ABSTRACT

An adversary looking to disrupt a power grid may look to target certain substations and sources of power generation to initiate a cascading failure that maximizes the number of customers without electricity. This is particularly an important concern when the enemy has the capability to launch cyber-attacks as practical concerns (i.e. avoiding disruption of service, presence of legacy systems, etc.) may hinder security. Hence, a defender can harden the security posture at certain power stations but may lack the time and resources to do this for the entire power grid. We model a power grid as a graph and introduce the cascading failure game in which both the defender and attacker choose a subset of power stations such as to minimize (maximize) the number of consumers having access to producers of power. We formalize problems for identifying both mixed and deterministic strategies for both players, prove complexity results under a variety of different scenarios, identify tractable cases, and develop algorithms for these problems. We also perform an experimental evaluation of the model and game on a real-world power grid network. Empirically, we noted that the game favors the attacker as he benefits more from increased resources than the defender. Further, the minimax defense produces roughly the same expected payoff as an easy-to-compute deterministic load based (DLB) defense when played against a minimax attack strategy. However, DLB performs more poorly than minimax defense when faced with the attacker’s best response to DLB. This is likely due to the presence of low-load yet high-payoff nodes, which we also found in our empirical analysis.

Categories and Subject Descriptors
I.2.11 [Artificial Intelligence]: Distributed Artificial Intelligence

General Terms
Algorithms Security

Keywords
power grid defense, game theory, complex networks

Copyright © 2014, International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.
An adversary looking to disrupt a power grid may look to target certain substations and sources of power generation to initiate a cascading failure that maximizes the number of customers without electricity. This is particularly an important concern when the enemy has the capability to launch cyber-attacks as practical concerns (i.e. avoiding disruption of service, presence of legacy systems, etc.) may hinder security. Hence, a defender can harden the security posture at certain power stations but may lack the time and resources to do this for the entire power grid. We model a power grid as a graph and introduce the cascading failure game in which both the defender and attacker choose a subset of power stations such as to minimize (maximize) the number of consumers having access to producers of power. We formalize problems for identifying both mixed and deterministic strategies for both players, prove complexity results under a variety of different scenarios, identify tractable cases, and develop algorithms for these problems. We also perform an experimental evaluation of the model and game on a real-world power grid network. Empirically, we noted that the game favors the attacker as he benefits more from increased resources than the defender. Further, the minimax defense produces roughly the same expected payoff as an easy-to-compute deterministic load-based (DLB) defense when played against a minimax attack strategy. However, DLB performs more poorly than minimax defense when faced with the attacker’s best response to DLB. This is likely due to the presence of low-load yet high-payoff nodes, which we also found in our empirical analysis.
<table>
<thead>
<tr>
<th>16. SECURITY CLASSIFICATION OF:</th>
<th>17. LIMITATION OF ABSTRACT</th>
<th>18. NUMBER OF PAGES</th>
<th>19a. NAME OF RESPONSIBLE PERSON</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. REPORT</td>
<td>Same as Report (SAR)</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>unclassified</td>
<td>unclassified</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b. ABSTRACT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>unclassified</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>c. THIS PAGE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>unclassified</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Standard Form 298 (Rev. 8-98)  
Prescribed by ANSI Std Z39-18
2. TECHNICAL PRELIMINARIES

Consider a power-grid network modeled as an undirected graph $G = (V, E)$. Let $V_{src}, V_{dst} \subseteq V$ be source (producers of power) and load (consumers of power) on the network. We shall use the notation $\text{disc}_{V_{src}, V_{dst}}(G)$ to denote the number of nodes in $V_{dst}$ which are not connected to any node in $V_{src}$ in graph $G$. Let $G$ be the set of all subgraphs of $G$. For a given node $i$, let $\mathcal{N}_G(i)$ be the set of nodes in $V_{src} - \{i\}$ that are closest to that node (based on path length in $G$).

From this, we define edge load (similar to the idea of edge betweenness [25]).

**Definition 2.1 (Edge Load).** Given edge $ij \in E$, the edge load, $\text{load}_{ij}(ij)$ is defined as follows:

$$\text{load}_{ij}(ij) = \sum_{t \in V_{src}} \sum_{s \in \mathcal{N}_G(t)} \frac{\sigma_G(s, t) \sigma_G(s, t)}{|\mathcal{N}_G(t)|},$$

where $\sigma_G(s, t)$ is the number of shortest paths between $s, t \in V$ and $\sigma_G(s, t)$ is the subset of these paths that pass through edge $ij \in E$.

Starting from initial network $G_0 = (V_0, E_0)$ we use $c_{ij}$ to denote the capacity edge $ij \in E_0$. In a real-world setting, we would expect to have this information. However, in this paper, we use the following proxy (similar to [8]).

$$c_{ij}(G_0) = (1 + \alpha) \text{load}_{ij}(ij)$$

where $\alpha$ is a non-negative real that specifies the excess capacity available on that line. We shall refer to $\alpha$ as the capacity margin. We assume that an edge $ij \in E$ fails in $G = (V, E)$, with $E \subseteq E_0$, if $\text{load}_{ij}(ij) > c_{ij}(G_0)$. Once nodes (and adjacent edges) in $V_0$ are removed from $G_0$, this results in a change of shortest paths between sources and loads, hence more edges will potentially fail. This cascading power failure is modeled by a “failure” operator denoted with $F$ (based on the failure model of [8] - though we note that our model is a new contribution due to the consideration of source and load nodes) that maps networks to networks. We define it as follows.

**Definition 2.2 (Failure Operator).** The failure operator, $F : G \rightarrow G$, is defined as follows:

$$F((V, E)) = (V, \{ij \in E | \text{load}_{ij}(ij) \leq c_{ij}(G_0)\})$$

Intuitively, one application of the failure operator removes all edges that have exceeded their maximum capacity. We can define multiple applications of this operator as follows:

$$F^i(G) = \begin{cases} G & \text{if } i = 0 \\ F(F^{i-1}(G)) & \text{otherwise} \end{cases}$$

Clearly, there must exist a fixed point that is reached in no more than $|E| + 1$ applications of $F$. Hence, we shall use the following notation:

$$F^*(G) = F^i(G) \text{ s.t. } F^i(G) = F^{i+1}(G)$$

We now consider two agents: an attacker and a defender. The attacker’s strategy is to destroy nodes (and their adjacent edges) in an effort to cause a cascading failure that maximizes the number of load nodes ($V_{dst}$) that are disconnected from all source nodes ($V_{src}$). Meanwhile, the defender’s strategy is to harden certain nodes such that the attacker is unable to destroy them - though these nodes can be taken offline as a result of the cascading failure\(^1\). The attacker can destroy $k_a$ nodes while the defender can harden $k_d$ nodes. Thus the strategy space of both the attacker and defender consists of all subsets $V_a, V_d \subseteq V$ of size $|V_a| \leq k_a$ ($|V_d| \leq k_d$ respectively). We denote these strategy spaces by $ATK (DEF)$ respectively, i.e., if we allow the attacker to consider all strategies of size $k_a$ or less we have:

$$ATK = \{S \in 2^V : |S| \leq k_a\}$$

We now have all of the components to define the payoff function.

**Definition 2.3 (Payoff Function).** Given initial network $G = (V, E)$ with edge capacities $c_{ij}(G)$, attack (defend) strategy $V_a(V_d)$, the payoff function is defined by

$$p_G(V_a, V_d) = \text{disc}_{V_{src}, V_{dst}}(F^*((V - (V_a - V_d), E))).$$

Now, in reality, the defender will have real-world limitations on the number of nodes (i.e. substations) he may harden. For instance, with regard to smart grid defense, applying the most up-to-date patches on all systems may not be realistic as it could potentially require system down-time affecting customer service. Further, it would also likely not make sense for the defender to only harden certain nodes and ignore others. Hence, it is reasonable to consider a situation where the defender can only harden certain nodes against attack (and may do so probabilistically - i.e. applying hardware or software updates according to a schedule). Therefore, we study mixed strategies. Such strategies will be specified by probability distributions $Pr_a, Pr_d$ for the attacker and defender respectively. We shall denote the number of strategies assigned a non-zero probability as $|Pr_a|, |Pr_d|$. We can define expected payoff as follows.

**Definition 2.4 (Expected Payoff).** Let $Pr_a, Pr_d$ be probability distributions over all subsets of $V$ of sizes $k_a$ (resp. $k_d$) or less. These probability distributions correspond to a mixed strategy for the attacker and defender respectively. Hence, given such probability distributions, the expected payoff can be computed as follows:

$$\text{ExP}(Pr_a, Pr_d) = \sum_{V_a \in 2^V} \sum_{V_d \in 2^V} Pr_a(V_a) Pr_d(V_d) p_G(V_a, V_d)$$

In this work our goal is to find the minimax strategy for the defender - that is the mixed strategy for the defender that minimizes the attacker’s maximum expected payoff - as well as deterministic “best responses” for both players given the other’s strategy.

3. COMPUTATIONAL COMPLEXITY

In this section, we analyze the computational complexity of determining the best response for each of the agents to a strategy of its opponent. First, we shall discuss the case for finding a deterministic strategy for the defender and attacker. Then we shall explore the computational complexity of finding a mixed strategy. We summarize our complexity results in Table 3.

\(^1\)Note that this would likely be the case where the attack and defense occurs in cyber-space, while the cascade occurs in the physical world.
<table>
<thead>
<tr>
<th>Opponent Strategy</th>
<th>Attacker</th>
<th>Defender</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mixed w. 1 resource</td>
<td>NP-Compl.</td>
<td>PTIME</td>
</tr>
<tr>
<td>Thm. 3</td>
<td>Prop. 3.2</td>
<td></td>
</tr>
<tr>
<td>Det. w. fewer resources</td>
<td>NP-Compl.</td>
<td>PTIME</td>
</tr>
<tr>
<td>Thm. 3</td>
<td>Prop. 3.1</td>
<td></td>
</tr>
<tr>
<td>Det. w. greater resources</td>
<td>NP-Compl.</td>
<td>NP-Compl.</td>
</tr>
<tr>
<td>Thm. 3</td>
<td>Thm. 1</td>
<td></td>
</tr>
<tr>
<td>Mixed w. fewer resources</td>
<td>NP-Compl.</td>
<td>NP-Compl.</td>
</tr>
<tr>
<td>Thm. 3</td>
<td>Thm. 2</td>
<td></td>
</tr>
<tr>
<td>Mixed w. greater resources</td>
<td>NP-Compl.</td>
<td>NP-Compl.</td>
</tr>
<tr>
<td>Thm. 3</td>
<td>Thm. 1</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Complexity Results for Finding a Deterministic Best Response

We frame the formal combinatorial problem of finding the best-response for the defender as follows:

**Grid-Defend Deterministic Best Response (GD-DBR)**

**INPUT:** Network $G = (V, E)$, attacker mixed strategy $\mathbf{Pr}_a$ (where each option is of size no greater than $k_a$), natural number $k_d$, real numbers $X, \alpha$

**OUTPUT:** “Yes” if there exists a set $V_d \subseteq V$ s.t. $|V_d| \leq k_d$ and $\sum_{v_d \in V_d} \mathbf{Pr}_a(v_d) pc(V_d, V_d) \leq X$ and “no” otherwise.

We shall study this case under several conditions. The first, and easiest case is when $\mathbf{Pr}_a = 1$ (the attacker uses a deterministic strategy) and $k_a \leq k_d$.

**Proposition 3.1.** When $k_a \leq k_d$ and $|\mathbf{Pr}_a| = 1$ then GD-DBR is solvable in polynomial time.

**Proof.** As the attacker plays only one strategy and the defender can defend at least as many nodes as are being attacked, the defender simply defends all the nodes in the attacker’s strategy. \(\blacksquare\)

However, even with $|\mathbf{Pr}_a| = 1$, the problem becomes NP-hard in the case where $k_a > k_d$.

**Theorem 1.** When $k_a > k_d$ then GD-DBR is NP-complete, even when $|\mathbf{Pr}_a| = 1$ and $X$ is an integer.

**Proof.** Clearly, checking if a given deterministic defender strategy $V_d$ meets the requirements of the “output” of GD-DBR can be completed in polynomial-time, providing membership in the class NP.

For NP-hardness consider the known NP-hard “set cover” problem [11] that takes as input a natural number $k$, set of elements $S = \{s_1, \ldots, s_n\}$, family of subsets of $S$, $H = \{h_1, \ldots, h_n\}$ and returns “yes” if there is a $k$-sized (or smaller) subset of $H$ s.t. their union is equal to $S$. We can embed Set Cover into an instance of GD-DBR in polynomial time with the following embedding: set $k_a = |H|$, $k_d = k$, $X = 0$, $\alpha = |H| + |S|$, create $G = (V, E)$ as follows:

- For each $h \in H$ create a node $v_h$ and for each $s \in S$ create node $v_s$.
- For each $s \in h$, create edge $(v_h, v_s)$, for each $ij \in E$.
- Set $V_{src} = \{v_h | h \in H\}$, $V_d = \{v_s | s \in S\}$, $V = V - V_{did}$.

Suppose, by way of contradiction (BWOC), that there is a “yes” answer to Set Cover but a “no” answer to GD-DBR. Consider set $H$’s subset of $H$ that is the certificate for Set Cover and the corresponding set $V' = \{v_h | h \in H'\}$ in the instance of GD-DBR. Suppose the defender utilizes this as a strategy. The attacker then effectively attacks the set $V - V_d - V'$. Note that as the graph is bi-bipartite, this does not cause any cascading failure. By the construction, each load node must be connected to a source node, hence the number of offline load nodes is $X$. This gives us a contradiction.

Suppose, BWOC, that there is a “yes” answer to GD-DBR but a “no” answer to the corresponding instance of Set Cover. Let $V'$ be the certificate for GD-DBR. We note that any element of $V_d \cap V'$ in $V'$ can be replaced by a neighboring node from $V_{src}$ without changing the size of this set and that such a set would still allow for all load nodes to remain online, let $V''$ be this new set. Consider the set $\{h | v_h \in V''\}$. By the contra-positive of the claim, this cannot be a cover of all elements of $S$. However, this would also imply that there is some element $v_s \in V_d$ that is not connected to $V''$ meaning that it fails (as the attacker successfully destroys all its neighbors). This means that the adversary has a payoff greater than 0 (which is what $X$ was set to) – hence a contradiction. \(\blacksquare\)

Hence, the presence of a more advantageous attacker is a source of complexity. The next question would be if the attacker’s behavior, i.e. deterministic vs. non-deterministic, also affects the complexity of the problem, even if the defender has the advantage. First, let us examine the case where the attacker has a mixed strategy with $k_a = 1$.

**Proposition 3.2.** When $k_a = 1$ then GD-DBR is solvable in polynomial time (w.r.t. $|\mathbf{Pr}_a|$), even when $|\mathbf{Pr}_a| \geq 0$.

**Proof.** In this case, we can re-write the payoff function as $pc_G(v) V_d = 0$ if $v \not\in V_d$ and $pc_G(v) V_d = pc_G(\{v\}, \emptyset)$ otherwise. Let $V' = \cup \{V_a \in ATK(\mathbf{Pr}_a(V_a) > 0\}$. Note that each element of $V'$ also is a strategy the attacker plays with a non-zero probability (as the attacker only plays singletons). Hence, the expected payoff can be re-written as $\sum_{v \in V'} \mathbf{Pr}_a(\{v\}) pc_G(\{v\}, \emptyset)$. Therefore, the best a defender can do is defend the top $k_d$ nodes in $V'$ where $\mathbf{Pr}_a(\{v\}) pc_G(\{v\}, \emptyset)$ is the greatest - which can be easily computed in polynomial time and allows us to determine the answer to GD-DBR. \(\blacksquare\)

However, if the defender is playing a mixed strategy with $k_a > 1$, then the problem again becomes NP-complete.

**Theorem 2.** When $|\mathbf{Pr}_a| > 1$ and $k_a > 1$, GD-DBR is NP-complete, even when $k_d > k_a$ and $X$ is an integer.

**Proof.** NP-completeness mirrors that of Theorem 1. For NP-hardness, we again consider a reduction from set-cover (defined in the proof of Theorem 1). The embedding can again be performed in polynomial time as follows: set $k_a = \max_{s \in S} |\{h | s \in h\}|$, set $k_d = k$, $X = 0$, $\alpha = |H| + |S|$, create $G = (V, E)$, $V_{src}$, and $V_{did}$ as per the construction in Theorem 1. We then set up the mixed strategy as follows: for each $s \in S$, set $V_s = |\{h | s \in h\}|$ and $\mathbf{Pr}_a(V_s) = 1/|S|$. Supposing, BWOC, that there is a “yes” answer to set cover and a “no” answer to the instance of GD-DBR. Consider set cover solution $H'$ and set $V_d = \{v_h | h \in H'\}$. Note that $V_d$ meets the cardinality requirement. Note that by the construction, a source node becomes disconnected only if all of the load nodes connected to it are attacked, hence there is some node in the set $V_{did}$ that is totally disconnected under at least one attacker strategy - let $v_h$ be this node.
However, as set $H'$ covers $S$, then regardless of the attacker strategy, there is always some node $v_h$ that is connected and never attacked (giving the attacker a payoff of zero) - hence a contradiction.

Suppose, BWOC, that there is a “yes” answer to GD-DBR and a “no” answer to the instance of set cover. Consider GD-DBR solution $V'$. We note that any element of $V_{id} \cap V'$ in $V'$ can be replaced by a neighboring node from $V_{src}$ without changing the size of this set and that such a set would still allow for all load nodes to remain online, let $V''$ be this new set. Consider the set $H' = \{ v_i | v_i \in V'' \}$. Note that $|H'| \leq k$. By the contra-positive, there must be at least one element of $S$ not covered by $H'$.

Let node $v_i$ be a node associated with uncovered element $s$. As GD-DBR returned “yes” then there is no attacker strategy where $v_i$ becomes disconnected from some node in $V_{src}$. As attack strategy $V_{a}''$ includes all nodes that are connected to $v_i$, then at least one of these nodes must be included in $V''$. Therefore, for every node $v_i \in V_{id}$ there is some node $v_h \in V_{id} \cap V''$ that is connected to it, which means, by the construction, that $H'$ must cover all elements of $S$ - a contradiction.}

Due to the use of covering problems for the complexity results in Theorems 1, 2, and 3, it may seem reasonable to frame the problem as a sub- or super-modularity optimization where the objective function is monotonic. However, here we show (unfortunately) that these properties do not hold for either player. First, we shall make statements regarding the monotonicity of the payoff function.

**Proposition 3.3.** If $\forall V_d$, $V_a \subseteq V_d$ : $pc(V_a, V_d) \leq pc(V_a, V_d')$ then $\forall V_a^*$, $V_a \subseteq V_d'$ : $pc(V_a^*, V_d') \geq pc(V_a^*, V_d)$.

The idea of submodularity can be thought of as “diminishing returns.” Given a set of elements $S$ and a function $f: 2^S \to \mathbb{R}$, we say a $f$ is submodular if for any sets $S_1 \subseteq S_2$ and element $s \notin S_2$, we have the following relationship:

$$f(S_1 \cup \{s\}) - f(S_1) \geq f(S_2) - f(S_2 - \{s\})$$

A complementary idea of supermodularity is also often studied - in this case the inequality is reversed. Unfortunately, when we fix the strategy for the defender, the attacker strategy is neither submodular nor supermodular - making the dynamics of this model significantly different from others (i.e. [24]). Let consider strategies $V_{a}, V_{d}$ where $V_{a}$ causes some load node $v \notin (V_{a} \cup V_{d}) \cap V_{id}$ to disconnect and any node the strategy $\{v\}$ causes to disconnect will also become disconnected with strategy $V_{a}$ (such a case is easy to contrive, particularly with a bi-partite network). Therefore, we get the following relationship:

$$pc(V_a \cup \{v\}, V_d) - pc(V_a, V_d) < pc(\{v\}, V_d) - pc(\emptyset, V_d)$$

This arises from the fact that the left-hand side of the above equation becomes zero and the right hand side of the equation is equal to $pc(\{v\}, V_d)$ which must be at least one. Now consider another example. Suppose we have a simple V-shaped network of three nodes. The angle of the V is a load node, while the other two nodes are source nodes. With $\alpha = 1$, the load node receives power if at least one of the source nodes is connected to it. However, it does not require both. Let $V_{a}$ be a strategy consisting of one source node and $v$ be the other source node, and $V_d$ consist of the load node. From this, we have the following relationship:

$$pc(V_a \cup \{v\}, V_d) - pc(V_a, V_d) > pc(\{v\}, V_d) - pc(\emptyset, V_d)$$

In this case, the right-hand side becomes zero while the left hand side becomes one. This leads us to the following fact:

**Fact 3.2.** When $V_d$ is fixed, $pc$ is neither submodular nor supermodular.

Now let us consider when we fix the attacker’s strategy. If the payoff is submodular when the attacker’s strategy is fixed, then we have the following for $V_a \subseteq V_d'$ and $v \notin V_d'$ if
the payoff subtracted from the number of nodes is submodular:
\[ pc(V_a, V'_a \cup \{v\}) - pc(V_a, V'_a) \geq pc(V_a, V_a \cup \{v\}) - pc(V_a, V_a) \]
This is equivalent to the following:
\[ pc(V_a - (V'_a \cup \{v\}), \emptyset) - pc(V_a - V'_a, \emptyset) \geq pc(V_a - (V_a \cup \{v\}), \emptyset) - pc(V_a - V_a, \emptyset) \]
Now let \( V''_a = V_a - (V'_a \cup \{v\}) \) and \( V''''_a = V'_a \cup (V'_a - V_a) \). Clearly \( V''''_a \supseteq V''_a \) and \( v \notin V''''_a \). Now we get the following:
\[ pc(V''_a, \emptyset) - pc(V''_a \cup \{v\}, \emptyset) \geq pc(V''''_a, \emptyset) - pc(V''''_a \cup \{v\}, \emptyset) \]
\[ pc(V''_a \cup \{v\}, \emptyset) - pc(V''_a, \emptyset) \leq pc(V''''_a \cup \{v\}, \emptyset) - pc(V''''_a, \emptyset) \]
Hence, submodularity of the payoff function when the attacker’s strategy is fixed would give us supermodularity of the payoff function when the defender’s strategy is fixed at the empty set. However, this clearly violates Fact 3.2 and gives rise to the following:

**Fact 3.3.** When \( V_a \) is fixed, \( pc \) is neither submodular nor supermodular.

### 4. Algorithms

In this section, we present heuristic algorithms for finding the deterministic best response of each player as the results of the previous section generally preclude a polynomial time algorithm for an exact solution. We first introduce a version of a “high load” strategy for the defender based on the ideas of [8]. Then we introduce a greedy heuristic for each player. This is followed by our approach for finding mixed strategies based on the double-oracle algorithm of [15].

**Hi-Load Node Approach.** In [8], the authors study “high load” nodes: nodes through which the greatest number of shortest paths pass. They show that attacks on these nodes tend to initiate cascading failures – suggesting that they should be a priority for defense. We formalize the definition of nodal load in our framework (essentially an extended definition of node betweenness [25]) by extending our function \( load_{\mathcal{C}} \) for nodes as follows.

**Definition 4.1 (Nodal Load).** For a given node, the **nodal load** is defined as the sum of the fraction of shortest paths for each pair that pass through that node. Formally:
\[
load_{\mathcal{C}}(i) = \sum_{s \in V_{src}, t \in V_{ld}} \frac{\sigma_{\mathcal{C}}(s,t|i)}{\sigma_{\mathcal{C}}(s,t)},
\]
where \( \sigma_{\mathcal{C}}(s,t|i) \) is the number of shortest paths between \( s, t \in V \) that pass through node \( i \).

Hence, we shall refer to the **Deterministic Load-Based** or **DLB** strategy for the defender as one in which he deterministically protects the \( k_d \) nodes with the greatest load. We note that this is not necessarily a “best response” but the intuition is that defense will occur at nodes that are perceived to be critical to the adversary. This intuition is similar to that of the “most vital arc” idea seen in other failure model games [2, 21].

**Greedy Heuristics for Finding Deterministic Strategies.** Here we present a simple greedy heuristic to find the defender’s best-response (**GREEDY_DEFENDER_RESP**). The analogous heuristic for the attacker is not shown due to space constraints, but we shall refer to it as **GREEDY_ATTACKER_RESP**. We note that while we do not make general approximation guarantees (due to the results in Section 3), we note that by Proposition 3.3, that nodes added in step 18 will always cause an increase in payoff to the defender (and in the analogous greedy approach for the attacker, this holds true as well). Further, by Proposition 3.2, when \( k_a \), we can be sure that **GREEDY_DEFENDER_RESP** returns an exact solution, even when the attacker has a mixed strategy. Unfortunately, by Theorem 3, the same cannot be said if the greedy heuristic is used for the attacker’s best response.

**Algorithm 1 GREEDY_DEFENDER_RESP**

**Require:** Mixed strategy \( \text{Pr}_a \), Natural number \( k_d \)

**Ensure:** Set of nodes \( V_{ld} \)

1: \( V_{ld} = \emptyset \)
2: Let \( \text{ATK} \) be the set of strategies associated with \( \text{Pr}_a \)
3: Set \( \text{flag} = \text{True} \), \( p^* = -\infty \)
4: while \( |V_{ld}| \leq k_d \) and \( \text{flag} \) and \( p^* < 0 \) do
5: \( p^* = -\sum_{v_a \in \text{ATK}} \text{Pr}_a(V_a) pc(V_a, V_{ld}) \)
6: \( \text{curBest} = \text{null}, \text{curBestScore} = 0, \text{haveValidScore} = \text{False} \)
7: for \( i \in V - V_{ld} \) do
8: \( \text{curScore} = p^* - \sum_{v_a \in \text{ATK}} \text{Pr}_a(V_a) pc(V_a, V_{ld} \cup \{i\}) \)
9: if \( \text{curScore} \geq \text{curBestScore} \) then
10: \( \text{curBest} = i \)
11: \( \text{curBestScore} = \text{curScore} \)
12: \( \text{haveValidScore} = \text{True} \)
13: end if
14: end for
15: if \( \text{haveValidScore} = \text{False} \) then
16: \( \text{flag} = \text{False} \)
17: else
18: \( V_{ld} = V_{ld} \cup \{\text{curBest}\} \)
19: end if
20: end while
21: return \( V_{ld} \).

**Finding Mixed Strategies.** If the attacker uses a mixed strategy that consists of uniformly attacking elements of \( \{S \subset V_{ld} : |S| = k_a\} \) then the best any pure defender strategy can do is defending \( V_a \subset V_{ld} \). The attacker’s strategy implies that any node in \( V_{ld} \) is attacked with probability \( \frac{k_a}{|V_{ld}|} \).

Each of the \( |V_{ld}| - k_a \) remaining nodes in \( V_{ld} \) is then disconnected with probability \( \frac{k_a}{|V_{ld}|} \), i.e., \( x \geq k_a (1 - \frac{k_a}{|V_{ld}|}) \). Clearly due to the cascading the value of the game will probably be higher, illustrating the disadvantage the defender has in this game. To determine both player’s optimal strategies and the value of the game we resort to an algorithmic approach. We find the defender’s optimal strategy with the following linear program. We can find minimax strategy for the defender with the following linear program. It simply assigns a probability to each of the attackers strategies in a manner that minimizes the maximum payoff for the adversary. As a consequence, the solution to the following linear program, **DEF\_LP** can provide the mixed minimax strategy for the defender. An analogous linear program, **ATK\_LP** (not shown), which mirrors **DEF\_LP**, will provide that result for the attacker.
**Definition 4.2** (DEF\_LP).

\begin{align}
\text{min } & \ p^* \\
\text{subject to} & \ \ p^* \geq \sum_{V_d \in \text{DEF}} X_{V_d} PC(V_a, V_d) \ \ \forall V_a \in \text{ATK} \\
& \ 1 = \sum_{V_d \in \text{DEF}} X_{V_d} \\
& \ X_{V_d} \in [0, 1] \ \ \forall V_d \in \text{DEF}
\end{align}

Note that the above linear program requires one variable for each of the defender’s strategies and one constraint for each of the attacker’s strategies. However, as there are a combinatorial number of strategies, even writing down such a linear program is not practical except for very small problem instances. To address this issue of intractability, we employ the double-oracle framework for zero-sum games in a linear program is not practical except for very small problem instances. To address this issue of intractability, we employ the double-oracle framework for zero-sum games.

**Algorithm 2 DOUBLE\_ORACLE**

**Require:** Network \( G = (V, E) \), natural number \( \text{maxIters} \)

**Ensure:** Mixed defender strategy \( Pr_d \)

1. Initialize \( \text{numIters} = 0 \), \( \text{flag} = \text{True} \)
2. Initialize the sets of strategies \( \text{ATK, DEF} \) to both be \( \{\emptyset\} \)
3. while \( \text{flag} \) and \( \text{numIters} \leq \text{maxIters} \) do
4. Create \( Pr_a, Pr_d \) based on the solutions to ATK\_LP and DEF\_LP respectively.
5. IF \( \text{numIters} < \text{maxIters} \) THEN let \( V_a \) be the attacker’s best response to \( Pr_d \) and \( V_d \) be the defender’s best response to \( Pr_a \).
6. IF \( V_a \in \text{ATK} \) and \( V_d \in \text{DEF} \) THEN \( \text{flag} = \text{False} \)
7. \( \text{numIters} + 1 \)
8. end while
9. return \( Pr_a \).

The intuition behind the above algorithm is that it iteratively creates mixed strategies for both the attacker and defender based on a solution to a linear program over the sets of current possible strategies for both players (ATK, DEF). This is followed by finding (for each player) the best deterministic response to its opponent’s strategy. If these new strategies are both already in the set of possible strategies for the respective players, the algorithm terminates. Otherwise, they are added to ATK, DEF respectively. We note that by Theorem 1 of [15] that the above algorithm will guarantee an exact solution if \( \text{maxIters} \) is set to the number of possible strategies. In practice, [15] demonstrates that the algorithm converges much faster.

In DOUBLE\_ORACLE, the finding the solutions to DEF\_LP, ATK\_LP will be tractable provided that the algorithm converges in a polynomial number of steps (either through convergence or after the specified \( \text{maxIters} \)). However, as we have shown, computing the best responses is usually computationally difficult. Although, we note in the case where \( k_a = 1 \), that by Proposition 3.2 and Fact 3.1, the double oracle algorithm will return an optimal solution, even if greedy approximations are used for the oracles (provided it runs until convergence).

**5. EXPERIMENTAL EVALUATION**

All experiments were run on a computer equipped with an Intel X5677 Xeon Processor operating at 3.46 GHz with a 12 MB Cache and 288 GB of physical memory. The machine was running Red Hat Enterprise Linux version 6.1.

Only one core was used for experiments. All algorithms were coded using Python 2.7 and leveraged the NetworkX library\(^2\) as well as the PuLP library for linear programming\(^3\). All statistics presented in this section were calculated using the R statistics software.

In our experiments, we utilized a dataset of an Italian 380 kV power transmission grid [20]. This power grid network consisted of 310 nodes of which 113 were source, 96 were load, and the remainder were transmission nodes. The nodes were connected with 361 edges representing the power lines.

In our initial experiments, we examined the properties of the model when no defense is employed. In Figure 1 (left) we show results concerning nodal load vs. the payoff achieved by the adversary if that node is attacked (and no others). Interestingly, we noticed a significant number of nodes with low nodal load yet high-payoff if attacked (see nodes in dashed box). This may suggest that the DLB strategy may be insufficient in some cases. Later we see how DLB fails to provide adequate in a defense against the attacker best response to DLB. This is likely due to these hi-payoff, low-load nodes. In Figure 1 (right) we examine \( \alpha \) (capacity margin) vs. attacker payoff for various settings of \( k_a \) (using the GREEDY\_ATTACKER\_RESP heuristic). Here we found that, in general, payoff decreases linearly with capacity margin \((R^2 \geq 0.84\) for each trial).

Next, we examined the relative performance of the minimax (mixed) defense strategy and the DLB strategy under different resource constraints and against the minimax (mixed) attack strategy as well as the attacker’s (deterministic) greedy response to the DLB defense. In these experiments, we considered the case where both players have equal resources, the attacker has one resource (which by Proposition 3.2 and Fact 3.1 we are guaranteed an optimal solution), and the defender has one resource. These results are displayed in Figure 2. In these trials we set the capacity margin \( \alpha = 0.5 \), meaning that all edges had an excess capacity of 50%. We did not use the \( \text{maxIters} \) parameter of the DOUBLE\_ORACLE algorithm, but instead allowed it to run until convergence.

With regard to the comparison between DLB and minimax defense, both performed comparably against the minimax attack strategy. In fact, an analysis of variance (ANOVA) indicated little variance between the two when faced with the minimax attacker \((p \geq 0.74\) for these trials). Yet, a defender known to be playing a single strategy would likely not face an attacker who plays the minimax strategy, but rather the

\[^2\]http://networkx.lanl.gov/
\[^3\]http://pythonhosted.org/PuLP/
Figure 2: Minimax and DBL defense strategies vs. minimax attack strategy (left) and the attacker’s greedy best response to DBL (right). Examined are the cases when $k_a = k_d$ (top), $k_a = 1$, $k_d$ varies (middle) and $k_d = 1$, $k_a$ varies (bottom).

best response to the DBL. In this case, DBL play resulted in significantly greater payoff to the attacker than the defender ($p < 0.29$ for these trials, the DBL defense results in 15.6 more disconnected nodes on average). This failure of the DBL strategy to perform well against a deterministic attacker best response is likely due to the presence of low-load yet high-payoff nodes as shown in Figure 1.

We also noticed that an increase in resources seems to favor the attacker more than the defender. When both players played their respective minimax strategy, the expected payoff for the attacker increased monotonically with the cardinality of the strategies. Further, when $k_d = 1$ and $k_a$ was greater, the attacker’s payoff tripled when his resources increased from 1 to 6. However, when $k_a = 1$ and $k_d$ was greater, the defender’s payoff only increased by a factor of 1.7. Hence, the attacker can cause more damage than the defender can mitigate with the same amount of extra resources. We suspect that this is likely because a defended node can still fail during a cascade - which would likely be the case if the attack and defense operations are restricted to cyber-space, where physical system failure may still be possible as the result of a cascade initiated by virtual means.

We also examined the run-time of our approach, as displayed in Figure 3 (left). Though run-time did seem to scale linearly with strategy size ($R^2 = 0.90 \pm 0.2$ for each experiment), it appears that run-time will in general prohibit the study of larger strategies or networks (our longest experiment ran for 12 days). In examining the iterations of the DOUBLEORACLE algorithm, Figure 3 (left), we find that run-time of an iteration of the algorithm progressively increases (note that this figure is showing run-time for each iteration, not a cumulative time). This increase is likely the combined result of the growing linear program and the growing size of the mixed strategies considered by the greedy approximation sub-routines. We are currently exploring reliable methods to limit the number of iterations while maintaining defender payoff.

6. RELATED WORK

Network security has received much attention from the research community in the past two decades. Recent incidents have shown that due to their internet connectedness such networks can come under cyber attack, causing severe problems. See [26] for a discussion of cyber-security issues relevant to smart grid grids.

The utilization of game theory in designing defense solutions seems ubiquitous. For instance [13] model the interaction between a DDoS attacker and the network administrator while [14] considers a game theoretic formulation for intrusion detection. Other formulations consist include stochastic games [17], signaling games [19], allocation games [4] and repeated games [3]. Game theory is also being used in monitoring and decision making in smart grids, see for instance [9] or the survey by Fadlullah et al. [10]. However to date no game theoretic approach has been given for the specific problem where the attacker explicitly sets of a cascading power failure to maximize the damage to the defender.

Cascading failure models applied to power grid infrastructure have been studied in the past [7, 8, 16]. The model of [8] introduces the idea of edge failure based on excessive loads. The goal of the research presented in these papers was to illustrate properties of the cascade, rather than explore strategies for attack and defense as this work does. There has been work on attack and defense of a power-grid network under the DC power-flow model [2, 21, 20, 6]. However, the DC power flow model is not designed to model the more rapid cascading failures (i.e. the 2003 cascading failure in the eastern United States [1]).

The application of game theory to security situations was made popular by [18] where it used for airport security patrol scheduling. Since then, other applications have emerged including port protection [23], finding weapons caches [22], and security checkpoint placement [12]. One that bears similarity to this work is [24] - studying games for controlling contagions on a network. However, as previously discussed, that model operates under very different dynamics.

http://www.wired.com/threatlevel/2009/10/smartgrid/
7. CONCLUSION

In this paper, we explored complexity, algorithmic, and implementation issues in a two-player security game where the attacker/defender look to create/mitigate cascading failure on a power grid. Future work includes an examination of scalability issues (larger networks and strategies), adding uncertainty to the model, and the consideration of more real-world information about the power grid network (i.e. actual line capacities, etc.) in order to create a richer model.

8. ACKNOWLEDGMENTS

We would like to thank D. Alderson for his input on related work and V. Rosato for providing us the power grid dataset. Some of the authors are supported by ARO project 2GDATXR042. The opinions in this paper are those of the funders, the U.S. Military Academy, or the U.S. Army.

9. REFERENCES


