_3, e_4\}, \{e_5\}\}$.

This generates the corresponding MPS formulation in Figure 4. Again, the feasible s-t cut solutions $\{e_1, e_2, e_5\}$ and $\{e_3, e_4, e_5\}$ violate constraints $x_3 + x_4 \geq 1$ and $x_1 + x_2 \geq 1$, respectively, in the MPS formulation.



$$(\text{MCS}) \quad \begin{aligned} \min \quad & \mathbf{cx} \\ \text{s.t.} \quad & x_5 \geq 1 \\ & x_2 + x_3 \geq 1 \\ & x_1 + x_4 \geq 1 \\ & \mathbf{x} \in \{0,1\}^E \end{aligned}$$

$$(\text{MPS}) \quad \begin{aligned} \min \quad & \mathbf{cx} \\ \text{s.t.} \quad & x_5 \geq 1 \\ & x_2 + x_3 \geq 1 \\ & x_1 + x_4 \geq 1 \\ & x_1 + x_2 \geq 1 \\ & x_3 + x_4 \geq 1 \\ & \mathbf{x} \in \{0,1\}^E \end{aligned}$$

Figure 4: $\mathcal{P}$-structures and supervalid inequalities in the shortest path interdiction.

In this example, all the minimal critical s-t paths are also minimal $\mathcal{P}$-structures. One particular difference with respect to the minimum spanning tree case is that here set $m(\hat{\Omega}_{\mathcal{P}})$contains two extra structures—$\{e_1, e_2\}$ and $\{e_3, e_4\}$—that are not subsets of any s-t path. △

Example 3 (Vertex Cover Interdiction, Cont.)**.** By definition, each edge in a graph $G = (V, E)$ has at least one endpoint in any vertex cover $T \subseteq V$. Thus, any vertex set $U \in \Pi$ that contains two adjacent vertices blocks all the vertex covers in G. Therefore, any feasible solution that does not satisfy this property must be an independent set. Following this logic, a bipartition property for the vertex cover interdiction can be defined as follows. For each $u \in \Pi$,

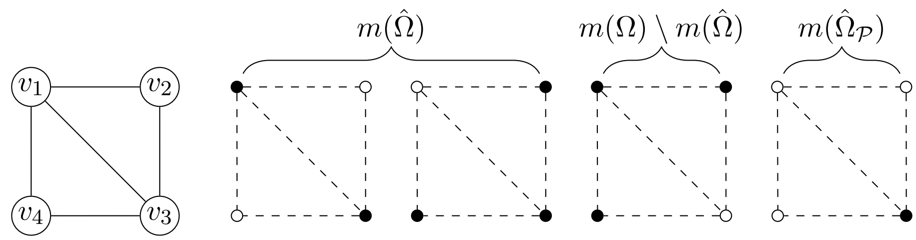
$$\mathcal{P}(U) = \begin{cases} 0, & U \text{ is an independent set,} \\ 1, & \text{otherwise.} \end{cases}$$

Figure 5 provides an example of the corresponding $\mathcal{P}$-structures for the vertex cover interdiction problem over a 4-vertex graph. Here, it is easy to verify that there are six vertex covers

$\{\{v_1, v_3\}, \{v_2, v_3, v_4\}, \{v_1, v_2, v_4\}, \{v_1, v_2, v_3\}, \{v_1, v_3, v_4\}, \{v_1, v_2, v_3, v_4\}\}$ comprising the follower's set Ω. We assume the first two are the critical ones. For convenience, in the figure, we only list the minimal vertex covers (in black). As before, we compute all the corresponding minimal $\mathcal{P}$-structures as follows.

- $\hat{\Omega} = \{\{v_1, v_3\}, \{v_2, v_3, v_4\}\}$.
- $m(\hat{\Pi}) = \{\{v_3\}, \{v_1, v_2\}, \{v_1, v_4\}\}$.
- $m(\hat{\Pi}_0) = \{\{v_3\}\}$, since the other two are not independent sets.
- $m(\hat{\Omega}_{\mathcal{P}}) = m(\mathcal{C}(m(\hat{\Pi}_0))) = \{\{v_3\}\}$.

This generates the corresponding MPS formulation in Figure 5. Again, the two feasible solutions $\{v_1, v_2\}$ and $\{v_1, v_4\}$ are removed by the supervalid constraints in the MPS formulation. Similarly to the minimum spanning tree example, every $\mathcal{P}$-structure here is a subset of some critical vertex cover. △



(MCS)

$$\begin{aligned} \min \quad & \mathbf{cx} \\ \text{s.t.} \quad & x_1 + x_3 \geq 1 \\ & x_2 + x_3 + x_4 \geq 1 \\ & \mathbf{x} \in \{0,1\}^V \end{aligned}$$

(MPS)

$$\begin{aligned} \min \quad & \mathbf{cx} \\ \text{s.t.} \quad & x_3 \geq 1 \\ & \mathbf{x} \in \{0,1\}^V \end{aligned}$$

Figure 5: $\mathcal{P}$-structures and supervalid inequalities in vertex cover interdiction.

In the next subsection, we will introduce an algorithm to solve the MPS formulation and prove its correctness.

3.4 MPS Algorithm

Since the MPS formulation only searches for an optimal solution from Π_0, we need to combine the results from Π_1 to obtain the global optimal solution for (2). We call this combined procedure the MPS algorithm (Algorithm 1). There are three subroutines invoked in this algorithm: (*i*) the EasySolve subroutine that solves the problem over the "easy-to-explore" solution space Π_1 and obtains the corresponding optimal solution therein; (*ii*) the MPS subroutine that solves a relaxed version of the MPS formulation defined over a set $\bar{\Omega}_{\mathcal{P}} \subseteq \Omega_{\mathcal{P}}$; and (*iii*) the MPSeparation subroutine that iteratively updates $\bar{\Omega}_{\mathcal{P}}$ by identifying new $\mathcal{P}$-structures in $\Omega_{\mathcal{P}} \setminus \bar{\Omega}_{\mathcal{P}}$. This iterative procedure terminates when either no new $\mathcal{P}$-structures can be produced or the objective value from MPS is at greater than or equal to the solution obtained by EasySolve. The following theorem proves the correctness and finite termination of Algorithm 1.

Algorithm 1: MPS Algorithm.

```
   Data: input c, 𝒫, Π, Ω̂
1  z̄, x̄ ←EasySolve(Π₁)            // get the optimal value and solution on Π₁
2  Ω̄_𝒫 ← ∅                       // initialize with an empty set of 𝒫-structures
3  z̲ ← −∞, x̲ ← 0                            // initialize the lower bound
4  while true do
5  |  z, x ←MPS(c, Π, Ω̄_𝒫)      // solve the MPS formulation with the current Ω̄_𝒫
6  |  if z ≥ z̄ then
7  |  |  z⋆ ← z̄, x⋆ ← x̄                       // stop due to worse than Π₁
8  |  |  break
9  |  else
10 |  |  Ω̄_x ← MPSeparation(c, Π, x)
11 |  |  if Ω̄_x = ∅ then
12 |  |  |  z⋆ ← z̲, x⋆ ← x̲                  // stop due to optimality on Π₀
13 |  |  |  break
14 |  |  else
15 |  |  |  Ω̄_𝒫 ← Ω̄_𝒫 ∪ Ω̄_x              // update the set of 𝒫-structures
16 |  |  |  z̲ ← z, x̲ ← x                          // update the lower bound
17 |  |  end
18 |  end
19 end
20 return z⋆, x⋆
```

Theorem 2. *The MPS algorithm terminates in finite iterations, and the returned solution $x^\star$ is optimal with respect to the MCS formulation.*

Proof. We first show that the number of iterations is finite. Let z_i and $\bar{\Omega}^i_{\mathcal{P}}$ denote the value of z and contents of $\bar{\Omega}_{\mathcal{P}}$ at Step 5 of the algorithm's ith iteration. Because of Step 15, we clearly have $z_i \leq z_{i+1}$ and $\bar{\Omega}^i_{\mathcal{P}} \subseteq \bar{\Omega}^{i+1}_{\mathcal{P}}$. Moreover, the latter inclusion is strict since, otherwise, we have $\bar{\Omega}_x = \emptyset$, and the procedure terminates due to Step 11–13. Since the set of $\mathcal{P}$-structures is finite, we have two possible cases: (i) at a certain iteration we get $\bar{\Omega}_{\mathcal{P}} = \hat{\Omega}_{\mathcal{P}}$ or (ii) the algorithm halts before (i) happens. For the former case, we have $\bar{\Omega}_x = \emptyset$ at the corresponding iteration, and, again, the algorithm terminates by Step 11–13, proving the claim.

For correctness, we again focus on the two cases introduced before. For (i), we have generated the full MPS formulation and have $z^\star < \bar{z}$. By Corollary 1, $x^\star$ is an optimal solution in Π_0. Moreover, $x^\star$ is better than the optimal solution in Π_1 as $z^\star < \bar{z}$. This proves $x^\star$ is an optimal solution over the entire solution space Π of the MCS formulation. For Case (ii), the algorithm may terminate in two different ways: (*ii.1*) $z \geq \bar{z}$; (*ii.2*) $\bar{\Omega}_x = \emptyset$. Case (ii.1) implies the optimal solution in Π_0 has an objective value that is greater than or equal to the optimal value in Π_1, which proves the optimality of $x^\star = \bar{x}$. Case (*ii.2*) means that solution x obtained at Step 5 is feasible for the MPS formulation and is optimal for a relaxation of the MPS formulation. Thus, x is also optimal to the MPS formulation, i.e., it is an optimal solution in Π_0. Again, because of $z < \bar{z}$, the solution x is optimal over the entire solution space Π. □

By assumption, subroutine EasySolve can be computed efficiently, and by Theorem 1 and Corollary 2, the MPS formulation has a strictly smaller solution space than the MCS formulation for any nontrivial bipartition property $\mathcal{P}$. Thus, the overall efficiency of the MPS algorithm depends on the trade-off between the strength of the constraints induced by the $\mathcal{P}$-structures and the complexity of the MPSeparation subroutine. We will devote the next section to the general theory of $\mathcal{P}$-structure separation.

4 $\mathcal{P}$-Structure Constraints Separation

Compared to the process of separating inequalities associated with the critical structures in $m(\hat{\Omega})$ (i.e., constraints (2b) of the MCS formulation), separating the constraints (3b) associated with a given set of $\mathcal{P}$-structures can be significantly harder. The main reason is that the elements in $m(\hat{\Omega})$ are feasible solutions for the follower that can be generated via the follower's optimization problem, whereas the $\mathcal{P}$-structures are arbitrary subsets of Δ without a direct method to produce them.

In this section, we focus on developing several methods for verifying whether any given set $S \subseteq \Delta$ is a $\mathcal{P}$-structure, which will naturally lead to constraint separation methods. Specifically, given a set $S \subseteq \Delta$, a set of critical structures $\hat{\Omega}$, and a bipartition property $\mathcal{P}$, a *$\mathcal{P}$-structure verifier* is a subroutine that returns true if $S \in \hat{\Omega}_{\mathcal{P}}$ and false otherwise. We say the verifier is *partial* if it only verifies a subset of $\hat{\Omega}_{\mathcal{P}}$, i.e., it may return a false negative but not a false positive.

A naive verification can be done by following the definition directly, as it was done for the three examples given in Section 3.3. However, this verification process is impractical even for modest-sized problems, as it requires knowing the entire feasible solution space $\hat{\Pi}$. In what follows, we restrict our attention to a special class of bipartition properties defined as follows.

Definition 5 (Null/Regular Properties)**.** Given the leader's strategy set Π and the follower's structure set Ω, the null property $\mathcal{P}_\emptyset$ is defined as follows. For every $U \in \Pi$,

$$\mathcal{P}_\emptyset(U) = \begin{cases} 1, & U \in \mathcal{C}(\Omega), \\ 0, & \text{otherwise.} \end{cases}$$

We say a property $\mathcal{P}$ is *regular* if and only if $\mathcal{P} \succeq \mathcal{P}_\emptyset$.

Hence, a strategy U belongs to $\Pi_1^{\mathcal{P}_\emptyset}$ if and only if it nullifies (intersects) every follower's structure in Ω (not just the critical structures in $\hat{\Omega}$). In fact, all three examples presented in Section 3.3 use the null property as their bipartition properties. Indeed, in the those examples, every edge-cut, s-t cut, and non-independent set intersects all the spanning trees, s-t paths, and vertex covers, respectively. We will soon to show that the $\mathcal{P}$-structures associated with null properties have some convenient characteristics that can be exploited for constraint separation purposes. Before diving into the verification algorithms, we need to provide some notation first.

4.1 Extended Sets

Definition 6. Given a family of sets Ω, we define

- $\Omega[S,k] := \{T \in \Omega \mid |T \cap S| \geq |S| - k\}$.
- $\Omega[S] := \Omega[S,0] = \{T \in \Omega \mid T \supseteq S\}$.

We call $\Omega[S,k]$ the set Ω extended on S with degree k and call $\Omega[S]$ the set Ω extended on S (with degree 0).

We have the following properties for the extended sets.

Proposition 3. *For sets $S \subseteq S'$ and numbers $k \leq k' \in \mathbb{N}$, we have*

1. $\Omega[S, k] = \Omega$ *for every* $k \geq |S|$.

2. $\Omega[S, k] \subseteq \Omega[S, k']$.

3. $\Omega[S, k] \supseteq \Omega[S', k]$.

Proof. Statements 1 can be verified directly by definition. For 2, give a fixed set S, we have

$$T \in \Omega[S, k] \Longrightarrow |T \cap S| \geq |S| - k \geq |S| - k' \Longrightarrow T \in \Omega[S, k'].$$

For 3, we know $|S'| = |S' \setminus S| + |S|$ since S' is a superset of S, and $|T \cap S'| \leq |T \cap S| + |S' \setminus S|$ because $T \cap S'$ contains at most everything in $S' \setminus S$ besides $T \cap S$. Then, we have

$$\begin{aligned} & T \in \Omega[S', k] \\ \Longrightarrow\ & |T \cap S'| \geq |S'| - k \\ \Longrightarrow\ & |T \cap S| + |S' \setminus S| \geq |S' \setminus S| + |S| - k \\ \Longrightarrow\ & |T \cap S| \geq |S| - k \\ \Longrightarrow\ & T \in \Omega[S, k]. \end{aligned}$$

This completes the proof. □

4.2 An Exact Verifier for $\mathcal{P}_\emptyset$

When focusing on null property $\mathcal{P}_\emptyset$, the following definition characterizes the corresponding set of $\mathcal{P}_\emptyset$-structures from a different perspective.

Definition 7 (Critical Nucleus)**.** A set S is a *critical nucleus* if for every $T \in (\uparrow \Omega)[S]$, there exists some $T' \in \hat{\Omega}$ such that $T' \subseteq T$.

By definition, $(\uparrow \Omega)[S]$ is an extended subposet. So, a set S is a critical nucleus if every superset of S in Ω contains some critical structures.

Theorem 3. *Given* $\hat{\Omega} \neq \emptyset$*,* $S \in \hat{\Omega}_{\mathcal{P}_\emptyset}$ *if and only if* S *is a critical nucleus.*

Proof. For sufficiency, towards a contradiction, suppose $S \notin \hat{\Omega}_{\mathcal{P}_\emptyset} = \mathcal{C}(\hat{\Pi}_0)$, then there exists some feasible strategy $U \in \hat{\Pi}_0$ such that $S \cap U = \emptyset$. By definition of $\mathcal{P}_\emptyset$, such U intersects all critical structures in $\hat{\Omega}$ but not all structures in Ω for otherwise, $U \in \Pi_1$. Hence , we can pick some $T \in \{T' \in \Omega \mid T' \cap U = \emptyset\}$, and we have $(T \cup S) \cap U = \emptyset$. Clearly, $T \cup S \in (\uparrow \Omega)[S]$, then the fact that S is a critical nucleus implies that we can find some critical structure $T' \in \hat{\Omega}$ such that $T' \subseteq T \cup S$. By construction, $T' \cap U = \emptyset$, which implies U is not a feasible strategy, a contradiction.

For necessity, suppose S is not a critical nucleus, i.e., there is some $T \in (\uparrow \Omega)[S]$ such that for all $T' \in \hat{\Omega}$, $T' \not\subseteq T$. Then, take $U = \bigcup_{T' \in \hat{\Omega}} T' \setminus T$, U intersects all the critical structures in $\hat{\Omega}$ hence is a feasible strategy. Moreover, T is from $(\uparrow \Omega)[S]$ implies T includes some structure $T_0 \in \Omega$, and $U \cap T_0 = \emptyset$ by construction. Thus, $\mathcal{P}_\emptyset(U) = 0$ as U does not block all the structures. So, we have $U \in \hat{\Pi}_0$ and $U \cap S = \emptyset$, since $S \subseteq T$ by our choice. This implies S is not a cut of $\hat{\Pi}_0$, thus is not a $\mathcal{P}_\emptyset$-structure. □

A direct implication of this theorem is the following corollary.

Corollary 3. *For every regular* $\mathcal{P}$*,* $\hat{\Omega}_{\mathcal{P}}$ *contains all the critical nuclei.*

Proof. Every regular $\mathcal{P}$ is stronger than $\mathcal{P}_\emptyset$. Thus, by Corollary 2, we have $\hat{\Omega}_{\mathcal{P}_\emptyset} \subseteq \hat{\Omega}_{\mathcal{P}}$. □

For every regular bipartition property $\mathcal{P}$, the above theorem and corollary provide a different (partial) characterization of the $\mathcal{P}$-structures that does not depend on the feasible solutions space $\hat{\Pi}$ of the MCS formulation. This enables us to develop verifiers based on this new characterization. In particular, we have the following exact verifier for $\mathcal{P}_\emptyset$-structures.

Theorem 4. *A set S is a critical nucleus if and only if the following formulation*

$$\max_{T\in m(\Omega)} \min_{\substack{T'\subseteq T\cup S\\ T'\in m(\Omega)}} w(T') \tag{4}$$

has an optimal value that is less than r.

Proof. It is sufficient to show that S is a critical nucleus if and only if for every $T \in m(\Omega)$, $T \cup S$ contains some critical structure $T' \in \hat{\Omega}$. The necessity is obvious as the set $T \cup S$ for some $T \in m(\Omega)$ belongs to $(\uparrow \Omega)[S]$. For sufficiency, we prove the contrapositive. Suppose S is not a critical nucleus, i.e., there exists some $T \in (\uparrow \Omega)[S]$ such that every defender's structure T' contained in $T \cup S$ is non-critical. Notice such T' always exists as we pick T from $(\uparrow \Omega)$. Moreover, either T' is minimal in Ω, or some subset of T' is. Without loss of generality, assume $T' \in m(\Omega)$, then by choice, $T' \cup S$ does not contain any critical structure, which completes the proof. □

According to this theorem, Formulation (4) provides an exact verifier for the $\mathcal{P}_\emptyset$-structures. Since (4) is a bilevel problem, optimization techniques such as dualize-and-merge and branch-and-bound can be applied to verify a given S exactly. We provide below several examples to illustrate this method.

Example 1 (Spanning Tree Interdiction, Cont.)**.** In this problem, $m(\Omega)$ is simply the set of spanning trees. Thus, we can use the established spanning tree formulations [35] to represent the constraints of Formulation (4). For the outer level problem, we can use the following constraints to represent the space of spanning trees. We assume $|V| = n$ and use $x = (x_e)_{e\in E}$ as the indicator vector for the selected spanning tree.

$$\sum_{e\in E} x_e = n - 1 \tag{5a}$$

$$\sum_{e\in L} x_e \le |L| - 1, \quad \forall L \subseteq V \tag{5b}$$

$$x_e \in \{0, 1\}, \quad \forall e \in E. \tag{5c}$$

The inner level problem of (4) is to solve the minimum spanning tree problem over the subgraph induced by the edge set $T \cup S$. Since the goal is to dualize the inner problem, we choose a compact linear formulation from [35] as follows,

$$\min \quad \sum_{e\in E} w_e y_e \tag{6a}$$

$$\sum_{\{j:\{i,j\}\in E\}} f^l_{ij} - \sum_{\{j:\{i,j\}\in E\}} f^l_{ji} = b^l_i \quad \forall i \in V,\, l \in V \setminus \{k\} \tag{6b}$$

$$y_e \ge f^l_{ij} + f^l_{ji} \qquad \forall e = \{i, j\} \in E,\, l \in V \setminus \{k\} \tag{6c}$$

$$\sum_{e\in E} y_e = n - 1 \tag{6d}$$

$$f^l_{ij}, f^l_{ji} \ge 0 \qquad \forall \{i, j\} \in E,\, l \in V \setminus \{k\} \tag{6e}$$

$$y_e \ge 0 \qquad \forall e \in E, \tag{6f}$$

where $(y_e)_{e\in E}$ is the indicator vector of the spanning tree to be selected. In this setting, a root vertex k is arbitrarily selected from V, and a commodity l is defined for every non-root vertex $l \in V \setminus \{k\}$. Then, one unit of each commodity l emanating from the root vertex k must be delivered to the vertex l. Thus, the parameter b_i^l takes values of 1, −1, or 0 whenever $i = k$, $i = l$, or $i \in V \setminus \{k, l\}$, respectively. Finally, we use the following linking constraints to combine formulations (5) and (6),

$$y_e \leq x_e + \mathbb{1}_S(e) \quad \forall e \in E, \tag{7}$$

where $\mathbb{1}_S$ is the indicator function of the input edge set S. Thus, an edge e can be used in the inner problem if either it has been selected as a part of the spanning tree by x_e or if it belongs to the input set S. Then, dualizing the inner problem produces a one-level mixed-integer problem (MIP). By Theorem 4, S is a critical nucleus if and only if this MIP formulation obtains an upper bound that is less than r. △

Example 2 (Shortest Path Interdiction, Cont.)**.** In this setting, the outer problem of (4) needs to impose a solution space of paths, and the inner problem is to solve for a shortest path on the subgraph restricted on $T \cup S$. Thus, we can use the following bilevel formulation.

$$\begin{aligned} \max_{x \in \mathcal{X}} \ \min \ & \sum_{e \in E} w_e y_e \\ & \sum_{e \in \delta^+(i)} y_e - \sum_{e \in \delta^-(i)} y_e = b_i, \quad \forall i \in V \\ & y_e \leq x_e + \mathbb{1}_S(e), \quad \forall e \in E \\ & y_e \geq 0, \quad \forall e \in E, \end{aligned}$$

where b_i takes values 1, −1, and 0 whenever $i = s$, $i = t$, and $i \in V \setminus \{s, t\}$, and $\mathcal{X}$ is the space of indicator vectors of paths as follows,

$$\begin{aligned} \sum_{e \in \delta^+(i)} x_e - \sum_{e \in \delta^-(i)} x_e = b_i, \quad \forall i \in V \\ x_e \in \{0, 1\}, \quad \forall e \in E. \end{aligned}$$

We make it an integer to prevent the inner problem from potentially using fractional edges. Again, we can dualize the inner problem to produce a one-level MIP serving as an exact verifier for the $\mathcal{P}_\emptyset$-structures in this specific problem. △

Example 3 (Vertex Cover Interdiction, Cont.)**.** Let $N(v)$ and $N[v]$ denote the open and closed neighbors of a vertex $v \in V$ in the given graph G, i.e., $N[v] = N(v) \cup \{v\}$, a minimal vertex cover can be exactly characterized by the following.

Proposition 4. *Given $G = (V, E)$, a vertex cover $T \subseteq V$ is minimal if and only if*

$$|T \cap N[v]| \leq |N(v)|$$

for every $v \in V$.

Proof. It is known that the complement of a vertex cover is an independent set. Thus, T is a minimal vertex cover if and only if its complement $\bar{T}$ is a maximal independent set. According to [34], an independent set $\bar{T}$ is maximal if and only if $N[v] \cap \bar{T} \neq \emptyset$, which completes the proof. □

Using this characterization, we can use the following constraints to describe the space of minimal vertex covers and denote it as $\mathcal{X}$.

$$x_v + x_u \geq 1, \quad \forall (v,u) \in E \tag{8a}$$

$$\sum_{u \in N[v]} x_u \leq |N(v)|, \quad \forall v \in V \tag{8b}$$

$$x_v \in \{0,1\}, \quad \forall v \in V. \tag{8c}$$

Accordingly, we can set up the following bilevel formulation.

$$\max_{x \in \mathcal{X}} \; \min_{y \in \mathcal{X}} \; \sum_{v \in V} w_v y_v \tag{9a}$$

$$y_v \leq x_v + \mathbb{1}_S(v), \quad \forall v \in V. \tag{9b}$$

This verifier is a bilevel integer linear program. We can use various bilevel branch-and-bound methods (e.g., [32, 61]) to solve this problem exactly. △

In general, suppose the solution space $m(\Omega)$ can be represented exactly by a solution space $\mathcal{X}$, then Formulation (4) can always be rewritten as (9) for the corresponding ground set Δ—V in the vertex cover example.

In all three examples, we have developed mathematical formulations to verify $\mathcal{P}_\emptyset$-structures. However, these verifiers are nowhere near practical since they need to be solved multiple times to separate new constraints. In fact, it is easy to see that solving some of these separation problems can be as challenging as solving the original problem.

To overcome such inefficiency, in the following subsection, we shift our focus to the development of partial verifiers that are allowed only to separate a subset of $\hat{\Omega}_{\mathcal{P}_\emptyset}$. To incorporate partial verifiers in Algorithm 1, we replace the MPS formulation with the MCS formulation in the algorithm, then generate supervalid inequalities using any given partial verifier. This variant algorithm searches for an optimal solution from some relaxation of Π_0, then compare this solution with the optimal solution obtained in Π_1. Hence, the overall correctness is still guaranteed.

4.3 A Partial Verifier for $\mathcal{P}_\emptyset$

Based on Theorem 3, in this section, we will provide two properties about the subsets of the ground set. We will prove that a structure $S \subset \Delta$ that holds both properties is guaranteed to be a $\mathcal{P}_\emptyset$-structure. This will lead to a more efficient procedure to identify $\mathcal{P}_\emptyset$-structures. The first concept is called *partial nuclei.*

Definition 8 (Partial Nucleus)**.** A set S is a partial nucleus of degree k, or a k-nucleus, if all elements in $m(\Omega)[S,k]$ are critical, i.e., $m(\Omega)[S,k] \subseteq \hat{\Omega}$. The set of k-nuclei is denoted by $\hat{\Omega}^c_k$.

Recall $m(\Omega)[S,k] = \{T \in m(\Omega) \mid |T \cap S| \geq |S| - k\}$ by Definition 6 for some $k \in \mathbb{N}$. In particular, if S is a 0-nucleus, then all minimal structures that contain S are critical, i.e., $m(\Omega)[S] \subseteq \hat{\Omega}$. Also, S is a k-nucleus for some $k \geq |S|$ if all minimal structures $m(\Omega)$ are critical, in which case all feasible solutions are in Π_1. By definition, we clearly have the following.

Proposition 5. *$\hat{\Omega}^c_k$ is an upper set and is decreasing on k. That is, for $k, k' \in \mathbb{N}$ such that $k \leq k'$, we have $\hat{\Omega}^c_k \supseteq \hat{\Omega}^c_{k'}$.*

Proof. For the first claim, for some $S \in \hat{\Omega}^c_k$ and any $S' \supseteq S$, we have $m(\Omega)[S', k] \subseteq m(\Omega)[S, k] \subseteq \hat{\Omega}$ by Proposition 3. Thus, $S' \in \hat{\Omega}^c_k$. For the second statement, $S \in \hat{\Omega}^c_{k'}$ means $m(\Omega)[S, k'] \subseteq \hat{\Omega}$. By Proposition 3, $k \leq k'$ implies

$$m(\Omega)[S, k] \subseteq m(\Omega)[S, k'] \subseteq \hat{\Omega}.$$

That is, $S \in \hat{\Omega}^c_k$. □

There are two reasons why the membership of a k-nucleus is much easier to verify than the critical nuclei. First, minimal elements $m(\Omega)$ are follower's structures and thus characterized by the follower's problem, whereas the elements in $m(\uparrow \Omega[S])$ can be arbitrary supersets of the follower's structures. Second, we can check whether any upper bound of the following

$$\max_{T \in m(\Omega)[S,k]} w(T) \tag{10}$$

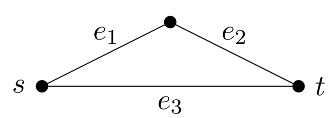
is less than the interdiction target r to verify a k-nucleus instead of solving a bilevel problem, as it is required to verify a critical nucleus. Unfortunately, being a k-nucleus does not suffice to guarantee $S \in \hat{\Omega}_{\mathcal{P}_\emptyset}$. The following proposition investigates the relationship between k-nuclei and critical nuclei.

Proposition 6. $\hat{\Omega}_{\mathcal{P}_\emptyset} \subseteq \hat{\Omega}^c_0$. *Furthermore, there exist binary interdiction games where this inclusion is strict.*

Proof. Notice that when $k = 0$, we have $m(\Omega)[S, k] = m(\Omega)[S]$.

To prove the first statement, for every $S \in \hat{\Omega}_{\mathcal{P}_\emptyset}$, consider the set $m(\Omega)[S]$. This set is either empty or not. For the former, we have $\emptyset \subseteq \hat{\Omega}$, which makes S a 0-nucleus. For the latter, for every $T \in m(\Omega)[S]$, we have $T \in (\uparrow \Omega)[S]$. Because S is a critical nucleus, by definition, T must contain some critical $T' \subseteq T$. But, T' must be taken as T; otherwise, it will contradict the minimality of T in Ω. This shows $m(\Omega)[S] \subseteq \hat{\Omega}$, i.e., $S \in \hat{\Omega}^c_0$.

For the second statement, we construct the following instance of the shortest path interdiction game as a counterexample.



We assume $\hat{\Omega} = \{\{e_1, e_2\}\}$. Clearly, $\{e_1\}$ is a 0-nucleus since $m(\Omega)[\{e_1\}] = \{\{e_1, e_2\}\} = \hat{\Omega}$. However, $\{e_1\}$ is not a critical nucleus since taking T as the path $\{e_3\}$, the union $T \cup \{e_1\}$ does not contain any critical paths. □

By this proposition, Formulation (10) is not a valid partial verifier as it may accept "false positive" structures. To resolve this issue, we use the following property.

Definition 9 (Regenerability)**.** A set S is k-regenerable in Ω, if for all $T \in \Omega$, there is a $T' \subseteq T \cup S$ such that $T' \in m(\Omega)[S, k]$. The family of k-regenerable sets is denoted by $\hat{\Omega}^r_k$.

Intuitively, a set S is k-regenerable if we combine it with any structure $T \in \Omega$ by the union will regenerate a minimal structure $T' \in m(\Omega)$ that intersects S with at least $|S| - k$ elements. In particular, 0-regenerable implies $S \cup T$ always regenerates some minimal structure T' that contains S as a subset, which is quite a strong condition. On the other hand, every set S is k-regenerable for all $k \geq |S|$ since $S \cup T$ must contain some minimal structure (either T or some minimal structure contained in T). This implies, in contrast to the k-nuclei, $\hat{\Omega}^r_k$ is increasing on k. To be exact, we have the following property of $\hat{\Omega}^r_k$.

Proposition 7. *$\hat{\Omega}_k^r$ is increasing on k. Moreover, for every $S \in \hat{\Omega}_k^r$ and $S' \supseteq S$, we have $S' \in \hat{\Omega}_{k+|S'|-|S|}^r$.*

Proof. To prove the first, we take any $S \in \hat{\Omega}_k^r$. By definition, for every $T \in \Omega$, there is some $T' \subseteq T \cup S$ such that $T' \in m(\Omega)[S,k]$. By Proposition 3, we have $m(\Omega)[S,k] \subseteq m(\Omega)[S,k']$ for $k' \geq k$. Thus, the same T' also exists in $m(\Omega)[S,k']$. By definition, $S \in \hat{\Omega}_{k'}^r$. For the second statement, for each $T \cup S'$, we can take the same T' that is regenerated by $T \cup S$, which intersects T' with at least

$$|S| - k = |S'| - (k + |S'| - |S|)$$

elements, which completes the proof. □

Combining both properties, we get the following verification theorem.

Theorem 5 (Partial Verification Theorem)**.** *$\hat{\Omega}_k^c \cap \hat{\Omega}_k^r \subseteq \hat{\Omega}_{\mathcal{P}_\emptyset}$ for all $k \in \mathbb{N}$.*

Proof. Given a set $S \in \hat{\Omega}_k^c \cap \hat{\Omega}_k^r$ for some k, suppose $S \notin \hat{\Omega}_{\mathcal{P}_\emptyset}$, i.e., S is not a cut of the corresponding $\hat{\Pi}_0$. Then, there is some feasible strategy $U \in \hat{\Pi}_0$ that does not intersect S. Because $\mathcal{P}_\emptyset(U) = 0$, by the definition of null property, some structure $T \in \Omega$ is not blocked by U. Thus, we have $(S \cup T) \cap U = \emptyset$. Then, S is k-regenerable in Ω implies there is some $T' \subseteq S \cup T$ such that $T' \in m(\Omega)[S,k]$. Moreover, S is a k-nucleus implies such T' must be critical. Note $U \cap T' = \emptyset$ by construction, which contradicts the feasibility of U. So, we are done. □

Corollary 4. *For any $k \in \mathbb{N}$, denote $\hat{\Omega}_k^{cr} := \hat{\Omega}_k^c \cap \hat{\Omega}_k^r$ and define $\hat{\Omega}^{cr} := \bigcup_{k\in\mathbb{N}} \hat{\Omega}_k^{cr}$. Then, $\hat{\Omega}^{cr} \subseteq \hat{\Omega}_{\mathcal{P}_\emptyset}$.*

We omit the proof as it is a trivial consequence of Theorem 5. In particular, this corollary provides an upper bound of $\hat{\Omega}^{cr}$. On the other hand, the following proposition provides a lower bound, which can be verified directly from the definitions.

Proposition 8. *Every critical structure $T \in m(\hat{\Omega})$ is a 0-nucleus and is 0-regenerable in Ω. In particular, $m(\hat{\Omega}) \subseteq \hat{\Omega}_0^{cr} \subseteq \hat{\Omega}^{cr}$.*

The relationship between all these sets forms a hierarchy map illustrated in Figure 6. From a practical perspective, Theorem 5 enables us to split the verification procedure into two parts. Given a set S, we only need to check: (i) whether it is a k-nucleus using Formulation (10) or any equivalent algorithms, (ii) whether it is k-regenerable for certain k between 0 and $|S|$. For the latter, we can often circumvent the verification step by starting with structures that are known to be k-regenerable. We provide several examples in the next subsection to illustrate these ideas.

4.4 Examples for the Partial Verifier

Given a set S, we can use the following general formulation to determine the smallest k required for S to be k-regenerable.

$$k = |S| - \min_{T \in m(\Omega)} \max_{\substack{T' \subseteq S \cup T \\ T' \in m(\Omega)}} |S \cap T'|.$$

However, this is again a bilevel optimization similar to the exact verifier (4). In practice, we can often skip this verification by identifying k-regenerable structures directly. Then, applying partial nucleus verifier (10) on these regenerable structures to check whether S is a $\mathcal{P}_\emptyset$-structure.

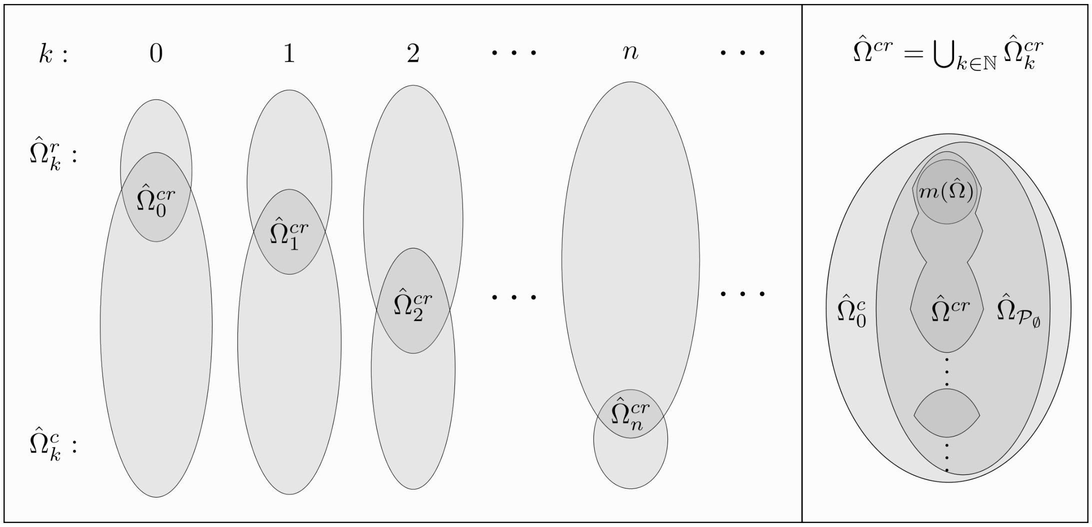


Figure 6: Hierarchy of Partial Nuclei $\hat{\Omega}^c_k$ and Regenerable Sets $\hat{\Omega}^r_k$. The left figure shows that $\hat{\Omega}^c_k$ and $\hat{\Omega}^r_k$ are decreasing and increasing on k, respectively. In the right figure, their intersections $\hat{\Omega}^{cr}_k$'s are combined into the set $\hat{\Omega}^{cr}$, which is sandwiched between $m(\hat{\Omega})$ and $\hat{\Omega}_{\mathcal{P}_\emptyset}$. In addition, the set of 0-nuclei contains all the critical nuclei $\hat{\Omega}_{\mathcal{P}_\emptyset}$ as a subset.

Example 1 (Spanning Tree Interdiction, Cont.)**.** In this case, every edge set S that does not contain a cycle is 0-regenerable due to the edge exchange property of spanning trees. Therefore, we can start with a critical spanning tree $T \in \hat{\Omega}$, then verify its proper subsets using any implementation of the partial nucleus verifier (10). This verifier computes the maximum weight of the spanning trees that contain S. A variant of Kruskal's algorithm has been implemented in [53]. In the same paper, the authors demonstrated that, with this $\mathcal{P}$-structure verifier for separating Constraint (3b), the resulting MPS formulation works more efficiently than the MCS formulation. △

Example 2 (Shortest Path Interdiction, Cont.)**.** In contrast to the minimum spanning tree case, not every subset of a s-t path is 0-regenerable. To identify the regenerable sets in this problem, we need the following definitions.

Definition 10 (Path Decomposition & Skeletons)**.** Given a graph $G = (V, E)$ and a source-terminal pair $s, t \in V$, a *path decomposition* is a sequence of vertex subsets $\{X_i\}_{i=0}^n$ with three properties:

- $s \in X_0$ and $t \in X_n$;
- for every $\{u, v\} \in E$, there exists some i such that $u, v \in X_i$;
- for every $i \le j \le k$, $X_i \cap X_k \subseteq X_j$.

For each X_i, the *source set* S_i and *terminal set* T_i are defined as

$$S_i = \begin{cases} \{s\}, & i = 0 \\ X_i \cap X_{i-1}, & \text{otherwise.} \end{cases} \qquad T_i = \begin{cases} \{t\}, & i = n \\ X_i \cap X_{i+1}, & \text{otherwise.} \end{cases}$$

For a set of vertices X, we define $E(X) := \{\{u, v\} \in E \mid u, v \in X\}$. Then, for any part X_i such that $S_i \cap T_i = \emptyset$, a *skeleton* K of X_i is a minimal edge set that preserves the connectivity between S_i and T_i, i.e., for every part $(s, t) \in S_i \times T_i$ that is connected through edges in $E(X_i) \setminus (E(S_i) \cup E(T_i))$,

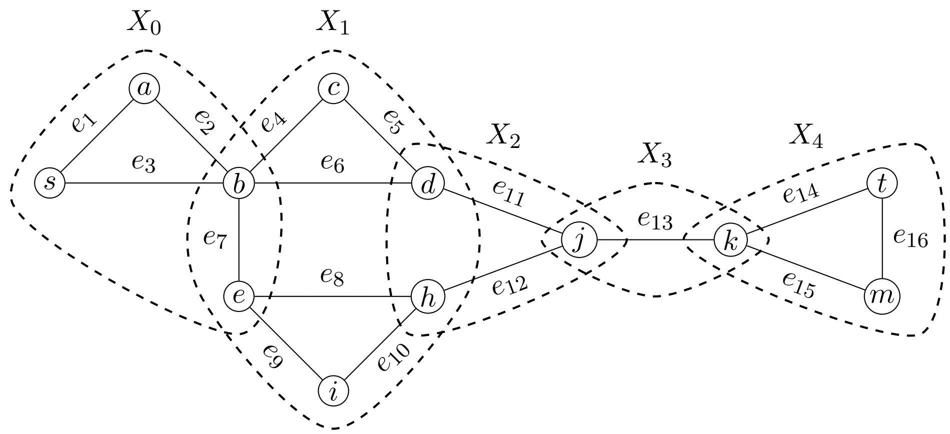


Figure 7: Path decomposition and skeletons. All the vertices are covered by the path decomposition $\{X_i\}_{i=0}^4$. For X_1, the source set S_1 and terminal set T_1 are $\{b, e\}$ and $\{d, h\}$, respectively. $K_1 = \{e_6, e_8\}$, $K_2 = \{e_4, e_5, e_9, e_{10}\}$ are both skeletons in X_1 since both are minimal edge sets that preserve all the connectivities between S_1 and T_1. By definition, we have $\lambda(K_1) = 1$ and $\lambda(K_2) = 2$. In contrast, $\{e_6\}$ and $\{e_4, e_5, e_6, e_8\}$ are not skeletons as the former breaks the connectivity between e and h while the latter is not minimal. Another example, for X_0, S_0 is $\{s\}$ and $T_0 = \{b, e\}$. Both $K_3 = \{e_1, e_2\}$ and $K_4 = \{e_3\}$ are skeletons with $\lambda(K_3) = 2$ and $\lambda(K_4) = 1$, while $\{e_3, e_7\}$ is not since it contains edges in T_0, thus is not minimal.

there is a path in K that connects them, and K is a minimal edge set that achieves this. Let $\mathcal{P}_K$ be all such paths in K, then we define

$$\lambda(K) := \min_{P \in \mathcal{P}_K} |P|,$$

i.e., the length of the shortest path (in terms of the number of edges) in K that connects S_i to T_i.

This definition is a variant of the path decomposition in [43], and we provide an example in Figure 7 for illustration. The intuition is that a path decomposition reduces the given graph G into a simple s-t path on the aggregated level. In particular, for every $i \le j \le k$, a path from some $u \in X_i$ to some $v \in X_k$ must take some subpath in X_j with a source vertex in S_i and a terminal vertex in T_i. This decomposition is related to regenerable sets by the following proposition.

Proposition 9. *In the shortest path interdiction, for an arbitrary path decomposition, every skeleton K induces a $(|K| - \lambda(K))$-regenerable set.*

Proof. For every s-t path P and a skeleton K with respect to a part X_i under some path decomposition, the subpath of P that passes through some X_i such that $S_i \cap T_i = \emptyset$ must connect some vertex $u \in S_i$ to some vertex in $v \in T_i$. Swapping this subpath with the unique (u, v)-path in K produces a new path from s to t. Moreover, the (u, v)-path in K has a length that is greater or equal to $\lambda(K)$. Hence, the new path intersects with K in at least $\lambda(K) = |K| - (|K| - \lambda(K))$ edges, which proves the claim. □

For instance, in Figure 7, $\{e_1, e_2\}, \{e_3\}, \{e_{13}\}$ are all 0-regenerable; $\{e_6, e_8\}$ and $\{e_{11}, e_{12}\}$ are both 1-regenerable; $\{e_4, e_5, e_8\}$ and $\{e_4, e_5, e_9, e_{10}\}$ are both 2-regenerable. Notice that some regenerable sets (e.g., $\{e_{11}, e_{12}\}$ or $\{e_{13}\}$) are uninteresting since they are k-nucleus if and only if all paths are critical. In general, it is easy to see that every s-t cut S is $(|S| - 1)$-regenerable. Yet, they are all trivial regenerable sets for the same reason.

Using this definition, we can randomly construct several path decompositions in this problem, then generate non-trivial skeletons from some parts. For each skeleton K, we can run Formulation

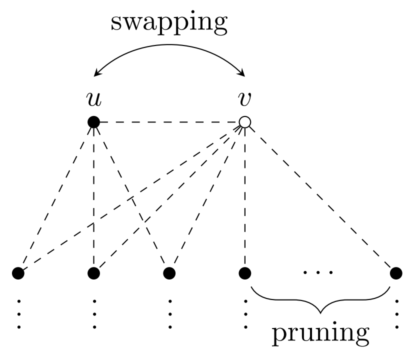


Figure 8: Swapping and pruning operations on the sociable vertex v.

(10) or any equivalent variant of the longest path algorithm to intersect K with at least $|K| - \lambda(K)$ edges. Suppose the resulting length is less than the predefined r, we separate a constraint of (3b) that is associated with K. We also note that these algorithms can be terminated early once any upper bound is found to be less than r. △

Example 3 (Vertex Cover Interdiction, Cont.)**.** Similar to the shortest path case, not every subset of a vertex cover is 0-regenerable. For instance, $T_1 = \{v_1, v_2, v_4, v_5\}$ and $T_2 = \{v_1, v_2, v_3, v_6\}$ are both minimal vertex covers in Figure 1. Take $\{v_5\} \subseteq T_1$, but $T_2 \cup \{v_5\}$ does not contain any minimal vertex cover that contains $\{v_5\}$ since all the neighbors of v_4 (i.e., elements in T_2) have to be included in a minimal vertex cover that excludes v_4. The following definition and lemma identify a certain type of vertex that is 0-regenerable. Recall that, for a vertex $v \in V$, we use $N(v)$ and $N[v]$ for the open and closed neighbors of v in the graph.

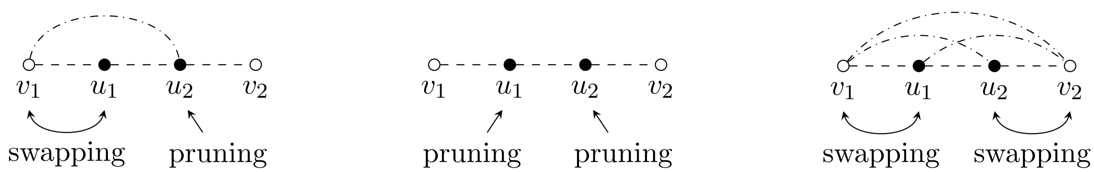
Definition 11 (Sociable Vertices)**.** In a simple graph $G = (V, E)$, a vertex $v \in V$ is called *sociable* if there exists $u \in N(v)$ such that $N[u] \subseteq N[v]$.

Lemma 1. *Every singleton* $\{v\}$ *for some sociable vertex* v *is* 0*-regenerable.*

Proof. Pick any minimal vertex cover $T \in m(\Omega)$, either $v \in T$ or not. The former is trivial as we can pick $T' = T \subseteq T \cup \{v\}$. Suppose otherwise, we have $N(v) \subseteq T$ to cover all the edges between v and its neighbors. Take one vertex $u \in N(v)$ such that $N[u] \subseteq N[v]$, we swap u with v to construct $T_0 = T \cup \{v\} \setminus \{u\}$. We will show that (i) T_0 is a vertex cover and (ii) it contains some minimal vertex cover T' that also contains v. For (i), notice that by this *swapping* operation (see Figure 8), only the coverage of edges $\{v\} \times N(v)$ and $\{u\} \times N(u)$ has been changed. We partition these edges into three parts,

1. (u, v);
2. $\{v\} \times (N(v) \setminus \{u\})$;
3. $\{u\} \times (N(u) \setminus \{v\})$.

This is clearly a partition. For 1, after swapping u and v, the edge $\{u, v\}$ is still covered; for 2, note that $N(v) \setminus \{u\} \subseteq T$ since $v \notin T$, and swapping u and v would not change this containment, i.e., $N(v) \setminus \{u\} \subseteq T_0$. Thus, all the edges in Case 2 are covered by T_0. Case 3 is trivial as $N[u] \subseteq N[v]$. To prove (ii), we use Proposition 4. That is, we will construct some T' that satisfies $v \in T' \subseteq T_0$ such that, for each $v \in V$, there exists some $v' \in N[v]$ does not belong to T'. Because only the



(a) Case 1 (b) Case 2 (c) Case 3

Figure 9: Three cases in Proposition 10. The dash-dotted edges exist due to $N[u_i] \subseteq N[v_i]$, which is implied by the corresponding swapping operations.

vertices in $N[v] \cup N[u]$ have been affected by the swapping operation, we partition these vertices as follows,

1. $\{u, v\}$;
2. $N(u) \setminus \{v\}$;
3. $N(v) \setminus N[u]$, which is empty if $N[v] = N[u]$.

This is a valid partition (that may contain empty sets) since $N[u] \subseteq N[v]$. For Case 1, both u and v satisfy the characterization as $u \notin T_0$. All the vertices in Case 2 also satisfy the characterization as they are neighbors of u. For Case 3, suppose $N(v) \setminus N[u]$ is nonempty (otherwise we are done), we can sort $N(v) \setminus N[u]$ in an arbitrary order. Then, we perform a *pruning* operation on $N(v) \setminus N[u]$ sequentially. Take each element v' from this set in order, either $N[v'] \subseteq T_0$ or not. For the latter, we do nothing and go to the next element in the set. For the former, we remove v' from T_0. Notice that $T_0 \setminus \{v'\}$ is still a vertex cover, since removing v' only affects its neighbors. But, the reason to remove v' is that all its neighbors are covered in T_0, which implies all the edges $\{v'\} \times N(v')$ are still covered by $T_0 \setminus \{v'\}$. Clearly, after scanning all the elements in $N(v) \setminus N[u]$, we are left with a set T' where every vertex satisfies the characterization, and T' is preserved all along as a vertex cover. Moreover, since the only operation is removing vertices in T_0 other than v, we have $v \in T' \subseteq T_0$, which implies v is 0-regenerable. □

However, a 0-regenerable singleton T is often weak in its quality since the set $\Omega[T]$ could be quite large, which is unlikely to be contained in $\hat{\Omega}$. The following proposition shows that these sociable vertices can be merged to a larger 0-regenerable set as long as they keep a certain distance from each other.

Proposition 10. *A set of sociable vertices $U \subseteq V$ is 0-regenerable if it is also an independent set.*

Proof. For every minimal vertex cover T and every sociable vertex $v \in U$, the swapping and pruning operations, if performed, will only add v into T and remove some neighbors in $N(v)$ from T (including the swapped u). Thus, we only need to show that, after sequentially performing the two operations on multiple sociable vertices in U, one of the two endpoints of every edge is still covered by the resulting T'. Since all the operations within each neighbor are safe by Lemma 1, the only possibility to cause some edge to be uncovered is that, for some $v_1, v_2 \in U$ such that $v_1 \neq v_2$, one vertex $u_1 \in N(v_1)$ removed from T along with some $u_2 \in N(v_2)$ removed from T form an edge (u_1, u_2). More specifically, we have the following three possible cases (see Figure 9):

1. without loss of generality, u_1 and u_2 are removed from T due to swapping and pruning operations, respectively;

2. both u_1 and u_2 are removed from T due to pruning operations;

3. both u_1 and u_2 are removed from T due to swapping operations.

We will show all three cases are impossible to occur. Consider we scan each $v \in U \setminus T$ in a given order and perform the corresponding swapping and pruning operations in each iteration. First, these operations will not remove any $U \cap T$ from T, because these operations only affect the neighbors of v while the minimum distance between vertices in U is at least two by the assumption that U is an independent set.

Then, for Case 1, either the swapping operation on v_1 is performed before or after the pruning operation on $N(v_2)$. For the former, after the swap, u_1, as a neighbor of u_2, has been removed from T. Thus, u_2 will not be pruned after as not all its neighbors are selected. For the latter, since v_1 is sociable and u_1 will be swapped with v_1, we have $N[u_1] \subseteq N[v_1]$ by definition. Clearly, the edge (u_1, u_2) is adjacent to u_1, thus we also have $u_2 \in N[v_1]$, i.e., $v_1 \in N(u_2)$. Before the swap, we further have $v_1 \notin T$. Thus, u_2 will not be removed from T by the pruning process as not all the neighbors of u_2 are selected (v_1 is not). This completes Case 1.

For Case 2, no matter which pruning operation happens first (say u_1), the other pruning operation will not be carried out since its neighbor u_1 is not selected in the constructed vertex cover.

For Case 3, since u_1 and u_2 in T will be swapped with v_1 and v_2, respectively. We have $N[u_i] \subseteq N[v_i]$ for $i = 1, 2$. In particular, $u_1 \in N[v_2]$ and $u_2 \in N[v_1]$. Moreover, v_1 is also a neighbor of u_2, so we have

$$v_1 \in N[u_2] \subseteq N[v_2],$$

which contradicts to the minimum distance between v_1 and v_2.

Therefore, all three cases are impossible to occur, which implies all the procedures of scanning $v \in U \setminus T$ and performing the corresponding swapping and pruning operations will never lead to any conflicts. Then, by Lemma 1, after the entire process, the resulting T' is still a vertex cover that satisfies the characterization of Proposition 4, thus is a minimal vertex cover that satisfies $U \subseteq T' \subseteq (U \cup T)$. This completes the proof. □

Accordingly, in this problem, we can first identify all (or part of) the sociable vertices U, then pre-compute their pairwise distance. Then, for every $U' \subseteq U$ that satisfies the distance requirement, we can run the following 0-nucleus verifier

$$\begin{aligned} \max_{x \in \mathcal{X}} \; & \sum_{v \in V} w_v x_v \\ & \text{(8a)–(8c)}, \\ & x_v = 1, \quad \forall v \in U'. \end{aligned}$$

Suppose the optimal value or any upper bound is less than the predefined target r, then U' is a $\mathcal{P}_\emptyset$-structure. Compared to the bilevel integer program (9), this verifier is easier to solve. △

4.5 General $\mathcal{P}_\emptyset$-Structure Separation Guidelines

In general, for a specific binary interdiction game, we can separate the $\mathcal{P}_\emptyset$-structures using the following general guidelines.

1. Identify the set of follower's structures Ω.

2. Identify a regular bipartition property $\mathcal{P}$ such that the optimal value $z(\Pi_1)$ can be efficiently determined.

3. Identify a class of regenerable sets in the corresponding problem.

4. Solve the MPS formulation (3) using Algorithm 1.

5. At each iteration of the separation step, heuristically select some regenerable set S, then run the corresponding k-nucleus verifier (10) on S. If an upper bound of (10) is obtained to be less than the interdiction target r, then we add the constraint $\sum_{a\in S} x_a \geq 1$ into the MPS formulation.

6. The algorithm terminates when the MPS formulation solves optimally, or its lower bound is no less than $z(\Pi_1)$.

All the previous examples can set up a separation subroutine based on these guidelines with deliberately designed implementations. For instance, in the minimum spanning tree interdiction [54], the authors developed an enumeration tree for the regenerable sets and implemented a binary search on each branch of the enumeration tree to separate the $\mathcal{P}_\emptyset$-structures. Though the regenerable set selection and k-nucleus verifier implementation for different binary interdiction games could be vastly different, a special class of structures, called the *greedoids*, shares similar features as the spanning trees in the $\mathcal{P}_\emptyset$-structure separation procedure.

5 Greedoid Interdiction Games

Previous examples imply that an efficient implementation of the partial verifier requires (i) a set of easy-to-identify regenerable sets and (ii) an efficient algorithm to obtain a tight upper bound for (10). In this section, we will show that, for the special class of problems called the *greedoid interdiction games*, both requirements are satisfied under mild assumptions.

5.1 Preliminaries on Greedoids

A greedoid is an abstract set system that connects algorithms, combinatoric optimizations, and classical analysis of mathematical structures [30]. Intuitively, greedoids extract the fundamental properties of problems where greedy algorithms can produce optimal solutions [57]. It also generalizes many other abstract set systems such as interval greedoids, antimatroids, and matroids. Many applications and research fields are associated with these set systems, including graph theory [51], electric network theory [42], assignment problems [19], game theory [1], semi-Markov process [21], poset analysis [5], and many more. We begin with the following definitions.

Definition 12 (Greedoid/Basis/Rank/Closure/Greedoid Optimization [30, p. 10, 44, 158, 160])**.** A set system $(\Delta, \widetilde{\Omega})$ with $\widetilde{\Omega} \subseteq 2^\Delta$ is called a *greedoid* if it satisfies the following two properties,

- *Accessibility*: for every $T \in \widetilde{\Omega}$ there exists $a \in T$ such that $T \setminus \{a\} \in \widetilde{\Omega}$;

- *Exchange property*: for every $T, T' \in \widetilde{\Omega}$ with $|T| < |T'|$, there is some $a \in T' \setminus T$ such that $T \cup \{a\} \in \widetilde{\Omega}$.

Given a greedoid $(\Delta, \widetilde{\Omega})$, all the maximal elements of $\widetilde{\Omega}$ form the set of *bases* denoted by Ω. The *rank function* $r : 2^\Delta \to \mathbb{N}$ and *closure operator* $\sigma : 2^\Delta \to 2^\Delta$ are defined as

$$r(S) = \max\left\{|T| \mid T \subseteq S,\ T \in \widetilde{\Omega}\right\},$$
$$\sigma(S) = \{a \in \Delta \mid r(S \cup \{a\}) = r(S)\}.$$

Algorithm 2: Greedy Algorithm for Greedoids [30, p. 160]

Data: input $\Delta, \widetilde{\Omega}, w(\cdot)$

$T \leftarrow \emptyset$ `// start with the minimal structure in` $\widetilde{\Omega}$

while $\Gamma(T) \neq \emptyset$ **do**

  $a^\star \leftarrow \max_{a \in \Gamma(T)} w(a)$ `// select a greedy choice`

  $T \leftarrow T \cup \{a^\star\}$

end

return T

A set S is *closed* if $\sigma(S) = S$. We say $\widetilde{\Omega}$ has the *strong exchange property* if for every $S \subseteq T \in \Omega$ with $S \in \widetilde{\Omega}$ and every $a \in \Delta \setminus B$ with $S \cup \{a\} \in \widetilde{\Omega}$, there exists some $b \in T \setminus S$ such that both $S \cup \{b\}$ and $(T \setminus \{b\}) \cup \{a\}$ are in $\widetilde{\Omega}$. The corresponding *greedoid optimization* is defined as

$$\max_{T \in \Omega} \; w(T),$$

where Ω is the set of bases of $\widetilde{\Omega}$ and $w(\cdot)$ is any weight function. We say $w(\cdot)$ is linear if $w(T) = \sum_{a \in T} w(a)$ for some weight assignment of the ground set $w : \Delta \to \mathbb{R}$.

By this definition, it is easy to verify that all bases of a greedoid $\widetilde{\Omega}$ have the same size, which is also called the rank of $\widetilde{\Omega}$. Because of the exchange property, every non-basis element in a greedoid $\widetilde{\Omega}$ can be extended gradually to a basis. That is, for every $T \in \widetilde{\Omega}$, define

$$\Gamma(T) := \{a \in \Delta \setminus T \mid T \cup \{a\} \in \widetilde{\Omega}\}.$$

Then, $\Gamma(T)$ is empty if and only if T is a greedoid basis. This observation enables a natural greedy algorithm (i.e., Algorithm 2) for solving the greedoid optimization problem. The following proposition gives one of the main results regarding the greedy algorithm on greedoids.

Proposition 11 ([30, p. 160])**.** *Let $(\Delta, \widetilde{\Omega})$ be a greedoid with Ω as its bases. Then the following are equivalent:*

- *For every linear objective function $w(\cdot)$, the greedy algorithm is optimal.*
- *$\downarrow \Omega$ is a matroid and every closed set in $(\Delta, \widetilde{\Omega})$ is also closed in $(\Delta, \downarrow \Omega)$.*
- *$(\Delta, \widetilde{\Omega})$ has the strong exchange property.*

5.2 Partial Verifier in Greedoid Interdiction Games

We can naturally associate a binary interdiction game with each greedoid structure $\widetilde{\Omega}$. We call this problem the *greedoid interdiction game* with respect to $\widetilde{\Omega}$ and is defined as follows.

Definition 13 (Greedoid Interdiction Games)**.** Given a greedoid $(\Delta, \widetilde{\Omega})$, the corresponding *greedoid interdiction* is a binary interdiction game where the set of minimal structures consists of all the bases in $\widetilde{\Omega}$.

The following theorem shows that the regenerable sets can be effortlessly identified in greedoid interdiction games.

Theorem 6. *Given a greedoid* $(\Delta, \widetilde{\Omega})$*, every* $S \in \widetilde{\Omega}$ *is* 0*-regenerable in the associated greedoid interdiction game.*

Proof. Let Ω be the set of bases in $\widetilde{\Omega}$. By the exchange property of greedoid, for every feasible set $S \in \widetilde{\Omega}$ and every basis $T \in \Omega$, $S \cup T$ contains some T' that satisfies $T' \supseteq S$, which implies S is 0-regenerable in the associated greedoid interdiction game. □

Therefore, for the binary interdiction game associated with a greedoid $\widetilde{\Omega}$, we can search for some proper subset $S \subseteq T \in m(\hat{\Omega})$ that satisfies $S \in \widetilde{\Omega}$, then run the 0-nucleus verifier (10) over S and accept it as a $\mathcal{P}_\emptyset$-structure if any upper bound of the verifier gets less than the predefined target value r. Moreover, according to the following theorem, suppose that the greedoid $\widetilde{\Omega}$ further satisfies the strong exchange property, then 0-nucleus verifier can be implemented as a greedy algorithm.

Theorem 7. *For the greedoid interdiction game associated with a greedoid* $(\Delta, \widetilde{\Omega})$ *that satisfies the strong exchange property, the corresponding* 0*-nucleus verifier* (10) *with a linear weight function* $w(\cdot)$ *adopts a greedy algorithm that solves for an optimal solution.*

Proof. Let Ω be the set of bases in the greedoid $\widetilde{\Omega}$. Then, the 0-nucleus verifier (10) for the associated greedoid interdiction game can be written as

$$\max_{T \in \Omega[S]} w(T). \tag{11}$$

By Proposition 11, the strong exchange property implies $\downarrow \Omega$ is a matroid and every closed set in $(\Delta, \widetilde{\Omega})$ is also closed in $(\Delta, \downarrow \Omega)$. Fixing partial solution S in the matroid $\downarrow \Omega$ is also called the contraction operation relative to S, which induces the contracted matroid $(\Delta \setminus S, (\downarrow \Omega)/S)$ with

$$(\downarrow \Omega)/S := \{T' \subseteq \Delta \setminus S \mid T' \cup T \in \downarrow \Omega \text{ for some } T \in \mathcal{B}(S)\},$$

where $\mathcal{B}(S)$ are the maximal elements in $\{T \subseteq S \mid T \in \widetilde{\Omega}\}$ (see [30, p. 15]). Thus, (11) is the greedoid optimization problem associated with this contracted matroid. Moreover, the closed sets in the contracted matroid are inherited from the closed sets in $\downarrow \Omega$. This means the contracted matroid satisfies the second condition in Proposition 11. Thus, the greedy algorithm is optimal to solve (11). □

These two theorems demonstrate that the $\mathcal{P}_\emptyset$-structure separation is often efficient for greedoid interdiction games.

6 Conclusion

This paper studies a particular class of supervalid inequalities to solve binary interdiction games. For an arbitrary bipartition of the leader's strategy space associated with a property $\mathcal{P}$, we identified a class of structures called the $\mathcal{P}$-structures, each of which induces a supervalid inequality for the corresponding problem. To design separation methods, we restricted our attention to the class of regular bipartition properties and derived a new characterization for the set of $\mathcal{P}_\emptyset$-structures, which results in an exact verification method. We further defined two special types of structures, the partial nuclei and the regenerable sets, based on which we developed a more efficient partial verifier. The classification of various types of structures also leads to a hierarchy map of the $\mathcal{P}_\emptyset$-structures for binary interdiction games.

We provided three general examples in which we apply our results to solve binary interdiction games targeting shortest paths, spanning trees, and vertex covers. Moreover, the realization of the regenerable sets in these problems reveals interesting network structures, such as skeletons of a path decomposition and sociable vertices in a graph, which may deserve further investigation due to their unique properties.

Finally, we had shown that every feasible set of a greedoid is 0-regenerable in the corresponding greedoid interdiction game. Moreover, if this greedoid also satisfies the strong exchange property, the associated 0-core verification subroutine can be implemented as a greedy algorithm. Therefore, the separation procedure is guaranteed to be efficient in the greedoid interdiction games.

For future work, it is interesting to explore whether other types of interactions between the two players will lead to a new class of supervalid inequalities and whether regenerable sets can be commonly identified in other structure-specific interdiction games.

References

[1] E. Algaba, J. M. Bilbao, R. van den Brink, and A. Jiménez-Losada. Cooperative games on antimatroids. *Discrete Mathematics*, 282(1-3):1–15, 2004.

[2] A. Arulselvan, C. W. Commander, L. Elefteriadou, and P. M. Pardalos. Detecting critical nodes in sparse graphs. *Computers and Operations Research*, 36(7):2193–2200, 2009.

[3] M. O. Ball, B. L. Golden, and R. V. Vohra. Finding the most vital arcs in a network. *Operations Research Letters*, 8(2):73–76, 1989.

[4] J. F. Bard. *Practical bilevel optimization: algorithms and applications*, volume 30. Springer Science & Business Media, 2013.

[5] M. Barnabei, G. Nicoletti, and L. Pezzoli. Matroids on partially ordered sets. *Advances in Applied Mathematics*, 21(1):78–112, 1998.

[6] N. O. Baycik and K. M. Sullivan. Robust location of hidden interdictions on a shortest path network. *IISE Transactions*, 51(12):1332–1347, 2019.

[7] C. Bazgan, S. Toubaline, and Z. Tuza. The most vital nodes with respect to independent set and vertex cover. *Discrete Applied Mathematics*, 159(17):1933–1946, 2011.

[8] C. Bazgan, S. Toubaline, and Z. Tuza. The most vital nodes with respect to independent set and vertex cover. *Discrete Applied Mathematics*, 159(17):1933–1946, 2011.

[9] C. Bazgan, S. Toubaline, and D. Vanderpooten. Critical edges/nodes for the minimum spanning tree problem: complexity and approximation. *Journal of Combinatorial Optimization*, 26(1):178–189, 2013.

[10] J. S. Borrero, O. A. Prokopyev, and D. Sauré. Sequential shortest path interdiction with incomplete information. *Decision Analysis*, 13(1):68–98, 2016.

[11] C. Contardo and J. A. Sefair. A progressive approximation approach for the exact solution of sparse large-scale binary interdiction games. *INFORMS Journal on Computing*, 34(2):890–908, 2022.

[12] K. J. Cormican, D. P. Morton, and R. K. Wood. Stochastic network interdiction. *Operations Research*, 46(2):184–197, 1998.

[13] S. Dempe and A. Zemkoho. *Bilevel Optimization*. Springer, 2020.

[14] M. Di Summa, A. Grosso, and M. Locatelli. Branch and cut algorithms for detecting critical nodes in undirected graphs. *Computational Optimization and Applications*, 53(3):649–680, 2012.

[15] T. N. Dinh, Y. Xuan, M. T. Thai, P. M. Pardalos, and T. Znati. On new approaches of assessing network vulnerability: Hardness and approximation. *IEEE/ACM Transactions on Networking*, 20(2):609–619, 2012.

[16] G. N. Frederickson and R. Solis-Oba. Increasing the weight of minimum spanning trees. In *Proceedings of the seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 539–546. Society for Industrial and Applied Mathematics, 1996.

[17] F. Furini, I. Ljubić, S. Martin, and P. San Segundo. The maximum clique interdiction problem. *European Journal of Operational Research*, 277(1):112–127, 2019.

[18] F. Furini, I. Ljubić, P. San Segundo, and Y. Zhao. A branch-and-cut algorithm for the edge interdiction clique problem. *European Journal of Operational Research*, 2021. `https://doi.org/10.1016/j.ejor.2021.01.030`.

[19] D. Gale. Optimal assignments in an ordered set: an application of matroid theory. *Journal of Combinatorial Theory*, 4(2):176–180, 1968.

[20] P. Ghare, D. C. Montgomery, and W. Turner. Optimal interdiction policy for a flow network. *Naval Research Logistics Quarterly*, 18(1):37–45, 1971.

[21] P. Glasserman and D. D. Yao. Generalized semi-markov processes: antimatroid structure and second-order properties. *Mathematics of operations research*, 17(2):444–469, 1992.

[22] N. Goldberg. Non-zero-sum nonlinear network path interdiction with an application to inspection in terror networks. *Naval Research Logistics (NRL)*, 64(2):139–153, 2017.

[23] R. E. Gomory and T. C. Hu. Multi-terminal network flows. *Journal of the Society for Industrial and Applied Mathematics*, 9(4):551–570, 1961.

[24] T. H. Grubesic, T. C. Matisziw, A. T. Murray, and D. Snediker. Comparative approaches for assessing network vulnerability. *International Regional Science Review*, 31(1):88–112, 2008.

[25] D. J. Houck, E. Kim, G. P. O'Reilly, D. D. Picklesimer, and H. Uzunalioglu. A network survivability model for critical national infrastructures. *Bell Labs Technical Journal*, 8(4):153–172, 2004.

[26] E. Israeli and R. K. Wood. Shortest-path network interdiction. *Networks: An International Journal*, 40(2):97–111, 2002.

[27] E. Jenelius, T. Petersen, and L.-G. Mattsson. Importance and exposure in road network vulnerability analysis. *Transportation Research Part A: Policy and Practice*, 40(7):537–560, 2006.

[28] A. Kheirkhah, H. Navidi, and M. M. Bidgoli. An improved benders decomposition algorithm for an arc interdiction vehicle routing problem. *IEEE Transactions on Engineering Management*, 63(2):259–273, 2016.

[29] T. Kleinert, M. Labbé, I. Ljubić, and M. Schmidt. A survey on mixed-integer programming techniques in bilevel optimization. Available at `http://www.optimization-online.org/DB_HTML/2021/01/8187.html`. Last access, May 2021., 2021.

[30] B. Korte, L. Lovász, and R. Schrader. *Greedoids*, volume 4. Springer Science & Business Media, 2012.

[31] C. Lim and J. C. Smith. Algorithms for discrete and continuous multicommodity flow network interdiction problems. *IIE Transactions*, 39(1):15–26, 2007.

[32] S. Liu, M. Wang, N. Kong, and X. Hu. An enhanced branch-and-bound algorithm for bilevel integer linear programming. *European Journal of Operational Research*, 291(2):661–679, 2021.

[33] L. Lozano and J. C. Smith. A backward sampling framework for interdiction problems with fortification. *INFORMS Journal on Computing*, 29(1):123–139, 2016.

[34] M. Luby. A simple parallel algorithm for the maximal independent set problem. *SIAM journal on computing*, 15(4):1036–1053, 1986.

[35] T. L. Magnanti and L. A. Wolsey. Optimal trees. *Handbooks in operations research and management science*, 7:503–615, 1995.

[36] F. Mahdavi Pajouh. Minimum cost edge blocker clique problem. *Annals of Operations Research*, 294:345 – 376, 2020.

[37] F. Mahdavi Pajouh, V. Boginski, and E. L. Pasiliao. Minimum vertex blocker clique problem. *Networks*, 64(1):48–64, 2014.

[38] T. C. Matisziw and A. T. Murray. Modeling s-t path availability to support disaster vulnerability assessment of network infrastructure. *Computers and Operations Research.*, 36:16–26, 2009.

[39] A. W. McMasters and T. M. Mustin. Optimal interdiction of a supply network. *Naval Research Logistics Quarterly*, 17(3):261–268, 1970.

[40] A. W. McMasters and T. M. Mustin. Optimal interdiction of a supply network. *Naval Research Logistics Quarterly*, 17(3):261–268, 1970.

[41] F. M. Pajouh, J. L. Walteros, V. Boginski, and E. L. Pasiliao. Minimum edge blocker dominating set problem. *European Journal of Operational Research*, 247(1):16 – 26, 2015.

[42] A. Recski. *Matroid theory and its applications in electric network theory and in statics*, volume 6. Springer Science & Business Media, 2013.

[43] N. Robertson and P. D. Seymour. Graph minors. i. excluding a forest. *Journal of Combinatorial Theory, Series B*, 35(1):39–61, 1983.

[44] J. Salmerón. Deception tactics for network interdiction: A multiobjective approach. *Networks*, 60(1):45–58, 2012.

[45] J. Salmeron, K. Wood, and R. Baldick. Analysis of electric grid security under terrorist threat. *IEEE Transactions on power systems*, 19(2):905–912, 2004.

[46] J. Salmeron, K. R. Wood, and R. Baldick. Analysis of electric grid security under terrorist threat. *Power Systems, IEEE Transactions on*, 19(2):905–912, 2004.

[47] J. A. Sefair and J. C. Smith. Dynamic shortest-path interdiction. *Networks*, 68(4):315–330, 2016.

[48] J. C. Smith and Y. Song. A survey of network interdiction models and algorithms. *European Journal of Operational Research*, 283(3):797–811, 2020.

[49] M. Stoer and F. Wagner. A simple min-cut algorithm. *Journal of the ACM (JACM)*, 44(4):585–591, 1997.

[50] Z. Tao, F. Zhongqian, and W. Binghong. Epidemic dynamics on complex networks. *Progress in Natural Science*, 16(5):452–457, 2005.

[51] W. T. Tutte. Matroids and graphs. *Transactions of the American Mathematical Society*, 90(3):527–552, 1959.

[52] A. Washburn and K. Wood. Two-person zero-sum games for network interdiction. *Operations research*, 43(2):243–251, 1995.

[53] N. Wei and J. L. Walteros. Integer programming methods for solving binary interdiction games. *European Journal of Operational Research*, 2022.

[54] N. Wei, J. L. Walteros, and F. M. Pajouh. Integer programming formulations for minimum spanning tree interdiction. *INFORMS Journal on Computing*, 2021.

[55] N. Wei, J. L. Walteros, M. R. Worden, and H. J. Ortiz-Peña. A resiliency analysis of information distribution policies over mobile ad hoc networks. *Optimization Letters*, 15(4):1081–1103, 2021.

[56] N. Wei, J. L. Walteros, M. R. Worden, and H. J. Ortiz-Peña. A resiliency analysis of information distribution policies over mobile ad hoc networks. *Optimization Letters*, 2021.

[57] N. White and N. M. White. *Matroid applications*. Number 40. Cambridge University Press, 1992.

[58] L. A. Wolsey and G. L. Nemhauser. *Integer and combinatorial optimization*, volume 55. John Wiley & Sons, 1999.

[59] R. Wood. Deterministic network interdiction. *Mathematical and Computer Modelling*, 17(2):1 – 18, 1993.

[60] X. Xie and F. Aros-Vera. An interdependent network interdiction model for disrupting sex trafficking networks. *Production and Operations Management*, 2022.

[61] P. Xu and L. Wang. An exact algorithm for the bilevel mixed integer linear programming problem under three simplifying assumptions. *Computers & operations research*, 41:309–318, 2014.

[62] J. Yang, J. S. Borrero, O. A. Prokopyev, and D. Sauré. Sequential shortest path interdiction with incomplete information and limited feedback. *Decision Analysis*, 18(3):218–244, 2021.

[63] R. Zenklusen. Matching interdiction. *Discrete Appl. Math.*, 158(15):1676–1690, 2010.