

An Exact Algorithm for Solving the Fortification Problem under Scenario Uncertainty

Joshua Ong^a, Andrew Mastin^a, Jean-Paul Watson^a

^a*Lawrence Livermore National Laboratory, 7000 East Avenue, Livermore, CA, 94550, USA,*

Abstract

The fortification problem (FP) is commonly formulated as a defender-attacker-defender (DAD) model. In this setting, a planner first hardens up to h components, then an attacker interdicts up to k components, and finally the defender solves a recourse problem to maintain system operations. A key limitation of the DAD model is the assumption that the planner has full knowledge of the attacker's capability and the operating condition of the system. To address these sources of uncertainty, we formulate the fortification problem under scenario uncertainty (FP-SU). Existing methods, such as column-and-constraint generation (CCG) algorithms, struggle to scale efficiently as the number of scenarios increases. We propose the Historically-Observed Precedent Sampling (HOPS) algorithm that exploits the scenario structure to generate valid lower bounds and implicitly enumerate candidate defenses. We prove that the HOPS algorithm converges in finitely many iterations to an optimal solution. Computational experiments on benchmark electrical power grid test cases demonstrate substantial improvements in scalability and runtime relative to existing methods.

Keywords: interdiction/fortification problems, multi-level optimization, power grid resilience

1. Introduction

We consider a fortification problem (FP) for optimizing system hardening investments, formulated as a defender-attacker-defender (DAD) model. In the first stage, a defender, acting as the planner, fortifies a subset of components in the system to enhance resilience. Next, an attacker interdicts unprotected components to maximize disruption. Finally, the defender, acting as the operator, solves a recourse problem to minimize performance impacts. Fortification and interdiction problems have extensive applications to critical infrastructure such as oil pipelines, public transportation, and power grids [1].

General-purpose techniques such as column-and-constraint generation (CCG) and Benders decomposition algorithms are commonly employed to solve DAD problems [2]. However, DAD problems remain computationally challenging and there is increasing interest in the development of faster algorithms for solving real-world-scale instances. Lozano and Smith [3] solve the DAD problem by approximating the recourse function

Email address: ong11@llnl.gov (Joshua Ong)

¹This work was funded by the US Department of Energy's Office of Electricity, under the North American Energy Resilience Model (NAERM) and Advanced Grid Modeling (AGM) programs. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC.

using a sampling-based approach. Mastin *et al.* [4] utilize the problem-specific structure of FPs to introduce strengthening cuts within an implicit enumeration algorithm.

While the FP generally assumes each decision maker has complete information, in practice, the planner often lacks precise knowledge of the attacker’s resources and capabilities. To address this, Xiang and Wang [5] consider multiple attack scenarios in which the defender minimizes the weighted sum of disruptions, with weights assigned by the perceived probabilities of the scenario occurring. We refer to this extension as the *FP under Interdiction Budget Uncertainty* (FP-IBU). Zang *et al.* [6] propose a framework for determining the weights of different attack scenarios and extend the formulation for node interdiction. The planner may also have to consider system uncertainty, encapsulated in the recourse problem or cost function. For example, applied to power grids, Xiang *et al.* [7] solve the FP with wind generation and load demand uncertainty sets. We refer to this extension as the *FP under Cost Function Uncertainty* (FP-CFU). All three studies [5], [6], and [7] solve their problem variants with a CCG algorithm that duplicates the recourse problem for every attack considered and for every operational condition and hence struggle to scale to larger problem instances.

We define a scenario as a joint realization of attacker capability and system state, and we refer to the resulting generalized problem as the *FP under Scenario Uncertainty* (FP-SU). In this work, we propose an algorithm to solve the FP-SU that exploits problem-specific structure to improve scalability. Our contributions are as follows:

1. We present an implicit enumeration algorithm that exploits the FP-SU structure called the Historically Observed Precedent Sampling (HOPS) algorithm and prove that the algorithm is optimal and converges in a finite number of iterations.
2. We present computational results on benchmark power grid test cases showing that the HOPS algorithm significantly outperforms the CCG algorithm as either the number of scenarios or the defender budget h increases.

In Section 2 we formulate the FP-SU and present the HOPS algorithm. We present computational results in Section 3. Section 4 concludes the paper with future work.

2. Problem Formulation and Algorithm

2.1. Problem Formulation

Let T denote the set of components in an interconnected system that a defender may harden or an attacker may target. The feasible sets of defenses and attacks are denoted as $\mathcal{D}^F \subseteq 2^T$ and $\mathcal{A}^F \subseteq 2^T$, respectively. Let $\mathcal{A}^k := \{A : A \in 2^T, |A| \leq k\}$ represent an attacker with limited capability of attacking at most k elements. Likewise let $\mathcal{D}^h := \{D : D \in 2^T, |D| \leq h\}$ represent a defender with a budget to harden h components. Let $C(A, D)$ denote the cost incurred by the system if attack A is performed after defense D has been implemented.

We rely on two conditions on the structure of the cost function introduced in [4]. The first condition states that an equivalent cost function can be constructed that only considers attacks on undefended components:

Condition 1. *The function $C : \mathcal{A}^F \times \mathcal{D}^F \rightarrow \mathbb{R}$ can be represented by an equivalent function $\tilde{C} : 2^T \rightarrow \mathbb{R}$ where for any attack $A \in \mathcal{A}^F$ and defense $D \in \mathcal{D}^F$, $C(A, D) = \tilde{C}(A \setminus D)$.*

The second condition states, for a given defense D , an attack can be constructed with equivalent cost that does not intersect with D :

Condition 2. For any $A \in \mathcal{A}^F$ and $D \in \mathcal{D}^F$ such that $|A \cap D| \geq 1$, there exists an $A' \in \mathcal{A}^F$ such that $A' \cap D = \emptyset$ and $C(A, D) = C(A', D)$.

Given a feasible attacker set $\mathcal{A}^F$ and a fixed defense D , let the *attacker's best response oracle* be defined as:

$$\text{ABR}_F(D) := \underset{\{A \in \mathcal{A}^F : A \cap D = \emptyset\}}{\operatorname{argmax}} C(A, D). \quad (1)$$

If the ABR function is formulated as a mixed integer program (MIP), commercial solvers routinely obtain additional feasible solutions. In this case, let the notation $A_0, \dots, A_j = \text{ABR}_F(D)$, denote an optimal solution A_0 and additional feasible solutions collected $A_1, \dots, A_j$ such that $\tilde{C}(A_0) \geq \dots \geq \tilde{C}(A_j)$. Note that the algorithm we propose is compatible with or without obtaining additional feasible solutions.

Let $\mathcal{S}$ be the set of scenarios. For each scenario $s \in \mathcal{S}$, let C_s denote the scenario-dependent cost function and let $\mathcal{A}^s$ denote the corresponding attacker feasibility region. Let the family of cost functions be defined as $\mathcal{C} := (C_s)_{s \in \mathcal{S}}$ and the family of attacker feasibility regions $\mathcal{A}^{\mathcal{S}} := (\mathcal{A}^s)_{s \in \mathcal{S}}$. Throughout this work, we assume that all scenarios are equally likely. We also assume the cost function is bounded. We define the *fortification problem under scenario uncertainty* (FP-SU) as:

$$\text{FP-SU}(\mathcal{A}^{\mathcal{S}}, \mathcal{D}^F, \mathcal{C}, \mathcal{S}) := \min_{D \in \mathcal{D}^F} \sum_{s \in \mathcal{S}} \max_{A \in \mathcal{A}^s} C_s(A, D). \quad (2)$$

2.2. Algorithm

We first describe a subroutine that selects the planner's candidate defense, called the new defense MIP (NDMIP), as described in equations (3a)–(3f). Let H denote the sequence of indices that orders all attacks observed by the algorithm in non-increasing order of cost. Let $\mathcal{A}_H^s := \{A_h^s \in \mathcal{A}^s : h \in H\}$ denote the history of observed attacks and $\mathcal{A}_H^{\mathcal{S}}$ be the family of all histories $(\mathcal{A}_H^s)_{s \in \mathcal{S}}$. Each attack A_h^s has an associated cost $c_h^s = \tilde{C}_s(A_h^s)$, where $c_h^s \geq c_{h'}^s$, whenever $h < h'$. Define $c_{\min}^s := \max_{A \in \mathcal{A}^s} C_s(A, D_1)$ which Lemma 1 shows is a valid lower bound for a cost function C_s . For every s in $\mathcal{S}$, denote the decision variables $b_h^s \in \{0, 1\}$ to represent that defense D covers attack A_h^s and η^s to represent the perceived cost given the observed history H .

$$(\text{NDMIP}(\mathcal{A}_H^{\mathcal{S}})) \quad \min_{b_h^s, \eta^s, D} \sum_{s \in \mathcal{S}} \eta^s \quad (3a)$$

subject to:

$$\eta^s \geq c_h^s - c_h^s b_h^s + c_{\min}^s b_h^s, \quad \forall h \in H, \quad \forall s \in \mathcal{S}, \quad (3b)$$

$$|D \cap A_h^s| \geq b_h^s, \quad \forall h \in H, \quad \forall s \in \mathcal{S}, \quad (3c)$$

$$b_h^s \in \{0, 1\}, \quad \forall h \in H, \quad \forall s \in \mathcal{S}, \quad (3d)$$

$$\eta^s \in \mathbb{R}, \quad \forall s \in \mathcal{S}, \quad (3e)$$

$$D \in \mathcal{D}^F. \quad (3f)$$

The NDMIP objective (3a) minimizes an approximation of the FP-SU given the limited information of attacks observed $\mathcal{A}_H^{\mathcal{S}} \subseteq \mathcal{A}^{\mathcal{S}}$. Constraint (3b) enforces the cost for

undefended attacks. Constraint (3c) ensures that b_h^s may only equal 1 if D intersects with A_h^s . Lastly constraints (3d) - (3f) define the feasible domains for the decision variables. In effect, NDMIP uses the stored attack history to construct a lower bound on the attacker's best response value for each candidate defense. Consequently, candidate defenses whose history-based lower bound is already worse than the incumbent solution can be excluded from further evaluation.

We present the *historically observed precedent sampling* (HOPS) algorithm in Algorithm 1. The algorithm is initialized in lines 2 - 5, where $D_\emptyset := \emptyset$ represents the empty set defense and $D_1 := T$ represents the full defense, corresponding to the all-ones vector. At each iteration, a new defense D is selected by solving the NDMIP in line 8. Each selected defense is evaluated by solving the ABR_s oracles for all s in $\mathcal{S}$ and newly observed attacks are added to the histories. If the evaluated defense improves the incumbent upper bound, then the upper bound is updated in line 14. This alternating process continues until the stopping criterion in line 7 is satisfied, yielding the optimal defense.

Algorithm 1 Historically Observed Precedent Sampling

Require: Set of scenarios $\mathcal{S}$, feasible set $\mathcal{D}^F$, family of feasible sets $\mathcal{A}^S$, family of cost functions $\mathcal{C}$, and attacker's best response oracles $\text{ABR}_s(\cdot)$ for $s \in \mathcal{S}$

Ensure: Optimal defense D^* , cost u

- 1: Initialize best defense $D^* \leftarrow \emptyset$, upper bound $u \leftarrow \infty$, lower bound $l \leftarrow -\infty$, history of observed attacks $\mathcal{A}_H^s \leftarrow \emptyset \forall s \in \mathcal{S}$
 - 2: **for** each s in $\mathcal{S}$ **do**
 - 3: Calculate attacker's best response $A_0, \dots, A_j = \text{ABR}_s(D_\emptyset)$, update $\mathcal{A}_H^s = \mathcal{A}_H^s \cup \left(\bigcup_{i=0}^j \{A_i\} \right)$
 - 4: Calculate attacker's best response $A_0, \dots, A_j = \text{ABR}_s(D_1)$, update $\mathcal{A}_H^s = \mathcal{A}_H^s \cup \left(\bigcup_{i=0}^j \{A_i\} \right)$
 - 5: **end for**
 - 6: Update upper bound $u \leftarrow \sum_{s \in \mathcal{S}} C_s(\text{ABR}_s(D_\emptyset), D_\emptyset)$
 - 7: **while** $l < u$ **do**
 - 8: Solve NDMIP($\mathcal{A}_H^S$) to obtain optimal objective value and defense l, D
 - 9: **for** each s in $\mathcal{S}$ **do**
 - 10: Calculate attacker's best response $A_0, \dots, A_j = \text{ABR}_s(D)$, update $\mathcal{A}_H^s = \mathcal{A}_H^s \cup \left(\bigcup_{i=0}^j \{A_i\} \right)$
 - 11: **end for**
 - 12: Evaluate defense to obtain cost $\tilde{u} = \sum_{s \in \mathcal{S}} C_s(\text{ABR}_s(D), D)$
 - 13: **if** $\tilde{u} < u$ **then**
 - 14: Update best defense $D^* \leftarrow D$ and best cost $u \leftarrow \tilde{u}$
 - 15: **end if**
 - 16: **end while**
-

Before we prove the correctness of Algorithm 1 in Theorem 1, we work towards establishing the NDMIP as a lower bound to the ABR problem in Lemma 1, Lemma 2, and Lemma 3. For each scenario $s \in \mathcal{S}$ and defense $D \subseteq T$, define the feasible non-intersecting set as

$$\mathcal{A}^s(D) := \{A \in \mathcal{A}^s : A \cap D = \emptyset\}.$$

Lemma 1. For any scenario $s \in \mathcal{S}$, the all-ones defense $D_1 := T$ satisfies

$$\min_{D \in \mathcal{D}^F} \max_{A \in \mathcal{A}^s} C_s(A, D) \geq \max_{A \in \mathcal{A}^s} C_s(A, D_1),$$

provided that Condition 1 and Condition 2 hold for all cost functions in the family $\mathcal{C}$.

Proof. For any $D \in \mathcal{D}^F$, by Conditions 1 and 2:

$$\max_{A \in \mathcal{A}^s} C_s(A, D) = \max_{A \in \mathcal{A}^s} \tilde{C}_s(A \setminus D) = \max_{A \in \mathcal{A}^s(D)} \tilde{C}_s(A). \quad (4)$$

By equation (4) and because $\mathcal{A}^s(D_1) \subseteq \mathcal{A}^s(D)$,

$$\max_{A \in \mathcal{A}^s} C_s(A, D_1) = \max_{A \in \mathcal{A}^s(D_1)} \tilde{C}_s(A) \leq \max_{A \in \mathcal{A}^s(D)} \tilde{C}_s(A) = \max_{A \in \mathcal{A}^s} C_s(A, D) \quad \forall D \in \mathcal{D}^F. \quad (5)$$

Since the pointwise inequality holds for arbitrary $D \in \mathcal{D}^F$, the claim follows. Note, although D_1 need not belong to $\mathcal{D}^F$, the quantity $C_s(\text{ABR}_s(D_1), D_1)$ is well defined and provides a valid lower bound. $\square$

Lemma 2. For any fixed defense $D \in \mathcal{D}^F$, for each scenario $s \in \mathcal{S}$, define $\tau^s(D) := \min\{h \in H : |D \cap A_h^s| = 0\}$ as the minimum index in the history at which D does not intersect an attack. Then there exists $b_h^s \in \{0, 1\}$ and $\eta^s \in \mathbb{R}$ such that $(b_h^s, \eta^s, D) \in \{b_h^s, \eta^s, D : (3b) - (3f)\}$ with resulting cost $\sum_{s \in \mathcal{S}} \eta^s = \sum_{s \in \mathcal{S}} c_{\tau^s(D)}^s$, provided that Algorithm 1 has completed the initialization step in lines 2-5.

Proof. Fix any defense $D \in \mathcal{D}^F$ and scenario $s \in \mathcal{S}$. By the initialization in lines 2 - 5, $\text{ABR}_s(D_1)$ is stored in the history $\mathcal{A}_H^s$. Because $\text{ABR}_s(D_1) \cap D = \emptyset$, the set $\{h \in H : |D \cap A_h^s| = 0\}$ is non-empty, and thus $\tau^s(D)$ is well-defined. Set

$$b_h^s = \begin{cases} 1 & \text{if } h < \tau^s(D) \\ 0 & \text{if } h \geq \tau^s(D) \end{cases} \quad \text{and} \quad \eta^s := c_{\tau^s(D)}^s. \quad (6)$$

By assumption, D satisfies constraint (3f). Next consider constraint (3c). First, suppose $h < \tau^s(D)$, then $|D \cap A_h^s| \geq 1$ by the construction of $\tau^s(D)$. If instead $h \geq \tau^s(D)$, then $|D \cap A_h^s| \geq 0$ trivially, because cardinality is non-negative. Thus constraint (3c) is satisfied since

$$\begin{cases} |D \cap A_h^s| \geq 1 = b_h^s & \text{if } h < \tau^s(D) \\ |D \cap A_h^s| \geq 0 = b_h^s & \text{if } h \geq \tau^s(D). \end{cases} \quad (7)$$

Lastly consider the cases for constraint (3b). If $h < \tau^s(D)$, then $b_h^s = 1$ and constraint (3b) is reduced to $\eta^s \geq c_{\min}^s$. This inequality holds by Lemma 1. Else if $h \geq \tau^s(D)$, $b_h^s = 0$, the constraint becomes $\eta^s \geq c_h^s$. Since the history is ordered in non-increasing cost, $c_{\tau^s(D)}^s \geq c_h^s$ for all $h \geq \tau^s(D)$. Altogether constraint (3b) is satisfied because

$$\begin{cases} \eta^s = c_{\tau^s(D)}^s \geq c_{\min}^s & \text{if } h < \tau^s(D) \\ \eta^s = c_{\tau^s(D)}^s \geq c_h^s & \text{if } h \geq \tau^s(D). \end{cases} \quad (8)$$

Repeating this construction for every $s \in \mathcal{S}$ we obtain the feasible point $(b_h^s, \eta^s, D) \in \{b_h^s, \eta^s, D : (3b) - (3f)\}$ with cost $\sum_{s \in \mathcal{S}} \eta^s = \sum_{s \in \mathcal{S}} c_{\tau^s(D)}^s$. $\square$

Lemma 3. Let l denote the optimal objective value of $\text{NDMIP}(\mathcal{A}_H^S)$, then

$$l \leq \min_{D \in \mathcal{D}^F} \sum_{s \in \mathcal{S}} \max_{A \in \mathcal{A}^s} C_s(A, D),$$

provided that Condition 1 and Condition 2 hold for all cost functions in the family $\mathcal{C}$ and Algorithm 1 has completed the initialization step in lines 2–5.

Proof. Fix an arbitrary defense $D \in \mathcal{D}^F$. By Lemma 2, there exists a feasible point (b_h^s, η^s, D) in $\text{NDMIP}(\mathcal{A}_H^S)$ with resulting objective value $\sum_{s \in \mathcal{S}} c_{\tau^s(D)}^s$. Since l is the optimal value for $\text{NDMIP}(\mathcal{A}_H^S)$, it follows that

$$l \leq \sum_{s \in \mathcal{S}} c_{\tau^s(D)}^s. \quad (9)$$

Next, for each $s \in \mathcal{S}$, let $A_{\tau^s(D)}^s$ be the attack with observed cost $c_{\tau^s(D)}^s$. Then $c_{\tau^s(D)}^s = \tilde{C}_s(A_{\tau^s(D)}^s) = C_s(A_{\tau^s(D)}^s, D)$, by Condition 1 and because $A_{\tau^s(D)}^s \cap D = \emptyset$. Because $A_{\tau^s(D)}^s$ is in $\mathcal{A}^s$,

$$c_{\tau^s(D)}^s = C_s(A_{\tau^s(D)}^s, D) \leq \max_{A \in \mathcal{A}^s} C_s(A, D). \quad (10)$$

Summing over all s in equation (10) and combining with equation (9) yields

$$l \leq \sum_{s \in \mathcal{S}} c_{\tau^s(D)}^s \leq \sum_{s \in \mathcal{S}} \max_{A \in \mathcal{A}^s} C_s(A, D). \quad (11)$$

Since the pointwise inequality holds for arbitrary $D \in \mathcal{D}^F$, it follows that

$$l \leq \min_{D \in \mathcal{D}^F} \sum_{s \in \mathcal{S}} \max_{A \in \mathcal{A}^s} C_s(A, D). \quad (12)$$

□

Theorem 1. Upon termination, Algorithm 1 returns an optimal solution to the $\text{FP-SU}(\mathcal{A}^S, \mathcal{D}^F, \mathcal{C}, \mathcal{S})$ that satisfies Condition 1 and Condition 2 for all cost functions in the family $\mathcal{C}$.

Proof. Assume Algorithm 1 terminates with the optimal objective value l and defense D obtained from $\text{NDMIP}(\mathcal{A}_H^S)$. Let u be the best upper bound generated from defense D^* . By the termination condition on line 7 and by Lemma 3,

$$u \leq l \leq \min_{D \in \mathcal{D}^F} \sum_{s \in \mathcal{S}} \max_{A \in \mathcal{A}^s} C_s(A, D). \quad (13)$$

Because $D^* \in \mathcal{D}^F$ and thus u is a valid upper bound,

$$\min_{D \in \mathcal{D}^F} \sum_{s \in \mathcal{S}} \max_{A \in \mathcal{A}^s} C_s(A, D) \leq \sum_{s \in \mathcal{S}} \max_{A \in \mathcal{A}^s} C_s(A, D^*) = u. \quad (14)$$

Combining equations (13) and (14) yields

$$l = u = \sum_{s \in \mathcal{S}} \max_{A \in \mathcal{A}^s} C_s(A, D^*) = \min_{D \in \mathcal{D}^F} \sum_{s \in \mathcal{S}} \max_{A \in \mathcal{A}^s} C_s(A, D). \quad (15)$$

□

Lemma 4. *At iteration t of Algorithm 1, let l_t and D_t be the optimal objective value and defense obtained from $\text{NDMIP}(\mathcal{A}_H^S)$. Further, for each s in $\mathcal{S}$, define the attacker's best responses $\tilde{A}^s = \text{ABR}_s(D_t)$. Then Algorithm 1 terminates at iteration t if the attacker best responses are in the history, $\tilde{A}^s \in \mathcal{A}_H^s$ for each s in $\mathcal{S}$.*

Proof. Fix any $s \in \mathcal{S}$. Let h_t denote the index at which the cost $\tilde{C}_s(\tilde{A}^s)$ is stored. Since $\tilde{A}^s \cap D_t = \emptyset$, this implies $b_{h_t}^s = 0$ because of constraint (3c). From constraint (3b), this implies $\eta^s \geq c_{h_t}^s = C_s(\tilde{A}^s, D_t)$. Since s was arbitrary, summing over all s yields

$$l_t = \sum_{s \in \mathcal{S}} \eta^s \geq \sum_{s \in \mathcal{S}} C_s(\tilde{A}^s, D_t). \quad (16)$$

At iteration t , $u_t = \sum_{s \in \mathcal{S}} C_s(\tilde{A}^s, D_t)$. Since the global upper bound u is less than all intermediate upper bounds,

$$u \leq u_t = \sum_{s \in \mathcal{S}} C_s(\tilde{A}^s, D_t). \quad (17)$$

Combining equations (16) and (17) yields

$$u \leq \sum_{s \in \mathcal{S}} C_s(\tilde{A}^s, D_t) \leq l_t, \quad (18)$$

the terminating condition of Algorithm 1. $\square$

Theorem 2. *Algorithm 1 terminates after finitely many iterations.*

Proof. Suppose the algorithm does not terminate at iteration t , then by the contrapositive of Lemma 4, there must exist at least one attacker's best response not in the history yet. Note that at least one new attack is added to the history each iteration in line 10. Since T is finite, the total number of attacks added to the histories is finite. Thus only finitely many nonterminal iterations are possible. $\square$

3. Computational Studies on Power Grid Interdiction

3.1. Experimental Setup

We present computational results for the FP-SU with application to the electrical power grid. We implemented the HOPS algorithm detailed in Algorithm 1 and the CCG algorithm formulated in [5] to compare computational performance on instances of the FP-SU. We construct a scenario set in which scenarios correspond to different attacker interdiction budgets. Let $\mathcal{K}$ denote the set of budget scenarios. The underlying power grid model, and hence the cost function, is identical across all scenarios, so that $C_s = C$ for all $s \in \mathcal{K}$. By contrast, the attacker feasibility region $\mathcal{A}^s$ varies with the interdiction budget.

Power flow is modeled with the direct-current optimal power flow (DCOPF) formulation and the planner and attacker harden and target branches, which correspond to edges in a network. The cost function in this application is the total amount of load shed in the electrical power grid. The ABR problem is formulated from [4] and we used $M_{\mu l} = 20$ and $M_{\eta l} = 4$ big-M values. We tested algorithms on standard test cases from the RTS-GMLC [8] and PGLIB v23 [9].

Table 1: Runtime and number of iterations to converge of the CCG and HOPS algorithms on the IEEE 118-bus test case for increasing defender budgets h .

$ \mathcal{K} , h$	CCG			HOPS			CCG/HOPS	
	Time (s)	Iterations	Time/Iter	Time (s)	Iterations	Time/Iter	Ratio Time	Ratio Iter
5,1	432.21	5	86.44	219.01	3	73.00	1.97	1.67
5,2	868.45	9	96.49	614.38	7	87.77	1.41	1.29
5,3	1631.62	14	116.54	1358.96	13	104.54	1.20	1.08
5,4	2843.34	20	142.17	2349.56	17	138.21	1.21	1.18
5,5	6870.10	35	196.29	5132.95	27	190.11	1.34	1.30
5,6	7749.58	27	287.02	5239.38	21	249.49	1.48	1.29
5,7	14930.29	38	392.90	9322.20	28	332.94	1.60	1.36
5,8	38040.71	67	567.77	19913.88	37	538.21	1.91	1.81
5,9	94166.85	100	941.67	31826.13	48	663.04	2.96	2.08
5,10	255916.28	150	1706.11	35489.93	43	825.35	7.21	3.49

Algorithms were written with Python 3.9.12. Mathematical programs were modeled with Pyomo [10] 6.8.2 and solved with Gurobi 12.0.3. To solve the ABR, Gurobi solver settings `PoolSolutions = 50` and `PoolSearchMode = 0` were used to collect additional feasible solutions which can be stored in the history. For all problem instances, the time limit was set to seven days of wall-clock time. All experiments were performed on a compute node with the Intel(R) Xeon(R) Gold 6140 CPU, 2.30GHz processor, 36 cores, and 192 GB of memory.

3.2. Scalability with Respect to Defender Budget h

In this set of experiments we fixed $\mathcal{K} = \{1, 3, 5, 7, 10\}$ and varied the defender budget h from one to ten to assess algorithmic scaling with respect to h . We present results comparing algorithms for all test cases in Tables 1-3.

Across all test instances in this experiment, the HOPS algorithm outperforms the CCG algorithm in runtime. While the CCG algorithm reaches the time limit on five instances after seven days, the HOPS algorithm solves all instances in under ten hours. For the IEEE 300-bus and the RTS-GMLC test cases, the HOPS algorithm improves runtime compared to the CCG algorithm by one to two orders of magnitude, up to a factor of 293. For the IEEE 118-bus test case, the performance gains are more modest, although the HOPS algorithm still converges with a runtime reduction by a factor of 7.21 for $h = 10$. We hypothesize that the modest performance gains are not due to the HOPS algorithm performing poorly on the IEEE 118-bus test case, but that the CCG algorithm performs significantly better on the IEEE 118-bus test case compared to other test cases. For example, for $h = 6$, the CCG algorithm converges in 27 iterations for the IEEE 118-bus test case but 124 and 103 iterations for the IEEE 300 and RTS-GMLC test cases respectively.

We conclude that the HOPS algorithm scales more efficiently with h compared to the CCG algorithm for the test cases included. Both the iteration count and the runtime per iteration of the HOPS algorithm grow substantially slower with h compared to the CCG algorithm. Since both algorithms solve the same ABR subproblem, the performance gains are attributable to the NDMIP structure compared to the CCG master problem.

Table 2: Runtime and number of iterations to converge of the CCG and HOPS algorithms on the IEEE 300-bus test case for increasing defender budgets h . Results marked with the $\dagger$ symbol failed to converge after 7 days.

$ \mathcal{K} , h$	CCG			HOPS			CCG/HOPS	
	Time (s)	Iterations	Time/Iter	Time (s)	Iterations	Time/Iter	Ratio Time	Ratio Iter
5,1	1177.22	6	196.20	287.65	2	143.82	4.09	3.00
5,2	3130.61	14	223.62	892.87	6	148.81	3.51	2.33
5,3	11186.09	35	319.60	2014.01	12	167.83	5.55	2.92
5,4	33648.43	69	487.66	3474.16	19	182.85	9.69	3.63
5,5	72799.10	95	766.31	5667.18	29	195.42	12.85	3.28
5,6	165380.84	124	1333.72	8723.88	38	229.58	18.96	3.26
5,7	423451.00	163	2597.86	13109.00	47	278.91	32.30	3.47
5,8	$\dagger$	$\dagger$	$\dagger$	15504.84	46	337.06		
5,9	$\dagger$	$\dagger$	$\dagger$	22256.84	51	436.41		
5,10	$\dagger$	$\dagger$	$\dagger$	30828.19	63	489.34		

Table 3: Runtime and number of iterations to converge of the CCG and HOPS algorithms on the RTS-GMLC test case for increasing defender budgets h . Results marked with the $\dagger$ symbol failed to converge after 7 days.

$ \mathcal{K} , h$	CCG			HOPS			CCG/HOPS	
	Time (s)	Iterations	Time/Iter	Time (s)	Iterations	Time/Iter	Ratio Time	Ratio Iter
5,1	143.63	3	47.88	126.30	3	42.10	1.14	1.00
5,2	847.95	16	53.00	330.65	8	41.33	2.56	2.00
5,3	1709.99	26	65.77	261.67	6	43.61	6.53	4.33
5,4	3423.58	42	81.51	647.48	14	46.25	5.29	3.00
5,5	11839.23	83	142.64	1159.75	23	50.42	10.21	3.61
5,6	22710.71	103	220.49	1202.50	23	52.28	18.89	4.48
5,7	47330.62	123	384.80	1424.63	27	52.76	33.22	4.56
5,8	466326.02	184	2534.38	1588.05	30	52.93	293.65	6.13
5,9	$\dagger$	$\dagger$	$\dagger$	2186.99	41	53.34		
5,10	$\dagger$	$\dagger$	$\dagger$	2285.42	40	57.14		

3.3. Scalability with Respect to Number of Scenarios $|\mathcal{K}|$

In this set of experiments, we fixed $h = 5$ and increased the number of attack scenarios in $\mathcal{K}$ from one to ten. The attack scenarios were constructed in a nested fashion, as shown in Table 4, such that scenarios were only added as $|\mathcal{K}|$ increases. We present experimental results in Tables 5 - 7.

For all test cases, adding a scenario does not necessarily increase runtime or the number of iterations for both algorithms. For example, in the IEEE 118-bus test case, both runtime and iteration count decrease from $|\mathcal{K}| = 7$ to $|\mathcal{K}| = 8$ for both algorithms. Across the three test cases, the CCG algorithm generally requires more iterations as $|\mathcal{K}|$ increases, whereas the HOPS algorithm generally requires fewer iterations to converge. We believe this behavior is because the HOPS algorithm is able to more efficiently lower bound scenarios without explicitly evaluating them as frequently in the NDMIP.

For smaller attacker scenario sets, the HOPS algorithm exhibits mixed performance compared to the CCG algorithm. For $|\mathcal{K}| = 1$ the runtime ratio of the HOPS algorithm relative to the CCG algorithm ranges from .59 to 1.78. By $|\mathcal{K}| = 3$ the HOPS algorithm outperforms for all test cases by a runtime factor of 1.19 to 8.7 times compared to the CCG algorithm. As the number of scenarios increases, the HOPS algorithm shows significantly increased scalability compared to the CCG algorithm. For $|\mathcal{K}| = 10$, the HOPS algorithm outperforms with 1.89 to 55.52 shorter runtimes. Consistent with the trends observed in previous experiments, these results suggest that HOPS scales relatively steadily across the test cases considered, while CCG appears to scale comparatively well on the IEEE 118-bus case but less favorably on the more challenging instances.

Table 4: Nested construction of attack budget scenarios. The FP-SU then solves $\mathcal{A}^k \forall k \in \mathcal{K}$.

$ \mathcal{K} $	1	2	3	4	5	6	7	8	9	10
Scenarios	{10}	{1, 10}	{1, 5, 10}	{1, 5, 7, 10}	{1, 3, 5, 7, 10}	{1, 3, 5, 7, 8, 10}	{1, 2, 3, 5, 7, 8, 10}	{1, 2, 3, 4, 5, 7, 8, 10}	{1, 2, 3, 4, 5, 6, 7, 8, 10}	{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

Table 5: Runtime and number of iterations to converge of the CCG and HOPS algorithms on the IEEE 118-bus test case as the number of attack scenarios is increased.

$ \mathcal{K} , h$	CCG			HOPS			CCG/HOPS	
	Time (s)	Iterations	Time/Iter	Time (s)	Iterations	Time/Iter	Ratio Time	Ratio Iter
1,5	2937.81	25	117.51	4951.50	52	95.22	0.59	0.48
2,5	4119.21	34	121.15	4934.03	42	117.48	0.83	0.81
3,5	4514.28	33	136.80	3801.44	28	135.77	1.19	1.18
4,5	5604.44	31	180.79	5195.76	28	185.56	1.08	1.11
5,5	6076.74	32	189.90	4166.23	22	189.37	1.46	1.45
6,5	10565.45	39	270.91	6455.20	26	248.28	1.64	1.50
7,5	9855.71	36	273.77	6417.78	25	256.71	1.54	1.44
8,5	7479.14	26	287.66	5373.99	20	268.70	1.39	1.30
9,5	9956.80	30	331.89	6026.12	21	286.96	1.65	1.43
10,5	16346.36	38	430.17	8664.71	20	433.24	1.89	1.90

Table 6: Runtime and number of iterations to converge of the CCG and HOPS algorithms on the IEEE 300-bus test case as the number of attack scenarios is increased.

$ \mathcal{K} , h$	CCG			HOPS			CCG/HOPS	
	Time (s)	Iterations	Time/Iter	Time (s)	Iterations	Time/Iter	Ratio Time	Ratio Iter
1,5	13039.74	76	171.58	7305.79	69	105.88	1.78	1.10
2,5	10864.00	56	194.00	5385.01	50	107.70	2.02	1.12
3,5	41246.21	94	438.79	4739.77	39	121.53	8.70	2.41
4,5	62927.41	95	662.39	5952.45	32	186.01	10.57	2.97
5,5	71362.48	94	759.18	5863.75	31	189.15	12.17	3.03
6,5	100305.79	94	1067.08	6955.45	26	267.52	14.42	3.62
7,5	89302.06	87	1026.46	7209.37	26	277.28	12.39	3.35
8,5	132180.14	94	1406.17	6808.53	23	296.02	19.41	4.09
9,5	128156.80	80	1601.96	8144.56	25	325.78	15.74	3.20
10,5	494545.00	95	5205.74	8907.54	22	404.89	55.52	4.32

Table 7: Runtime and number of iterations to converge of the CCG and HOPS algorithms on the RTS-GMLC bus test case as the number of attack scenarios is increased.

$ \mathcal{K} , h$	CCG			HOPS			CCG/HOPS	
	Time (s)	Iterations	Time/Iter	Time (s)	Iterations	Time/Iter	Ratio Time	Ratio Iter
1,5	1181.82	41	28.82	705.40	32	22.04	1.68	1.28
2,5	1772.39	48	36.92	885.17	38	23.29	2.00	1.26
3,5	3694.27	60	61.57	966.61	31	31.18	3.82	1.94
4,5	6666.90	68	98.04	883.14	20	44.16	7.55	3.40
5,5	12302.55	83	148.22	958.88	18	53.27	12.83	4.61
6,5	10770.15	66	163.18	1155.56	18	64.20	9.32	3.67
7,5	22050.69	90	245.01	1118.18	17	65.78	19.72	5.29
8,5	22293.38	84	265.40	1135.84	15	75.72	19.63	5.60
9,5	37814.40	103	367.13	1407.80	16	87.99	26.86	6.44
10,5	47863.96	102	469.25	1638.65	16	102.42	29.21	6.38

3.4. Sensitivity Analysis of Optimal Defenses to Scenario Set Reduction

In this set of experiments, we analyze solution quality when solving the FP-SU with a subset of scenarios. This experiment is motivated by practical settings in which data availability or computational resources may limit the number of scenarios considered during optimization. We fixed $h = 5$ and $|\mathcal{K}| = 10$, with scenarios defined as in Table 4. Consider a subset of attack scenarios $U \subseteq \mathcal{K}$. Let D_U^* be the optimal defense obtained by solving the FP-SU($\mathcal{A}^F, \mathcal{D}^h, \mathcal{C}, U$). To evaluate the performance of D_U^* , we evaluate it across all scenarios by solving $\sum_{k \in \mathcal{K}} \max_{A \in \mathcal{A}^k} C(A, D_U^*)$ where D_U^* is fixed. We consider scenario subsets $|U| = 1$ and $|U| = 5$ from Table 4. For comparison, we also evaluate the no-defense baseline D_0^* , in which no defense actions are taken.

Results for all test cases are shown in Figures 1–3. Each point corresponds to an attacker budget k and the associated cost $\max_{A \in \mathcal{A}^k} C(A, D)$ for a fixed defense. Overall, we find that a small number of scenarios is often sufficient to recover defenses that perform near-optimally across all attacker budgets for these test cases. Although adding more

scenarios typically improves performance, the FP-SU formulation occasionally over-tailors the defense to the selected scenario set, resulting in degraded performance for excluded attacker budgets.

For the RTS-GMLC test case, the defense D_1^* obtained from a single scenario coincides with the defense D_{10}^* obtained from all scenarios. This behavior may be attributable to a substantial overlap in the effective attack patterns across attacker budgets relative to the other test cases. We also observe that the D_5^* solution is overly tailored to the included scenarios: the optimal defense yields less load shed for the included scenario $k = 3$, $k = 5$, and $k = 7$, but performs worse for excluded scenarios $k = 6$, $k = 8$, and $k = 9$. In contrast, the IEEE 118-bus and IEEE 300-bus test cases exhibit a more regular pattern. For these test cases, the defense D_1^* is near-optimal and the defense D_5^* improves further and matches the optimal solution.

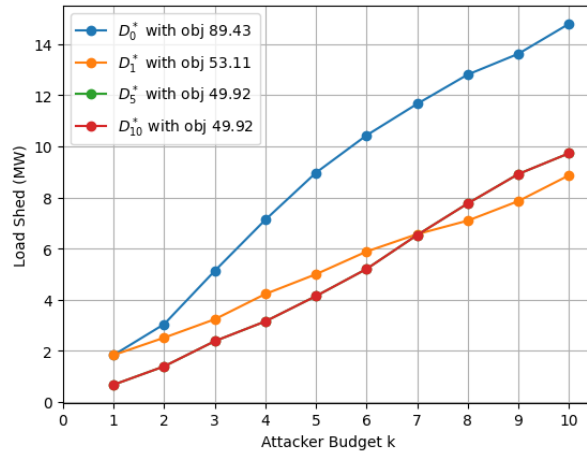


Figure 1: A comparison of solving the FP-SU on reduced scenario sets compared to all scenarios for the IEEE 118-bus test case.

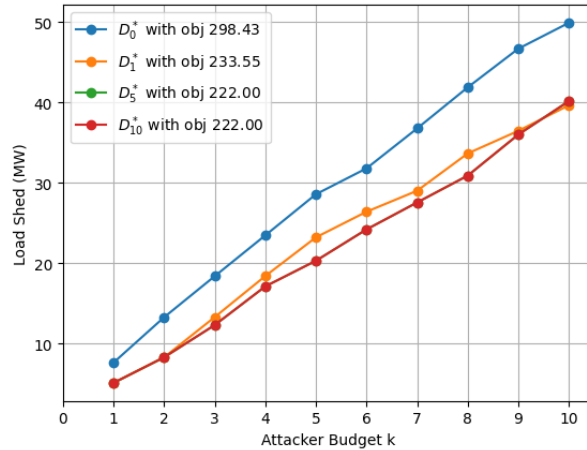


Figure 2: A comparison of solving the FP-SU on reduced scenario sets compared to all scenarios for the IEEE 300-bus test case.

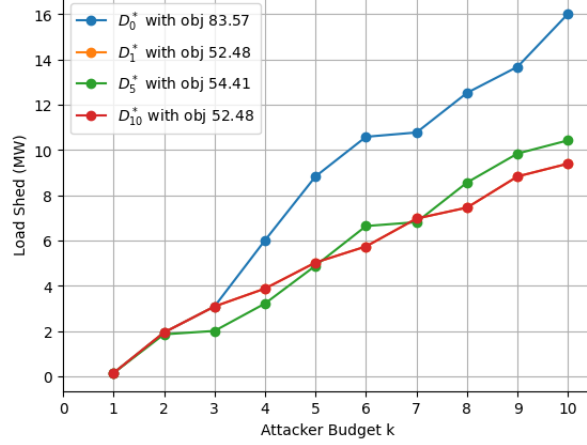


Figure 3: A comparison of solving the FP-SU on reduced scenario sets compared to all scenarios for the RTS-GMLC test case.

4. Future Research

Several directions for future research arise from this work. From an algorithmic standpoint, the HOPS framework could be combined with existing strengthening cuts for DAD problems such as those introduced in [4]. In addition, the structure of the FP-SU is well suited for the design of parallel algorithms. Another promising direction is the development of principled scenario reduction methods for the FP-SU. Our computational results indicate that only a small number of attack budget scenarios may be sufficient to recover near-optimal defenses, motivating both heuristic and theoretically grounded approaches for selecting representative scenario subsets. Finally, several theoretical questions remain open, including characterizing the additional computational complexity introduced by interdiction budget uncertainty relative to the deterministic fortification problem.

References

- [1] G. G. Brown, W. M. Carlyle, J. Salmerón, and K. Wood, “Analyzing the Vulnerability of Critical Infrastructure to Attack and Planning Defenses,” in *Emerging Theory, Methods, and Applications*, ser. INFORMS TutORials in Operations Research. INFORMS, Sep. 2005, pp. 102–123. [Online]. Available: <https://pubsonline.informs.org/doi/abs/10.1287/educ.1053.0018>
- [2] J. C. Smith and Y. Song, “A survey of network interdiction models and algorithms,” *European Journal of Operational Research*, vol. 283, no. 3, pp. 797–811, Jun. 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0377221719305156>
- [3] L. Lozano and J. C. Smith, “A Backward Sampling Framework for Interdiction Problems with Fortification,” *INFORMS Journal on Computing*, vol. 29, no. 1, pp. 123–139, Jan. 2017. [Online]. Available: <https://pubsonline.informs.org/doi/10.1287/ijoc.2016.0721>
- [4] A. Mastin, A. Baxter, A. Musselman, and J.-P. Watson, “Best Response Intersection: An Optimal Algorithm for Interdiction Defense,” *INFORMS Journal on Optimization*, vol. 5, no. 2, pp. 172–190, Apr. 2023. [Online]. Available: <https://pubsonline.informs.org/doi/abs/10.1287/ijoo.2022.0081>

- [5] Y. Xiang and L. Wang, “An Improved Defender–Attacker–Defender Model for Transmission Line Defense Considering Offensive Resource Uncertainties,” *IEEE Transactions on Smart Grid*, vol. 10, no. 3, pp. 2534–2546, May 2019. [Online]. Available: <https://ieeexplore.ieee.org/document/8286941>
- [6] T. Zang, Y. Xiao, Y. Liu, S. Wang, Z. Wang, and Y. Zhou, “Defense Strategy Against Cyber Attacks on Substations Considering Attack Resource Uncertainty,” *Journal of Modern Power Systems and Clean Energy*, vol. 13, no. 4, pp. 1335–1346, Jul. 2025. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/10847935>
- [7] Y. Xiang, X. Zhang, D. Shi, Y. Jin, Z. Wang, and L. Wang, “A Robust Power Grid Defense Model Considering Load Demand and Wind Generation Uncertainties,” in *2018 IEEE Power & Energy Society General Meeting (PESGM)*, Aug. 2018, pp. 1–5. [Online]. Available: <https://ieeexplore.ieee.org/document/8585823>
- [8] C. Barrows, A. Bloom, A. Ehlen, J. Ikäheimo, J. Jorgenson, D. Krishnamurthy, J. Lau, B. McBennett, M. O’Connell, E. Preston, A. Staid, G. Stephen, and J.-P. Watson, “The IEEE Reliability Test System: A Proposed 2019 Update,” *IEEE Transactions on Power Systems*, vol. 35, no. 1, pp. 119–127, Jan. 2020. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/8753693>
- [9] S. Babaeinejadsarookolaee, A. Birchfield, R. D. Christie, C. Coffrin, C. DeMarco, M. Ferris, S. Greene, R. Huang, C. Jozs, R. Korab, B. Lesieutre, J. Maeght, T. W. K. Mak, D. K. Molzahn, T. J. Overbye, P. Panciatici, B. Park, J. Snodgrass, A. Tbaileh, P. V. Hentenryck, and R. Zimmerman, “The Power Grid Library for Benchmarking AC Optimal Power Flow Algorithms,” Jan. 2021, arXiv:1908.02788. [Online]. Available: <http://arxiv.org/abs/1908.02788>
- [10] M. L. Bynum, G. A. Hackebeil, W. E. Hart, C. D. Laird, B. L. Nicholson, J. D. Siirola, J.-P. Watson, and D. L. Woodruff, *Pyomo — Optimization Modeling in Python*, ser. Springer Optimization and Its Applications. Springer International Publishing, 2021, vol. 67. [Online]. Available: <http://link.springer.com/10.1007/978-3-030-68928-5>