Submitted to *Operations Research*
manuscript (Please, provide the manuscript number!)

Authors are encouraged to submit new papers to INFORMS journals by means of a style file template, which includes the journal title. However, use of a template does not certify that the paper has been accepted for publication in the named journal. INFORMS journal templates are for the exclusive purpose of submitting to an INFORMS journal and should not be used to distribute the papers in print or online or to submit the papers to another publication.

A Decomposition Algorithm with Fast Identification of Critical Contingencies for Large-Scale Security-Constrained AC-OPF

Frank E. Curtis
Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA, USA, frank.e.curtis@lehigh.edu

Daniel K. Molzahn
School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA, molzahn@gatech.edu

Shenyinying Tu, Andreas Wächter
Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, USA, shenyinyingtu2021@u.northwestern.edu, waechter@iems.northwestern.edu

Ermin Wei
Department of Electrical and Computer Engineering, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, USA, ermin.wei@northwestern.edu

Elizabeth Wong
Department of Mathematics, University of California, San Diego, La Jolla, CA, USA, elwong@ucsd.edu

A decomposition algorithm for solving large-scale security-constrained AC optimal power flow problems is presented. The formulation considered is the one used in the ARPA-E Grid Optimization (GO) Competition, Challenge 1, held from November 2018 through October 2019. Algorithmic strategies are proposed for contingency selection, fast contingency evaluation, handling complementarity constraints, avoiding issues related to degeneracy, and exploiting parallelism. The results of numerical experiments are provided to demonstrate the effectiveness of the proposed techniques as compared to alternative strategies.

Key words: nonlinear optimization; network optimization; security-constrained AC optimal power flow; complementarity constraints; decomposition methods

1. Introduction

This paper presents algorithmic methodologies for solving large-scale security-constrained AC optimal power flow (SC-AC-OPF) problems such as those adhering to the formulation considered for the ARPA-E Grid Optimization (GO) Competition, Challenge 1; see Aravena et al. (2022). The algorithm that we propose uses a derivative-based local-search interior-point optimization algorithm as its core, and confronts large problem sizes by iteratively enforcing the constraints associated with contingencies that the algorithm identifies to be important. Our algorithm also applies several customized contingency screening and parallel processing techniques to identify quickly what seem to be the most important contingencies. Additionally, our algorithm utilizes tailored heuristics designed to manage the complementarity conditions and avoids certain degeneracies by modifying the original problem formulation.

For the GO Competition, Challenge 1, our algorithm successfully solved SC-AC-OPF problems for 17 synthetic networks (each with 20 scenarios) and 3 actual industrial networks (each with 4 scenarios) for a total of 352 SC-AC-OPF problems. The network sizes for these systems ranged from 500 to 30,000 buses, i.e., transmission network nodes (ARPA-E 2019c). For an illustration of an instance involved in the competition, see Figure 1 below. The operating cost and constraint violation penalties achieved by our algorithm resulted in a second-place finish overall among the 27 teams in the GO Competition, Challenge 1. Our algorithm obtained the best scores for 20% of the problems, the best or second-best scores for 53% of the problems, and top-ten scores for almost all ($> 99\%$) of the problems. The results of this competition, by our algorithm and others involved in the competition, indicate that nonlinear optimization techniques such as those used in the algorithm described in this paper are effective at solving industrially relevant, large-scale SC-AC-OPF problems. Hence, our contributions in this article pertain to the details of our algorithm, which has already proved to be effective for the GO Competition, Challenge 1, and we contend would be effective for solving other similar optimization problem formulations.

This paper is organized as follows. Section 2 introduces the problem formulation and discusses several key modeling decisions. Section 3 describes our proposed solution algorithm, including an overall decomposition strategy along with our contingency selection and evaluation procedures. Section 4 provides the results of numerical experiments that isolate the effectiveness of various components of our overall algorithm. Section 5 provides concluding remarks.

2. Problem Statement

We present the details of our algorithmic methodologies in the context of solving the problem formulation used for the GO Competition, Challenge 1. This problem seeks the minimum-cost operating point, in terms of both generation cost and penalized soft constraint violations, that satisfies constraints from both a base case with all components in service and multiple contingency cases that model the failures of transmission lines, transformers, and generators. A complete description of the problem statement can be found on the GO Competition website (ARPA-E 2019c), as well as the introductory paper Aravena et al. (2022) included in this journal issue. For ease of reference, the elements of the model to which we refer are restated below, including the line flow definitions

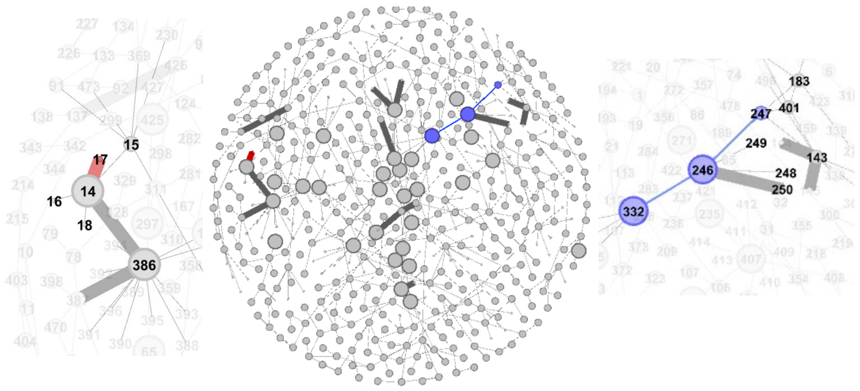

Figure 1 Visualization of Network 1 with 500 nodes. The thick lines represent parallel lines or transformers and the size of each node indicates the voltage of the bus. Bus 17 (see the red edge, with a close-up view shown on the left) is associated with the generator contingency that has the highest penalty value. Transformer 247 to 246 and line 246 to 332 (both in blue, with a close-up view shown on the right) are the transformer and line contingencies associated with the highest penalty values. All three of the indicated contingencies are located near a parallel line and close to a high-voltage bus.

(2.1), transformer flow definitions (2.2), bus power balance equations (2.3), line current ratings constraints (2.4), transformer power ratings constraints (2.5), and generator active (2.6) and reactive (2.7) power contingency response (complementarity) constraints, respectively. Equations (2.6) and (2.7) link the base case variables and constraints with the variables and constraints associated with each contingency.

$$\begin{aligned} p_e^o &= G_e v_{i_e^o}^2 - (G_e \cos(\theta_{i_e^o} - \theta_{i_e^d}) + B_e \sin(\theta_{i_e^o} - \theta_{i_e^d})) v_{i_e^o} v_{i_e^d} \\ q_e^o &= -(B_e + B_e^{CH}/2) v_{i_e^o}^2 + (B_e \cos(\theta_{i_e^o} - \theta_{i_e^d}) - G_e \sin(\theta_{i_e^o} - \theta_{i_e^d})) v_{i_e^o} v_{i_e^d}, \end{aligned} \tag{2.1}$$

$$\begin{aligned} p_f^o &= (G_f/\tau_f^2 + G_f^M) v_{i_f^o}^2 - (G_f/\tau_f \cos(\theta_{i_f^o} - \theta_{i_f^d} - \theta_f) + B_f/\tau_f \sin(\theta_{i_f^o} - \theta_{i_f^d} - \theta_f)) v_{i_f^o} v_{i_f^d} \\ q_f^o &= -(B_f/\tau_f^2 + B_f^M) v_{i_f^o}^2 + (B_f/\tau_f \cos(\theta_{i_f^o} - \theta_{i_f^d} - \theta_f) - G_f/\tau_f \sin(\theta_{i_f^o} - \theta_{i_f^d} - \theta_f)) v_{i_f^o} v_{i_f^d}, \end{aligned} \tag{2.2}$$

$$\begin{aligned} \sum_{g \in \mathcal{G}_i} p_g - p_i^L - G_i^{FS} v_i^2 - \sum_{e \in \mathcal{E}_i^o} p_e^o - \sum_{e \in \mathcal{E}_i^d} p_e^d - \sum_{f \in \mathcal{F}_i^o} p_f^o - \sum_{f \in \mathcal{F}_i^d} p_f^d = \sigma_i^{P+} - \sigma_i^{P-} \\ \sum_{g \in \mathcal{G}_i} q_g - q_i^L + (B_i^{FS} + b_i^{CS}) v_i^2 - \sum_{e \in \mathcal{E}_i^o} q_e^o - \sum_{e \in \mathcal{E}_i^d} q_e^d - \sum_{f \in \mathcal{F}_i^o} q_f^o - \sum_{f \in \mathcal{F}_i^d} q_f^d = \sigma_i^{Q+} - \sigma_i^{Q-}. \end{aligned} \tag{2.3}$$

$$\begin{aligned} (p_e^o)^2 + (q_e^o)^2 \le (\overline{R}_e v_{i_e^o} + \sigma_e^S)^2 \\ (p_e^d)^2 + (q_e^d)^2 \le (\overline{R}_e v_{i_e^d} + \sigma_e^S)^2. \end{aligned} \tag{2.4}$$

$$\begin{aligned} (p_f^o)^2 + (q_f^o)^2 \le (\overline{s}_f + \sigma_f^S)^2 \\ (p_e^d)^2 + (q_e^d)^2 \le (\overline{s}_f + \sigma_f^S)^2. \end{aligned} \tag{2.5}$$

$$\begin{aligned} 0 \le (p_{gk} - \underline{p}_g) \perp (p_g + \alpha_g \Delta_k - p_{gk}) \le 0 \\ \text{or} \quad 0 \ge (p_{gk} - \overline{p}_g) \perp (p_g + \alpha_g \Delta_k - p_{gk}) \ge 0. \end{aligned} \tag{2.6}$$

$$\begin{aligned} & 0 \leq (q_{gk} - \underline{q}_g) \perp (v_{i_g} - v_{i_g k}) \leq 0 \\ \text{or} \quad & 0 \geq (q_{gk} - \overline{q}_g) \perp (v_{i_g} - v_{i_g k}) \geq 0. \end{aligned} \tag{2.7}$$

2.1. High-level model

At a high level, the problem can be expressed in the manner shown in (2.8). As defined in more detail below, let u_0 be the control variables in the base case, y_0 be the state variables in the base case, and $(\varsigma_0, \varsigma_0^+, \varsigma_0^-)$ be the slack variables in the base case. Similarly, for each contingency $k \in \mathcal{K}$, let u_k be the control variables, y_k be the state variables, and $(\varsigma_k, \varsigma_k^+, \varsigma_k^-)$ be the slack variables.

$$\min_{\substack{(u_0, y_0, \varsigma_0, \varsigma_0^+, \varsigma_0^-) \\ \{(u_k, y_k, \varsigma_k, \varsigma_k^+, \varsigma_k^-)\}_{k \in K}}} \; f_0(u_0, y_0) + \psi_0(\varsigma_0, \varsigma_0^+, \varsigma_0^-) + \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} \psi_k(\varsigma_k, \varsigma_k^+, \varsigma_k^-)$$
$$\text{s.t.} \begin{cases} a_0(u_0, y_0) = 0 & \\ b_0(u_0, y_0) = \varsigma_0^+ - \varsigma_0^- & \\ c_0(u_0, y_0) \leq \varsigma_0 & \\ \underline{u}_0 \leq u_0 \leq \overline{u}_0 & \\ \underline{y}_0 \leq y_0 \leq \overline{y}_0 & \\ (\varsigma_0, \varsigma_0^+, \varsigma_0^-) \geq 0 & \\ a_k(u_k, y_k) = 0 & \text{for all } k \in K \\ b_k(u_k, y_k) = \varsigma_k^+ - \varsigma_k^- & \text{for all } k \in K \\ c_k(u_k, y_k) \leq \varsigma_k & \text{for all } k \in K \\ \underline{u}_k \leq u_k \leq \overline{u}_k & \text{for all } k \in K \\ \underline{y}_k \leq y_k \leq \overline{y}_k & \text{for all } k \in K \\ (\varsigma_k, \varsigma_k^+, \varsigma_k^-) \geq 0 & \text{for all } k \in K \\ 0 \leq \tau_k^L(u_0, y_0, u_k, y_k) \perp \tau_k^R(u_k, y_k) \geq 0 & \text{for all } k \in K. \end{cases} \tag{2.8}$$

At a more detailed level, the base case control variables are the active and reactive power produced by the generators and the controllable shunt susceptances at the buses, whereas the only control variables in the contingencies are the controllable shunt susceptances at the buses. All other variables are state variables except those that are specified as slack variables that are penalized in the objective function. The objective consists of a base case objective f_0—representing the cost of active power generation—along with penalty terms for the base case (ψ_0) and contingency slack variables ($\{\psi_k\}_{k \in \mathcal{K}}$). The base case objective is a convex piecewise linear cost of generation, the definition of which involves some auxiliary variables; see the official problem statement.

The constraints come in a few different types. Besides simple bounds on the control, state, and penalty variables, they are as follows. The constraint functions a_0 and $\{a_k\}_{k\in\mathcal{K}}$ capture the line and transformer flow definitions (2.1)–(2.2), where a_0 fixes a single phase angle in the base case (to zero) in order to set a reference angle. The constraint functions b_0 and $\{b_k\}_{k\in\mathcal{K}}$ capture the bus power balance (2.3). The constraint functions c_0 and $\{c_k\}_{k\in\mathcal{K}}$ capture the line current and transformer power ratings constraints (2.4)–(2.5). Finally, the constraint functions $\{\tau_k^L\}_{k\in\mathcal{K}}$ and $\{\tau_k^R\}_{k\in\mathcal{K}}$ capture the complementarity constraints (2.6)–(2.7).

It is worth emphasizing that only the complementarity constraints involve a combination of base case and contingency variables, and *no* single constraint involves variables from more than one contingency at a time. This makes the problem nearly separable, which is a property that needs to be exploited in a solution approach for it to be efficient. We also note that for the specific SC-AC-OPF problems that we solve in our experiments, many of the upper and lower bounds on the control and state variables are consistent across all contingencies, but we write them all as contingency-dependent in (2.8) for the sake of generality when describing our algorithmic approach.

2.2. Modeling decisions

The aforementioned formulation allows some flexibility, and the specific modeling choices that are made can have a significant impact on the performance of a solution algorithm. In this subsection, we comment on the modeling choices that we believe are most consequential in our approach.

First, as is common when solving many types of optimization problems, one has the option of eliminating a potentially large number of constraints. For example, in the problem at hand, one could eliminate the line and transformer flow definition constraints (2.1)–(2.2) and simply plug the expressions for the variables being defined by these constraints into the other constraints. This would have the effect of eliminating a number of variables and number of constraints on the order of the number of lines plus the number of transformers, multiplied by the number of contingencies. However, the downside of such elimination is a large increase in the density of the constraint Jacobians. Hence, rather than perform this elimination explicitly, we formulate the problem with all of these variables and constraints and allow the linear system solver in the nonlinear optimization method to exploit the sparsity of the resulting linear systems.

Second, the line current and transformer power ratings constraints (2.4)–(2.5) are essentially upper bounds on the norms of two-dimensional vectors involving active and reactive power at an origin or destination bus. The norm is a nonsmooth function, which might be problematic for an interior-point method whose theoretical guarantees depend on smoothness of the problem functions. Hence, the constraints that we state in (2.4)–(2.5) are squared versions of the constraints stated in the official problem statement so that c_0 and $\{c_k\}_{k\in\mathcal{K}}$ are smooth.

Third, the generator active and reactive power contingency responses can be modeled in various ways, including a *logical formulation* in which the responses are captured by complementarity constraints, a *projection formulation*, and a *mixed integer programming* formulation. As can be seen in (2.6)–(2.7), we choose to follow the logical formulation involving complementarity constraints.

Finally, we note that, as is typical for steady-state transmission system analyses, both the GO competition and our algorithm consider a balanced single-phase equivalent representation of the power system. Extensions to unbalanced three-phase models would be most relevant in the context of optimizing distribution systems. While many ideas presented in this paper (and the competition more broadly) could be applicable to distribution systems, there are a number of modeling aspects that would need to be considered for such applications that are beyond the scope of this paper.

3. Algorithm

A high-level view of our algorithm is presented in Figure 2. Following the terminology used for the GO Competition, Challenge 1, we refer to our main algorithm for solving (2.8) as being contained in *Code 1*, which determines the base case solution, i.e., values of the variables associated with nominal operation, and *Code 2*, which produces the contingency response solutions for every contingency scenario. More precisely, *Code 1* observes the combined base case and contingencies problem in order to optimize the base case solution subject to the contingencies, but is only asked to provide a base case solution. Afterwards, *Code 2* is run to produce contingency response solutions using the base case solution that is determined by the run of *Code 1*. For the remainder of the paper, unless stated otherwise, we focus on the algorithm contained in *Code 1*.

After a preprocessing phase (about which we leave discussion until Section 3.6), a base case solution is obtained by solving (2.8) with *no* contingencies. Given this base case solution, a scheme is employed to determine an initial ranking of contingencies according to their estimated importance (see Section 3.1), after which a parallel process is initiated to start providing better estimates of each contingency's importance (see Section 3.2). This process involves a specialized procedure for handling the complementarity constraints (see Section 3.3) and one for fast (approximate) contingency evaluation (see Section 3.4). At this point, a few contingencies are selected and the algorithm enters an iterative process during which a *master problem* is solved—i.e., a problem of the same form as (2.8), but with only those contingencies that have been selected as important—and contingencies are continually evaluated and possibly selected for inclusion in the master problem in subsequent iterations. Many of the procedures in this loop are performed in a coordinated fashion in parallel (see Section 3.8 about parallel processing in *Code 1* and *Code 2*). We also mention in this section the techniques that we employed for avoiding the selection of *dominated* contingencies (see Section 3.5) as well as a summary of our entire algorithm (see Section 3.7).

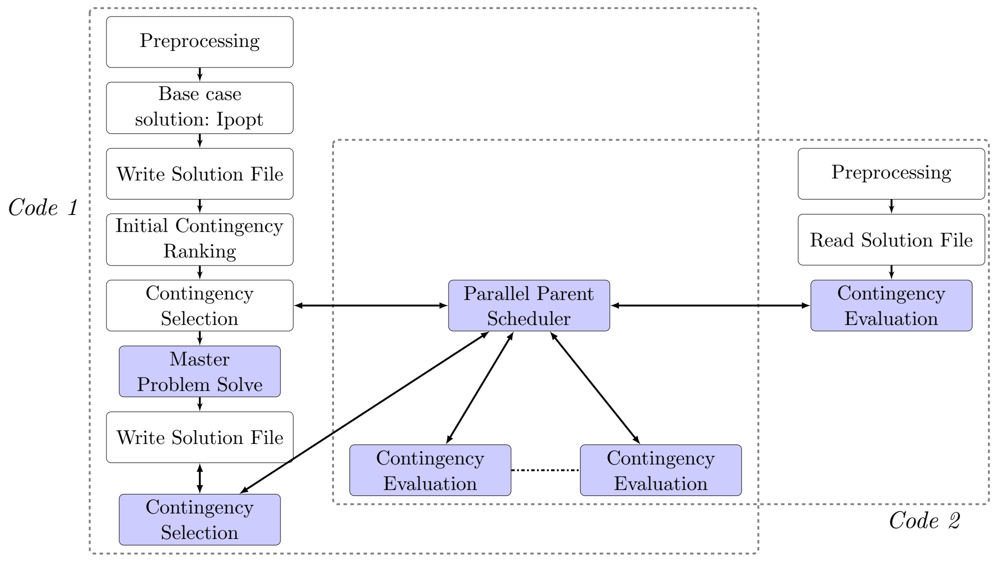

Figure 2 High-level view of algorithm.

Overall, our algorithmic strategy is to identify quickly what are the *important* contingencies to include in the master problem in order to obtain a solution that is as close as possible to the solution that would be obtained if all contingencies were to be included. By the importance of a contingency, we mean its effect on the optimal value of the overall problem. Generally speaking, if an important contingency is not included in the master problem, then this means that the solution obtained for the master problem would represent a poor solution for the overall problem, whereas if a less important contingency is ignored, then the solution would not be much poorer. A complicating factor is that, in some cases, two contingencies could be individually important, although as soon as one is included in the master problem, the other becomes less important. For example, this may be the case for contingencies corresponding to parallel lines in the transmission network. Our overall strategy attempts to determine as few contingencies as possible such that, once these are included in the master problem, the remaining contingencies are unimportant. We note that similar algorithmic strategies are used in previous power systems literature; see, e.g., (Ejebe and Wollenberg 1979, Stott et al. 1987, Capitanescu 2016).

All optimization problems in the algorithm—including the instances of the master problem and penalty minimization problems for contingency evaluation—are solved with the interior-point method implemented in the Ipopt software (Wächter and Biegler 2006). Interior-point methods were first employed to solve power system problems in the early 1990s for the purpose of state estimation (Clements et al. 1991). Subsequent work over the next three decades applied interior-point methods to solve various OPF problems; see, e.g., (Wu et al. 1994, Wang et al. 2007, Capitanescu et al. 2011, Capitanescu and Wehenkel 2013). The algorithm implemented in Ipopt offers theoretical convergence guarantees under assumptions that are common for local-search algorithms for solving nonconvex constrained continuous optimization problems; see (Wächter and Biegler 2006). Since our problems of interest are nonconvex and the Ipopt algorithm is a derivative-based local-search algorithm, convergence to a (global or local) minimizer is not guaranteed; rather, the theoretical guarantees only pertain to convergence to first-order stationarity. That said, given the time constraints for solving large-scale SC-AC-OPF problems problems in real-world scenarios, even state-of-the-art algorithms with global optimization guarantees are generally inapplicable.

3.1. Initial Ranking of Contingencies

For a given base case solution, evaluation of the penalty associated with contingency $k \in \mathcal{K}$ is relatively expensive computationally, since it involves solving a nonlinear optimization problem that contains all of contingency k's variables; see Section 3.2. Hence, after a base case solution is obtained, it may be detrimental to wait until all contingencies have been evaluated in this manner before solving a new master problem involving a few contingencies to obtain an improved base case solution. For this reason, we employ a contingency ranking heuristic in order to identify potentially important contingencies more quickly. These heuristics are deployed immediately after we obtain the first base case solution. The contingencies that are identified as potentially being the most important are evaluated first using the contingency evaluation and selection strategies that we describe in subsequent subsections. Through this process, which exploits parallel computations, not all contingencies are evaluated before a new master problem is solved.

For several decades, contingency ranking has been an extensively studied subject in the power systems literature (see, e.g,. (Ejebe and Wollenberg 1979) and (Stott et al. 1987) as well as the more recent survey in (Wu et al. 2017)). While we were conceptually inspired by this literature, differences in the problem formulation (e.g., constraint violation penalties), recent availability of many SC-AC-OPF test cases, and capabilities of the computing platform (e.g., extensive parallel computing resources) motivated us to create a new contingency ranking heuristic. We now describe how our heuristic has been tailored for the problem formulation, dataset characteristics, and computing time requirements of the GO Competition, Challenge 1, which provides a guide for how it may be tailored for other related real-world settings as well.

Our contingency ranking heuristic uses features of the transmission network topology as well as the values obtained from the initial base case solution. To develop this heuristic, we first identified a collection of potential features, such as topology information in the neighborhood of a contingency and generation or transmission flow loss introduced by the contingency. We then trained a ridge regression model, i.e., a least-squares loss function with an ℓ_2-norm regularization term, using a linear combination of these features to predict the penalty associated with each contingency—whether it be defined by a generator, line, or transformer failure—in any scenario using the datasets provided by the competition. The dataset was divided into training and testing groups. The model parameters were trained on the training dataset and tested on the rest of the networks including both new scenarios for the networks in the training dataset as well as new networks that were not included in the training set. We repeated the cross validation process 10 times and took averages across these repetitions to get the final model parameters. During the execution of our algorithm, we use the obtained model to generate penalty values that predict the importance of each contingency. Sorting the contingencies in descending order of their penalty values provides our initial contingency ranking. We next describe the features we use to generate this ranking.

Studying the testing datasets offered by the competition organizers suggests that there is often a direct correlation between the importance of a contingency and the degree (i.e., number of neighboring buses) of the buses near the contingency. We consider the voltage ratings of either the bus(es) associated with a contingency (i.e., the bus where the generator contingency occurs or the end points of either line or transformer contingencies) or the *neighboring* buses, with the idea that a generator, line, or transformer contingency may be more important if one of these ratings is sufficiently large. The power flow losses introduced by the contingencies are also identified as important. Specifically, after considering many combinations of potential features, for each contingency $k \in \mathcal{K}$, the best predicted importance is given by a linear combination of the following features: $\{t_k^g, t_k^l, t_k^t, l_k^p, l_k^s, l_k^c, v_k^d, d_k^o, d_k^d, \pi_k\}$. The first three features, namely, $\{t_k^g, t_k^t, t_k^l\}$ are binary variables indicating the type of contingency: for a generator contingency, $t_k^g = 1$ and $t_k^t = t_k^l = 0$; for a line contingency, $t_k^l = 1$ and $t_k^g = t_k^t = 0$; and for a transformer contingency, $t_k^t = 1$ and $t_k^g = t_k^l = 0$. The remainder of the features have different interpretations for generator versus line or transformer contingencies. Intuitively, the features related to l are used to denote lost power (i.e., the base case power generated by or flowing through the component that is failed in the contingency), v for voltage, d for degree of the corresponding buses and π reflects a local topological property, as described next.

For contingency $k \in \mathcal{K}$ associated with generator g, we define three features related to the loss in power generation induced by the generator failure relative to the base case solution: the lost active power generation $l_k^p = p_g$, the lost apparent power generation $l_k^s = \sqrt{p_g^2 + q_g^2}$, and the lost apparent power relative to the generator capacity $l_k^c = \frac{\sqrt{p_g^2+q_g^2}}{\sqrt{\overline{p}_g^2+\overline{q}_g^2}}$. We also define the following two features related to the neighboring buses of the generator: v_k^d is the highest voltage rating prior to per-unit normalization (given by the field BASKV in the dataset) of any neighboring buses and d_k^o is the degree of the bus where the generator g is located. The rest of the features are set to zero (i.e., $d_k^t = \pi_k = 0$) for all generator contingencies.

For contingency $k \in \mathcal{K}$ associated with line e or transformer f connecting origin bus o and destination bus d, we similarly define three features related to the lost flow relative to the base case solution. In particular, we define the lost active power flow as $l_k^p = \max\{p_e^o, p_e^d\}$ for line e (and similarly for transformer f), the lost apparent power as $l_k^s = \max\left\{\sqrt{(p_e^o)^2 + (q_e^o)^2}, \sqrt{(p_e^d)^2 + (q_e^d)^2}\right\}$ for line e (and similarly for transformer f), and the lost apparent power relative to the flow capacity as $l_k^c = \max\left\{\sqrt{\frac{(p_e^o)^2+(q_e^o)^2}{\overline{R}_e v_{i_e^o}}}, \sqrt{\frac{(p_e^d)^2+(q_e^d)^2}{\overline{R}_e v_{i_e^d}}}\right\}$ for line e and $l_k^c = \max\left\{\sqrt{\frac{(p_e^o)^2+(q_e^o)^2}{\bar{s}_f}}, \sqrt{\frac{(p_e^d)^2+(q_e^d)^2}{\bar{s}_f}}\right\}$ for transformer f. The following four features related to the network topology also enhance the quality of predicted penalty values: v_k^d, the voltage rating of bus d; d_k^o, the degree of bus o; d_k^d, the degree of bus d; and π_k, a weight set to 10 if the line/transformer has a parallel counterpart, i.e., there exists another line/transformer connecting the same pair of origin and destination buses, or set to 0 otherwise. The model's performance does not improve when we include additional features such as the degree of the origin and destination buses for the lines/transformers.

Among these features, the lost apparent power l_k^s has the largest coefficient. This observation agrees with our numerical studies in Section 4.1, where we show that this feature alone has very good predictive power. One interesting property of our model is that it is universal to all networks regardless of their sizes. Introducing network size as a feature does not improve the predictive power of the model. We suspect this is due to the effect of a contingency often being either local or well-modeled by the power loss features alone.

We remark in passing that the aforementioned model uses a simple linear combination of the features. Alternative approaches (e.g., multiplicatively combining the features, using additional features, etc.) generally led to similar or inferior empirical performance in our tests. Finally, we note that some contingencies are often identified as important (in terms of their typical penalty values) across many scenarios for the same system. For each scenario for a particular system, we recorded the top contingencies which incurred a large penalty value. We then took the union of all of these top contingencies to form a candidate list. The initial contingency ranking in a run of our algorithm combines this candidate list with the generator, line, and transformer ranking values described above to create an initial prioritized list of the contingencies.

3.2. Contingency Evaluation

Given a base case solution (u_0, y_0), a solution for contingency $k \in \mathcal{K}$ is obtained by solving the (continuous and nonlinear) penalty minimization problem

$$
\begin{aligned}
\min_{u_k, y_k, \varsigma_k, \varsigma_k^+, \varsigma_k^-} \quad & \psi_k(\varsigma_k, \varsigma_k^+, \varsigma_k^-) \\
\text{s.t.} \quad & \begin{cases}
a_k(u_k, y_k) = 0 \\
b_k(u_k, y_k) = \varsigma_k^+ - \varsigma_k^- \\
c_k(u_k, y_k) \leq \varsigma_k \\
\underline{u}_k \leq u_k \leq \overline{u}_k \\
\underline{y}_k \leq y_k \leq \overline{y}_k \\
(\varsigma_k, \varsigma_k^+, \varsigma_k^-) \geq 0 \\
0 \leq \tau_k^L(u_0, y_0, u_k, y_k) \perp \tau_k^R(u_k, y_k) \geq 0,
\end{cases}
\end{aligned}
\tag{3.1}
$$

where the variables and problem functions are defined as in (2.8). We note that one might consider warm-starting the solve for (3.1), e.g., using the base case solution values or a solution of the problem obtained in a previous iteration of the overall algorithm. However, we did not find warm-starting to work well in our experiments, which may be due to the well-known difficulty inherent in effectively warm-starting interior-point methods.

The complementarity constraints in (3.1) are not included as explicit complementarity constraints when the problem is solved in our algorithm. Rather, they are replaced by a set of (smooth) constraints that depend on an active-set prediction regarding which of the terms involved in the complementarity constraints are active and which are free. Overall, in our solution algorithm, (3.1) is solved approximately through a loop in which the complementarity predictions are updated iteratively. We describe this process in further detail in the next subsection.

3.3. Handling Complementarities

Each pair of complementarity constraints in (2.6) and (2.7) involves a variable, call it χ (representing p_{gk} in (2.6) and q_{gk} in (2.7)), and a linear expression, call it ρ (representing $p_g + \alpha_g \Delta_k - p_{gk}$ in (2.6) and $v_{i_g} - v_{i_g k}$ in (2.7)). Each constraint requires that at least one of the following holds:

- χ equals a lower bound $\underline{\chi}$ and $\rho \leq 0$,
- χ equals an upper bound $\overline{\chi}$ and $\rho \geq 0$, or
- $\chi \in [\underline{\chi}, \overline{\chi}]$ and $\rho = 0$.

Our strategy for handling complementarities may be referred to as an active set approach, wherein we iteratively (i) make a prediction about which of these conditions holds at an optimal solution, (ii) solve the resulting problem (that involves no complementarity constraints), then (iii) update our prediction based on multiplier values obtained from the solution obtained in step (ii).

Let us refer to the condition $\chi = \underline{\chi}$ as the *lower* segment, $\chi \in [\underline{\chi}, \overline{\chi}]$ as the *middle* segment, and $\chi = \overline{\chi}$ as the *upper* segment of a complementarity constraint. The corresponding restrictions on the linear expression are $\rho \leq 0$, $\rho = 0$, and $\rho \geq 0$, respectively. Our procedure for updating a prediction for a complementarity is based on the same strategy in all cases. For example, let us consider the case when the prediction in step (i) is that the lower segment is optimal. In step (ii), the problem (without complementarities) is solved with the constraints $\chi = \underline{\chi}$, $\rho - s = 0$, and $s \leq 0$, where s is a slack variable. Let $\lambda \geq 0$ be the Lagrange multiplier for $s \leq 0$. If $\lambda > 0$ and $-s/\lambda < 10^{-6}$, then the constraint $s \leq 0$ is deemed to be active and the prediction is altered to say that the middle segment is optimal; otherwise, the prediction is not changed. Using such a procedure, a prediction may also be altered from middle to lower, middle to upper, or upper to middle.

Our active-set estimates for the complementarity constraints are updated in this manner during contingency evaluation only. In other words, the active-set estimates are not updated during a master problem solve. We considered updating the predictions iteratively for the master problem as well, but this turned out to be too time-consuming due to the greater expense of solving each master problem, and might not be worthwhile anyway since the predictions are updated during the next contingency evaluations in any case. During contingency evaluation, the predictions are updated all-at-once in this manner until either a time limit is reached or the improvement in the optimal value of (3.1) is too small (or negative). (In terms of evaluating the objective value of (3.1), it is important to evaluate the objective value explicitly, rather than consider the objective value reported by Ipopt, since the latter is influenced by internal relaxations of the bound constraints.)

For the first time that a contingency is evaluated, the active-set predictions for the complementarity constraints are initialized differently depending on the type of complementarity and contingency. With respect to active power (i.e., (2.6)) in a generator contingency, a one-dimensional bisection search is performed over the perturbation variable Δ_k. For each generator, a new production level is computed in a way that satisfies the complementarity constraints (2.6). The search is terminated when the overall added active power generation equals 1.01 times the amount of active power lost by the generator contingency. The factor 1.01 is used as an estimate that we would incur a 1% increase in losses due to the rerouting of power flows. The active constraints resulting from this search are used to initialize the active-set predictions for each generator. In all other cases, the active-set predictions are initialized to the middle segment.

We also tried reformulating the complementary constraints as penalty terms in the objective function. Observe that each complementarity constraint in (2.6)–(2.7) can be written as $0 \leq \eta \perp \mu \geq 0$, which is equivalent to saying that $(\eta, \mu) \geq 0$ while $\eta\mu = 0$. An approach that has been explored in the literature is to include the bound constraints $(\eta, \mu) \geq 0$ and add a term of the form $-\beta(\eta\mu)$ to the objective function, where β is a positive penalty parameter (Hu and Ralph 2004). This reformulation did not work well in our experiments, which we attribute to the negative curvature that such a penalty term introduces into the objective function.

3.4. Fast Contingency Evaluation

Evaluating each contingency as described in the preceding subsections can be expensive computationally, especially due to the combinatorial nature of problem (3.1) from the presence of the complementarity constraints. In an attempt to mitigate the computational costs that would be incurred by performing a full evaluation of each contingency during each iteration of the complementarity update loop described in the previous section, we employ a *fast* contingency evaluation scheme that is able to quickly produce an upper bound on the optimal value of (3.1). This is done by solving a reduced contingency evaluation problem, then setting values of the remaining variables in a way that results in a feasible solution of problem (3.1), which gives such an upper bound.

Each reduced problem involves the power flow equations (2.1)–(2.3), but without the corresponding inequality constraints (i.e., physical bounds) and without the slack variables. After fixing the controllable shunt susceptances (where, for our purposes, we use their values from the base case solution) and explicitly choosing the active complementarity segments, one obtains a square system of nonlinear equations for contingency $k \in \mathcal{K}$, which involves the equations:

$$\begin{cases} a_k(u_k, y_k) = 0 \\ b_k(u_k, y_k) = 0 \\ \tau_k(u_0, y_0, u_k, y_k) = 0. \end{cases} \tag{3.2}$$

Here, τ_k is derived from either τ_k^L or τ_k^R depending on the active-set prediction. To solve (3.2), we use a Newton method; in particular, we call Ipopt to solve the system of equations.

After (3.2) is solved, we update the active-set predictions for the complementarities based on any violated bounds, then project the voltage and generation values according to their bounds, adjust the remaining state variables, and compute slack variables accordingly to get a feasible solution to the full contingency problem. One could stop there and employ the upper bound offered by this feasible solution. In our approach, however, we continue the fast evaluation in an iterative manner. In particular, using the updated active-set predictions for the complementarities, we repeat the aforementioned steps until the penalty value no longer decreases, a time limit is reached, or the penalty value falls below a threshold value that is tightened iteratively by the overall algorithm.

If, for a given contingency, the penalty value obtained after termination of this fast evaluation process is greater than the threshold value mentioned above, then we transition to the full evaluation of this contingency by solving (3.1) directly. Otherwise, the contingency is deemed not to be worth a full evaluation at this stage in the overall solution algorithm.

3.5. Avoiding Selection of Dominated Contingencies

Iteratively throughout our solution algorithm, we select a specified number (three, in our implementation) of additional contingencies to include in subsequent master problems. While our choice of contingencies is informed by their associated penalty values, simply selecting the contingencies with the largest penalties may result in redundant contingencies being added to the master problem. In other words, adding the constraints associated with one contingency may implicitly result in the (near) satisfaction of all constraints associated with another contingency (or contingencies). Hence, the latter contingencies are *dominated* by the former.

Contingencies consisting of failures of nearby generators, transmission lines, or transformers can often be (nearly) redundant. Some of these redundant contingencies can be identified easily. For instance, the failure of an identical parallel line or identical generator at the same bus is clearly redundant since the failure of one unit will have the same impact on the system as the failure of its counterpart. In our solution algorithm, we preprocess the set of contingencies to eliminate these trivially redundant contingencies before we start our algorithm.

Identifying other near-redundancy in contingencies is more challenging. However, since each contingency added to the master problem introduces a significant computational burden, it is worthwhile to make efforts to avoid inclusion of nearly redundant contingencies. To this end, (Capitanescu et al. 2007) introduced the concept of an *individually dominated contingency*, which attempts to identify a redundant contingency by observing its corresponding constraint violations.

Definition 3.1. (Capitanescu et al. 2007) **Individually Dominated Contingency.**
Contingency $k \in \mathcal{K}$ is ***individually dominated*** *by contingency $j \in K$ if*

$$\varsigma_j \geq \varsigma_k, \quad \varsigma_j^+ \geq \varsigma_k^+, \quad \varsigma_j^- \geq \varsigma_k^- \quad \textit{(component-wise)} \tag{3.3}$$

for the optimal penalty variables.

The approach in (Capitanescu et al. 2007) enforces the constraints associated with all contingencies that are not individually dominated. Our initial numerical experiments suggested that this approach added many nearly-redundant contingencies (i.e., with nonzero, but small penalty values) to the master problem. Due to the large number of constraints within each contingency, it was nearly impossible for one contingency to strictly dominate another. Therefore, we relax condition (3.3) by only considering the most violated constraint for each contingency, rather than all constraints, through the introduction of the concept of a *maximum violation dominated contingency.*

Definition 3.2. **Maximum Violation Dominated Contingency.**
Let $\bar{i}_k$ be the index of the constraint in contingency $k \in \mathcal{K}$ with the largest violation (i.e., the index of the constraint with the largest associated slack variable ς_k, ς_k^+, or ς_k^-), and let $\bar{\varsigma}_k$ be the corresponding violation. We say contingency k is ***maximum violation dominated*** *by contingency $j \in \mathcal{K}$ if*

$$\bar{i}_j = \bar{i}_k \quad \textit{and} \quad \bar{\varsigma}_j > \bar{\varsigma}_k. \tag{3.4}$$

During the selection of contingencies to include in subsequent master problems, we make sure to select a set of contingencies that are not maximum violation dominated.

3.6. Preprocessing and Handling Degeneracy

To achieve faster convergence for the interior-point method, we preprocess the input data to eliminate degeneracies. Some of these degeneracies result from parallel lines that have identical electrical parameters (i.e., G_e, B_e, and $\overline{R}_e$). Since they have the same terminal voltages, the power flows on these parallel lines are also identical. Hence, the line flow limits (2.4) for these lines are redundant and all but one can be eliminated. Flow limits for identical parallel transformers are similarly redundant. However, when dealing with the contingency associated with one of these lines, we have to make sure that at least one copy of the flow limit constraints is present when one of the lines is removed from the network.

Our preprocessing step also eliminates degeneracies associated with generators that are located at the same bus. In particular, we reduce the number of variables and variable bounds by aggregating the reactive power outputs of all generators at the same bus.

3.7. Overall Algorithm

Our overall solution algorithm consists of three major components, all occurring in concert in parallel: solving master problems, evaluating contingencies, and applying ranking schemes to identify important contingencies. The algorithm maintains a priority list of contingencies that determines in which order they should be evaluated and which top three should be included in subsequent master problems. This list is updated at various steps of our algorithm.

Our algorithm is summarized in the steps below, which we refer to as **Algorithm GO-SNIP**. It is implemented as *Code 1*, which is depicted on the left-hand side of Figure 2.

1. Remove trivially redundant contingencies (Section 3.5) and preprocess the data (Section 3.6).
2. Solve the base case problem to obtain a base case solution. For this problem, the variables are initialized with a *flat start*: active power generation values are set to their upper limits; reactive power generation values are set to zero; voltages and controllable shunt susceptances at the buses are initialized to the midpoints within their described limits; and line and transformer flow variables and voltage angles are initialized to zero.
3. Write the current base case solution to a file.
4. Initialize the contingency priority list (Section 3.1).
5. Apply fast contingency evaluation (Section 3.4) to the contingencies, including complementarity updates (Section 3.3), according to the priority list until a time limit (of one minute) is reached or until all contingencies have been evaluated. Re-sort the priority list in order from largest to smallest penalty value.

6. Perform full contingency evaluations (Section 3.2) starting from the top of the list, given an overall time limit (e.g., a limit of 30 seconds was used for the GO Competition, Challenge 1). Re-sort the priority list in order from largest to smallest penalty value.
7. Add the top contingencies (e.g., for the GO Competition, Challenge 1, the top 3 were selected) from the list to the master problem, avoiding dominated contingencies (Section 3.5).
8. Solve the master problem, with no adjustments to the complementarity segments, while at the same time continuing to evaluate contingencies in the order of the priority list, with preference given to those that have not yet been evaluated. Each evaluation of a contingency starts with the fast evaluation (Section 3.4) including complementarity updates. If the penalty value after fast evaluation is still above a threshold, then full evaluation (Section 3.2) with complementarity updates (Section 3.3) is performed.
9. On completion of the solve of the master problem, write the new base case solution to a file and terminate the concurrent contingency evaluation procedure.
10. Update the priority list based on the most recently computed penalty values. If the priority list is not empty and time remains, return to Step 6. Otherwise, stop.

3.8. Implementation for the GO Competition, Challenge 1

In the GO Competition, Challenge 1, *Code 1* was given a time limit of 10 minutes or 45 minutes, depending on the competition category, and terminated once the time limit was exceeded. The most recent base case solution, written to a file, is the product of *Code 1*. Our implementation of *Code 2* is depicted in the right-hand side of Figure 2. After the base case solution produced by *Code 1* is read from the file, all contingencies are evaluated in parallel. For each contingency, the fast evaluation in Section 3.4 including complementarity updates is used first. If the penalty value after this evaluation is above a threshold, it is further reduced by a full contingency evaluation as described in Section 3.2.

For the competition, it was crucial to compute results for *all* contingencies within the time limit for *Code 2*, set by the competition organizers to (2 $\times$ number of contingencies) seconds. To ensure timely termination, at the beginning of a new contingency evaluation, we calculated how much total computing time was left, accumulated over all threads combined, and divided this total remaining time by the number of unevaluated contingencies. If, from that time onward, all contingencies would require that much time, it would be guaranteed that *Code 2* would finish within the overall time limit for *Code 2*. However, since most contingency evaluations are finished faster, the remaining time per contingency was expected to increase over the run of *Code 2*, with the largest time limits observed at the very end. To make the best use of this strategy, we processed the contingencies in the reverse order obtained by the ranking procedure described in Section 3.1.

Since contingencies for which the evaluation requires more time were typically ranked earlier in our list, they are processed at the final stages of *Code 2* and thus receive the largest time limit.

Participants had different options as to how each submitted the implementation of their algorithm to the evaluation platform. We chose to write *Code 1* and *Code 2* in C++ and submitted Linux executables. As previously mentioned, the nonlinear interior-point optimization software known as Ipopt (Wächter and Biegler 2006) was used to solve the base case problem and the master problems with selected contingencies; these problems all have the form in (2.8). Ipopt was also used to solve the contingency evaluation problems as described in Section 3.2, as well as the power flow equations for the fast contingency evaluation strategy described in Section 3.4.

Given the short amount of time available to compute a solution, it was crucial to utilize all computational resources provided by the organizers. To do so, we implemented a master/worker framework. The main thread was run on the head node, and was responsible for overall coordination as well as the solution of the base case and the master problems. The remaining threads were distributed across the compute nodes, and were responsible for continuously performing contingency evaluations. The parallelization was implemented using both MPI and pthreads. Each node ran one multi-threaded process. Communication between nodes took place with MPI, and within each process, the threads used shared memory to exchange data. The master process ran the base case and the master problem optimizations exclusively, even though these computations required only a single thread. Since these were the most crucial tasks in terms of the time limit, reserving an entire compute node for this thread meant that it was not slowed down by sharing hardware resources with other threads.

In the overall algorithm described in Section 3.7, some parameters had to be chosen. For example, we decided that the number of new contingencies added in each iteration was three (see Step 7). Similarly, in Step 6, we gave the algorithm 30 seconds for full contingency evaluation before the top three contingencies were chosen. These values seemed to perform well in the experiments during the competition. It is likely that other values, potentially adjusted to the size of the problem, would have performed better. However, here, as well as with respect to other aspects of the algorithm, the submission deadline imposed by the competition organizers forced us to make practical compromises that might not have been acceptable in a purely academic setting.

4. Numerical Results

For the preparation of this paper, we did not have access to the GO Competition, Challenge 1, computing platform. Instead, we ran our codes on a single Linux server node with two Intel Xeon CPUs running 40 (hyperthreaded) threads each and 256GB RAM.

We believe that the official results published by the competition organizers already present an unbiased comparison of the approaches and software implementation of the individual teams. Therefore, our numerical results presented here concentrate on the benefits of the contingency selection strategies described in the previous section. Specifically, we assess the contribution of each of the following of these heuristics: initial contingency ranking (see Section 4.1), complementarity update (see Section 4.2), fast evaluation (see Section 4.3), and contingency selection (see Section 4.4), with different heuristic choices being switched on and off.

The test cases in our experiments were taken from the Challenge 1 offline datasets (ARPA-E 2019a). We evaluated all scenarios for the selected networks. We also used the sets of contingencies given in these datasets. Following typical industry practices, these sets typically included the $N-1$ contingencies (failures of individual generators, transmission lines, and transformers) as well as some other multiple-component contingencies that were identified as important by the dataset creators. No time limits were imposed during these experiments because the hardware used was significantly less powerful than that available during the competition. In addition, since our purposes here are to isolate the effect of each of the aforementioned heuristics, we want to show results that are unaffected by a time limit.

4.1. Comparison: Contingency Initial Rankings

We employ a regression model to provide an initial ranking of contingencies; recall Section 3.1. This ranking prescribes the order in which the contingencies are initially evaluated in Step 4 of **Algorithm GO-SNIP**. In the competition, only a limited number of contingencies could be evaluated within the time constraint. Thus, it is critical that the most important contingencies are near the top of the initial contingency ranking so that they are the ones that are evaluated within the time constraints. In this section, we compare the heuristic we included in the algorithm with three other simpler heuristics that involve fewer features. In particular, we compare the following four ranking heuristics using the initial base case solution:

- Heuristic l^p: rank the contingencies by the associated amount of active power loss (p_g for a generator contingency and $\max\{p_e^o, p_e^d\}$ for a line/transformer contingency).
- Heuristic l^s: rank the contingencies by the associated amount of apparent power loss ($\sqrt{p_g^2+q_g^2}$ for a generator contingency and $\max\{\sqrt{(p_e^o)^2+(q_e^o)^2}, \sqrt{(p_e^d)^2+(q_e^d)^2}\}$ for a line (and similarly for a transformer) contingency).
- Heuristic l^c: rank the contingencies by the associated apparent power loss relative to the corresponding capacity $\left(\sqrt{\frac{p_g^2+q_g^2}{\overline{p}_g^2+\overline{q}_g^2}}\right.$ for a generator contingency and $\max\left\{\sqrt{\frac{(p_e^o)^2+(q_e^o)^2}{\overline{R}_e v_{i_e^o}}}, \sqrt{\frac{(p_e^d)^2+(q_e^d)^2}{\overline{R}_e v_{i_e^d}}}\right\}$ for a line (and similarly for a transformer) $\left.\text{contingency}\right)$.

- Heuristic l^w: rank the contingencies by the regression model as in Section 3.1.

To compare the effectiveness of these alternatives, we solved the base case problem and obtained the initial rankings with each of these ranking schemes. As a base line, we also performed full penalty minimization by solving (3.1) for each contingency to obtain their smallest penalty values.

In Step 7 of **Algorithm GO-SNIP**, three contingencies are selected after the contingency evaluation. Therefore, we only focus on the predictions of the top three contingencies using the above four ranking schemes. From the top of the ranking generated by each heuristic, we compute the penalty values of the top three contingencies as a percentage of the total penalty values of all contingencies. Figure 3 compares the ranking heuristics for four networks with different sizes. One representative scenario is selected for each network. On the horizontal axis, we increase the number of contingencies that are considered from the initial ranking. For each number of contingencies considered, we determine which are the top three using the different ranking schemes and compute the aforementioned percentage. In this manner, the penalty percentage monotonically increases as one moves to the right in the horizontal axis. A jump in the graph for a given heuristic indicates that a contingency with large penalty value has been identified once one more contingency is considered.

Figure 3 shows that all four ranking schemes identify important contingencies at an early stage of the contingency evaluation. However, our ranking heuristic l^w generally captures more important contingencies earlier in the list, as these representative figures illustrate. Overall, given any number of contingencies in the ranked list that have been evaluated, the top three contingencies identified via the regression model consistently have higher penalty values compared to those identified by the other three heuristics.

In the competition, the parallel execution of the initial penalty evaluations in Steps 5 and 6 made it possible to compute the correct (minimal) penalty value of several hundreds of contingencies within the time limits. The results in Figure 3 indicate that the most important contingencies could often be correctly identified and were added to the first master problem.

4.2. Comparison: Complementarity Updates

The choices of the active segments for the complementarity constraints (2.6) and (2.7) are updated using an active-set approach during contingency evaluation; recall Section 3.3. To illustrate the effectiveness of this approach, we next compare the contingency evaluation results obtained with and without complementarity updates.

In this experiment, we solve the base case problem (without contingencies) using *Code 1*, then perform contingency evaluations using *Code 2* with the base case solution from *Code 1*. In this contingency evaluation, we compare two approaches:

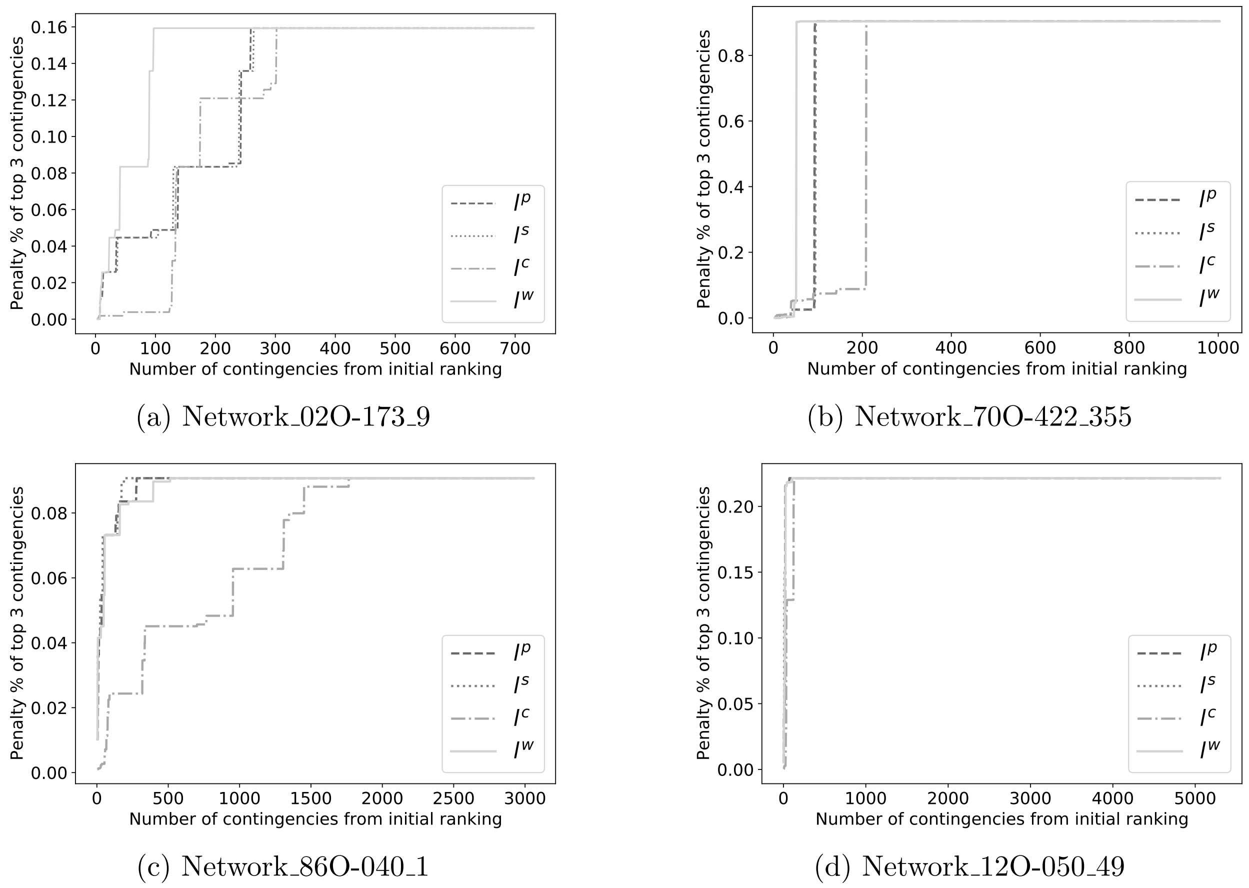

Figure 3 Comparison of different initial contingency ranking schemes: As one moves to the right along the horizontal axis, one considers adding more contingencies from the initially ranked list. Along the vertical axis is the penalty captured from the top 3 selected contingencies as a percentage of the total penalty values of all contingencies.

1. Without complementarity updates: each contingency is evaluated by solving (3.1) using the default complementarity initialization.
2. With complementarity updates: adjustments to the complementarity constraints are made iteratively within the contingency evaluations until the decrease in the penalty value is sufficiently small (as described in Section 3.3).

Figure 4 shows the factors by which the penalty values obtained without complementarity updates are worse than those obtained with complementarity updates. The result for each network is obtained by averaging over the scenarios. Note that since the complementarity updates allow for further optimization in the contingency evaluation, the penalty values obtained with complementarity updates are always less than the penalty values obtained without complementarity updates; hence, each bar in the figure is greater than or equal to 1. Allowing complementarity updates often results in significantly reduced penalty values for each network.

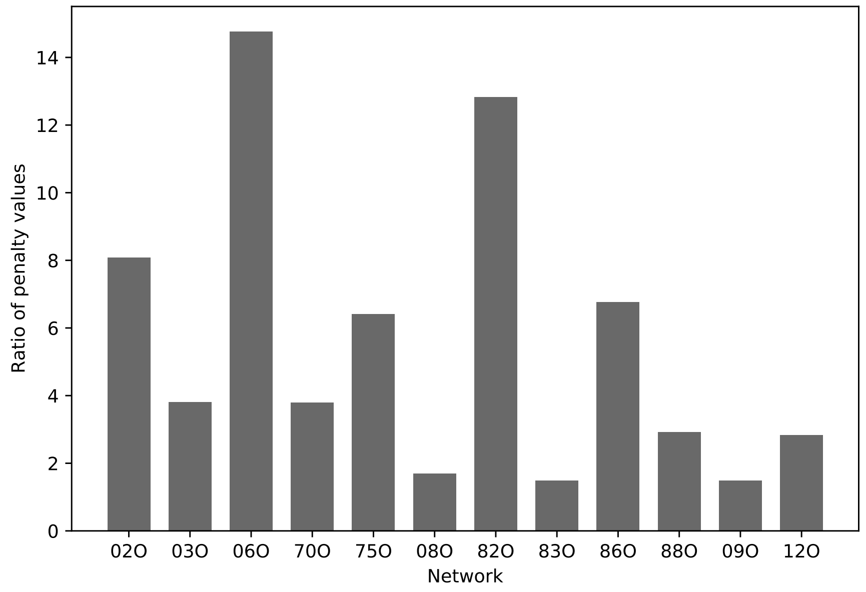

Figure 4 Ratio of penalty values obtained without complementarity updates to penalty values obtained with complementarity updates.

4.3. Comparison: Fast Contingency Evaluation

Our fast contingency evaluation scheme (see Section 3.4) is intended to mitigate the heavy computational cost required for full contingency evaluation. Indeed, in many cases, fast evaluation serves as a pre-screening step for filtering out feasible contingencies (i.e., with low penalty) from requiring a full evaluation. To assess the computational improvements resulting from this pre-screening step, we conduct a comparison of contingency evaluation with and without fast evaluation. Similar to the preceding experiment, we perform contingency evaluation in *Code 2* based on the base case solution obtained from *Code 1*. Two approaches are compared here:

1. With pre-screening: fast evaluation is performed first, followed by full evaluation if the penalty value obtained after fast evaluation is larger than a threshold value.
2. Without pre-screening: each contingency is fully evaluated without fast evaluation.

We set the threshold value for both fast and full evaluations such that the constraint violations are less than 2×10^{-2} per unit. (This constraint violation corresponds to the first breakpoint in the piecewise linear penalty function used in the competition.) Note that complementarity updates are included in both approaches.

Figure 5 shows the percentage of contingencies that are feasible after pre-screening. For most of the networks, fast evaluation eliminates over half of the contingencies. Among those contingencies that are feasible after pre-screening, we compare the average computation times with and without pre-screening in Figure 6. This figure shows significant improvement in computational time through

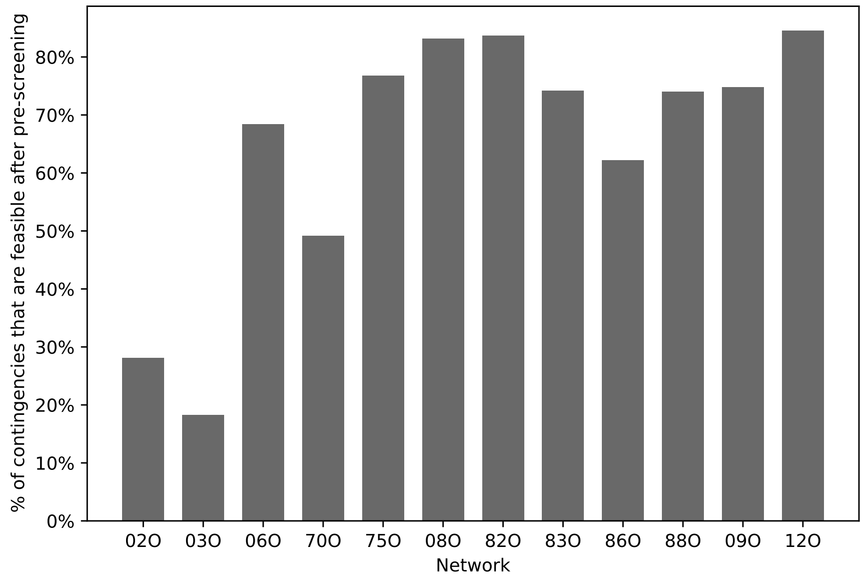

Figure 5 Percentage of contingencies with penalty value below a threshold after pre-screening.

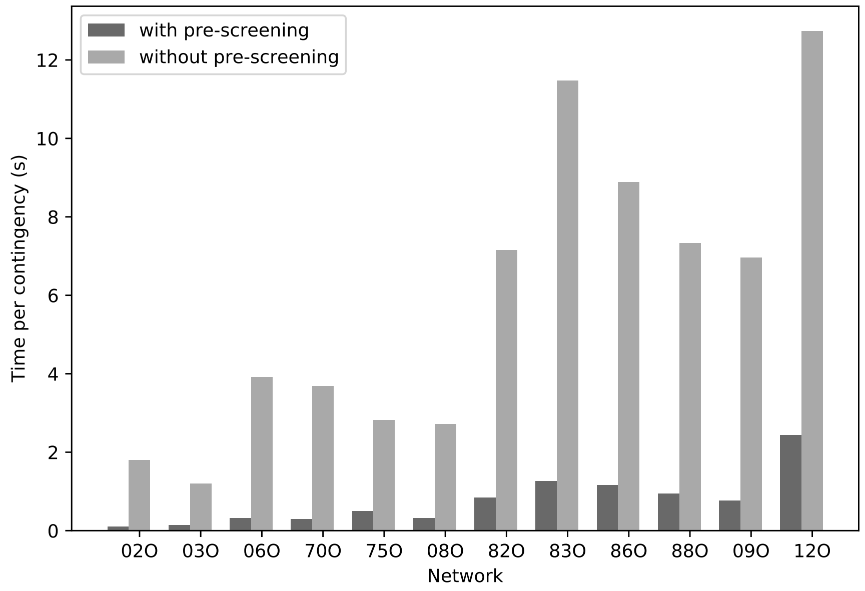

Figure 6 Comparison of average evaluation time among contingencies that are determined to be (sufficiently) feasible after pre-screening.

pre-screening. Figure 7 compares the total contingency evaluation times with and without pre-screening. This figure shows that pre-screening allows the evaluation of more contingencies within a limited time, thus identifying more important contingencies to include in the master problem.

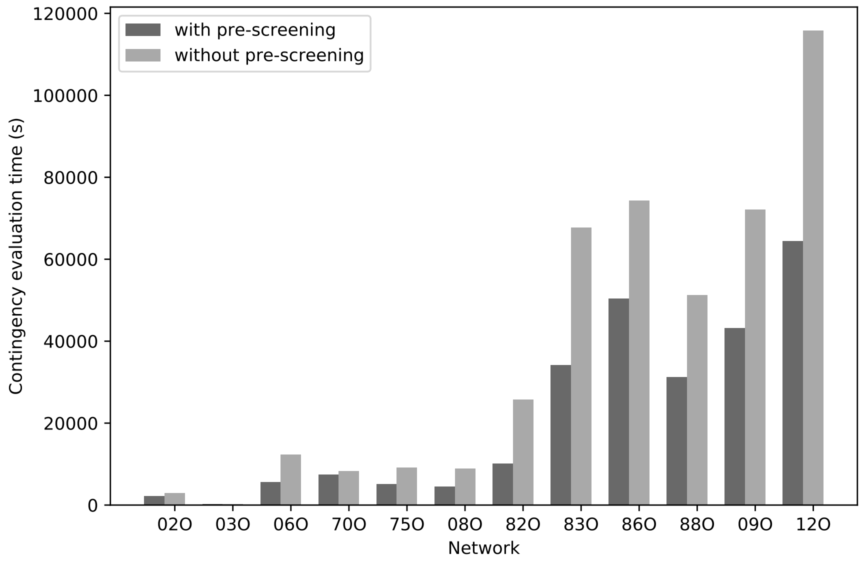

Figure 7 Comparison of contingency evaluation times with and without pre-screening.

4.4. Comparison: Contingency Selection

As discussed in Section 3, it is important to identity the most important contingencies to include in the master problem. Even though the initial ranking scheme gives a good approximation of the true penalties, the top contingencies captured by the initial ranking might not be the most important. Therefore, it is necessary to evaluate the contingencies to identify those that are most important. In **Algorithm GO-SNIP**, fast evaluation is applied first to contingencies in the order of the initial ranking. This eliminates a large percentage of unimportant contingencies within a short amount of time. The algorithm proceeds with full evaluation only for the remaining contingencies. The dominance filtering technique proposed in Section 3.5 helps avoid adding redundant contingencies into the master problem.

To show how each step in this procedure helps in identifying important contingencies, we compare four different contingency selection approaches, each of which adds an additional step to the previous approach:

1. Initial ranking: contingencies are ranked using our initial ranking scheme.
2. Fast evaluation: fast evaluation is applied to the contingencies in the order of the initial ranking. Contingencies are then ranked by their penalty values computed by fast evaluation.
3. Full evaluation: full evaluation is performed if the penalty value resulting from fast evaluation is above a threshold value. Contingencies are then ranked by their penalty values computed by full evaluation.
4. Dominance filtering: the dominance filtering technique is applied after the full evaluation of the contingencies. Dominated contingencies are eliminated from the ranking.

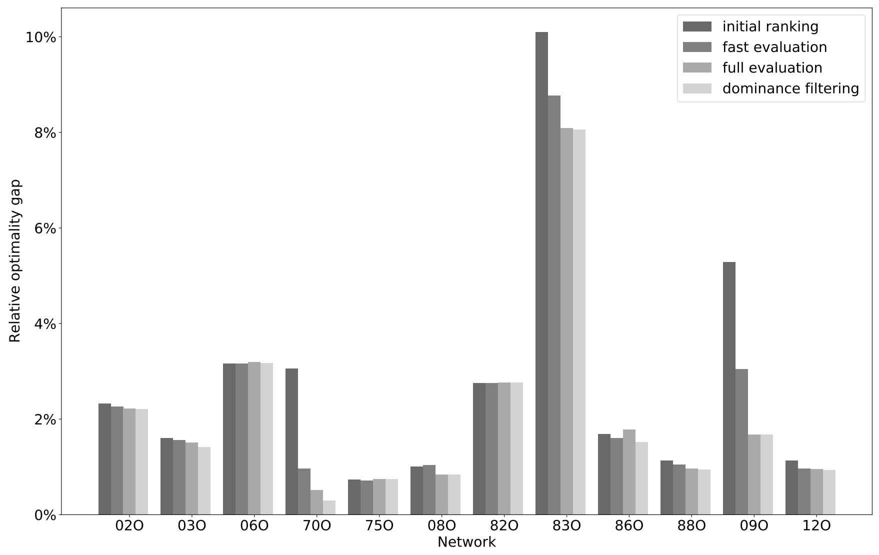

Figure 8 Comparison of combined contingency selection schemes.

For each of these approaches, we augment the base case with the constraints from the top three contingencies to form four master problems. We then solve these master problems and evaluate all contingencies for the new base case solution to compute the corresponding objective values including the penalty costs. To characterize the quality of the solutions obtained by just including three contingencies in the master problem, we compare these objective values with the best score from the final round of the competition. The results for each network are obtained by taking the average of the results over each scenario. Figure 8 shows the optimality gaps between these scores and the best scores from the competition. In nearly all cases, sequentially augmenting the contingency selection scheme to include the initial ranking strategy, fast evaluation, full evaluation, and dominance filtering monotonically reduces the total cost, in some cases significantly.

5. Conclusion

This paper presents the algorithm that our team, "GO-SNIP," submitted for the ARPA-E GO Competition, Challenge 1. The key features of our algorithm are strategies for contingency selection, fast contingency evaluation, handling complementarity constraints, avoiding issues related to degeneracy, and exploiting parallelism. The results of the numerical experiments that are presented isolate the effectiveness of important features of our solution algorithm. Our submission for the GO Competition, Challenge 1, along with those of the other participating teams, demonstrate that tailored nonlinear optimization techniques are capable of solving large-scale SC AC-OPF problems within industrially relevant time limits.

Ongoing efforts to improve upon the SC AC-OPF algorithms developed for the ARPA-E Grid Optimization Challenge 1 competition benefit from recently developed extensive repositories of large-scale realistic power system datasets. References to the specific datasets used in Challenge 1 as well as the subsequent Challenge 2 of the GO Competition are provided in (ARPA-E 2019a) and (ARPA-E 2019b), respectively. Other large repositories of power system data include DR POWER (DR POWER 2021) and The GRID DATA Repository (The BetterGrids Foundation 2021). Additionally, researchers often use the curated set of AC-OPF problems in the PGLib-OPF repository for algorithmic benchmarking purposes (Babaeinejadsarookolaee et al. 2019).

Acknowledgments

The authors would like to acknowledge Xingyi He for the training of the regression model for the initial ranking of contingencies. The authors also thank the reviewers and editors for their comments and suggestions, and gratefully acknowledge the organizers of the ARPA-E Grid Optimization competition. The information, data, or work presented herein was funded in part by the Advanced Research Projects Agency-Energy (ARPA-E), U.S. Department of Energy, under Award Number DE-AR0001073.

References

Aravena I, Molzahn DK, Zhang S, Petra CG, Curtis FE, Tu S, Wächter A, Wei E, Wong E, Gholami A, Sun K, Sun XA, Elbert ST, Holzer JT, Veeramany A (2022) Recent developments in security-constrained AC optimal power flow: Overview of Challenge 1 in the ARPA-E Grid Optimization Competition. *arXiv:2206.07843* .

ARPA-E (2019a) ARPA-E GO Competition Challenge 1 datasets. https://gocompetition.energy.gov/challenges/22/datasets.

ARPA-E (2019b) ARPA-E GO Competition Challenge 2 datasets. https://gocompetition.energy.gov/challenges/1/datasets.

ARPA-E (2019c) ARPA-E GO Competition website. https://gocompetition.energy.gov/.

Babaeinejadsarookolaee S, Birchfield A, Christie RD, Coffrin C, DeMarco CL, Diao R, Ferris M, Fliscounakis S, Greene S, Huang R, Josz C, Korab R, Lesieutre BC, Maeght J, Molzahn DK, Overbye TJ, Panciatici P, Park B, Snodgrass J, Zimmerman RD (2019) The power grid library for benchmarking AC optimal power flow algorithms. *arXiv:1908.02788* URL https://github.com/power-grid-lib/pglib-opf.

Capitanescu F (2016) Critical review of recent advances and further developments needed in AC optimal power flow. *Electric Power Systems Research* 136:57–68.

Capitanescu F, Glavic M, Ernst D, Wehenkel L (2007) Contingency filtering techniques for preventive security-constrained optimal power flow. *IEEE Transactions on Power Systems* 22(4):1690–1697.

Capitanescu F, Ramos JLM, Panciatici P, Kirschen D, Marcolini AM, Platbrood L, Wehenkel L (2011) State-of-the-art, challenges, and future trends in security constrained optimal power flow. *Electric Power Systems Research* 81(8):1731–1741.

Capitanescu F, Wehenkel L (2013) Experiments with the interior-point method for solving large scale optimal power flow problems. *Electric Power Systems Research* 95:276–283.

Clements KA, Davis PW, Frey KD (1991) An interior point algorithm for weighted least absolute value power system state estimation. *IEEE Power & Energy Society Winter Meeting* (New York, NY, USA).

DR POWER (2021) Data repository for power system open models with evolving resources (DR POWER). `https://egriddata.org/`.

Ejebe GC, Wollenberg BF (1979) Automatic contingency selection. *IEEE Transactions on Power Apparatus and Systems* PAS-98(1):97–109.

Hu XM, Ralph D (2004) Convergence of a penalty method for mathematical programming with complementarity constraints. *Journal of Optimization Theory and Applications* 123:365–390.

Stott B, Alsaç O, Monticelli AJ (1987) Security analysis and optimization. *Proceedings of the IEEE* 75(12):1623–1644.

The BetterGrids Foundation (2021) The GRID DATA repository. `https://www.bettergrids.org/`.

Wächter A, Biegler LT (2006) On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming* 106(1):25–57.

Wang H, Murillo-Sanchez CE, Zimmerman RD, Thomas RJ (2007) On computational issues of market-based optimal power flow. *IEEE Transactions on Power Systems* 22(3):1185–1193.

Wu L, Gao J, Wang Y, Harley RG (2017) A survey of contingency analysis regarding steady state security of a power system. *North American Power Symposium (NAPS)*, 1–6.

Wu YC, Debs AS, Marsten RE (1994) A direct nonlinear predictor-corrector primal-dual interior point algorithm for optimal power flows. *IEEE Transactions on Power Systems* 9(2):876–883.