The N - k Problem in Power Grids: New Models, Formulations and Numerical Experiments [1]

Daniel Bienstock and Abhinav Verma
Columbia University, New York

May 2008; Revised February 2010

Abstract

Given a power grid modeled by a network together with equations describing power flows, power generation and consumption, the so-called $N-k$ problem asks whether there exists a set of k or fewer arcs whose removal will cause the system to fail. The case where k is small is of practical interest. We present theoretical and numerical results involving a mixed-integer linear model and a continuous nonlinear model related to this problem.

1 Introduction

Recent large-scale power grid failures have highlighted the need for effective computational tools for discovering vulnerabilities of electrical transmission networks. We consider the *vulnerability evaluation* problem: given a power grid, is there a small set of power lines whose removal will lead to system failure? Here, the number of disabled power lines is denoted by k, and experts have called for small values of k, such as $k = 3$ or 4.

A variety of names have been used for this problem: network *interdiction*, network *inhibition*, and others, though the "N - k problem" terminology is common in the industry. One could consider the failure of different types of components (generators, transformers) in addition to lines; however, in this paper "N" represents the number of lines in the network. The $N-k$ problem is intractable, even for small values of k – the pure enumeration approach is impractical. In addition to the combinatorial explosion, a significant difficulty is the need to model the laws of physics governing power flows in a sufficiently accurate and yet computationally tractable manner.

In our experience, when k is large the problem tends to become easier, possibly because many solutions exist, but on the other hand one can argue that it is unrealistic to use large k. The simplest case, $k = 1$, can be addressed by enumeration but may yield insufficient information. The middle range, $2 \leq k \leq 5$, is both relevant and difficult, and is our primary focus.

We present results using two optimization models. The first (Section 2.1) employs integer programming techniques to model a fictional attacker seeking to disable the network, and a controller who tries to prevent a collapse by selecting which generators to operate and adjusting generator outputs and demand levels.

The second approach (Section 3) is given by a new, continuous nonlinear programming formulation that models a fictional attacker with the ability to modify the physical attributes of the power lines in order to expose underlying vulnerabilities. While both formulations provide substantial savings over a pure enumerational approach, the second formulation appears much more effective, enabling us to handle models an order of magnitude larger.

1.0.1 Previous work on vulnerability problems

A survey of recent work on optimization methods applied to blackout-related problems is given in [22]. Typically work has focused on identifying a small set of arcs whose removal – used to model complete failure – will result in a network unable to deliver a minimum amount of demand. A problem of this type, when small enough, can be solved using mixed-integer programming techniques techniques, see [3], [25], [4]. See Section 2.0.5. Also see [23], which considers a discrete model for conductances (inverse of resistances) from an empirical perspective.

[1] Partially funded by award NSF-0521741 and award DE-SC000267

A different line of research focuses on attacks with certain structural properties, see [8], [22]. An example of this approach is used in [22], where as an approximation to the $N-k$ problem with AC power flows, the following problem is solved: remove a minimum number of arcs, such that in the resulting network there is a partition of the nodes into two sets, N_1 and N_2, such that

$$D(N_1) + G(N_2) + cap(N_1, N_2) \leq Q^{min}. \tag{1}$$

Here $D(N_1)$ is the total demand in N_1, $G(N_2)$ is the total generation capacity in N_2, $cap(N_1, N_2)$ is the total capacity in the (non-removed) arcs between N_1 and N_2, and Q^{min} is a minimum amount of demand that needs to be satisfied. The quantity in the left-hand side in the above expression is an upper-bound on the total amount of demand that can be satisfied – the upper-bound can be strict because it may be unattainable under power flow laws. [22] describes the solution of this approximate problem on a network with over 16 thousand arcs.

Finally, we mention that the most sophisticated models for the behavior of a grid under stress attempt to capture the multistage nature of blackouts, and are thus more comprehensive than the static models considered above and in this paper. See, for example, [11]-[14], and [7].

1.0.2 Basic definitions and properties of power flows

Here we provide a brief introduction to the *linearized*, or *DC* power flow model. In what follows, a grid is represented by a directed network $\mathcal{N}$, with arc-set E and node-set V, where:

- Each node corresponds to a "generator" (a supply node), or a "load" (a demand node), or a node that neither generates nor consumes power. We denote the set of generator nodes by $\mathcal{G}$ and the set of demand nodes by $\mathcal{D}$.
- If node i corresponds to a generator, then there are values $0 \leq P_i^{min} \leq P_i^{max}$. If the generator is operated, then its output must be in the range $[P_i^{min}, P_i^{max}]$; if the generator is not operated, then its output is zero. In general, we expect $P_i^{min} > 0$.
- For each demand node i there is a value D_i^{nom}, the "nominal" demand value at node i.
- The arcs of $\mathcal{N}$ represent power lines. For each arc (i, j), we are given a parameter $x_{ij} > 0$ (the resistance) and a parameter u_{ij} (the capacity).

Given a set $\mathcal{C}$ of operating generators, a *power flow* is a solution to the system $P(\mathcal{N}, \mathcal{C})$ given by equations (2)-(6) below. In this system, for each arc (i, j), we use a variable f_{ij} to represent the flow on (i, j) – possibly $f_{ij} < 0$, in which case power is effectively flowing from j to i. In addition, for each node i we will have a variable θ_i, the "phase angle" at i. Finally, if i is a generator node, then we have a variable P_i, while if i represents a demand node, we have a variable D_i. Given a node i, we represent with $\delta^+(i)$ ($\delta^-(i)$) the set of arcs oriented out of (respectively, into) i.

Constraints (2), (5) and (6) are typical for network flow models (for background see [1]) and model, respectively: flow balance (i.e., net flow leaving a node equals net supply at that node), generator and demand node bounds. Constraint (3) is a commonly used linearization of equations describing power flow physics; see [2] and Section 1.0.3.

$$\sum_{(i,j)\in\delta^+(i)} f_{ij} - \sum_{(j,i)\in\delta^-(i)} f_{ji} = \begin{cases} P_i & i \in \mathcal{C} \\ -D_i & i \in \mathcal{D} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$\theta_i - \theta_j - x_{ij} f_{ij} = 0 \quad \forall (i, j) \tag{3}$$

$$|f_{ij}| \leq u_{ij}, \quad \forall (i, j) \tag{4}$$

$$P_i^{min} \leq P_i \leq P_i^{max} \quad \forall i \in \mathcal{C} \tag{5}$$

$$0 \leq D_j \leq D_j^{nom} \quad \forall j \in \mathcal{D} \tag{6}$$

One could also impose explicit upper bounds on the quantities $|\theta_i - \theta_j|$ (over the arcs (i,j)). However, note that our model already does so, implicitly: because of constraint (3), imposing an upper bound on $|f_{ij}|$ is equivalent to imposing an upper bound on $|\theta_i - \theta_j|$.

A useful property is given by the following result, whose proof is routine.

Lemma 1.1 *Let $\mathcal{C}$ be given, and suppose $\mathcal{N}$ is connected. For each choice of nonnegative values P_i (for $i \in \mathcal{C}$) and D_i (for $i \in \mathcal{D}$) such that $\sum_{i\in\mathcal{C}} P_i = \sum_{i\in\mathcal{D}} D_i$, system (2)-(3) has a unique solution in the f_{ij}. Denoting by b the vector with an entry for each node, where $b_i = P_i$ for $i \in \mathcal{C}$, $b_i = -D_i$ for $i \in \mathcal{D}$, and $b_i = 0$ otherwise, by $\bar{N}$ the matrix obtained from N by removing an arbitrarily chosen row, and by $\bar{b}$ the corresponding subvector of b, we have $f = X^{-1}\bar{N}^T(\bar{N}X^{-1}\bar{N}^T)^{-1}\bar{b}$, where $X = \mathbf{diag}\{x_{ij}\}$.*

Remark 1.2 *Lemma 1.1 concerns the subsystem of $P(\mathcal{N},\mathcal{C})$ consisting of (2) and (3). In particular, the "capacities" u_{ij} play no role in the determination of solutions.*

Definition 1.3 *If (f,θ,P,D) is a feasible solution to $P(\mathcal{N},\mathcal{C})$, its* **throughput** *is*

$$\frac{\sum_{i\in\mathcal{D}} D_i}{\sum_{i\in\mathcal{D}} D_i^{nom}}. \tag{7}$$

The throughput of $\mathcal{N}$ is the maximum throughput of any feasible solution to $P(\mathcal{N},\mathcal{C})$.

1.0.3 DC and AC power flows, and other modeling issues

In the case of an AC network (the relevant case when dealing with power grids) constraint (3) only approximates a complex system of nonlinear equations (see [2]). We provide a brief summary of how the linearized model arises. First, AC models typically include equations of the form

$$x_{ij}f_{ij} \;=\; \sin(\theta_i - \theta_j), \quad \forall (i,j), \tag{8}$$

as opposed to (3). In normal operation, one would expect that $\theta_i \approx \theta_j$ for any arc (i,j) and thus (8) can be linearized to yield (3). The linearization is valid if we impose that $|\theta_i - \theta_j|$ be very small, but this requirement is relaxed when considering regimes other than a normal operating mode.

Constraints such as (8) give rise to complex models and as a result studies that require multiple power flow computations tend to rely on the linearized formulation. This will be the approach we take in this paper, so as to focus more explicitly on the basic combinatorial complexity that underlies the $N-k$ problem. In contrast, an approach incorporating a better representation of the physics might not be able to tackle the combinatorial complexity of the problem quite as effectively, for the simple reason that the theory and computational machinery for linear programming are far more mature, effective and scalable than those for nonlinear, nonconvex optimization.

A final point is that whether we use an AC or DC power flow model, the resulting problems have a far more complex structure than traditional single- or multi-commodity flow models arising in computer science or operations research. This is because of side-constraints such as (3), which give rise to counter-intuitive behavior reminiscent of Braess's Paradox [10].

2 The "N - k" problem

Suppose a fictional *attacker* wants to remove a small number of arcs from $\mathcal{N}$ so that the resulting network has low throughput. A *controller* responds to an attack by choosing the set $\mathcal{C}$ of operating generators, their output levels, and the demands D_i, so as to feasibly obtain high throughput. The controller's adjustment of demands corresponds to the traditional notion of "load shedding" in the power systems literature.

The attacker seeks to defeat *all possible courses of action* by the controller; in other words, we are modeling the problem as a Stackelberg game between the attacker and the controller, where the attacker moves first. To cast our model in a precise way we will use the following definition. We let $0 < T^{min} < 1$ be a given value.

Definition 2.1

- *An* **attack** $\mathcal{A}$ *is a set of arcs removed by the attacker, and* $\mathcal{N} - \mathcal{A}$ *denotes the subnetwork of* $\mathcal{N}$ *made up by the remaining arcs.*
- *A* **configuration** *is a set* $\mathcal{C}$ *of generators.*
- *We say that an attack* $\mathcal{A}$ **defeats** *a configuration* $\mathcal{C}$*, if either (a) the maximum throughput of any feasible solution to* $P(\mathcal{N} - \mathcal{A}, \mathcal{C})$ *is strictly less than* T^{min}*, or (b) no feasible solution to* $P(\mathcal{N} - \mathcal{A}, \mathcal{C})$ *exists. Otherwise we say that* $\mathcal{C}$ *defeats* $\mathcal{A}$*.*
- *We say that an attack is* **successful***, if it defeats* **every** *configuration.*
- *The* **min-cardinality T^{min}-throughput attack problem** *is that of finding a successful attack* $\mathcal{A}$ *with* $|\mathcal{A}|$ *minimum. For brevity, we will call this the min-cardinality problem.*

Note that the minimum cardinality of a successful attack could vary substantially as a function of T^{min}. Consequently *it might appear* more fruitful to focus on the problem of removing a given number k (or fewer) arcs so that the resulting network has minimum throughput. We will refer to this as the **budget-k min-throughput problem**. Given k, the budget-k min-throughput problem can be reduced to a finite number of minimum cardinality problems, using binary search on T^{min}, and viceversa. However, there are reasons why we prefer to focus on the min-cardinality problem.

(a) For a given k, the throughput of the network when k arcs are removed could be much larger than when $(k+1)$ arcs are removed. For this reason, and given the approximate nature of the models and underlying data, one would want to test various values of k – this issue is obviously related to what percentage of demand loss would be considered tolerable, in other words, the parameter T^{min}.

(b) From an operational perspective it should be straightforward to identify reasonable values for the quantity T^{min}; whereas the value k is more obscure and bound to models of how much power the adversary can wield.

Returning to our model, there are three different ways for an attack $\mathcal{A}$ to defeat a configuration $\mathcal{C}$.

(i) Consider a partition of the nodes of $\mathcal{N}$ into two classes, N^1 and N^2. Write, for $k = 1, 2$,

$$D^k = \sum_{i \in \mathcal{D} \cap N^k} D_i^{nom} \quad \text{and} \quad P^k = \sum_{i \in \mathcal{C} \cap N^k} P_i^{max}, \tag{9}$$

e.g. the total (nominal) demand in N_k and the maximum power generation in N_k, respectively. The following condition, should it hold, would guarantee that $\mathcal{A}$ defeats $\mathcal{C}$:

$$T^{min} \sum_{j \in \mathcal{D}} D_j^{nom} - \min\{D^1, P^1\} - \min\{D^2, P^2\} > \sum_{(i,j) \notin \mathcal{A} : i \in N^1, j \in N^2} u_{ij} + \sum_{(i,j) \notin \mathcal{A} : i \in N^2, j \in N^j} u_{ij}. \tag{10}$$

To understand (10), note that for $k = 1, 2$, $\min\{D^k, P^k\}$ is the maximum demand within N^k that could possibly be met using power flows that do not leave N^k. Consequently the left-hand side of (10) is a lower bound on the amount of flow that must travel between N^1 and N^2, whereas the right-hand side of (10) is the total capacity of arcs between N^1 and N^2 under attack $\mathcal{A}$. In other words, condition (10) amounts to a mismatch between demand and supply. A special case of (10) is that where in $\mathcal{N} - \mathcal{A}$ there are no arcs between N^1 and N^2, i.e. the right-hand side of (10) is zero. Condition (11) is similar, but not identical, to (1).

(ii) Consider a partition of the nodes of $\mathcal{N}$ into two classes, N^1 and N^2. Then $\mathcal{A}$ defeats $\mathcal{C}$ if

$$\sum_{i \in \mathcal{D} \cap \in N^1} D_i^{nom} + \sum_{(i,j) \notin \mathcal{A} : i \in N^1, j \in N^2} u_{ij} < \sum_{i \in \mathcal{C} \cap N^1} P_i^{min}, \tag{11}$$

i.e., the minimum power output within N^1 exceeds the maximum demand within N^1 plus the sum of arc capacities leaving N^1. Note that (ii) may apply even if (i) does not.

(iii) Even if (i) and (ii) do not hold, it may still be the case that the system (2)-(6) does not admit a feasible solution, as follows. Suppose that for every choice of demand values $0 \leq D_i \leq D_i^{nom}$ (for $i \in \mathcal{D}$) and supply values $P_i^{min} \leq P_i \leq P_i^{max}$ (for $i \in \mathcal{C}$) such that $\sum_{i \in \mathcal{C}} P_i = \sum_{i \in \mathcal{D}} D_i$ the unique solution to system (2)-(3) in network $\mathcal{N} - \mathcal{A}$ (as per Lemma 1.1) does *not* satisfy the "capacity" inequalities $|f_{ij}| \leq u_{ij}$ for all arcs $(i,j) \in \mathcal{N} - \mathcal{A}$. Then $\mathcal{A}$ defeats $\mathcal{C}$. This is the most subtle case of all; it may hold even if (i) or (ii) do not.

Note that in particular in case (ii), the defeat condition is unrelated to throughput and is instead due to having positive P_i^{min}. This makes the min-throughput problem difficult to model. Nevertheless, from a practical perspective, it is important to handle models with positive P_i^{min}. It is also important to model standby generators that are turned on when needed, and to model the turning off of generators that are unable to dispose of their minimum power output, post-attack. All these features add complexity through a possibly exponential number of control possibilities.

As far as we can tell, most (or all) prior work in the literature does require that the controller must always use the configuration $\bar{\mathcal{G}}$ consisting of all generators. It is simple to create examples with positive P_i^{min} where there exist attacks $\mathcal{A}$ such that $P(\mathcal{N} - \mathcal{A}, \bar{\mathcal{G}})$ is infeasible (see Figure 1). Because of this fact, algorithms that rely on direct application of Benders' decomposition or bilevel programming are problematic, and invalid formulations can be found in the literature.

For the case where $P_i^{min} = 0$ for all i, our approach yields a mixed-integer program simple enough that a commercial integer programming solver can directly handle instances larger than previously reported in the literature relying on integer programming methods.

2.0.4 Non-monotonicity of attack severity as a function of cardinality

In this section we show that the severity of the worst attack of a given cardinality may decrease if the cardinality is increased. Consider the example in Figure 1, where we assume $T^{min} = 0.3$. Notice that there are two parallel copies of arcs $(2,4)$ and $(3,5)$, each with capacity 10 and resistance 1. The network with no attack is feasible: we operate generator 1 and not operate generators 2 and 3, and send 3 units of flow on paths $1-6-2-4$ and $1-6-3-5$. Thus the flow on the two parallel $(2,4)$ arcs, and on the two parallel $(3,5)$ arcs, is evenly split.

However, the attack consisting of arc $(1,6)$ is successful. To see this, note that under this attack, the controller cannot operate both generators 2 and 3, since their combined minimum output exceeds the total demand. Without loss of generality suppose that only generator 3 is operated, and assume by contradiction that a feasible solution exists – then this solution must route at most 3 units of flow along $3-6-2-4$, and since $P_3^{min} = 8$, at least 5 units of flow on $(3,5)$ (both copies altogether). In such a case, the phase angle drop from 3 to 5 is at least 2.5, whereas the drop from 3 to 4 is at most 1.56. So $\theta_4 - \theta_5 \geq 0.94$, and we will have $f_{45} \geq 0.94$ – thus, the net inflow at node 5 is at least $5.94 > D_5^{nom}$. Hence the attack is indeed successful.

However, there is no successful attack consisting of arc $(1,6)$ and another arc. To see this, note that if one of $(2,6)$, $(3,6)$ or $(4,5)$ are also removed then the controller can just operate one of the two generators 2 and 3 and meet eight units of demand. Suppose that one of the two copies of $(3,5)$ is removed in addition to $(1,6)$. Then the controller operates generator 2, sending 2.5 units of flow on each of the two parallel $(2,4)$ arcs; thus $\theta_2 - \theta_4 = 2.5$. The controller also routes 3 units of flow along $2-6-3-5$, and therefore $\theta_2 - \theta_5 = 3.06$. So $\theta_4 - \theta_5 = .56$ and $f_{45} = .56$, resulting in a feasible flow which satisfies 4.44 units of demand at 4 and 3.56 units of demand at 5.

In fact, no successful attack of cardinality 2 exists. To see this, assume that $(1,6)$ is not attacked (we just handled the case when it is). We observe that in the original network, the smallest cut

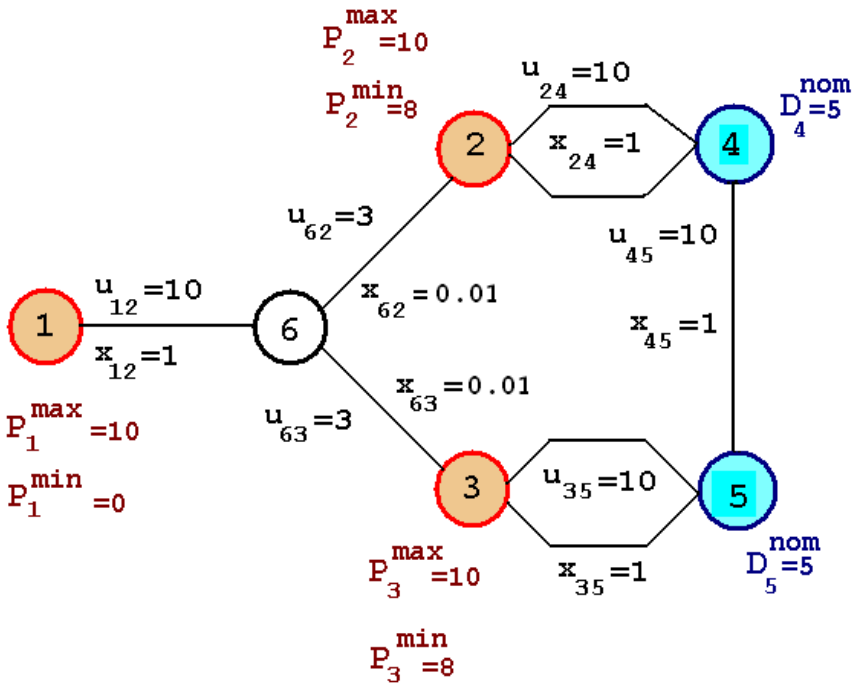

Figure 1: An example where there is a successful attack with $k = 1$ but none with $k = 2$.

simultaneously separating 6 from both 4 and 5 has cardinality 2, and there is only one such cut, namely $\{(2,6), (3,6)\}$. If the attack consists of $(2,6)$ and $(3,6)$ then we only operate generator 2 and deliver 5 units of demand to 4 and 3 units to 5. And as per the observation, in every other attack of cardinality 2 that does not include arc $(1,6)$, there is a (surviving) path from 1 to at least one of 4 and 5 – in such a case we only operate generator 1 and use it to output 3 units of demand. Since every arc capacity is at least 3 the resulting flow will feasibly deliver 3 units of demand. This concludes the analysis.

By elaborating on this example, one can create cases with arbitrary patterns in the cardinality of successful attacks. One can also generate examples that exhibit non-monotone behavior in response to controller actions. In both cases, the non-monotonicity can be viewed as a manifestation of "Braess's Paradox" [10].

2.0.5 Brief review of work related to the min-cardinality problem

The min-cardinality problem, as defined above, can be viewed as a bilevel program where both the master problem and the subproblem are mixed-integer programs – the master problem corresponds to the attacker (who chooses the arcs to remove) and the subproblem to the controller (who chooses the generators to operate). In general, bilevel programs are extremely challenging. A recent general-purpose algorithm for such integer programs is given in [16].

Alternatively, each configuration can be viewed as a "scenario". In this sense our problem resembles a stochastic program, although without a probability distribution. Recent work [19] considers a single commodity max-flow problem under attack by an interdictor with a limited attack budget; where an attacked arc is removed probabilistically, leading to a stochastic program (to minimize the expected max flow). A deterministic, multi-commodity version of the same problem is given in [20].

Previous work on power grid vulnerability models has focused on cases where either the generator lower bounds P_i^{min} are all zero, or all generators must be operated (the single configuration case). Algorithms for these problems have either relied on heuristics, or on mixed-integer programming techniques, usually a direct use of Benders' decomposition or bilevel programming. [3] considers a version of the min-throughput problem with $P_i^{min} = 0$ for all generators i, and presents an algorithm using Benders' decomposition (also see references therein). They analyze the so-called

IEEE One-Area and IEEE Two-Area test cases, with, respectively, 24 nodes and 38 arcs, and 48 nodes and 79 arcs. Also see [25]. [4] studies the IEEE One-Area test case, and allows $P_i^{min} > 0$, but does not allow generators to be turned off; the authors present a bilevel programming formulation which unfortunately is incorrect, due to reasons outlined above.

2.1 A mixed-integer programming algorithm for the min-cardinality problem

In this section we will describe an algorithm for the min-cardinality attack problem. As a preliminary step, we first solve a simpler problem: given an attack $\mathcal{A}$ and a configuration $\mathcal{C}$, does $\mathcal{A}$ defeat $\mathcal{C}$? In the LP below, $z_{ij}^{\mathcal{A}} = 1$ if $(i,j) \in \mathcal{A}$ and $z_{ij}^{\mathcal{A}} = 0$ otherwise; similarly for each generator i we set $y_i^{\mathcal{C}} = 1$ if $i \in \mathcal{C}$ and $y_i^{\mathcal{C}} = 0$ otherwise. "t" is an auxiliary variable. To the left of each constraint we indicate the corresponding dual variable.

$$\boldsymbol{K_{\mathcal{C}}^*(\mathcal{A})}: \qquad t_{\mathcal{C}}^*(z^{\mathcal{A}}) \quad \doteq \quad \min\, t \tag{12}$$

Subject to:

$$(\boldsymbol{\alpha_i^{\mathcal{C}}}) \qquad \sum_{(i,j)\in\delta^+(i)} f_{ij} - \sum_{(j,i)\in\delta^-(i)} f_{ji} = \begin{cases} P_i & i \in \mathcal{G} \\ -D_i & i \in \mathcal{D} \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

$$(\boldsymbol{\beta_{ij}^{\mathcal{C}}}) \qquad \theta_i - \theta_j - x_{ij} f_{ij} = 0 \quad \forall\ (i,j) \notin \mathcal{A} \tag{14}$$

$$(\boldsymbol{p_{ij}^{\mathcal{C}}}, \boldsymbol{q_{ij}^{\mathcal{C}}}) \qquad u_{ij}\ t\ -\ |f_{ij}| \quad \geq \quad 0, \quad \forall\ (i,j) \notin \mathcal{A} \tag{15}$$

$$(\boldsymbol{\omega_{ij}^{\mathcal{C}+}}, \boldsymbol{\omega_{ij}^{\mathcal{C}-}}) \qquad t\ -\ |f_{ij}| \quad \geq \quad 1, \quad \forall\ (i,j) \in \mathcal{A} \tag{16}$$

$$(\boldsymbol{\gamma_i^{\mathcal{C}+}}, \boldsymbol{\gamma_i^{\mathcal{C}+}}) \qquad P_i^{min} y_i^{\mathcal{C}} \ \leq P_i \ \leq P_i^{max} y_i^{\mathcal{C}} \quad \forall i \in \mathcal{G} \tag{17}$$

$$(\boldsymbol{\mu^{\mathcal{C}}}) \qquad \sum_{j\in\mathcal{D}} D_j \ \geq \ T^{min} \left(\sum_{j\in\mathcal{D}} D_j^{nom} \right) \tag{18}$$

$$(\boldsymbol{\Delta_j^{\mathcal{C}}}) \qquad D_j \ \leq D_j^{nom} \quad \forall j \in \mathcal{D} \tag{19}$$

$$P \geq 0, \quad D \geq 0. \tag{20}$$

Note that (15) can be represented as two linear inequalities, and the same applies to (16).

Lemma 2.2 *$t_{\mathcal{C}}^*(z^{\mathcal{A}}) \geq 1$, and $\mathcal{A}$ defeats $\mathcal{C}$ if and only if $t_{\mathcal{C}}^*(z^{\mathcal{A}}) > 1$.*

Proof. By (16), $t_{\mathcal{C}}^*(z^{\mathcal{A}}) \geq 1$, and $t_{\mathcal{C}}^*(z^{\mathcal{A}}) = 1$ iff there is a solution with $f_{ij} = 0$ for all $(i,j) \in \mathcal{A}$ and $f_{ij} \leq u_{ij}$ for all $(i,j) \notin \mathcal{A}$. ■

In our algorithm, we will replace $t_{\mathcal{C}}^*(z^{\mathcal{A}}) > 1$ with the weaker condition $t_{\mathcal{C}}^*(z^{\mathcal{A}}) \geq 1 + \epsilon$, where $\epsilon > 0$ and small. The use of this parameter gives more power to the controller. We are thus not solving the optimization problem to exact precision; nevertheless we expect our relaxation to have negligible impact so long as ϵ is small. A deeper issue here is how to interpret truly borderline attacks that are successful according to our strict model but not when we use the weaker condition; we expect that in practice such attacks would be ambiguous and that the approximations incurred in modeling the grid and the numerical sensitivity of the integer and linear solvers being used would have a far more significant impact on precision.

Slightly extending our notation, the min-cardinality problem can thus be written as follows:

$$\min \sum_{(i,j)} z_{ij} \tag{21}$$

$$t_{\mathcal{C}}^*(z) \ \geq \ 1 + \epsilon, \quad \forall\ \mathcal{C} \subseteq \mathcal{G}, \tag{22}$$

$$z_{ij} \ = \ 0 \text{ or } 1, \ \forall\ (i,j). \tag{23}$$

2.1.1 A cutting-plane algorithm for the min-cardinality problem.

We solve problem (21)-(23) by means of a cutting-plane algorithm. To produce such an algorithm, we will show, given an attack $\mathcal{A}$, how to test if $\mathcal{A}$ is successful, and if not, how to produce a an inequality valid for the set (22)-(23) which cuts-off $z^{\mathcal{A}}$.

First, an attack $\mathcal{A}$ is not successful if and only if we can find $\mathcal{C} \subseteq \mathcal{G}$ with $t_{\mathcal{C}}(z^{\mathcal{A}}) < 1 + \epsilon$; we test for this condition by solving the **controller's problem**:

$$\min_{\mathcal{C} \subseteq \mathcal{G}} \; t_{\mathcal{C}}(z^{\mathcal{A}}). \tag{24}$$

This is done by replacing, in formulation (12)-(20), each parameter y_i^C with a 0/1 variable y_i (where $y_i = 1$ iff the controller operates generator i), i.e. by replacing constraint (17) with the system

$$P_i^{min} y_i \; \leq P_i \; \leq P_i^{max} y_i, \quad y_i \; = 0 \text{ or } 1, \quad \forall i \in \mathcal{G}. \tag{25}$$

Thus, we are solving a mixed-integer program with a 0/1 variable for each generator.

Next we turn to the issue of computing valid cutting-planes that cut-off unsuccessful attacks. An attack $\mathcal{A}$ is successful against $\mathcal{C}$ iff $t^*_{\mathcal{C}}(\mathcal{A}) \geq 1 + \epsilon$, or equivalently iff the dual of $K^*_{\mathcal{C}}(\mathcal{A})$ has value at least $1 + \epsilon$. Suppressing (for clarity) the index $\mathcal{C}$ from the variables, an LP equivalent to the dual of $K^*_{\mathcal{C}}(\mathcal{A})$ can be represented as follows, where $\mathbf{M} > 0$ is a large enough constant to be specified later.

$$\mathbf{\Lambda}_{\mathcal{C}}(\mathcal{A}) : \max \sum_{i \in \mathcal{G}} y_i^{\mathcal{C}} P_i^{min} \gamma_i^- - \sum_{i \in \mathcal{G}} y_i^{\mathcal{C}} P_i^{max} \gamma_i^+ - \sum_{j \in \mathcal{D}} D_j^{nom} \Delta_j + \sum_{j \in \mathcal{D}} D_j^{nom} \mu + \sum_{(i,j) \in E} (\omega_{ij}^+ + \omega_{ij}^-)$$

Subject to:

$$(f_{ij}) \quad \alpha_i - \alpha_j - x_{ij}\beta_{ij} - p_{ij} + q_{ij} + \omega_{ij}^+ - \omega_{ij}^- \; = \; 0 \;\; \forall (i,j) \in E \tag{26}$$

$$(\theta_i) \quad \sum_{(i,j) \in \delta^+(i)} \beta_{ij} - \sum_{(j,i) \in \delta^-(i)} \beta_{ji} \; = \; 0 \;\; \forall i \in V \tag{27}$$

$$(t) \quad \sum_{(i,j) \in E} u_{ij}(p_{ij} + q_{ij}) + \sum_{(i,j) \in E} (\omega_{ij}^+ + \omega_{ij}^-) \; \leq \; 1 \tag{28}$$

$$(P_i) \quad -\alpha_i - \gamma_i^- + \gamma_i^+ \; = \; 0 \;\; \forall i \in \mathcal{G} \tag{29}$$

$$(D_j) \quad \alpha_j + \mu - \Delta_j \; \leq 0 \quad \forall j \in \mathcal{D} \tag{30}$$

$$(\xi_{ij}^+, \xi_{ij}^-) \quad x_{ij}^{1/2} |\beta_{ij}| \; \leq \; \mathbf{M}(1 - z_{ij}^{\mathcal{A}}) \;\; \forall (i,j) \in E \tag{31}$$

$$(\varrho_{ij}) \quad p_{ij} + q_{ij} \; \leq \; \frac{1}{u_{ij}}(1 - z_{ij}^{\mathcal{A}}) \;\; \forall (i,j) \in E \tag{32}$$

$$(\eta_{ij}) \quad \omega_{ij}^+ + \omega_{ij}^- \; \leq \; z_{ij}^{\mathcal{A}} \;\; \forall (i,j) \in E \tag{33}$$

$$\omega_{ij}^+ \geq 0, \;\; \omega_{ij}^- \geq 0, \;\; p_{ij} \; \geq \; 0, \;\; q_{ij} \geq 0 \;\; \forall (i,j) \in E$$

$$\gamma_i^+, \gamma_i^- \; \geq \; 0 \;\; \forall i \in \mathcal{G}, \quad \Delta_j \geq 0 \;\; \forall j \in \mathcal{D}$$

$$\mu \geq 0, \quad \delta_{ij}, \;\; \beta_{ij} \text{ free } \;\; \forall (i,j) \in E, \quad \alpha_i \text{ free } \;\; \forall i \in V.$$

As before, for each constraint we indicate the corresponding dual variable. Also note that we are scaling β_{ij} by $x_{ij}^{1/2}$ – this is allowable since $x_{ij}^{1/2} > 0$; using this scaling helps yield a simple expression for $\mathbf{M}$ below. The purpose of inequalities (31) and (32) is to force $\beta_{ij} = p_{ij} = q_{ij} = 0$ whenever $(i,j) \in \mathcal{A}$ (and similarly with constraints (33)). We stress that this linear program is equivalent to the dual of $K^*_{\mathcal{C}}(\mathcal{A})$, but is not the dual itself (it includes redundant variables and constraints). A critical point is that only the right-hand side vector of $\Lambda_{\mathcal{C}}(\mathcal{A})$ depends on $\mathcal{A}$. The following result is central to our algorithm; inequality (34) is a Benders cut [B62].

Lemma 2.3 *Let $(\bar{f}, \bar{\theta}, \bar{t}, \bar{P}, \bar{D}, \bar{\xi}^+, \bar{\xi}^-, \bar{\varrho}, \bar{\eta})$ be optimal dual variables for problem $\Lambda_{\mathcal{C}}(\mathcal{A})$. The inequality*

$$\bar{t} + \sum_{(i,j) \in E} ((\bar{\xi}_{ij}^+ + \bar{\xi}_{ij}^-)\mathbf{M}(1 - z_{ij})) + \sum_{(i,j) \in E} (\frac{1}{u_{ij}} \bar{\varrho}_{ij}(1 - z_{ij})) + \sum_{(i,j) \in E} \bar{\eta}_{ij} z_{ij} \geq 1 + \epsilon, \tag{34}$$

is valid for the min-cardinality problem, and cuts-off $z^{\mathcal{A}}$ if $\mathcal{C}$ defeats $\mathcal{A}$.

Proof. Let $\hat{\mathcal{A}}$ be an attack. As noted above, the constraints for the dual of $\Lambda_{\mathcal{C}}(\mathcal{A})$ are independent of $\mathcal{A}$, and so $(\bar{f}, \bar{\theta}, \bar{t}, \bar{P}, \bar{D}, \bar{\xi}^+, \bar{\xi}^-, \bar{\varrho}, \bar{\eta})$ is feasible for the dual of $\Lambda_{\mathcal{C}}(\hat{\mathcal{A}})$. Further, the left-hand side of (34), evaluated at $z^{\hat{\mathcal{A}}}$, is the objective value attained by $(\bar{f}, \bar{\theta}, \bar{t}, \bar{P}, \bar{D}, \bar{\xi}^+, \bar{\xi}^-, \bar{\varrho}, \bar{\eta})$ in the dual of $\Lambda_{\mathcal{C}}(\hat{\mathcal{A}})$. By duality, this constitutes an upper bound on the value of problem $\Lambda_{\mathcal{C}}(\hat{\mathcal{A}})$, which is at least $1+\epsilon$ if $\hat{\mathcal{A}}$ defeats $\mathcal{C}$. So (34) is valid. And if $\mathcal{C}$ defeats $\mathcal{A}$, by strong duality the left-hand side of (34), evaluated at $z^{\mathcal{A}}$, equals $t^*_{\mathcal{C}}(z^{\mathcal{A}}) < 1+\epsilon$, and (34) cuts-off $z^{\mathcal{A}}$. ■

We now present an iterative algorithm for the min-cardinality problem. Our algorithm maintains a "master attacker" linear mixed-integer program (MIP) which relaxes the exponentially large set of constraints $t_{\mathcal{C}}(z) \geq 1+\epsilon \;\; \forall\, \mathcal{C}$. We initialize the master attacker MIP as $\min\{\sum_{(i,j)} z_{ij} : z_{ij} = 0 \text{ or } 1, \;\forall\, (i,j)\}$. In a given iteration the solution of the master attacker MIP yields an attack $\mathcal{A}$; either $\mathcal{A}$ is defeated by some configuration $\mathcal{C}$, which is tested by solving the controller's problem (24), or, as we will see, $\mathcal{A}$ is an optimal attack. In the former case, the Benders' cut (34) is added to the master attacker MIP, excluding $\mathcal{A}$. Formally, our algorithmic template is as follows:

Algorithm for min-cardinality problem

1. Attacker: Solve master attacker MIP, obtaining attack $\mathcal{A}$.

2. Controller: Search for a configuration $\mathcal{C}$ such that $t^*_{\mathcal{C}}(z^{\mathcal{A}}) < 1+\epsilon$.

(2.a) If no such $\mathcal{C}$ exists, **EXIT**: $\mathcal{A}$ is a min-cardinality successful attack.

(2.b) Otherwise, add to the master the Benders' cut (34). **Go to 1.**

Inductively, by Lemma 2.3, at each iteration the master attacker MIP is a relaxation of the minimum-cardinality problem and thus the value of the master at any execution of Step 1, i.e. the value $\sum_{(i,j)} z^{\mathcal{A}}_{ij}$, is a lower bound on the cardinality of any successful attack. Thus the exit condition in Step 2.a is correct, since it proves that $\mathcal{A}$ is successful. Finally, the algorithm is finite since at each Step 2.b we cut-off one new 0/1 vector.

The template above can be enhanced in many ways through the use of heuristics and by strengthening the cuts. As per Lemma 2.3, any vector dual-feasible for $\Lambda_{\mathcal{C}}(\mathcal{A})$ yields a valid inequality of the form (34). For any given configuration $\mathcal{C}$, the collection of all such cuts is equivalent to the dual of $\Lambda_{\mathcal{C}}(\mathcal{A})$, for every $\mathcal{A}$. This dual can be represented in compact form by introducing auxiliary variables and constraints; adding this dual to the master MIP considerably strengthens the formulation in one step, but at the cost of substantially enlarging it. See [28] for details.

In order to make formulation $\Lambda_{\mathcal{C}}(\mathcal{A})$ precise we must choose a value for $\mathbf{M}$. Integer programming folklore dictates that $\mathbf{M}$ should be chosen small to improve the tightness of the linear programming relaxation of the master problem, that is to say, how close an approximation to the integer program is provided by its LP relaxation. We have also found that popular optimization packages show significant numerical instability when solving power flow linear programs. In our experience this is the primary reason for choosing $\mathbf{M}$ as small as possible; in particular $\mathbf{M}$ should not grow with network size since this would yield an approach that very quickly becomes unmanageable. In the Appendix, we prove that a valid choice for $\mathbf{M}$ is $\max_{(i,j)\in E}\left\{\frac{1}{\sqrt{x_{ij}}\, u_{ij}}\right\}$.

2.2 Numerical experiments with the min-cardinality model

In the experiments reported in this section we used a 3.4 GHz Xeon machine with 2 MB L2 cache and 8 GB RAM. All experiments were run using a single core. The LP/IP solver was Cplex v. 10.01, with default settings.

2.2.1 Data sets

For our experiments we used problem instances of two types; all problem instances are available for download (http://www.columbia.edu/∼dano/research/pgrid/Data.zip).

(a) Two of the IEEE "test cases" [18]: the "57 bus" case (57 nodes, 78 arcs, 4 generators) and the "118 bus" case (118 nodes, 186 arcs, 17 generators).

(b) Two artificial examples were also created. One was a "square grid" network with 49 nodes and 84 arcs, 4 generators and 14 demand nodes. We also considered a modified version of this data set with 8 generators but same sum $\sum_{i\in\mathcal{G}} P_i^{max}$. Generator and demand nodes, as well as generator output bounds and nominal demand amounts, were chosen at random. We point out that square grids frequently arise as difficult networks for combinatorial problems (they are sparse but highly symmetric and have high *tree-width*, see [24], [6]). We created a second artificial network by taking two copies of the 49-node network and adding a random set of arcs to connect the two copies; with resistances (resp. capacities) equal to the average in the 49-node network plus a small random perturbation. This yielded a 98- node, 204-arc network, with 28 demand nodes, and we used 10, 12, and 15 generator variants.

In all cases, each of the generator output lower bounds P_i^{min} was set to a random fraction (but never higher than 80%) of the corresponding P_i^{max}.

An important consideration involves the capacities u_{ij} – should capacities be too small, or too large, the problem we study tends to become quite easy. For example, if a generator accounts for 20% of all demand then in a tightly capacitated situation the removal of just one arc incident with that node could constitute a successful attack for T^{min} large. For the purposes of our study, we assumed constant capacities for the two networks in (a) and the initial network in (b); these constants were scaled, through experiments with our algorithm, precisely to make the problems we solve more difficult. A topic of further research would be to analyze the $N-k$ problem under regimes where capacities are significantly different across arcs, possibly reflecting a condition of pre-existing stress. In Section 3.4, which addresses experiments involving the second model in this paper, we consider some variations in capacities.

2.2.2 Goals of the experiments

The experiments focus, primarily, on the workload incurred by our algorithm on the problem instances described above. A second issue we examine whether the number of generators exponentially impact performance – does the algorithm need to enumerate a large fraction of all the configurations? In general, what features of a problem instance adversely affect the algorithm?

As noted above, previous studies (see [4], [3], [25]) involving integer programming methods applied to the $N-k$ problem have considered examples with up to 79 arcs, and sparse. In this paper, in addition to considering significantly larger examples we also face the added combinatorial complexity caused by having many configurations. Potentially, our algorithm could rapidly break down – thus, our focus on performance.

2.2.3 Results

Tables 1 and 2 refer to the 57-node and 118-node case, respectively, Table 3 considers the artificial 49-node case and Table 4 considers the 98-node case. In the tables, each row corresponds to a value of the minimum throughput T^{min}, while each column corresponds to an attack cardinality. For each (row, column) combination, the corresponding cell is labeled "Not Enough" if using any attack of the corresponding cardinality or smaller the attacker will not be able to reduce demand below the stated throughput. "Success" means that some attack of the given cardinality or smaller does succeed. Further, we also indicate the number of iterations that the algorithm took in order to prove the given outcome (shown in parentheses) as well as the corresponding CPU time in seconds. Thus, for example, in Table 2, the algorithm proved that using an attack of size 3 or smaller we cannot reduce total demand below 75% of the nominal value; this required 4 iterations which overall took 267 seconds. At the same time, in 7 iterations (6516 seconds) the algorithm found a successful attack of cardinality 4.

Comparing the 57- and 118-node cases, the significantly higher CPU times for the second case could be explained by the much larger number of arcs. The larger number of generators could also be a cause – however, the number of configurations in the second case is more than eight thousand times larger than that of the first; much larger than the actual slowdown shown by the tables.

Table 1: ***Runs of first model on 57-node, 78-arc, 4-generator test case***

Entries show: (iteration count), CPU seconds, Attack status (**F** = cardinality too small, **S** = attack success)					
	Attack cardinality				
Min. throughput	**2**	**3**	**4**	**5**	**6**
0.75	(1), 2, **F**	(2), 3, **S**			
0.70	(1), 1, **F**	(3), 7, **F**	(48), 246, **F**	(51), 251, **S**	
0.60	(2), 2, **F**	(3), 6, **F**	(6), 21, **F**	(6), 21, **S**	
0.50	(2), 2, **F**	(3), 7, **F**	(6), 13, **F**	(6), 13, **F**	(6), 13, **S**
0.30	(1), 1, **F**	(2), 3, **F**	(2), 3, **F**	(2), 3, **F**	(2), 3, **F**

Table 2: ***Runs of first model on 118-node, 186-arc, 17-generator test case***

Entries show: (iteration count), CPU seconds, Attack status (**F** = cardinality too small, **S** = attack success)			
	Attack cardinality		
Min. throughput	**2**	**3**	**4**
0.92	(4), 18, **S**		
0.90	(5), 180, **F**	(6), 193, **S**	
0.88	(4), 318, **F**	(6), 595, **S**	
0.84	(2), 23, **F**	(6), 528, **F**	(148), 6562, **S**
0.80	(2), 18, **F**	(5), 394, **F**	(7), 7755, **F**
0.75	(2), 14, **F**	(4), 267, **F**	(7), 6516, **F**

Table 3 presents experiments with our algorithm on the 49-node, 84-arc network, first using 4 and then 8 generators. Not surprisingly, the network with 8 generators proves more resilient (even though total generator capacity is the same) – for example, an attack of cardinality 5 is needed to reduce throughput below 84%, whereas the same can be achieved with an attack of size 3 in the case of the 4-generator network. Also note that the running-time performance does not significantly degrade as we move to the 8-generator case, even though the number of generator configurations has grown by a factor of 16. Not surprisingly, the most time-consuming cases are those where the adversary fails, since here the algorithm must prove that this is the case (i.e. prove that no successful attack of a given cardinality exists) while in a "success" case the algorithm simply needs to find *some* successful attack of the right cardinality.

Table 4 describes similar tests on the 98-node, 204-arc network. Note that in the 15 generator case there are over 30000 generator configurations that must be examined, at least implicitly, in order to certify that a given attack is successful. But, as in the case of Table 3, the number of generators does not have an exponential impact on the overall running time. Thus, the experiments appear to show that the number of generators plays a second-order role in the complexity of the algorithm; the total number of iterations depends weakly on the total number of generator configurations, and the primary agent behind complexity is the topological structure of the network.

With a few exceptions, the running time tends to decrease, for a given attack cardinality, once the minimum throughput is sufficiently past the threshold where no successful attack exists. This can be explained as follows: as the minimum throughput decreases the controller has more ways to defeat the attacker, and this is leveraged by our algorithm – the cuts added in step (2.b) of our

Table 3: ***Runs of first model on 49-node, 84-arc network***

Entries show: (iteration count), CPU seconds, Attack status (**F** = cardinality too small, **S** = attack success)				
4 generators				
	Attack cardinality			
Min. throughput	**2**	**3**	**4**	**5**
0.84	(4), 129, **F**	(4), 129, **S**		
0.82	(4), 364, **F**	(35), 1478, **F**	(36), 1484, **S**	
0.78	(4), 442, **F**	(4), 442, **F**	(26), 746, **S**	
0.74	(4), 31, **F**	(11), 242, **F**	(168), 4923, **F**	(168), 4923, **S**
0.70	(3), 31, **F**	(4), 198, **F**	(10), 1360, **F**	(203), 3067, **S**
0.62	(4), 86, **F**	(4), 86, **F**	(131), 2571, **F**	(450), 34298, **F**
8 generators				
	Attack cardinality			
Min. throughput	**2**	**3**	**4**	**5**
0.90	(1), 13, **F**	(3), 133, **S**		
0.86	(1), 59, **F**	(5), 357, **F**	(13), 1291, **S**	
0.84	(1), 48, **F**	(4), 227, **F**	(41), 2532, **F**	(43), 2535, **S**
0.80	(1), 14, **F**	(4), 210, **F**	(8), 1689, **F**	(50), 2926, **S**
0.74	(1), 8, **F**	(3), 101, **F**	(10), 1658, **F**	(68), 23433, **F**

algorithm enable us to prove an effective lower bound on the minimum attack cardinality needed to obtain a successful attack. Also (consider the cases corresponding to cardinality = 4; and we have 12 or 15 generators) CPU time increases with decreasing minimum throughput so long as a successful attack does exist. This can also be explained, as follows: in order for the algorithm to terminate it must generate a successful attack, but this task becomes more difficult as the minimum throughput decreases (the controller has more options). Roughly speaking, in summary, we would expect the problem to be "easiest" (for a given attack cardinality) near extreme values of the minimum demand threshold; the experiments overall confirm this expectation.

3 A continuous, nonlinear attack problem

In this section we study a new attack model, motivated by the sense (see e.g. [27]) that recent real-world blackouts were not simply the result of discrete line failures, and, rather, the system as a whole was already under stress when the failures took place. It seems difficult to derive a comprehensive model that accounts for each of the many forms of 'stress' and 'noise' in the operation of a grid. Instead we seek a generic modeling methodology that can serve to expose system vulnerabilities.

We posit that the performance of a stressed line can be approximated by perturbing that line's resistance (or, for AC models, the susceptance, etc.). For example, by significantly increasing the resistance of a line we will, in general, force the power flow on that line to zero. This paradigm becomes particularly effective when the resistances of many lines are simultaneously altered in an adversarial fashion. Our second model is as follows:

(I) The attacker sets the resistance x_{ij} of any arc (i, j).

(II) The attacker is constrained: we must have $x \in F$ for a certain known set F.

(III) The output of each generator i is fixed at a given value P_i, and similarly each demand value D_i is also fixed at a given value.

Table 4: ***Runs of first model on 98-node, 204-arc network***

Entries show: (iteration count), CPU seconds, Attack status (**F** = cardinality too small, **S** = attack success)			
10 generators			
	Attack cardinality		
Min. throughput	**2**	**3**	**4**
0.89	(2) 177, **F**	(30) 555, **S**	
0.86	(2), 195, **F**	(12), 5150, **F**	(14), 5184, **S**
0.84	(2), 152, **F**	(11), 7204, **F**	(35), 223224, **F**
0.82	(2), 214, **F**	(9), 11458, **F**	(16), 225335, **F**
0.75	(2), 255, **F**	(9), 5921, **F**	(17), 151658, **F**
0.60		(1), 4226, **F**	N/R
12 generators			
	Attack cardinality		
Min. throughput	**2**	**3**	**4**
0.92	(2), 318, **F**	(11), 7470, **F**	(14), 11819, **S**
0.90	(2), 161, **F**	(11), 14220, **F**	(18), 16926, **S**
0.88	(2), 165, **F**	(10), 11178, **F**	(15), 284318, **S**
0.84	(2), 150, **F**	(9), 4564, **F**	(16), 162645, **F**
0.75	(2), 130, **F**	(9), 7095, **F**	(15), 93049, **F**
15 generators			
	Attack cardinality		
Min. throughput	**2**	**3**	**4**
0.94	(2), 223, **F**	(11), 654, **S**	
0.92	(2), 201, **F**	(11), 10895, **F**	(18), 11223, **S**
0.90	(2), 193, **F**	(11), 6598, **F**	(16), 206350, **S**
0.88	(2), 256, **F**	(9), 15445, **F**	(18), 984743, **F**
0.84	(2), 133, **F**	(9), 5565, **F**	(15), 232525, **F**
0.75	(2), 213, **F**	(9), 7550, **F**	(11), 100583, **F**

(IV) The objective of the attacker is to maximize the overload of any arc, that is to say, the attacker wants to solve

$$\max_{x\in F}\ \max_{(i,j)}\left\{\frac{|f_{ij}|}{u_{ij}}\right\}, \tag{35}$$

where $f = f(x)$ is the (unique) flow vector under resistances x (refer to Lemma 1.1).

As a comment on (III), suppose that e.g. the value of (35) equals 1.25. Then even if we allow demands to be reduced, but insist that this be done under a *fair* demand-reduction discipline (one that decreases all demands by the same factor) the system will lose 25% of the total demand if line overloads are to be avoided (and it is not surprising that the same qualitative conclusion holds even if demands are "unfairly" reduced to minimize maximum overload; see Table 13). Thus we expect that the impact of (III), under this model, may not be severe.

It will be more convenient to deal with the inverses of resistances, the so-called "conductances." For each $(i,j) \in E$, write $y_{ij} = 1/x_{ij}$, and let y be the vector of y_{ij}. Likewise, instead of considering a set $F \subseteq \mathcal{R}^E$ of allowable values x_{ij} we consider a set Γ describing the conductance values that the adversary is allowed to use. The problem of interest becomes

$$\max_{y\in\Gamma}\ \max_{(i,j)}\left\{\frac{|f_{ij}(y)|}{u_{ij}}\right\}, \tag{36}$$

where as just discussed the notation $f_{ij}(y)$ is justified. A relevant example of a set Γ is given by:

$$\sum_{(i,j)} \frac{1}{y_{ij}} \leq B, \qquad \frac{1}{x_{ij}^U} \leq y_{ij} \leq \frac{1}{x_{ij}^L} \quad \forall\,(i,j), \tag{37}$$

where B is a given 'budget', and, for any arc (i,j), x_{ij}^L and x_{ij}^U and indicates a minimum and maximum value for the resistance at (i,j). Suppose the initial resistances x_{ij} are all equal to some common value $\bar{x}$, and we set $x_{ij}^L = \bar{x}$ for every (i,j), and $B = k\,\theta\,\bar{x} + (|E| - k)\bar{x}$, where $k > 0$ is an integer and $\theta > 1$ is large. Then (among other choices) the adversary can make the resistance of (up to) k arcs "very large", a situation reminiscent of the classical $N - k$ problem.

If the objective in (36) is convex then the optimum will take place at some extreme point. In general, the objective is not convex; but experiments show that we tend to converge to points that are either extreme points, or very close to extreme points (see the Section 3.4.3). Problem (36) differs from the standard N-k problem (but see Lemma 3.4) however in our opinion we obtain a better approach for modeling noise. Additionally, in our numerical experiments we were able to handle much larger problems (on the order of 1000 arcs) than with our first model.

3.1 Solution methodology

Problem (36) is not smooth. However, it is equivalent to:

$$\max_{y,p,q} \quad \sum_{(i,j)} \frac{f_{ij}(y)}{u_{ij}}(p_{ij} - q_{ij}) \tag{38}$$

$$\text{s.t.} \quad \sum_{(i,j)} (p_{ij} + q_{ij}) \;=\; 1, \tag{39}$$

$$y \in \Gamma, \quad p, q \geq 0. \tag{40}$$

We sketch a proof of the equivalence. Suppose (y^*, p^*, q^*) is an optimal solution to (38)-(40); let $(\hat{i}, \hat{j})$ be such that $|f_{\hat{i}\hat{j}}(y^*)|/u_{\hat{i}\hat{j}} = \max_{(i,j)} |f_{ij}(y^*)|/u_{ij}$. Then without loss of generality if $f_{\hat{i}\hat{j}}(y^*) > 0$ ($f_{\hat{i}\hat{j}}(y^*) \leq 0$) we have $p^*_{\hat{i}\hat{j}} = 1$ (resp., $q^*_{\hat{i}\hat{j}} = 1$) and all other p^*, q^* equal to zero. This proves the equivalence once way and the converse is similar.

Our approach will use this formulation and rely on a more explicit representation of the functions $f_{ij}(y)$. This will require a sequence of technical results given in the following section; however a brief discussion of our approach follows.

The objective function (38), although smooth, is not concave. A relatively recent research thrust, which we will leverage, has focused on adapting techniques of (convex) nonlinear programming to nonconvex problems; resulting in a very large literature with interesting and useful results; see [17], [5]. Since one is attempting to solve non-convex minimization problems, there is no guarantee that a global optimum will be found by these techniques. One can sometimes assume that a global optimum is approximately known; and the techniques then are likely to converge to the optimum from an appropriate guess.

In any case, (a) the use of nonlinear models allows for much richer representation of problems, (b) the very successful numerical methodology backing convex optimization is brought to bear, and (c) even though only a local optimum may be found, we are relying on an agnostic optimization technique as opposed to a pure heuristic or a method that makes structural assumptions about the nature of the optimum in order to simplify the problem.

3.1.1 Some comments on the algorithmic framework

Suppose we consider a a linearly constrained problem of the form $\min\{\mathcal{F}(x) : Ax \geq b\}$. Implementations such as LOQO [26] or IPOPT [29] require, in addition to some representation of the linear constraints $Ax \geq b$, subroutines for computing, at any given point $\hat{x}$, (a) the functional value $\mathcal{F}(\hat{x})$, (b) the gradient $\nabla\mathcal{F}(\hat{x})$, and, ideally, (c) the Hessian $\nabla^2\mathcal{F}(\hat{x})$.

If routines for the computation of the gradient or Hessian are not available, then automatic differentiation may be used. At each iteration the algorithms will evaluate the subroutines and perform additional work such as matrix computations. The cumulative run-time accrued in the computation of (a)-(c) could prove significant and it is important to develop fast routines especially in large-scale settings.

We will construct an explicit representation of the functions $f_{ij}(y)$ given above, such that the three evaluation steps in (a)-(c) indeed admit efficient implementations using sparse linear algebra techniques. The approach is "compact" in that, essentially, the only variables we deal with are the y_{ij}. A potential drawback concerns the density of the Hessian that we compute.

In what follows, we will first provide a review of some relevant material in linear algebra (Section 3.1.2). This material is used to make some structural remarks in Section 3.1.3. Section 3.2 describes our algorithms for computing the gradient and Hessian of the objective function for problem (38)-(40). Finally, Section 3.3 presents details of our implementation, and Section 3.4 describes our numerical experiments.

3.1.2 Laplacians

In this section we present some background material on linear algebra and Laplacians of graphs. See [9] for relevant material. As before we have a connected, directed network G with n nodes and m arcs and with node-arc incidence matrix N. For a positive diagonal matrix $Y \in \mathcal{R}^{m\times m}$ write

$$L \;=\; NYN^T, \qquad J \;=\; L + \frac{1}{n}\,\mathbf{1}\mathbf{1}^T, \qquad \mathcal{P} \;=\; I - J.$$

where $\mathbf{1} \in \mathcal{R}^n$ is the vector $(1, 1, \ldots, 1)^T$. L is called a *generalized Laplacian*. L is symmetric positive-semidefinite; if $\lambda_1 \le \lambda_2 \le \ldots \le \lambda_n$ are its eigenvalues and L, and $v^1, v^2, \ldots, v^n$ are the corresponding unit-norm eigenvectors,

$$\lambda_1 = 0, \quad \text{but} \quad \lambda_i > 0 \quad \text{for } i > 1,$$

because G is connected, and thus L has rank $n-1$. Similarly, since $N\mathbf{1} = 0$, we can assume $v^1 = n^{-1/2}\,\mathbf{1}$, and therefore $\mathbf{1}^T v^i = 0$ for $2 \le i \le n$. Lemmas 3.1, 3.2 and 3.3 follow from basic properties of Laplacians, see [9] and references therein.

Lemma 3.1 *L and J have the same eigenvectors, and all but one of their eigenvalues coincide. Further, J is invertible.*

Lemma 3.2 *Let $b \in \mathcal{R}^n$. Any solution to the system of equations $L\alpha = b$ is of the form*

$$\alpha = J^{-1}b + \delta\mathbf{1},$$

for some $\delta \in \mathcal{R}$.

Note that the eigenvalues of $\mathcal{P}$ are 0 and $1 - \lambda_i$, $2 \le i \le n$; thus if we have

$$\sum_{(u,v)} y_{uv} \;<\; 1/2, \quad \text{for all } u, \tag{41}$$

then it is not difficult to show that

$$0 \;<\; 1 - \lambda_i < 1, \qquad \text{for all } i \ge 2. \tag{42}$$

(See [21] for related background). In such a case we can write

$$J^{-1} \;=\; (I - \mathcal{P})^{-1} \;=\; \sum_{k=0}^{\infty} \mathcal{P}^k. \tag{43}$$

Lemma 3.3 *For any integer $k > 0$, $\mathcal{P}^k \;=\; (I - NYN^T)^k - \frac{1}{n}\mathbf{1}\mathbf{1}^T$.*

3.1.3 Observations

Consider problem (38)-(40), where, as per our modeling assumption (III), b denotes the (fixed) net supply vector, i.e. $b_i = P_i$ for a generator i , $b_i = -D_i$ for a demand node i, and $b_i = 0$ otherwise. Writing $Y = \mathbf{diag}\{y_{ij}\}$, we have that the flows f and angles θ are obtained by solving the system

$$\begin{aligned} N^T\theta - Y^{-1}f &= 0 \quad (44)\\ Nf &= b. \quad (45)\end{aligned}$$

In what follows, it will be convenient to assume that condition (42) holds, i.e. $1 - \lambda_i < 1$ for each i. Next we argue that without loss of generality we can assume that this holds.

As noted above, this condition will be satisfied if $\sum_{(u,v)} y_{uv} < 1/2$ for all u (eq. (41) above). But Lemma 1.1 implies that if all y_{uv} are scaled by a common constant $\mu > 0$, the flows f that solve (44)-(45) do not change. Thus, if we assume that the set Γ in our formulation (38)-(40) is bounded (as is the case if we use (37)) then, without loss of generality, (41) indeed holds. Consequently, in what follows we will assume that

$$\exists\ r < 1 \ \text{ such that } 1 - \lambda_i < r \text{ for } 2 \le i \le n. \quad (46)$$

By Lemma 3.2 each solution to (44)-(45) is of the form

$$\begin{aligned} &\theta = J^{-1}b + \delta\mathbf{1} \quad \text{for some } \delta \in \mathcal{R}, \quad (47)\\ &f = YN^TJ^{-1}b. \end{aligned}$$

For each arc (i,j) denote by ν_{ij} the column of N corresponding to (i,j). Using (43) we thus have

$$\begin{aligned} f_{ij} &= y_{ij}\nu_{ij}^TJ^{-1}b \;=\; y_{ij}\nu_{ij}^T\left[\sum_{k=0}^{\infty}\mathcal{P}^k\right]b, \quad \forall (i,j), \quad \text{and} \quad (48)\\ \theta_i - \theta_j &= \nu_{ij}^TJ^{-1}b \;=\; \nu_{ij}^T\left[\sum_{k=0}^{\infty}\mathcal{P}^k\right]b, \end{aligned}$$

Below we will be handling expressions with infinite series such as the above. In order to facilitate the analysis we need a 'uniform convergence' argument, as follows. For $y \in \Gamma$, write

$$\mathcal{P} \;=\; \mathcal{P}(y) \;=\; U(y)\Lambda(y)U(y)^T,$$

where $U(y)$ is a unitary matrix and $\Lambda(y)$ is the diagonal matrix containing the eigenvalues of $\mathcal{P}(y)$. Hence, for any $k \ge 1$ and any arc (i,j) (and dropping the dependence on y for simplicity),

$$|\nu_{ij}^T\mathcal{P}^kb| \;=\; |\nu_{ij}^TU\Lambda^kU^Tb| \;<\; \rho^k, \quad (49)$$

for some $\rho < 1$, by (46). We will rely on this bound below.

The following result provides a parallel between the continuous interdiction model we consider here the discrete model of network vulnerability considered in Section 2.1.

Lemma 3.4 *Let S be a set of arcs whose removal does not disconnect G. Suppose we fix the values $y_{ij} = 1/x_{ij}$ for each arc $(i,j) \notin S$, and we likewise set $y_{st} = \epsilon$ for each arc $(s,t) \in S$. Let $(f(y), \theta(y))$ denote the resulting power flow on G, and $(\bar{f}, \bar{\theta})$ the power flow on $G - S$. Then*

(a) $\lim_{\epsilon\to 0} f_{st}(y) = 0$, for all $(s,t) \in S$,

(b) For any $(u,v) \notin S$, $\lim_{\epsilon\to 0} f_{uv}(y) = \bar{f}_{uv}$.

(c) For any (u,v), $\lim_{\epsilon\to 0}(\theta_u(y) - \theta_v(y)) = \bar{\theta}_u - \bar{\theta}_v$.

Proof. (a) Let $\tilde{G} = G - (s,t)$, let $\tilde{N}$ be node-arc incidence matrix of $\tilde{G}$, $\tilde{Y}$ the restriction of Y to all arcs other (s,t), and $\tilde{\mathcal{P}} = I - \tilde{N}\tilde{Y}\tilde{N}^T - \frac{1}{n}\mathbf{1}\mathbf{1}^T$. For any integer $k \geq 1$ we have by Lemma 3.3

$$\lim_{\epsilon \to 0} \mathcal{P}^k = \lim_{\epsilon \to 0} (I - NYN^T)^k - \frac{1}{n}\mathbf{1}\mathbf{1}^T = (I - \tilde{N}\tilde{Y}\tilde{N}^T)^k - \frac{1}{n}\mathbf{1}\mathbf{1}^T = \tilde{\mathcal{P}}^k.$$

Consequently, by (48), for any $(s,t) \in S$,

$$\lim_{\epsilon \to 0} f_{st} \;=\; \lim_{\epsilon \to 0} \left[y_{st} \nu_{st}^T \left(\sum_{k=0}^{\infty} \mathcal{P}^k \right) b \right] \;=\; \sum_{k=0}^{\infty} \left[\lim_{\epsilon \to 0} y_{st} \left(\nu_{st}^T \mathcal{P}^k b \right) \right] \;=\; 0;$$

the exchange between summation and limit is valid by (49). The proofs of (b), (c) are similar. ■

Lemma 3.4 can be interpreted as describing a particular attack pattern – make x_{ij} very large for $(i,j) \in S$ and leave all other x_{ij} unchanged. Our numerical experiments show a pattern similar to that assumed by the Lemma: the attacker tends to concentrate most of the budget on a small number of arcs.

3.2 Efficient computation of the gradient and Hessian

Here we give efficient closed-form expressions for the gradient and Hessian of the objective in (36). As before, we denote by ν_{ij} the column of the node-arc incidence matrix of the network corresponding to arc (i,j). First we present a technical result. This will be followed by the development of formulas for the gradient (eqs. (50)-(51)) and the Hessian (eqs. (52)-(54)).

Lemma 3.5 *For any integer $k > 0$, any arc (i,j) and any $\tilde{b} \in \mathcal{R}^n$,*

$$\begin{aligned} &(a) \quad \mathbf{1}^T \mathcal{P}^k \;=\; 0, \\ &(b) \quad \frac{\partial}{\partial y_{ij}} \left[\mathcal{P}^k \tilde{b} \right] \;=\; \mathcal{P} \frac{\partial}{\partial y_{ij}} \left[\mathcal{P}^{k-1} \tilde{b} \right] - \nu_{ij} \nu_{ij}^T \mathcal{P}^{k-1} \tilde{b}. \end{aligned}$$

Proof. Note that $\mathbf{1}^T \mathcal{P} = \mathbf{1}^T (I - J) = \mathbf{1}^T (I - NYN^T - \frac{1}{n}\mathbf{1}\mathbf{1}^T) = 0$. Hence $\mathbf{1}^T \mathcal{P}^k = 0$.

$$\begin{aligned}
\frac{\partial}{\partial y_{ij}} \left[\mathcal{P}^k \tilde{b} \right] &= \frac{\partial}{\partial y_{ij}} \left[\mathcal{P}^{k-1} \tilde{b} \right] \;=\; \frac{\partial}{\partial y_{ij}} \left[\left(I - \sum_{(u,v) \in E} y_{uv}\, \nu_{uv} \nu_{uv}^T - \frac{1}{n}\mathbf{1}\mathbf{1}^T \right) \mathcal{P}^{k-1} \tilde{b} \right] \\
&= \frac{\partial}{\partial y_{ij}} \left[\mathcal{P}^{k-1} \tilde{b} \right] - \frac{\partial}{\partial y_{ij}} \left[\left(\sum_{(u,v) \in E} y_{uv}\, \nu_{uv} \nu_{uv}^T \right) \mathcal{P}^{k-1} \tilde{b} \right] - \frac{\partial}{\partial y_{ij}} \left[\frac{1}{n}\mathbf{1}\mathbf{1}^T \mathcal{P}^{k-1} \tilde{b} \right] \\
&= \frac{\partial}{\partial y_{ij}} \left[\mathcal{P}^{k-1} \tilde{b} \right] - \frac{\partial}{\partial y_{ij}} \left[\left(\sum_{(u,v) \in E} y_{uv}\, \nu_{uv} \nu_{uv}^T \right) \mathcal{P}^{k-1} \tilde{b} \right] \\
&= \frac{\partial}{\partial y_{ij}} \left[\mathcal{P}^{k-1} \tilde{b} \right] - \sum_{(u,v) \in E} \frac{\partial}{\partial y_{ij}} \left[y_{uv}\, \nu_{uv} \nu_{uv}^T \mathcal{P}^{k-1} \tilde{b} \right] \\
&= \frac{\partial}{\partial y_{ij}} \left[\mathcal{P}^{k-1} \tilde{b} \right] - \sum_{(u,v) \in E} \left[\frac{\partial y_{uv}}{\partial y_{ij}} \right] \nu_{uv} \nu_{uv}^T \mathcal{P}^{k-1} \tilde{b} - \sum_{(u,v) \in E} y_{uv} \frac{\partial}{\partial y_{ij}} \left[\nu_{uv} \nu_{uv}^T \mathcal{P}^{k-1} \tilde{b} \right] \\
&= \frac{\partial}{\partial y_{ij}} \left[\mathcal{P}^{k-1} \tilde{b} \right] - \nu_{ij} \nu_{ij}^T \mathcal{P}^{k-1} \tilde{b} - \sum_{(u,v) \in E} y_{uv}\, \nu_{uv} \nu_{uv}^T \frac{\partial}{\partial y_{ij}} \left[\mathcal{P}^{k-1} \tilde{b} \right] \\
&= \left[I - \sum_{(u,v) \in E} y_{uv}\, \nu_{uv} \nu_{uv}^T \right] \frac{\partial}{\partial y_{ij}} \left[\mathcal{P}^{k-1} \tilde{b} \right] - \nu_{ij} \nu_{ij}^T \mathcal{P}^{k-1} \tilde{b} \\
&= \left[\mathcal{P} + \frac{1}{n}\mathbf{1}\mathbf{1}^T \right] \frac{\partial}{\partial y_{ij}} \left[\mathcal{P}^{k-1} \tilde{b} \right] - \nu_{ij} \nu_{ij}^T \mathcal{P}^{k-1} \tilde{b} \\
&= \mathcal{P} \frac{\partial}{\partial y_{ij}} \left[\mathcal{P}^{k-1} \tilde{b} \right] - \nu_{ij} \nu_{ij}^T \mathcal{P}^{k-1} \tilde{b} + \frac{\partial}{\partial y_{ij}} \left[\frac{1}{n}\mathbf{1}\mathbf{1}^T \mathcal{P}^{k-1} \tilde{b} \right] \;=\; \mathcal{P} \frac{\partial}{\partial y_{ij}} \left[\mathcal{P}^{k-1} \tilde{b} \right] - \nu_{ij} \nu_{ij}^T \mathcal{P}^{k-1} \tilde{b}.
\end{aligned}$$

where the third and the last equality follow from (a). ■

Let $\widetilde{\nabla}_{ij} := \frac{\partial}{\partial y_{ij}} J^{-1} b = \frac{\partial}{\partial y_{ij}} \sum_{k=0}^{\infty} \mathcal{P}^k b$, by (43). For $k \geq 1$, $\frac{\partial}{\partial y_{ij}}[\mathcal{P}^k b] = -\sum_{h=1}^{k} \mathcal{P}^{k-h} \nu_{ij} \nu_{ij}^T \mathcal{P}^{h-1} b$ (by Lemma 3.5(b)), and so

$$\widetilde{\nabla}_{ij} = -\left(\sum_{k=0}^{\infty} \mathcal{P}^k\right) \nu_{ij} \nu_{ij}^T \left(\sum_{k=0}^{\infty} \mathcal{P}^k\right) b = -J^{-1}\, \nu_{ij} \nu_{ij}^T\, \theta,$$

where the last equality follows from (47) and (43), and the fact that $\nu_{ij}^T \mathbf{1} = 0$. Using (48), the gradient of function $f_{uv}(y)$ with respect to the variables y_{ij} can be written as:

$$\frac{\partial f_{uv}}{\partial y_{ij}} = y_{uv}\, \nu_{uv}^T \frac{\partial}{\partial y_{ij}} J^{-1} b = y_{uv}\, \nu_{uv}^T\, \widetilde{\nabla}_{ij}, \quad (i,j) \neq (u,v) \tag{50}$$

$$\frac{\partial f_{ij}}{\partial y_{ij}} = \nu_{ij}^T \theta + y_{ij}\, \nu_{ij}^T \frac{\partial}{\partial y_{ij}} J^{-1} b = \nu_{ij}^T \theta + y_{ij} \nu_{ij}^T \widetilde{\nabla}_{ij}. \tag{51}$$

We similarly develop closed-form expressions for the second-order derivatives. For $(u,v) \neq (i,j)$ and $(u,v) \neq (s,t)$,

$$\begin{aligned} \frac{\partial^2 f_{uv}}{\partial y_{ij} \partial y_{st}} &= y_{uv} \nu_{uv}^T \left[\, J^{-1}\, \nu_{ij} \nu_{ij}^T\, J^{-1} \nu_{st} \nu_{st}^T + J^{-1}\, \nu_{st} \nu_{st}^T\, J^{-1}\, \nu_{ij} \nu_{ij}^T \,\right] \theta \\ &= -y_{uv} \nu_{uv}^T J^{-1} \left[\nu_{ij} \nu_{ij}^T \widetilde{\nabla}_{st} + \nu_{st} \nu_{st}^T \widetilde{\nabla}_{ij} \right]. \end{aligned} \tag{52}$$

Similarly, the remaining terms are:

$$\frac{\partial^2 f_{uv}}{\partial y_{uv}^2} = 2\, \nu_{uv}^T \widetilde{\nabla}_{uv} - 2\, y_{uv} \nu_{uv}^T J^{-1} \nu_{uv} \nu_{uv}^T \widetilde{\nabla}_{uv}, \tag{53}$$

$$\frac{\partial^2 f_{uv}}{\partial y_{uv} \partial y_{ij}} = \nu_{uv}^T \widetilde{\nabla}_{ij} - y_{uv}\, \nu_{uv}^T J^{-1} \left[\nu_{ij} \nu_{ij}^T \widetilde{\nabla}_{uv} + \nu_{uv} \nu_{uv}^T \widetilde{\nabla}_{ij} \right] \tag{54}$$

3.3 Implementation details

Our approach was applied to problem (38)-(40), using

$$\Gamma = \left\{ y \geq 0 : \sum_{(i,j)} \frac{1}{y_{ij}} \leq B,\ \frac{1}{x_{ij}^U} \leq y_{ij} \leq \frac{1}{x_{ij}^L}\ \forall\, (i,j) \right\}.$$

At each iteration we compute the gradient and Hessian using (50), (51), (52)-(54). This requires the computation and storage of quantities $\nu_{uv}^T J^{-1} \nu_{ij}$ for each pair of arcs (i,j), (u,v); at any given iteration, this can be done in $O(n^2 + nm)$ space. In order to compute $\nu_{uv}^T J^{-1} \nu_{ij}$, for given (i,j) and (u,v), we simply solve the sparse linear system on variables κ, λ:

$$N^T \kappa - Y^{-1} \lambda = 0, \qquad N\lambda = \nu_{ij}. \tag{55}$$

As in (47), we have $\kappa = J^{-1} \nu_{ij} + \delta \mathbf{1}$ for some real δ. But then $\nu_{uv}^T \kappa = \nu_{uv}^T J^{-1} \nu_{ij}$, the desired quantity. In order to solve (55) we use Cplex (any efficient linear algebra package would suffice). In our implementation we use an iteration limit, but apply additional stopping criteria:

(1) If both primal and dual are feasible, we consider the relative error between the primal and dual values, $\epsilon = \frac{\text{PV - DV}}{\text{DV}}$, where 'PV' and 'DV' refer to primal and dual values respectively. If the relative error ϵ is less than some desired threshold we stop, and report the solution as "ϵ-locally-optimal."

(2) If on the other hand we reach the iteration limit without stopping, then we consider the last iteration at which we had both primal and dual feasible solutions. If such an iteration exists, then we report the corresponding configuration of resistances along with the associated maximum congestion value. If such an iteration does not exist, then we report the run as unsuccessful.

Our implementation relies on LOQO [26] to carry out the underlying search. LOQO uses a primal-dual interior-point method to solve a sequence of quadratic approximations to a given optimization problem. LOQO stops if at any iteration the primal and dual problems are feasible and with objective values that are *close* to each other, in which case a local optimal solution is found. Additionally, LOQO uses an upper bound on the overall number of iterations it is allowed to perform.

Finally, we used the starting point $y_{ij} = 1/x_{ij}^L$ for each arc (i, j).

3.4 Experiments

In the experiments reported in this section we used a 2.66 GHz Xeon machine with 2 MB L2 cache and 16 GB RAM. All experiments were run using a single core. Altogether we report on 37 runs of the algorithm.

For our tests we used the 57- and 118-node test cases as in Section 2.2 with some variations on the capacities; as well as the 49-node "square grid" example and three larger networks created using the replication technique described at the start of Section 2.2: a 300-node, 409-arc network, a 600-node, 990-arc network, and a 619-node, 1368-arc network. Additional artificial networks were created to test specific conditions. All data sets are available for download (http://www.columbia.edu/∼dano/research/pgrid/Data.zip).

We considered several three constraint sets Γ as in (37):

(1) $\mathbf{\Gamma(1)}$, where for all (i, j), $x_{ij}^L = 1$ and $x_{ij}^U = 5$,

(2) $\mathbf{\Gamma(2)}$, where for all (i, j), $x_{ij}^L = 1$ and $x_{ij}^U = 10$,

(3) $\mathbf{\Gamma(3)}$, where for all (i, j), $x_{ij}^L = 1$ and $x_{ij}^U = 20$.

In each case, we set $B = \sum_{(i,j)} x_{ij}^L + \Delta\mathrm{B}$, where ΔB represents an "excess budget". Note that for example in the case $\Delta\mathrm{B} = 30$, under $\Gamma(2)$, the attacker can increase the resistance of up to 3 arcs by a factor of 10 from their minimum value, with 3 units of budget left over. And under $\Gamma(1)$, up to 6 arcs can have their resistance increased by a factor of 5. In either case we have a situation reminiscent of the $N - k$ problem, with small k.

3.4.1 Focus of the experiments

In these experiments, we first study how the behavior of the algorithm changes as network size increases up to roughly 1000 arcs and as ΔB increases. An additional point we study concerns the structure of the solutions produced by the algorithm – what is the distribution of the x_{ij} obtained at termination, and is there a logic to that distribution? A final set of experiments carry out a comparison with results obtained using the standard $N - k$ model.

3.4.2 Basic run behavior

Tables 5-10 present results for different networks and scenarios. Each column corresponds to a different value of $\mathbf{\Delta B}$. For each run, **"Max Cong"** is the numerical value of the maximum arc congestion (as in (35)) at termination. Additionally, we present the CPU time (in seconds) taken by the algorithm, the number of iterations, and the termination criterion, which is indicated by "Exit Status", with the following interpretation:

(1) '**ϵ-L-opt.**': the algorithm computed an ϵ-locally-optimal solution.

(2) '**PDfeas, Iter: lastItn**': the algorithm reached the iteration limit without finding an ϵ-locally-optimal solution, but there was an iteration at which both primal and dual problems were feasible. 'lastItn' gives the last iteration at which both primal and dual solutions were feasible.

(3) '**opt.**': the algorithm attained LOQO's internal optimality tolerance.

Tables 5 and 6 contain results for the 57- and 118-node networks, respectively, both using set $\Gamma(2)$. Tables 7 and 8 handle the 49-node, 84-arc network, with 14 demand nodes and 4 generators that we considered in section 2.2, using sets $\Gamma(1)$ and $\Gamma(2)$ respectively.

Table 5: ***Runs of second model on 57 nodes, 78 arcs, constraint set*** $\mathbf{\Gamma(2)}$
Iteration Limit: **700,** $\boldsymbol{\epsilon = 0.01}$

	ΔB			
	9	**18**	**27**	**36**
Max Cong	1.070	1.190	1.220	1.209
Time (sec)	8	19	19	19
Iterations	339	Limit	Limit	Limit
Exit Status	ϵ-L-opt.	PDfeas. Iter: 700	PDfeas. Iter: 700	PDfeas. Iter: 700

Table 6: ***Runs of second model on 118 nodes, 186 arcs, constraint set*** $\mathbf{\Gamma(2)}$
Iteration Limit: **700,** $\boldsymbol{\epsilon = 0.01}$

	ΔB			
	9	**18**	**27**	**36**
Max Cong	1.807	2.129	2.274	2.494
Time (sec)	88	200	195	207
Iterations	Limit	578	Limit	Limit
Exit Status	PDfeas. Iter: 302	ϵ-L-opt.	PDfeas. Iter: 700	PDfeas. Iter: 700

Table 9 presents similar results for the network with 300 nodes and 409 arcs (42 generators and 172 loads). Note that for the runs $\Delta B \geq 20$ the maximum load value is identical; the optimal solution values x_{ij} were nearly identical, independent of the initial point given to LOQO.

Table 10 contains the results for the network with 600 nodes and 990 arcs (344 demand nodes and 98 generators) under set $\Gamma(2)$. We observed an interesting issue in the case where $\Delta B = 10$. Here, LOQO terminated with a solution in which, for some arc (i, j), both $p_{ij} > 0$ and $q_{ij} > 0$ (refer to formulation (38)-(40)). In parentheses we give the true value of the maximum congestion obtained by solving the controller's problem using the resistance values (x_{ij}) given by LOQO.

Finally, Table 11 presents experiments on the network with 649 nodes and 1368 arcs. Here, exit status 'DF' means that dual feasibility was achieved, but not primal feasibility. In such a case, the budget constraint (37) was violated – the largest (scaled) violation we observed was 1e-03. Even though this is a small violation, LOQO's threshold for primal feasibility is 1e-06; we simply scaled down any resistance value $x_{ij} > x_{ij}^{min}$ so as to obtain a solution satisfying (37). In Table

Table 7: ***Runs of second model on 49 nodes, 84 arcs, constraint set* $\Gamma(1)$**
Iteration Limit: 800, $\epsilon = 0.01$

	ΔB					
	5	**10**	**15**	**20**	**25**	**30**
Max Cong	0.673054	0.750547	0.815623	0.865806	0.901453	0.951803
Time (sec)	12	15	18	19	28	22
Iterations	258	347	430	461	Limit	492
Exit Status	ϵ-L-opt.	ϵ-L-opt.	ϵ-L-opt.	ϵ-L-opt.	PDfeas Iter: 613	ϵ-L-opt.

Table 8: ***Runs of second model on 49 nodes, 84 arcs, constraint set* $\Gamma(2)$**
Iteration Limit: 800, $\epsilon = 0.01$

	ΔB					
	5	**10**	**15**	**20**	**25**	**30**
Max Cong	0.67306	0.751673	0.815584	0.8685	0.91523	0.9496
Time (sec)	9	13	34	3	29	30
Iterations	177	295	Limit	Limit	Limit	Limit
Exit Status	ϵ-L-opt.	ϵ-L-opt.	PDfeas Iter: 800	PDfeas Iter: 738	PDfeas Iter: 624	PDfeas Iter: 656

11, the quantity following the parenthesis in the "Max Cong" line indicates the resulting maximum congestion, obtained by solving a controller's problem on the network using the reduced resistance values.

Comments: The algorithm runs fairly reliably in cases with up to approximately 1000 arcs; at that point the internal solver (LOQO) starts to develop difficulties. For any given network, note that the computed solution does vary as a function of the parameter ΔB, and in the expected manner, as reflected by the "Max Cong" values. However the performance of the algorithm (running time or number of iterations) appears stable as a function of ΔB. By "stable" what we mean is that even though larger ΔB values correspond to larger numbers of arcs that could be maximally interdicted, the workload incurred by the algorithm *does not* increase "combinatorially" as a function of ΔB. This a significant difference between this algorithm and the algorithm presented above for the $N-k$ problem. In particular, our algorithm appears to allow for practicable analyses of the impact of multiple choices of ΔB; this is a critical feature that parameterizes the risk-aversion of the model.

3.4.3 Distribution of attack weights

A significant question in the context of our model and algorithm concerns the structure of the attack chosen by the adversary. The adversary is choosing continuous values with great leeway; potentially, for example, the adversary could choose them uniformly equal (which, we would argue, would make the model quite uninteresting). The experiments in this section address these issues.

Table 12 describes the distribution of x_{ij} values at termination of the algorithm, for a number of networks and attack budgets. For each test we show first (in parentheses) the number of nodes and arcs, followed by the the attack budget and constraint set. The data for each test shows, for each range of resistance values, the number of arcs whose resistance falls in that range.

Table 9: ***Runs of second model on 300 nodes, 409 arcs, constraint set*** $\mathbf{\Gamma(2)}$
Iteration Limit: 500, $\boldsymbol{\epsilon = 0.01}$

	ΔB			
	9	**18**	**27**	**36**
Max Cong	0.590690	0.694101	0.771165	0.771165
Time (sec)	208	1248	981	825
Iterations	91	Limit	406	320
Exit Status	opt.	PDfeas Iter: 318	opt.	opt

Table 10: ***Runs of second model on 600 nodes, 990 arcs, constraint set*** $\mathbf{\Gamma(2)}$
Iteration Limit: **300,** $\boldsymbol{\epsilon = 0.01}$

	ΔB				
	10	**20**	**27**	**36**	**40**
Max Cong	0.082735 (0.571562)	1.076251	1.156187	1.088491	1.161887
Time (sec)	11848	7500	4502	11251	7800
Iterations	Limit	210	114	Limit	208
Exit Status	PDfeas Iter: 300	ϵ-L-opt.	ϵ-L-opt.	PDfeas Iter: 300	ϵ-L-opt.

Note that in each test case the adversary can increase the resistance of up to (roughly) three arcs to their maximum value. The pattern we observe in the table is that in all three cases (i) many resistances take relatively small values and (ii) a small number of arcs have high resistance. Recall that for set $\Gamma(2)$ we always have $x_{ij}^{max} = 10$, thus in the case of the $(300, 409)$ network exactly three arcs are in the top range, while for the $(600, 990)$ network two are in the top range and one more has relatively high resistance. In the case of the small network there is also a concentration 'at the top' though not in the very highest segment. We have observed this type of behavior in many runs.

In summary, thus, the solutions produced by the algorithm appear to superimpose two separate effects. As argued in the next section, both effects play an important role.

3.4.4 Comparison with the minimum-cardinality attack model

The experiments in this section have as a first goal to effect a comparison with the $N - k$ model as embodied by the mixed-integer programming approach considered in Section 2.1. A direct comparison on a case-by-case basis is not possible for a number of reasons (more on this below) but the purpose of the tests is to investigate whether on "similar" data the two models behave in similar ways.

A second goal of the experiments is to investigate the impact of one of our modeling assumptions (assumption (III) in Section 3), namely that demands and supplies are fixed. Ideally, our model should be robust, that is to say, the attack computed in a run of the algorithm should remain effective even if the controller has the power to adjust demands.

A common thread runs through both goals. Turning to the first goal, it turns out that the modeling assumption (III) is, in fact, what makes a direct comparison with the $N - k$ model

Table 11: ***Runs of second model on 649 nodes, 1368 arcs, constraint set*** $\mathbf{\Gamma(2)}$
Iteration Limit: 500, $\boldsymbol{\epsilon = 0.01}$

	ΔB			
	20	**30**	**40**	**60**
Max Cong	(0.06732) 1.294629	1.942652	(0.049348) 1.395284	2.045111
Time (sec)	66420	36274	54070	40262
Iterations	Limit	374	Limit	Limit
Exit Status	DF	ϵ-L-opt.	DF	PDfeas Iter: 491

Table 12: **Solution histograms for three representative runs**

$(49, 90)$ $\Delta B = 57, \Gamma(3)$		$(300, 409)$ $\Delta B = 27, \Gamma(2)$		$(600, 990)$ $\Delta B = 36, \Gamma(2)$	
Range	**Count**	**Range**	**Count**	**Range**	**Count**
[1, 1]	8	[1, 1]	1	[1, 1]	14
(1, 2]	72	(1, 2]	405	(1, 2]	970
(2, 3]	4	(2, 9]	0	(2, 5]	3
(5, 6]	1	(9, 10]	3	(5, 6]	0
(6, 7]	1			(6, 7]	1
(7, 8]	4			(7, 9]	0
(8, 20]	0			(9, 10]	2

difficult. In principle, in the model in Section 2.1 one could set the desired minimum throughput to 100%, i.e. set $T^{min} = 1.0$. But in that case an attack that disconnects a demand node, even one with tiny demand, would be considered a success for the attacker.

To deal with these issues and still obtain a meaningful comparison, we set an example with 49 nodes and 88 arcs, and an example with 49 nodes and 90 arcs, in which no demand or generator node can be disconnected from the rest by removing up to three arcs. In each case there are 4 generators and 14 demand nodes. A family of problem instances was then obtained by scaling up all capacities by a common constant.

In terms of the mixed-integer programming model, in each instance we constructed a single configuration problem (generator lower bounds = 0) with $T^{min} = 1$, with the goal of investigating its vulnerability should up to three arcs be removed. Here we remind the reader that the algorithms in Section 2.1 seek a minimum-cardinality attack that defeats the controller, and not the most severe attack of a given cardinality – at termination, the optimal attack is certified to be successful (and of minimum-cardinality), but not necessarily the most severe attack of that cardinality. Nevertheless, formulation $\Lambda_{\mathcal{C}}(\mathcal{A})$ (Section 2.1.1) is easily modified so as to produce an approximation (in general, close) to the highest severity of any attack of cardinality $\leq k$. This is done by making the z parameters into 0/1 variables, and adding the constraint $\sum_{(i,j)} z_{ij} \leq k$, where k (= 3, in our test) is the number of arcs that the attacker can remove. A final detail is that since 3 lines will not disconnect the demands from the generators, the "severity" of a given attack is the maximum arc congestion post-attack; thus putting the problem on a common ground with the nonlinear models we consider.

For our tests we used $\Gamma(3)$ (which allows each resistance to increase by up to a factor of 20) with an excess budget of 60, on the network with 49 nodes, 90 arcs, 4 generators and 14 demand nodes. Note that the parameters allow the attacker to concentrate the budget on three arcs.

Table 13 contains the results. Each row corresponds to a different experiment, where the value

indicated by $\boldsymbol{\sigma}$ was used to scale all capacities from their values in the original network. As σ increases the network becomes progressively more difficult to interdict.

In the 'MIP' section, the column headed 'Cong' indicates the congestion (max. arc overload) in the network obtained by removing the arcs produced by the mixed-integer programming model, and the column headed 'ATTACK' indicates which arcs were removed by the MIP. In the 'NONLINEAR' section, 'Cong' indicates the maximum congestion resulting from the increase in resistances computed by the model. We also list the six arcs with highest resistance (and the resistance values). The column headed 'Impact' indicates the maximum congestion obtained by deleting the three arcs with maximum resistance (as computed by the model), while leaving all other resistances unchanged.

Note that the values in the 'Cong' column are consistent with increasing values of $\boldsymbol{\sigma}$ – higher values yield networks that are more difficult to interdict with a given budget, and so the objective value (maximum overload) decreases. In the last two cases the computed attack fails to achieve maximum overload greater than 1.0.

We also performed additional tests with the goal of testing the robustness of our solutions with respect to decreased demand levels. In the first test, we removed the top three (post-attack) highest resistance arcs, while keeping all other resistances unchanged, while allowing the controller to reduce total demand by up to 10% with the objective of minimizing the maximum congestion. This computation can be formulated as a linear program; the resulting minimum congestion value is shown in the column labeled 'I-10%'. Note that to some degree this test also addresses the comparison with the $N - k$ model. Similarly, but now using *all* resistance values as computed by the nonlinear model, and without removing any arcs, we allowed the controller to reduce total demand by up to 10%, again with the objective of minimizing the maximum congestion. The column labeled 'C-10%' shows the resulting congestion value.

Table 13: **Comparison between the two optimization models**

$\boldsymbol{\sigma}$	**MIP**		**NONLINEAR**				
	Cong	**Attack**	**Cong**	**Top 6 Arcs**	**Impact**	**I-10%**	**C-10%**
1.0	1.44088	29,32,45	2.14967	29(7.79), 27(7.20), 41(7.03), 67(7.02), 54(6.72), 79(5.71)	1.71758	1.33454	1.67145
1.2	1.43132	27,29,41	1.78687	29(8.28), 27(7.72), 41(7.32), 67(7.19), 54(6.92), 79(5.78)	1.43132	1.11211	1.38642
1.4	1.22685	27,29,41	1.55634	29(8.31), 27(7.74), 41(7.53), 67(7.48), 54(7.18), 79(6.15)	1.22685	0.95324	1.21329
1.6	1.07349	27,29,41	1.35995	29(8.18), 27(7.58), 41(7.53), 67(7.58), 54(7.22), 79(6.25)	1.07349	0.83409	1.05458
1.8	0.692489	18,57,60	1.20271	29(8.43), 27(7.90), 41(7.53), 67(7.48), 54(7.18), 79(6.12)	0.95421	0.74141	0.93595
2.0	0.68630	20,89,45	1.07733	29(7.87), 27(7.29), 41(7.04), 67(7.01), 54(6.70), 79(5.63)	0.85889	0.66727	0.83878

Comments. As before, we see that the solutions to the nonlinear model tend to concentrate the attack on a relatively small number of lines, while at the same time investing small portions of the attack budget on other lines. This helps highlight the significant overlap between the results from the two models. Note that in the cases for $\sigma = 1.2, 1.4, 1.6$ the set of attacked lines show high correlation. Moreover, the two models are consistent: the severity of the attack as measured by the maximum congestion levels (the 'Cong' parameters), for both models, decrease as the scale increases (as one should expect).

The last three columns of the table address our second set of questions – they appear to show that the solution computed by the nonlinear model is robust; even as the controller reduces total demand, the congestion level is proportionally reduced. Finally, note that the congestion values in

the 'Impact' column are significantly smaller than the corresponding values in the 'Cong' column; similarly, the 'C-10%' values are higher than the 'I-10%' values – thus, the low x_{ij} arcs in the nonlinear attack do play a significant role.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications.* Prentice Hall, NJ (1993).

[2] G. Andersson, *Modelling and Analysis of Electric Power Systems.* Lecture 227-0526-00, Power Systems Laboratory, ETH Zürich, March 2004. Download from http://www.eeh.ee.ethz.ch/uploads/tx_ethstudies/modelling_hs08_script_02.pdf.

[3] R. Alvarez, Interdicting Electric Power Grids, Masters' Thesis, U.S. Naval Postgraduate School, 2004.

[4] J. Arroyo and F. Galiana, On the Solution of the Bilevel Programming Formulation of the Terrorist Threat Problem, *IEEE Trans. Power Systems*, Vol. 20 (2005), 789–797.

[B62] J.F. Benders, Partitioning procedures for solving mixed variables programming problems, *Numerische Mathematik* **4** (1962) 238-252.

[5] H. Y. Benson, D. F. Shanno and R. J. Vanderbei, Interior-point methods for nonconvex nonlinear programming: jamming and comparative numerical testing, *Math. Programming* **99**, 35 – 38 (2004).

[6] Algorithmic implications of the Graph Minors project (with M. Langston), in *Handbook of Operations Research* (Ball, Magnanti, Monma, Nemhauser, eds.), North-Holland (1995).

[7] D. Bienstock and S. Mattia, Using mixed-integer programming to solve power grid blackout problems , *Discrete Optimization* **4** (2007), 115–141.

[8] V.M. Bier, E.R. Gratz, N.J. Haphuriwat, W. Magua, K.R. Wierzbickiby, Methodology for identifying near-optimal interdiction strategies for a power transmission system, Reliability Engineering and System Safety **92** (2007), 1155–1161.

[9] S. Boyd, Convex Optimization of Graph Laplacian Eigenvalues, *Proc. International Congress of Mathematicians* **3** (2006), 1311–1319.

[10] D. Braess, Über ein Paradox der Verkerhsplannung, Unternehmenstorchung Vol. 12 (1968) 258–268.

[11] B.A. Carreras, V.E. Lynch, I. Dobson, D.E. Newman, Critical points and transitions in an electric power transmission model for cascading failure blackouts, Chaos, vol. 12, no. 4, 2002, 985-994.

[12] B.A. Carreras, V.E. Lynch, D.E. Newman, I. Dobson, Blackout mitigation assessment in power transmission systems, 36th Hawaii International Conference on System Sciences, Hawaii, 2003.

[13] B.A. Carreras, V.E. Lynch, I. Dobson, D.E. Newman, Complex dynamics of blackouts in power transmission systems, Chaos, vol. 14, no. 3, September 2004, 643-652.

[14] B.A. Carreras, D.E. Newman, I. Dobson, A.B. Poole, Evidence for self organized criticality in electric power system blackouts, IEEE Transactions on Circuits and Systems I, vol. 51, no. 9, Sept. 2004, 1733- 1740.

[15] ILOG CPLEX 11.0. ILOG, Inc., Incline Village, NV.

[16] S.T. DeNegre and T.K Ralphs, A Branch-and-cut Algorithm for Integer Bilevel Linear Programs, COR@L Technical Report, Lehigh University (2008).

[17] R. Fletcher, N. I. M. Gould, S. Leyffer, Ph. L. Toint, and A. Wächter, Global convergence of trust-region SQP-filter algorithms for general nonlinear programming, *SIAM J. Optimization* **13**, 635–659 (2002).

[18] The IEEE reliability test system–1996, IEEE Trans. Power Syst., vol. 14 (1999) 1010 - 1020.

[19] U. Janjarassuk and J. T. Linderoth, Reformulation and Sampling to Solve a Stochastic Network Interdiction Problem, to appear, *Networks* (2008).

[20] C. Lim and J.C. Smith, Algorithms for Discrete and Continuous Multicommodity Flow Network Interdiction Problems, *IIE Transactions* **39**, 15-26, 2007.

[21] B. Mohar, The Laplacian spectrum of graphs, in: Y. Alavi, G. Chartrand, O. Oellermann, A. Schwenk (Eds.), *Graph Theory, Combinatorics, and Applications*, London Math. Soc. Lecture Notes, Wiley-Interscience, 871-898 (1991).

[22] A. Pinar, J. Meza, V. Donde, and B.C. Lesieutre, Optimization Strategies for the Vulnerability Analysis of the Power Grid, submitted to *SIAM Journal on Optimization* (2007).

[23] A. Pinar, A. Reichert, and B.C. Lesieutre, Computing criticality of lines in power systems, in IEEE Int. Symp. Circuits and Systems (ISCAS 2007), New Orleans, LA, 2007, 65 – 68.

[24] N. Robertson and P. D. Seymour, Graphs minors. III. Planar tree-width, *J. Combinatorial Theory, Ser. B.* **36** (1984), 49 - 64.

[25] J. Salmeron, K. Wood and R. Baldick, Analysis of Electric Grid Security Under Terrorist Threat, *IEEE Trans. Power Systems* **19** (2004), 905–912.

[26] Vanderbei, R. 1997. LOQO User's manual, *Statistics and Operations Research* Technical report No SOR-97-08, Princeton University.

[27] *Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations*, U.S.-Canada Power System Outage Task Force, April 5, 2004. Download from https://reports.energy.gov.

[28] A. Verma, "Power grid security analysis: an optimization approach," Ph.D. thesis, Columbia University (2009). Download from http://www.columbia.edu/ dano/theses/verma.pdf.

[29] A. Wächter and L. T. Biegler, On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming, *Mathematical Programming* **106** (2006), 25 – 57.

A Appendix - Choosing M

Lemma A.1 *In formulation* $\mathbf{A}_{\mathcal{C}}(\mathcal{A})$*, a valid choice for* $\mathbf{M}$ *is*

$$\mathbf{M} = \max_{(i,j)\in E} \left\{ \frac{1}{\sqrt{x_{ij}}\, u_{ij}} \right\}. \tag{56}$$

Proof. Given an attack $\mathcal{A}$, consider a connected component K of $\mathcal{N} - \mathcal{A}$. For any arc (i, j) with both ends in K, $\omega_{ij}^{+} + \omega_{ij}^{-} = 0$ by (33). Hence we can rewrite constraints (26)-(27) over all arcs with both ends in K as follows:

$$N_K^T \alpha_K - X_K \beta_K = p_K - q_K, \tag{57}$$

$$N_K \beta_K = 0. \tag{58}$$

Here, N_K is the node arc incidence matrix of K, $\alpha_K, \beta_K, p_K, q_K$ are the restrictions of α, β, p, q to K, and X_K is the diagonal matrix $diag\{x_{ij} : (i,j) \in K\}$. From this system we obtain

$$N_K X_K^{-1} N_K \alpha_K \;=\; N_K X_K^{-1}(p_K - q_K). \tag{59}$$

The matrix $N_K X_K^{-1} N_K$ has one-dimensional null space and thus we have one degree of freedom in choosing α_K. Thus, to solve (59), we can remove from N_K an arbitrary row, obtaining $\tilde{N}_K$, and remove the same row from α_K, obtaining $\tilde{\alpha}_K$. Thus, (59) is equivalent to:

$$\tilde{N}_K X_K^{-1} \tilde{N}_K \tilde{\alpha}_K \;=\; \tilde{N}_K X_K^{-1}(p_K - q_K), \tag{60}$$

The matrix $\tilde{N}_K X_K^{-1} \tilde{N}_K$ and thus (60) has a unique solution (given $p_K - q_K$); we complete this to a solution to (59) by setting to zero the entry of α_K that was removed. Moreover,

$$X_K^{-1/2} N_K^T \alpha_K \;=\; X_K^{-1/2} \tilde{N}_K^T \tilde{\alpha}_K \;=\; X_K^{-1/2} \tilde{N}_K^T (\tilde{N}_K X_K^{-1} \tilde{N}_K^T)^{-1} \tilde{N}_K X_K^{-1}(p_K - q_K). \tag{61}$$

The matrix

$$H \;:=\; X_K^{-1/2} \tilde{N}_K^T \; (\tilde{N}_K X_K^{-1} \tilde{N}_K^T)^{-1} \; \tilde{N}_K X_K^{-1/2}$$

is symmetric and idempotent, e.g. $HH^T = I$. Thus, from (61) we get

$$\|X_K^{-1/2} N_K^T \alpha_K\|_2 \;\leq\; \|H\|_2 \, \|X_K^{-1/2}(p_K - q_K)\|_2 \;\leq\; \|X_K^{-1/2}(p_K - q_K)\|_2, \tag{62}$$

where the last inequality follows from the idempotent attribute. Because of constraints (28), (32) and (33), we can see that the square of the right-hand side of (62) is upper-bounded by the value of the convex maximization problem,

$$\max \quad \sum_{(i,j)\in E} x_{ij}^{-1}(p_{ij} - q_{ij})^2 \tag{63}$$

$$\text{s.t.} \quad \sum_{(i,j)\in E} u_{ij}(p_{ij} + q_{ij}) \leq 1 \tag{64}$$

$$p_{ij} \geq 0, \; q_{ij} \geq 0, \tag{65}$$

which equals $\max_{(i,j)\in E} \left\{ \frac{1}{x_{ij} u_{ij}^2} \right\}$. ■