European Journal of Operational Research 279 (2019) 704–720

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

Designing networks with resiliency to edge failures using two-stage robust optimization

Logan R. Matthews[a], Chrysanthos E. Gounaris[b,*], Ioannis G. Kevrekidis[a,c]

[a] Department of Chemical and Biological Engineering, Princeton University, Princeton, NJ, USA
[b] Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA, USA
[c] Department of Chemical and Biomolecular Engineering, Johns Hopkins University, Baltimore, MD, USA

ARTICLE INFO

Article history:
Received 22 February 2018
Accepted 10 June 2019
Available online 13 June 2019

Keywords:
Networks
Network design
Network resiliency
Two-stage robust optimization
Column and constraint generation

ABSTRACT

We study the design of resilient single-commodity flow networks that can remain robust against multiple concurrent edge failures. We model these failures as binary random variables, allowing us to formally formulate the network design problem as a two-stage robust optimization problem. With an objective of minimizing the overall cost of building and operating the network, the capacities of the edges are decided in the first stage, while the optimal flows are determined in the second stage once the uncertainty has been realized. We first examine the standard affine decision rules approach and show that it is not a viable approach when two or more edges are allowed to fail at the same time. We then propose a *column and constraint generation algorithm* that we tailor to this application. Since the problem does not satisfy the *relatively complete recourse* assumption, we employ an *oracle with two subproblems*: one to determine edge failure scenarios that render the required demand satisfaction infeasible, and if no such scenario exists, a second one to determine the flow rerouting plan of highest cost. Our column and constraint generation algorithm is applied to networks adapted from the Survivable Network Design Library. For each instance, we determine sequences of fully adaptive, robust optimal solutions for various levels of resiliency, identifying also the maximum number of concurrent edge failures that can be sustained by these networks. Finally, we demonstrate how our algorithm can be applied to a defender versus attacker context, via the use of a decision-dependent uncertainty set.

1. Introduction

Networks are prevalent in a vast variety of contexts, and it comes as no surprise that their optimal design has arisen as a prime research question that has been contemplated extensively in the open literature. Whether it be a supply chain (Snyder, Scaparra, Daskin, & Church, 2006), telecommunications (Orlowski, Wessäly, Pióro, & Tomaszewski, 2010), or transportation (Alderson, Brown, & Carlyle, 2015; Zhang, Lawphongpanich, & Yin, 2009) application, to name but a few, obtaining optimal designs for such networks is a very important task. Network design problems come in many flavors, such as problems involving a single versus multiple commodities, problems in which the demands are to be met exactly versus allowing for demand shortfall and pursuing some sort of penalization scheme, as well as problems in which the network is to be designed from scratch versus the case where a pre-existing network is to be expanded. The nominal case for a network design problem is often straightforward: decisions must be made on the capacities of and/or flows through each edge of the network such that demand is satisfied at each node. Designing nominal networks in all of the above settings can typically be achieved via solving some monolithic, often mixed-integer linear, optimization formulation that utilizes flow balance constraints at its core. Mixed-integer linear optimization has also been shown to be an effective tool to control the levels of network connectivity, spread and assortativity by explicitly imposing constraints on collective properties of the network (Gounaris, Rajendran, Kevrekidis, & Floudas, 2011; 2016).

The problem of designing networks, however, becomes significantly harder to tackle when uncertainty comes into play. Such uncertainty may appear in a variety of forms, with the most common ones being uncertainty in the costs of the network or the demand at each node. Bertsimas and Sim (2003) addressed cost uncertainty through the use of robust optimization (Ben-Tal, El Ghaoui, & Nemirovski, 2009) with a uncertainty set of polyhedral type. The popular uncertainty sets of Bertsimas and Sim (2004) have since been commonly applied to network design problems as well, as

* Corresponding author.
E-mail address: gounaris@cmu.edu (C.E. Gounaris).

https://doi.org/10.1016/j.ejor.2019.06.021
0377-2217/

seen for example in the handling of demand uncertainty in virtual private networks by Altın, Amaldi, Belotti, and Pınar (2007). Uncertain costs in the form of travel times as well as demand uncertainties were also addressed using conic uncertainty sets by Ordóñez and Zhao (2007) in a capacity expansion version of the robust network flow problem. More recently, Cacchiani, Jünger, Liers, Lodi, and Schmidt (2016) and Gounaris and Schmidt (2019) considered the case when uncertain demand may be represented through a finite number of scenarios or using Hose uncertainty sets, and Poss and Raack (2013) considered affine decision rules for handling uncertain demand. Many additional studies have elucidated the use of robust optimization in the context of network design problems under cost and/or demand uncertainty (Atamtürk & Zhang, 2007; Lee, Lee, & Park, 2013; Álvarez Miranda, Cacchiani, Lodi, Parriani, & Schmidt, 2014; Mudchanatongsuk, Ordóñez, & Liu, 2008), and have demonstrated applications of such problems in a wide range of fields. For instance, Gong, Garcia, and You (2016) and Gong and You (2017) formulate problems of selecting optimal biomass conversion pathways, addressing price, supply and demand uncertainties in their networks using a column-and-constraint generation algorithm (Zeng & Zhao, 2013). It is clear that cost and demand uncertainty is an important application area that has to-date attracted a significant amount of research effort.

This manuscript will focus on network design under another ominous form of uncertainty, namely, the possibility for failure of edges in the network topology. The failure of an edge may prevent some or all flow along that edge, potentially cutting off nodes that supply or require commodities and eliminating any feasible way to satisfy the overall commodity supply and demand requirements of the network. The design of networks that are robust against such uncertainty is often referred to as *survivable*, or *resilient*, network design, and can be addressed in a variety of ways. An early approach by Aneja, Chandrasekaran, and Nair (2001) considered the case where the network contains one source and one sink that must be connected, and assumed that flow may not be rerouted after the failure. In this case, the goal was to maximize the remaining flow in the network given a single edge failure. Tomaszewski, Pióro, and Żotkiewicz (2010) described a "flow-restoration" problem in which extra capacity is reserved along each edge such that, for every edge failure scenario, there exists enough capacity for flow to be redirected along installed edges that did not fail. The authors note that this problem is NP-hard for cases with multiple simultaneous edge failures, and argue that path generation techniques could be used to tackle it efficiently. Koster and Kutschka (2011) also considered demand uncertainty in a single source and sink setting, achieving survivability under an edge failure by limiting the percentage of flow along each path, which forces flow diversification over disjoint paths. Many robust and adaptive maximum flow or minimum cut formulations are provided by Bertsimas, Nasrabadi, and Stiller (2013) when there exists one source and one sink, while Boginski, Commander, and Turko (2009) consider a conditional value-at-risk formulation for the network flow problem with an objective of minimizing cost when each edge has a probability of failure.

The fact that the network designs are chosen prior to the edges failing leads naturally to min-max-min formulations, which correspond nicely to two-stage robust optimization problems, which we also contemplate later in this manuscript. In the context of supply chains, Snyder et al. (2006) provided many formulations for network design, focusing primarily on the cases when facilities or sources may fail. This is similar to the problem of edge failures in that, if an entire node fails, then essentially each edge connected to that node has failed. Such failure of a node would not be acceptable in the application of this manuscript, which mandates that *demands of all nodes must be met under all failure scenarios*. A column and constraint generation algorithm for robust network design was proposed by Simchi-Levi, Wang, and Wei (2017) for cases with uncertain demands and uncertain capacities, modeled through continuous polyhedral uncertainty sets. The effectiveness of the algorithm was demonstrated through two case studies involving uncertain demands in the context a robust lot-sizing problem and a production postponement problem. Finally, Chen and Phillips (2013) studied the design of multi-commodity networks that are resilient to multiple edge failures. In their setting, each commodity has one source and sink, while pre-determined capacity values must be applied for each edge that is chosen to be built. The objective involves the minimization of investment costs, and optimal network designs are found using a Benders' decomposition approach and a column-and-cut algorithm.

Survivability against edge failures has also been approached through the protection of edges before uncertainty is realized, viewing the problem via an attacker-defender lens. Yuan et al. (2016a) utilized two-stage robust optimization to decide which edges in a power grid network should be hardened to protect against hurricane damage through a min-max-min model that will minimize the load shedding due to failed edges. Allowing for such load shedding imbues relatively complete recourse to the problem, which can then be addressed with the column-and-constraint generation algorithm proposed by Zeng and Zhao (2013). Alderson et al. (2015) formulated defender-attacker-defender models to assess the operational resilience of transportation network infrastructure. In this approach, edges may be protected and new edges may be built at pre-set capacities up to a given defense budget, in order to minimize the total cost of meeting the demand of a single-commodity in a network. Their models allow for penalized demand shortfall in their objective function due to edge failures. Finally, Lou and Zhang (2011) considered the protection of transportation networks from attacks via a min-max-min formulation for which a defender may either minimize the total amount of unmet demand or maximize the total amount of flow through the network. In this problem, each edge has a designated capacity which may be decreased or eliminated due to an attacker's decision, given that the edge is not protected by the defender. The problem is solved via a mixture of a cutting-plane and active set algorithm, which is common in discrete network design problems in the transportation sector in which discrete modifications, such as added lanes, are made to existing infrastructures (Wang & Pardalos, 2017; Zhang et al., 2009).

Another approach that one could potentially follow in a two-stage robust optimization context is adjustable robust optimization. This is a popular technique for addressing uncertainty through the explicit definition of second-stage recourse decisions as functions of the uncertain parameters, often in affine relationships (Ben-Tal, Goryashko, Guslitzer, & Nemirovski, 2003). The use of affine decision rules has been shown to be effective in a variety of applications, from process scheduling (Lappas & Gounaris, 2016) to network capacity assignment with uncertain demands (Babonneau, Vial, Klopfenstein, & Ouorou, 2013) to unit commitment problems (Álvaro Lorca, Sun, Litvinov, & Zheng, 2016). Although there have been cases where affine decision rules yield optimal, fully adaptive solutions (Bertsimas, Iancu, & Parrilo, 2010; Gounaris, Wiesemann, & Floudas, 2013), it should be highlighted that this is not always guaranteed, warranting the use of an alternative solution framework such as the column and constraint generation employed in a number of the above studies.

In this paper, we explore the suitability and effectiveness of adjustable robust optimization with affine decision rules as well as column and constraint generation in solving a two-stage robust optimization formulation of the single-commodity network design problem under the possibility of multiple concurrent edge failures, which we model as binary random variables, as we primarily focus on situations when edges would only fail completely, such as

power lines shorting out due to fallen trees, network cables going offline, pipeline leakages warranting a complete shutdown due to environmental zero tolerance policy, or railroad tracks being blocked by obstacles, to name but a few examples. In our problem setting, a network must be either designed from scratch or expanded through the choice of capacity and flow variables for each edge. These decisions must be made such that the network remains feasible (i.e., demands are satisfied at each node) for any realization of an uncertainty set that allows up to k concurrent edge failures. Consequently, we assume at first that a node cannot be cut off completely by edge failures, as otherwise the problem would be guaranteed to be robust infeasible, though we later demonstrate how to lift this assumption in the context of a defender-attacker setting. Once the capacities are chosen in a first stage, we allow uncertainty to realize, and we choose the flows in a second stage after the edge failures have occurred.

The resulting two-stage robust optimization approach is a very natural one for handling uncertainty of this nature. Rather than *a priori* generating all (or the most important subset of) failure scenarios of a multiple edge network so as to, say, support a stochastic optimization approach, one can simply identify all edge failures that might be realized in the network and appropriately correlate them in an intuitive uncertainty set. Then, as one tries to maintain overall network feasibility, one is not concerned with any probabilistic interpretation of edge failures, but simply cares to ensure a working network in the worst case. Two-stage robust optimization also naturally encapsulates the order of events, with design decisions that happen initially followed by uncertainty realizations and resulting flow decisions.

The major contributions of this manuscript are outlined as follows.

1. We formalize the design of networks that maintain their ability to meet demand requirements after any k concurrent edge failures as a two-stage robust optimization problem considering costs for both capacities and flows.
2. We show that the popular affine decision rules approach is exact when $k = 1$, but inadequate to admit robust feasible solutions when $k > 1$.
3. We propose a *tailored column and constraint generation algorithm* to solve our robust network design problem to guaranteed optimality; this algorithm *does not require* the common assumption of relatively complete recourse.
4. We conduct a comprehensive set of computational studies to measure the numerical tractability of our column and constraint generation algorithm, elucidating in the process the price of robustness required to insure network designs against various numbers of concurrent edge failures.
5. We utilize a decision-dependent uncertainty set and demonstrate how our algorithm can be applied on a defender versus attacker network design context.

The remainder of this manuscript is structured as follows. The two-stage robust network design problem is outlined in Section 2, along with an analysis of the applicability and limitations of using the affine decision rules approach to address it. Section 3 then presents the proposed column and constraint generation algorithm that we develop to obtain fully adaptive solutions to the two-stage problem. The computational efficiency is demonstrated in Section 4, and the problem is expanded to the defender-attacker paradigm in Section 5. We finally conclude in Section 6.

2. Problem definition

The single-commodity network design problem considered in this paper is concerned with maintaining the ability to meet the demand at each node despite a certain number, k, of concurrent edge failures. In our context, an edge failure completely removes the ability for any flow to move along the edge. Given that edge failures may occur, the optimum network topology and edge capacities should be such that flow may be rerouted along the remaining edges and meet the demands at every node.

Let $G = (V, E)$ be a network superstructure, which we assume without loss of generality (w.l.o.g.) to be a connected graph.[1] Each edge $e \in E$ is defined with origin node $o(e) \in V$ and destination node $d(e) \in V$. Flow is allowed to travel in either direction along the edge, with f_e^f representing forward flow along the edge from node $o(e)$ to node $d(e)$, and f_e^b representing backward flow, from the destination node to the origin node, along the edge. The flows must be assigned to the edges such that the demands $b \in \mathbb{R}^{|V|}$ are satisfied for each node $i \in V$.[2] We remark that, w.l.o.g., we require $\sum_{i \in V} b_i = 0$ for the instance to be well-defined, as well as $\max_{i \in V}\{b_i\} > 0$ for the instance to be non-trivial.

Obviously, for flow to occur along an edge, the edge must have some capacity. Whereas pre-installed capacities p_e may be available on certain edges of the network, we always allow for new capacities u_e to be added on any edge. Given these definitions, a deterministic single-commodity flow network design problem may be modeled via the formulation below.

$$
\begin{aligned}
\min_{u, f^b, f^f \in \mathbb{R}} \quad & \sum_{e \in E} c_e^u u_e + \sum_{e \in E} c_e^f \left(f_e^f + f_e^b\right) & & \\
\text{s.t.} \quad & f_e^f \le u_e + p_e & & \forall\ e \in E \\
& f_e^b \le u_e + p_e & & \forall\ e \in E \\
& \sum_{e: d(e) = i} \left(f_e^f - f_e^b\right) + \sum_{e: o(e) = i} \left(f_e^b - f_e^f\right) = b_i & & \forall\ i \in V \\
& u_e,\ f_e^b,\ f_e^f \ge 0 & & \forall\ e \in E
\end{aligned} \tag{1}
$$

The objective function in Formulation (1) is to minimize the total cost to build the network and operate it by sending flows along its edges. The parameters c_e^u represent the capacity-dependent investment costs,[3] amortized over the lifetime of the network, while parameters c_e^f represent the costs to flow along the edges. The first and second sets of constraints ensure that the flows along each edge, either forward or backward, do not exceed the total capacity installed on the edge,[4] while the third constraint set corresponds to the standard flow balance constraints, which ensure that the sum of all flows into and out of each node equals the demand of the node.

Formulation (1) may be solved to provide an optimal solution to the network design problem for normal, or *nominal*, operation when the totality of the network's edges remain functional. However, at the nominal solution, any edge failure may prevent the network from being able to satisfy all node demands. Therefore, the possibility for edge failures should be incorporated into the model in order to gain a robust network design that would satisfy all edge failure scenarios at the minimum cost. To that end, given random binary variables ξ_e to indicate whether edges $e \in E$ fail, an uncertainty set is defined to allow up to k edges in the network to simultaneously be in a state of failure at any moment during

[1] If G is not connected, then every one of its components defines an independent problem that could be addressed separately.

[2] A negative demand represents the net amount of the commodity that the node must supply to the rest of the network.

[3] We remark that it is straightforward to include fixed investment costs in this formulation through the presence of a binary variable indicating whether a new edge is built or not, albeit at an increase in computational cost. For simplicity of exposition, this extension is not considered in this work, but the same column and constraint generation algorithm would still apply.

[4] Note that, in this single-commodity flow problem where $c_e^f > 0$, it will never be the case at an optimal solution that both f_e^f and f_e^b are nonzero; hence, these two sets of constraints suffice to enforce the edge capacity restrictions.

network operation, as follows:

$$\Xi := \left\{ \xi \in \{0,1\}^{|E|} : \begin{array}{l} \sum_{e \in E} \xi_e \leq k \\ \sum_{e:d(e)=i} \xi_e + \sum_{e:o(e)=i} \xi_e \leq D_i - 1 \quad \forall i \in V_T \end{array} \right\}, \tag{2}$$

where D_i is the degree of a node $i \in V$, and $V_T \subseteq V$ is the subset of *terminal* nodes, i.e., nodes that have nonzero demands, and hence, must either receive or transmit some amount of flow at any feasible solution.

Consequently, the second constraint in the uncertainty set mandates that at least one edge connecting to a terminal node must be allowed to remain in operation. Had this constraint not been incorporated in the uncertainty set, we would allow for scenarios where all edges incident to some terminal node become inoperable, making the problem trivially robust infeasible for any $k \geq \min_{i \in V_T} D_i$. Besides this "numerical" motivation for adding this constraint, we remark that restricting the uncertainty set in this manner conforms well with many real-world operational contexts, where extra care would be taken to ensure the last operating edge of a node stays online any time all other edges connecting to that node are down. If, however, one is dedicated to allowing the complete elimination of a node's connectivity to other nodes, one could add a penalty term to the objective function associated with missed demand, through the incorporation of slack variables into the flow balance constraint. In this paper, we choose to focus on maintaining completely demand feasible networks, which also makes for a more theoretically interesting two-stage robust optimization setting by preventing relatively complete recourse.

We also remark that, for certain cases where it is appropriate to allow for partial edge failures, the latter could be modeled by allowing random variables $\xi \in [0, 1]$, rather than ξ being binary. Whereas this is a relaxation of the uncertainty set Ξ, it actually makes the problem easier to solve computationally, as it reduces the (otherwise MILP) separation problem to a mere LP.

Given the uncertainty set defined above, Formulation (3) may be constructed for the robust network design problem. In this a two-stage robust optimization model, the decisions of what capacities to install are taken in the first-stage minimization, while the actual operating flows are considered in the second-stage minimization, after potential edge failures have occurred via the maximization over the uncertainty set Ξ. Note how, if an edge fails, $\xi_e = 1$ and the right-hand-sides of the two corresponding capacity constraints in the inner problem become zero, denying any flow in that edge.

$$\begin{array}{lll} \min_{u \in \mathbb{R}} & \sum_{e \in E} c_e^u u_e + \max_{\xi \in \Xi} \min_{f \in \mathbb{R}} & \sum_{e \in E} c_e^f \left(f_e^f + f_e^b \right) \\ \text{s.t.} & u_e \geq 0 \quad \forall\, e \in E \ \text{s.t.} & f_e^b \leq (1 - \xi_e)(u_e + p_e) \qquad \forall\, e \in E \\ & & f_e^f \leq (1 - \xi_e)(u_e + p_e) \qquad \forall\, e \in E \\ & & \sum_{e:d(e)=i} \left(f_e^f - f_e^b \right) + \sum_{e:o(e)=i} \left(f_e^b - f_e^f \right) = b_i \qquad \forall\, i \in V \\ & & f_e^b, f_e^f \geq 0 \qquad \forall\, e \in E \end{array} \tag{3}$$

2.1. An affine decision rule approach and its limitations

Solving a two-stage robust optimization problem like (3), which features continuous second-stage variables (the flows), may be approached in a variety of ways. Perhaps the most common approach pertains to the use of affine decision rules, where one opts to enforce an affine relationship between the second-stage variables and the observed realization of the uncertain random variables, in an effort to gain a possibly conservative, yet tractable, solution to the problem.

In order to apply the affine decision rules approach in our robust network design problem, the flow variables f_e^b and f_e^f are redefined using new variables to represent the slopes ($f_{e,l}^b$, $f_{e,l}^f$ $\forall e \in E, l \in E$) and intercepts (f_e^{b0}, f_e^{f0} $\forall e \in E$) of the postulated affine relationships:

$$\begin{aligned} f_e^b &\leftarrow f_e^{b0} + \sum_{l \in E} \xi_l f_{e,l}^b \qquad \forall e \in E \\ f_e^f &\leftarrow f_e^{f0} + \sum_{l \in E} \xi_l f_{e,l}^f \qquad \forall e \in E. \end{aligned} \tag{4}$$

After replacing the original flow variables with these expressions, and after converting the inner maximization problem into a semi-infinite form (introducing an epigraph variable τ in the process to reformulate the second-stage minimization operator), we acquire Formulation (5). Using standard techniques, further manipulation of this formulation will yield a linear programming model that can be addressed directly via standard linear optimization codes. The final model as well as some remarks on its derivation are presented in Appendix A. Note that this derivation requires a relaxation of the binary random variables to continuous ones, in order to allow for dualization of the inner problem; this is equivalent to allowing partial edge failures, yet under integer values for k, this mathematical relaxation will not alter the optimal solutions.

$$\begin{array}{lll} \min_{u, f, \tau \in \mathbb{R}} & \sum_{e \in E} c_e^u u_e + \tau & \\ \text{s.t.} & \sum_{e \in E} c_e^f \left(f_e^{f0} + \sum_{l \in E} \xi_l f_{e,l}^f + f_e^{b0} + \sum_{l \in E} \xi_l f_{e,l}^b \right) \leq \tau & \forall\, \xi \in \Xi \\ & f_e^{b0} + \sum_{l \in E} \xi_l f_{e,l}^b \leq (1 - \xi_e)(u_e + p_e) & \forall\, \xi \in \Xi,\ \forall\, e \in E \\ & f_e^{f0} + \sum_{l \in E} \xi_l f_{e,l}^f \leq (1 - \xi_e)(u_e + p_e) & \forall\, \xi \in \Xi,\ \forall\, e \in E \\ & f_e^{b0} + \sum_{l \in E} \xi_l f_{e,l}^b \geq 0 & \forall\, \xi \in \Xi,\ \forall\, e \in E \\ & f_e^{f0} + \sum_{l \in E} \xi_l f_{e,l}^f \geq 0 & \forall\, \xi \in \Xi,\ \forall\, e \in E \\ & \sum_{e:d(e)=i} \left(f_e^{f0} + \sum_{l \in E} \xi_l f_{e,l}^f - f_e^{b0} - \sum_{l \in E} \xi_l f_{e,l}^b \right) + \sum_{e:o(e)=i} & \\ & \times \left(f_e^{b0} + \sum_{l \in E} \xi_l f_{e,l}^b - f_e^{f0} - \sum_{l \in E} \xi_l f_{e,l}^f \right) = b_i & \forall\, \xi \in \Xi,\ \forall\, i \in V \\ & u_e \geq 0 & \forall\, e \in E \end{array} \tag{5}$$

Despite the widespread use of affine decision rules for a variety of problems featuring continuous recourse variables, it is worth examining the suitability of the affine relationship assumption in the context of network design problems with multiple concurrent edge failures. In fact, the following propositions show that, while affine decision rules provide exact optimal solutions without any adaptivity gap when at most one edge failure occurs, such rules are not appropriate when two or more edges are allowed to fail concurrently.

Proposition 1. *For the robust network design problem defined in Formulation (3) under the uncertainty set defined in (2) with $k \leq 1$, using affine decision rules of the form (4) will yield the optimum solution, if one exists.*

Proof. The case $k = 0$ is trivial, since the uncertainty set admits only a single scenario, namely the nominal realization $\xi_e = 0$ for all

$e \in E$. These parameter values can thus be imposed on the problem, simplifying it to a formulation that is equivalent to (1) under variable transformations $f^b \leftarrow f^{b0}$ and $f^f \leftarrow f^{f0}$. Solving this formulation will obviously yield the same result as the deterministic problem, which qualifies as the robust problem in this case.

In the case $k = 1$, the uncertainty set admits the above nominal realization, as well as additional realizations corresponding to individual edge failure scenarios. Let $E^1 \subseteq E$ be the subset of edges that are allowed to fail (alone) in some scenario admitted by the applicable set. W.l.o.g., let us also assume that $|E^1| \neq 0$, since otherwise the instance is equivalent to the trivial $k = 0$ case addressed above.

The only scenario that involves the failure of some edge $e' \in E^1$, namely the scenario where $\xi_{e'} = 1$ and $\xi_l = 0$ for all $l \in E \backslash \{e'\}$, directly impacts the upper bounds of the corresponding edge's flow variables. More specifically, any robust feasible solution must satisfy

$$0 \leq f^{b0}_{e'} + \sum_{l \in E} \xi_l f^b_{e',l} = f^{b0}_{e'} + f^b_{e',e'} \leq (1 - \xi_{e'})(u_{e'} + p_{e'}) = 0$$
$$0 \leq f^{f0}_{e'} + \sum_{l \in E} \xi_l f^f_{e',l} = f^{f0}_{e'} + f^f_{e',e'} \leq (1 - \xi_{e'})(u_{e'} + p_{e'}) = 0. \quad (6)$$

Thus, it is obvious that $f^{b0}_{e'} + f^b_{e',e'} = 0$ and $f^{f0}_{e'} + f^f_{e',e'} = 0$ for all $e' \in E^1$, since the final solution must be robust to any such edge failing.

The robust solution must also satisfy the demand constraints for any scenario, namely

$$\sum_{e:d(e)=i} \left(f^{f0}_e + \sum_{l \in E} \xi_l f^f_{e,l} - f^{b0}_e - \sum_{l \in E} \xi_l f^b_{e,l} \right) + \sum_{e:o(e)=i} \times \left(f^{b0}_e + \sum_{l \in E} \xi_l f^b_{e,l} - f^{f0}_e - \sum_{l \in E} \xi_l f^f_{e,l} \right) = b_i \quad \forall\, \xi \in \Xi,\ \forall\, i \in V, \quad (7)$$

which may be expanded over the $1 + |E^1|$ possible realizations encompassed by the uncertainty set. For the nominal scenario, $\xi_e = 0$ for all $e \in E$, this yields the nominal demand satisfaction constraints

$$\sum_{e:d(e)=i} \left(f^{f0}_e - f^{b0}_e \right) + \sum_{e:o(e)=i} \left(f^{b0}_e - f^{f0}_e \right) = b_i \quad \forall\, i \in V. \quad (8)$$

For the remaining scenarios where edges $e' \in E^1$ fail alone, and taking also into account the above proven fact that $f^{b0}_{e'} + f^b_{e',e'} = 0$ and $f^{f0}_{e'} + f^f_{e',e'} = 0$, the solution must also satisfy constraints

$$\sum_{e:d(e)=i, e \neq e'} \left(f^{f0}_e + f^f_{e,e'} - f^{b0}_e - f^b_{e,e'} \right) + \sum_{e:o(e)=i, e \neq e'} \times \left(f^{b0}_e + f^b_{e,e'} - f^{f0}_e - f^f_{e,e'} \right) = b_i \quad \forall\, e' \in E^1,\ \forall\, i \in V. \quad (9)$$

Examining constraints (8) and (9) reveals the presence of an effective flow variable for each edge and scenario combination. More specifically, for general failure of edge e', the flow along edge e is simply $f^{b0}_e + f^b_{e,e'}$ or $f^{f0}_e + f^f_{e,e'}$, where the values of f^{b0}_e and f^{f0}_e are equal to the nominal scenario flows, and $f^b_{e,e'}$ and $f^f_{e,e'}$ constitute the flow adjustments necessary under the specific scenario when edge e' fails. Thus, there effectively exists a unique set of flow variables for each failure scenario, and if a robust feasible solution exists for a given first-stage decision u_e, it will be found.

Furthermore, a robust optimal solution will be found with no adaptivity gap. Consider again the failure of edge e'. Whereas the effective flow variables for edge e' shall be equal to zero, correctly capturing the requirement that no flow shall occur along a failed edge, the effective flow variables for all other edges e will only be restricted within the desirable bounds

$$0 \leq f^{b0}_e + f^b_{e,e'} \leq u_e + p_e \quad \forall\, e \in E \setminus \{e'\}$$
$$0 \leq f^{f0}_e + f^f_{e,e'} \leq u_e + p_e \quad \forall\, e \in E \setminus \{e'\}. \quad (10)$$

Consequently, picking values for these variables does not restrict the range of possible first-stage decisions; once the latter have been selected, each of the effective flow variables may be picked perfectly so as to minimize the second-stage costs given the applicable failure scenario, resulting in the absence of an adaptivity gap in the final solution. □

Proposition 2. *For the robust network design problem defined in Formulation (3) under the uncertainty set defined in (2) with $k \geq 2$, and given the following two mild assumptions that*

1. *this uncertainty set admits at least one scenario in which two edges have failed, and*
2. *a feasible network flow solution cannot be gained by solely utilizing edges that are never allowed to fail in the scenarios admitted by this uncertainty set,*

using affine decision rules of the form (4) will never yield a feasible solution, even if one exists.

Proof. We shall first prove the proposition for the setting $k = 2$. In this case, the uncertainty set encompasses all scenarios admitted in the $k = 1$ case, as well as additional scenarios that correspond to two edges having failed at the same time. Let $E^1 \subseteq E$ be the subset of edges that are allowed to fail alone, and let $E^2 \subseteq E^1 \times E^1$ be the subset of pairs of edges that are allowed to have failed at the same time, in some scenario admitted by the applicable set. Note how the proposition requires in its first assumption that $|E^2| \neq 0$.

Let a pair of edges $(e', e'') \in E^2$, and note how this also implies that $e' \in E^1$. Hence, any robust feasible solution should satisfy the upper bounds of the flow variables of edge e' for both the scenario where this edge has failed alone, as well as the scenario where it has failed in conjunction with edge e''; that is,

$$\begin{aligned} f^{b0}_{e'} + f^b_{e',e'} &= 0 \\ f^{f0}_{e'} + f^f_{e',e'} &= 0 \\ f^{b0}_{e'} + f^b_{e',e'} + f^b_{e',e''} &= 0 \\ f^{f0}_{e'} + f^f_{e',e'} + f^f_{e',e''} &= 0. \end{aligned} \quad (11)$$

Combining the above, we deduce that $f^b_{e',e''} = 0$ and $f^f_{e',e''} = 0$, which should hold true for any edge e'' that may have failed at the same time with edge e'. It immediately follows that affine decision rules for the two directional flows along any edge $e \in E^1$ must be restricted to the form

$$\begin{aligned} f^b_e &\leftarrow f^{b0}_e (1 - \xi_e) \\ f^f_e &\leftarrow f^{f0}_e (1 - \xi_e). \end{aligned} \quad (12)$$

We shall now characterize the form of decision rules that are possible for edges $e \in E \backslash E^1$, i.e., edges that are not allowed to fail in any scenario we wish to insure against, which according to the definition of our uncertainty set (2), are exclusively the edges that are adjacent to terminal nodes of degree one. Let node $t \in V_T$ be such a terminal node with degree $D_t = 1$, and let e be its only adjacent edge (w.l.o.g., consider the edge e defined with node t being its destination node, i.e., $d(e) = t$). Applying the robust demand satisfaction constraint (7) for node t yields

$$\left(f^{f0}_e - f^{b0}_e \right) + \sum_{l \in E^1} \xi_l \left(f^f_{e,l} - f^b_{e,l} \right) = b_t \quad \forall\, \xi \in \Xi. \quad (13)$$

This constraint needs to remain feasible for the nominal scenario when no edge fails, as well as any scenario where one or more

edges from within the set E^1 fail. It is easy to show that these conditions result into the restrictions $f_e^{f0} - f_e^{b0} = b_t$ and $f_{e,l}^{f} - f_{e,l}^{b} = 0$, for all $l \in E^1$, where the latter relationships can be further reduced to $f_{e,l}^{f} = f_{e,l}^{b} = 0$ without loss of network flow feasibility. In summary, the affine decision rules for flows along edges e that are not allowed to fail must be restricted to only their constant, non-adjustable parts, f_e^{f0} and f_e^{b0}. Noting also the fact that, for any edge $e \in E \backslash E^1$, $\xi_e = 0$ applies for all scenarios admitted by the uncertainty set (by definition), we remark that the form of the affine decision rules presented in (12) in fact applies for these edges as well. Hence, the form applies for all edges $e \in E$.

Substituting the restricted functional forms (12) into the robust demand satisfaction constraint (7) for node $d(e')$, for some $e' \in E^1$, yields

$$\sum_{e:d(e)=d(e')} \left(f_e^{f0}(1-\xi_e) - f_e^{b0}(1-\xi_e)\right) + \sum_{e:o(e)=d(e')} \left(f_e^{b0}(1-\xi_e) - f_e^{f0}(1-\xi_e)\right) = b_{d(e')} \quad \forall\, \xi \in \Xi. \tag{14}$$

Clearly, this constraint removes much of our flexibility in deciding the flows for e'. More specifically, the above constraint must be satisfied for the nominal scenario where no edge fails,

$$\sum_{e:d(e)=d(e')} \left(f_e^{f0} - f_e^{b0}\right) + \sum_{e:o(e)=d(e')} \left(f_e^{b0} - f_e^{f0}\right) = b_{d(e')}, \tag{15}$$

as well as for the scenario where the edge e' fails alone,

$$\sum_{e:d(e)=d(e')} \left(f_e^{f0} - f_e^{b0}\right) - \left(f_{e'}^{f0} - f_{e'}^{b0}\right) + \sum_{e:o(e)=d(e')} \left(f_e^{b0} - f_e^{f0}\right) = b_{d(e')}. \tag{16}$$

Combining the above, we deduce that $f_{e'}^{f0} - f_{e'}^{b0} = 0$, which can be further reduced to $f_{e'}^{f0} = f_{e'}^{b0} = 0$ without loss of network flow feasibility.

This result means that affine decision rules cannot route net flow through any edge that can potentially fail in a scenario under consideration. To that end, given the proposition's second assumption, which states that a network flow solution satisfying all demands may not be gained by utilizing only edges that are not allowed to fail, affine decision rules cannot produce a robust feasible solution for the setting of $k = 2$. It trivially follows that this result holds also for the setting $k \geq 3$, as the uncertainty sets in those instances shall be supersets of the $k = 2$ uncertainty set. □

We highlight that the two assumptions stated in Proposition 2 are indeed very mild, since they both imply applicable network superstructures of very limited nature. In particular, if the uncertainty set does not admit any scenario in which two edges have failed, then the setting $k = 1$ is practically equivalent to $k \geq 2$, leading to the same collection of scenarios, and one should consult Proposition (1) instead. With regards to the second assumption, it would be arguably rare to consider an instance for which a feasible network flow may be gained by using only edges protected due to their being adjacent to terminal nodes of degree one. Consider, for example, a network with a star graph superstructure, with $|V| - 1$ terminal nodes of degree one and 1 central node connecting them together. All edges in this instance are protected from failure by construction of the uncertainty set; hence, the nominal deterministic solution in this case would be robust feasible (in fact, optimal), and the affine decision rules with adjustment variables attaining the value of zero would be able to reproduce it. Other similar cases can be envisioned, but these are rather extreme and, in most cases, not relevant for practical survivable network design problems.

Given the above described limitations of affine decision rules in our context, another approach must be followed. To that end, we shall next present a tailored column and constraint generation algorithm to address the general case of the network design problem under study. The proposed approach will either provide robust optimal solutions or prove robust infeasibility for any k value.

3. Column and constraint generation

Column and constraint generation (CCG) was originally introduced by Zeng and Zhao (2013), and has been shown to be an effective methodology for solving two-stage robust optimization problems. In CCG, a master problem is formulated and solved to select values for the first-stage decisions. These decisions are then passed to an oracle subproblem to determine random variable values that are significant to consider, either because they make the second stage infeasible, or because they cause the overall objective function value to be higher than what was reported by the master problem. Once the subproblem identifies such random variable realizations, the master problem is augmented with new constraints and variables to specifically account for those, and the process iterates. In this scheme, the master problem *provides a lower bound* on the two-stage optimal solution, while the subproblem allows for an upper bound to be computed. Iterations proceed until the two bounds converge, at which point the algorithm terminates with a certificate of two-stage robust optimality (a fully adaptive solution).

Unlike a Benders dual cutting-plane approach (Thiele, Terry, & Epelman, 2009), which only provides an implicit guarantee of robust feasibility, CCG works directly in the primal space and provides us with an explicit solution for the second stage. Furthermore, it can in principle handle the general case of mixed-integer recourse, although one might have to pay the price of solving increasingly larger subproblems. In the case when the second stage problem is a linear program, Zeng and Zhao (2013) demonstrate how KKT conditions may be utilized to reformulate the subproblem into a single-level problem that can be addressed directly via a linear optimization solver. This treatment is valid only when *relatively complete recourse* is assumed, meaning the subproblem is assumed to be always feasible regardless of the decisions made in the first stage or the random variable realizations. Arguably, in the original presentation of the CCG algorithm, there had been a significant emphasis on this assumption for practical implementations. However, it should be highlighted that this assumption is not so reasonable in the context of the network design problem, in which the subproblems determine flows to satisfy the demands given an edge failure scenario. If the master problem does not provide enough extra capacity on edges, then certain edge failures will cause the flow problem to be infeasible. Thus, the CCG algorithm must be broadened to account for the possibility of uncertain realizations that render the second stage infeasible in light of given first-stage decisions.

In fact, the CCG algorithm that we implement for purposes of our work differs from the original variant (Zeng & Zhao, 2013) in a few key ways. Firstly, since the relatively complete recourse is not assumed, a generalized algorithmic step is proposed to address the oracle through the use of two distinct subproblems, namely a *feasibility* subproblem to seek uncertain parameter realizations that would render the second stage infeasible, and if such realizations do not exist, an *optimality* subproblem to determine realizations that have the most detrimental impact on the overall objective function value. While this generalized CCG algorithm is presented below in the network design context, the method of utilizing both a feasibility and an optimality subproblem is applicable for a variety of two-stage robust optimization problems for which the relatively complete recourse assumption does not hold. For

example, a similar idea to employ two subproblems in the oracle was also utilized in the context of unit commitment problems (Lee, Liu, Mehrotra, & Shahidehpour, 2014) and robust power distribution networks (Lee, Liu, Mehrotra, & Bie, 2015). Secondly, while the original CCG approach utilizes KKT conditions to reformulate the second stage problems, in the following we will utilize strong linear programming duality to derive subproblems that are generally more tractable and can be solved more efficiently towards identifying the new columns and constraints to add to the master problem. In general, linear duality results in simpler second stage reformulations than the KKT-based approach, as evidenced by the fact that it has been preferred in the past for a variety of two-stage robust optimization problems solved with CCG implementations (An & Zeng, 2015; Jabr, Džafić, & Pal, 2015; Lee et al., 2015; Lee et al., 2014; Ye & Li, 2016; Zhao & Zeng, 2012). Thirdly, from a practical standpoint, our implementation avoids a common shortfall of previous approaches that pertains to the improper use of big-M constraints for linearizing bilinear terms that arise in the CCG subproblems. We will later argue why this is the case and propose an appropriate implementation remedy.

3.1. Master problem

Formulation (17) constitutes our master problem. This model seeks to determine capacities that perform the best in light of the collection of edge failure scenarios that have been identified as significant to explicitly account for.

$$
\begin{aligned}
\min_{u,f,\zeta\in\mathbb{R}} \quad & \sum_{e\in E} c_e^u u_e + \zeta \\
\text{s.t.} \quad & \sum_{e\in E} c_e^f \left(f_e^{f,l} + f_e^{b,l}\right) \le \zeta && \forall\, l \in \{0,1,\ldots,n\} \\
& f_e^{b,l} \le (1-\xi_e^{*,l})(u_e + p_e) && \forall\, e\in E,\ \forall\, l \in \{0,1,\ldots,n\} \\
& f_e^{f,l} \le (1-\xi_e^{*,l})(u_e + p_e) && \forall\, e\in E,\ \forall\, l \in \{0,1,\ldots,n\} \\
& \sum_{e:d(e)=i} \left(f_e^{f,l} - f_e^{b,l}\right) + \sum_{e:o(e)=i} \left(f_e^{b,l} - f_e^{f,l}\right) = b_i \\
& && \forall\, i\in V,\ \forall\, l \in \{0,1,\ldots,n\} \\
& u_e \ge 0 && \forall\, e\in E \\
& f_e^{b,l}, f_e^{f,l} \ge 0 && \forall\, e\in E,\ \forall\, l \in \{0,1,\ldots,n\}
\end{aligned}
\tag{17}
$$

The flow variables, $f_e^{b,l}$ and $f_e^{f,l}$, now feature extra indices l, which correspond to the iteration of the CCG algorithm for which they were generated. Hence, if n is the current iteration of the algorithm, there are a total of $n+1$ different sets of these variables and corresponding constraints that have been added in the master problem. Fixed values for the random variables, $\xi_e^{*,l}$, based on the results of the subproblems solved in prior iterations, are referenced as parameters in the master problem in order to ensure that important uncertain parameter realizations are considered in the master problem. Iteration $n=0$ is reserved for initializing the algorithm via some pre-defined realization $\xi_e^{*,0}$, which is usually the nominal case ($\xi_e^{*,0}=0,\ \forall e\in E$), or some other, easily inferred, "bad" scenario that is part of the uncertainty set. The formulation is thus initialized with variables $f_e^{b,0}$ and $f_e^{f,0}$ to represent flows for this initially considered scenario, while the corresponding solution becomes the first lower bound inside the overall algorithm. Solving the master problem also yields provisional first-stage capacities u_e^*, which are then referenced as parameters in the oracle subproblems.

3.2. Oracle

The general form of the oracle subproblem is presented in problem (18), where the minimum cost of the flows is maximized over all possible edge failure realizations allowed by the uncertainty set.

$$
\begin{aligned}
\max_{\xi\in\Xi} \quad \min_{f^b,f^f\in\mathbb{R}} \quad & \sum_{e\in E} c_e^f \left(f_e^f + f_e^b\right) \\
\text{s.t.} \quad & f_e^b \le (1-\xi_e)(u_e^* + p_e) && \forall\, e\in E \\
& f_e^f \le (1-\xi_e)(u_e^* + p_e) && \forall\, e\in E \\
& \sum_{e:d(e)=i} \left(f_e^f - f_e^b\right) + \sum_{e:o(e)=i} \left(f_e^b - f_e^f\right) = b_i && \forall\, i\in V \\
& f_e^b, f_e^f \ge 0 && \forall\, e\in E
\end{aligned}
\tag{18}
$$

The goal of solving the oracle is to identify the values of ξ_e that cause the most damage for our provisional network design, u_e^*, either by rendering the entire problem infeasible or by maximizing the second-stage costs. Once such ξ_e values are found, they are sent back to master problem as parameters $\xi_e^{*,n}$, where n is the appropriate iteration identifier. Due to the bi-level nature of the oracle, we reformulate the inner minimization by utilizing strong linear programming duality to gain a tractable, single-level formulation. However, since there may exist realizations of ξ_e that render the inner level infeasible due to our not assuming relatively complete recourse, there exists a possibility for the outer maximization to be unbounded, providing a numerical hurdle for obtaining solutions to feed back into the master problem. Furthermore, an often overlooked possibility in the CCG literature involves uncertainty realizations that render the inner minimization problem both primal- and dual-infeasible. If we were to impose the inner level's dual formulation on the outer-level decision maker, then such realizations would be implicitly restricted from the latter's search space, leading possibly to the incorrect conclusion that the provisional capacities are robust feasible, when in fact they are not. To that end, we follow an approach that addresses the oracle in two distinct steps. In the first step, we solve a *feasibility* variant of the oracle in order to determine if ξ_e values exist that would cause an infeasible second stage problem. If such realizations do exist, then the master problem is restricted accordingly so that these realizations are no longer second-stage infeasible. Otherwise, the original oracle (18) is solved (in what we later refer to as the *optimality* subproblem) to determine worst-case second-stage costs.

3.2.1. Feasibility subproblem

The original subproblem (18) is modified through the introduction of positive slack variables σ_e^b, σ_e^f, s_i^+, and s_i^-. The former two sets of variables add slack to the capacity constraints, while the latter two sets add slack to the demand equality constraints. The inner objective function is now to minimize the values of these slack variables, as shown in problem (19).

$$
\begin{aligned}
\max_{\xi\in\Xi} \quad \min_{f^b,f^f,\sigma^b,\sigma^f,s^+,s^-\in\mathbb{R}} \quad & \sum_{e\in E} \left(\sigma_e^b + \sigma_e^f\right) + \sum_{i\in V} \left(s_i^+ + s_i^-\right) \\
\text{s.t.} \quad & f_e^b - \sigma_e^b \le (1-\xi_e)(u_e^* + p_e) && \forall\, e\in E \\
& f_e^f - \sigma_e^f \le (1-\xi_e)(u_e^* + p_e) && \forall\, e\in E \\
& \sum_{e:d(e)=i} \left(f_e^f - f_e^b\right) + \sum_{e:o(e)=i} \left(f_e^b - f_e^f\right) + s_i^+ - s_i^- = b_i && \forall\, i\in V \\
& f_e^b, f_e^f, \sigma_e^b, \sigma_e^f \ge 0 && \forall\, e\in E \\
& s_i^+, s_i^- \ge 0 && \forall\, i\in V.
\end{aligned}
\tag{19}
$$

Note how this problem does possess relatively complete recourse by construction. Hence, we can apply linear duality to reformulate it into a singe-level problem. This results in (20), which constitutes a quadratic programming problem due to the presence

in the objective function of bilinear terms between the random variables ξ_e and the dual variables v_e^b and v_e^f,

$$\begin{aligned}
\max_{\substack{\xi\in\Xi \\ v^b,v^f,\mu\in\mathbb{R}}} \quad & \sum_{e\in E}(1-\xi_e)(u_e^*+p_e)v_e^b \\
& +\sum_{e\in E}(1-\xi_e)(u_e^*+p_e)v_e^f+\sum_{i\in V}b_i\mu_i \\
\text{s.t.} \quad & v_e^b+\mu_{o(e)}-\mu_{d(e)}\le 0 && \forall\ e\in E \\
& v_e^f-\mu_{o(e)}+\mu_{d(e)}\le 0 && \forall\ e\in E \\
& -1\le v_e^b, v_e^f\le 0 && \forall\ e\in E \\
& -1\le \mu_i\le 1 && \forall\ i\in V. \qquad (20)
\end{aligned}$$

Due to the fact that ξ_e are binary, the bilinear terms may be effectively eliminated from the problem by introducing new auxiliary variables z_e^b and z_e^f and by imposing that, at an optimal solution, those be equal to the products $v_e^b\xi_e$ and $v_e^f\xi_e$, respectively. A common approach to achieve this is by applying the well-known "Glover" linearization technique, which calls for utilizing big-M constraints. An alternative approach followed here is to reformulate these bilinear terms as explicit implications. The apodoses of the latter can then be enforced during the branch and bound search directly as cuts, whenever the corresponding protases are activated (see, e.g., the concept of *logical constraints* in modern mixed-integer linear solvers).

$$\begin{aligned}
\max_{\substack{\xi\in\Xi \\ v^b,v^f,\mu,z^b,z^f\in\mathbb{R}}} \quad & \sum_{e\in E}(u_e^*+p_e)(v_e^b-z_e^b) \\
& +\sum_{e\in E}(u_e^*+p_e)(v_e^f-z_e^f)+\sum_{i\in V}b_i\mu_i \\
\text{s.t.} \quad & v_e^b+\mu_{o(e)}-\mu_{d(e)}\le 0 && \forall\ e\in E \\
& v_e^f-\mu_{o(e)}+\mu_{d(e)}\le 0 && \forall\ e\in E \\
& \xi_e=0\Rightarrow z_e^b=0 && \forall\ e\in E \\
& \xi_e=1\Rightarrow z_e^b=v_e^b && \forall\ e\in E \\
& \xi_e=0\Rightarrow z_e^f=0 && \forall\ e\in E \\
& \xi_e=1\Rightarrow z_e^f=v_e^f && \forall\ e\in E \\
& -1\le v_e^b, v_e^f\le 0 && \forall\ e\in E \\
& -1\le \mu_i\le 1 && \forall\ i\in V \qquad (21)
\end{aligned}$$

If the optimal objective value of problem (21) is nonzero, then the chosen values of u^* are robust infeasible, and the optimal ξ_e solution stemming from this problem can be used to further restrict our first stage decisions in the master problem. In doing so, the values of ξ_e^* representing the optimal solution to problem (21) are stored and explicitly used to generate new rows in the master problem, Formulation (17). However, if the optimal objective is zero, then the current solution from the master problem is robust feasible, and the original form of the subproblem – now referred to as the optimality subproblem – must be solved to determine true worst-case costs of the provisional first-stage decisions.

3.2.2. Optimality subproblem

The optimality subproblem is utilized after we have solved the feasibility subproblem and have confirmed that the master problem has selected a robust feasible design. Thus, an upper bound on the overall two-stage problem may be calculated by finding the worst-case second-stage flow costs. Since the inner minimization is linear in the flow variables f_e^b and f_e^f, strong linear programming duality may again be used to reformulate the original subproblem (18), as follows.

$$\begin{aligned}
\max_{\substack{\xi\in\Xi \\ v^b,v^f,\mu\in\mathbb{R}}} \quad & \sum_{e\in E}(1-\xi_e)(u_e^*+p_e)v_e^b \\
& +\sum_{e\in E}(1-\xi_e)(u_e^*+p_e)v_e^f+\sum_{i\in V}b_i\mu_i \\
\text{s.t.} \quad & v_e^b+\mu_{o(e)}-\mu_{d(e)}\le c_e^f && \forall\ e\in E \\
& v_e^f-\mu_{o(e)}+\mu_{d(e)}\le c_e^f && \forall\ e\in E \\
& v_e^b, v_e^f\le 0 && \forall\ e\in E \qquad (22)
\end{aligned}$$

Again, the resulting bilinear terms $\xi_e v_e^b$ and $\xi_e v_e^f$ must be linearized. However, we should highlight that, unlike the case of the feasibility subproblem, the utilization of big-M constraints here would not be appropriate. This is due to the fact that proper selection of the big-M coefficients requires the knowledge of valid bounds on the dual variables, which are generally not available in the case of the optimality subproblem. In fact, the use of arbitrary big-M values poses risks onto the rigorousness of the solutions. More specifically, solving an overly restricted subproblem constitutes an effective relaxation of the overall two-stage problem such that the first-stage decisions are not properly evaluated in terms of the worst-case total costs. Thus, the choice of a large but not rigorous big-M coefficient in this context has the potential to admit first-stage decisions which are suboptimal in the robust sense. In order to avoid this situation, it is imperative that the bilinear products are handled via explicit implication constraints, which are then implemented as logical constraints in the employed mixed-integer linear optimization packages.

$$\begin{aligned}
\max_{\substack{\xi\in\Xi \\ v^b,v^f,\mu,z^b,z^f\in\mathbb{R}}} \quad & \sum_{e\in E}(u_e^*+p_e)(v_e^b-z_e^b) \\
& +\sum_{e\in E}(u_e^*+p_e)(v_e^f-z_e^f)+\sum_{i\in V}b_i\mu_i \\
\text{s.t.} \quad & v_e^b+\mu_{o(e)}-\mu_{d(e)}\le c_e^f && \forall\ e\in E \\
& v_e^f-\mu_{o(e)}+\mu_{d(e)}\le c_e^f && \forall\ e\in E \\
& \xi_e=0\Rightarrow z_e^b=0 && \forall\ e\in E \\
& \xi_e=1\Rightarrow z_e^b=v_e^b && \forall\ e\in E \\
& \xi_e=0\Rightarrow z_e^f=0 && \forall\ e\in E \\
& \xi_e=1\Rightarrow z_e^f=v_e^f && \forall\ e\in E \\
& v_e^b, v_e^f\le 0 && \forall\ e\in E \qquad (23)
\end{aligned}$$

The optimal ξ_e solution stemming from the optimality subproblem corresponds to the edge failure scenario that imposes the highest possible costs to route flows given the chosen capacities u_e^*. This scenario shall then be incorporated in the master problem so as to further restrict the first stage decisions, by adding new sets of variables and constraints appearing in Formulation (17) as $\xi_e^{*,l}$ in the second and third constraint blocks, where l represents a counter for the current iteration of the algorithm (see details in next section).

3.3. Full algorithm

Now that we have defined the master and two subproblem formulations, we can outline in more detail the CCG algorithm that we developed so as to address the robust network design problem under study. Given a choice of k and an optimality tolerance ε, the algorithm below will either yield a robust optimal solution, or a certificate that the instance is robust infeasible for k (or more) edge failures.

1. Initialize optimal first-stage decisions u_e^{opt} to empty, bounds $UB=+\infty$ and $LB=-\infty$, iteration counter $n=0$, and set up the nominal realization as $\xi_e^{*,0}=0$ for all $e\in E$.

2. Solve the master problem (17).
   (a) If the master problem is infeasible, return. The instance is robust infeasible.
   (b) Obtain provisional first-stage decisions u_e^*, and set LB equal to the master problem's optimal objective value.
   (c) If $UB - LB \leq \varepsilon$, return. The current solution u_e^{opt} is robust optimal.
3. Solve the feasibility subproblem (21).
   (a) If the optimal objective function value is zero, then go to Step 4.
   (b) Set $\xi_e^{*,n+1}$ in the master problem to be equal to the feasibility subproblem's optimal ξ_e values, and go to Step 5.
4. Solve the optimality subproblem (23), and let ϕ be its optimal objective value.
   (a) If $\sum_{e\in E} c_e^u u_e^* + \phi < UB$, then update incumbent $u_e^{\text{opt}} = u_e^*$ and set $UB = \sum_{e\in E} c_e^u u_e^* + \phi$.
   (b) If $UB - LB \leq \varepsilon$, return. The current solution u_e^{opt} is robust optimal.
   (c) Set $\xi_e^{*,n+1}$ in the master problem to be equal to the optimality subproblem's optimal ξ_e values.
5. Iterate counter $n \leftarrow n+1$, and go to Step 2.

4. Computational results

The proposed CCG algorithm from Section 3 was implemented and tested on instances from the Survivable Network Design Library (SNDLib) (Orlowski et al., 2010). SNDLib conveniently provides base graphs for many different instances, but since these are constructed for a telecommunications framework, each demand in the SNDLib networks is associated with a specific source node and target node. In order to adapt the instances to the current single-commodity network problem, the demands are aggregated as if there is only one commodity. More specifically, given SNDLib data β_{jk} to denote demands from a source node $j \in V$ to a destination node $k \in V$, the demand b_i will account for any demands with node i as target less any demands with node i as source; that is,

$$b_i := \sum_{j\in V} \beta_{ji} - \sum_{k\in V} \beta_{ik}. \tag{24}$$

The remaining necessary parameters, such as pre-installed capacities and costs, were extracted from the SNDLib database as follows. Pre-installed capacities p_e are directly listed in the database for every edge, and those values were adopted (many datasets simply list $p_e = 0$). Regarding costs, we note that the edges in the SNDLib datasets are listed with an associated "module capacity" and "module cost." As the capacities in this formulation are continuous rather than discrete, the cost per capacity c_e^u was simply calculated as a ratio between the module cost and the module capacity. Furthermore, since flow costs c_e^f are not provided in the SNDLib context, we define them as a fraction of the capacity costs, i.e., $c_e^f = \rho\, c_e^u$, where ρ is a fixed parameter to control the relative influence of each of the two types of costs in the objective function. In our computational experiments, we used the settings $\rho \in \{0.5, 1.0, 2.0\}$. It is also worth noting that, in some cases, multiple pairs of module costs and capacities are provided; as a rule, the first pair in the order listed is always the one adopted to define the cost parameters for our study. In the rare case that no module capacity and cost were provided for an edge, the capacity variable u_e is fixed to zero, and only pre-installed capacity may be utilized for that edge (the pre-installed capacity in these instances can still be subject to an edge failure).

In total, 23 network datasets were utilized from SNDLib to provide a variety of different sized and shaped initial graphs.[5] In each case, all edges are assumed to be bi-directional, such that flow could be chosen in either direction. However, as the initial graphs were analyzed, it became clear that some preprocessing simplifications could be made to the instances in order to reduce the complexity of the problems. For instance, if a node has a demand of zero and degree two, with the same pre-installed capacity on each edge, the two edges connecting to this node may be effectively represented as one common edge, with the intermediate node removed from the problem formulation. This and other similar observations were used to construct a preprocessing algorithm for the networks. This algorithm is presented in Appendix B.

All optimization models in this work were solved using CPLEX 12.7.0 via the CPLEX Callable Library (C API) (IBM, 2016). All runs were performed in the Tiger Supercomputing Cluster at Princeton University, and were allocated two 2.6 gigahertz Sandybridge cores, each with 4 gigabytes of RAM, allowing CPLEX to utilize two threads. The MIP absolute gap tolerance was set to 10^{-7} and the MIP integrality tolerance was set to 0.

For reference, Table 1 presents the input network sizes and the optimal objective function values for the deterministic versions of these instances ($k = 0$). We highlight that optimizing the deterministic instances was very tractable, in general, as all these runs concluded in less than 0.06 wall clock seconds.

Table 1
Network superstructure sizes and deterministic optimal costs (objective function values) for various ρ settings.

Network	$\|V\|$	$\|E\|$	$\rho = 0.5$	$\rho = 1.0$	$\rho = 2.0$
abilene	12	15	51384.72	69367.03	105328.93
atlanta	15	22	7782567.50	15399035.00	30406980.00
brain	161	166	2146.77	2862.36	4293.54
dfn-bwin	10	45	16139.41	21519.21	32278.82
dfn-gwin	11	46	3583.78	4778.37	7167.56
di-yuan	11	42	3447150.00	4596200.00	6894300.00
france	25	45	3890.16	5186.88	7780.32
geant	22	36	72718.43	96957.90	145436.85
germany50	50	88	379939.13	506585.50	759878.25
giul39	39	86	28.92	38.56	57.84
india35	35	80	1171.20	1561.60	2342.40
newyork	16	49	13090.80	17454.40	26181.60
nobel-eu	28	41	808680.00	1078240.00	1617360.00
nobel-germany	17	26	178702.50	238270.00	357405.00
nobel-us	14	21	1945173.00	2593564.00	3890346.00
norway	27	51	18009.21	24012.28	36018.42
pdh	11	34	13328338.80	17771118.40	26656677.60
pioro40	40	89	181091.19	241454.92	362182.37
polska	12	18	13121.97	17495.96	26243.94
sun	27	51	521.45	695.26	1042.89
ta1	24	51	5970763.75	7961018.33	11941527.49
ta2	65	108	8111296.78	14300028.77	25956798.12
zib54	54	80	331471.58	441962.10	662943.16

4.1. The effect of increasing number of edge failures

For each instance, the value of k was continually increased until a value was reached for which the network could no longer sustain flows meeting the applicable demand requirements at the worst case.[6] For each run (value of k), we allowed a time limit of three hours[7] for the algorithm to either return the optimal network topology and associated objective function value, or to certify that the network is robust infeasible.

[5] We remark that the SNDLib contains a total of 26 datasets. However, since the aggregation of demands for networks 'cost266', 'janos-us-ca' and 'janos-us' created networks for which $b_i = 0$ for all $i \in V$, only a trivial solution with all capacities and flows at 0 would be found, and thus these 3 datasets were excluded from consideration.

[6] The collection of detailed results can be found in the electronic "Supplementary Materials" that are provided as part of this article.

[7] Convergence within the allotted time was not an issue, as each run finished well below the time limit for every network.

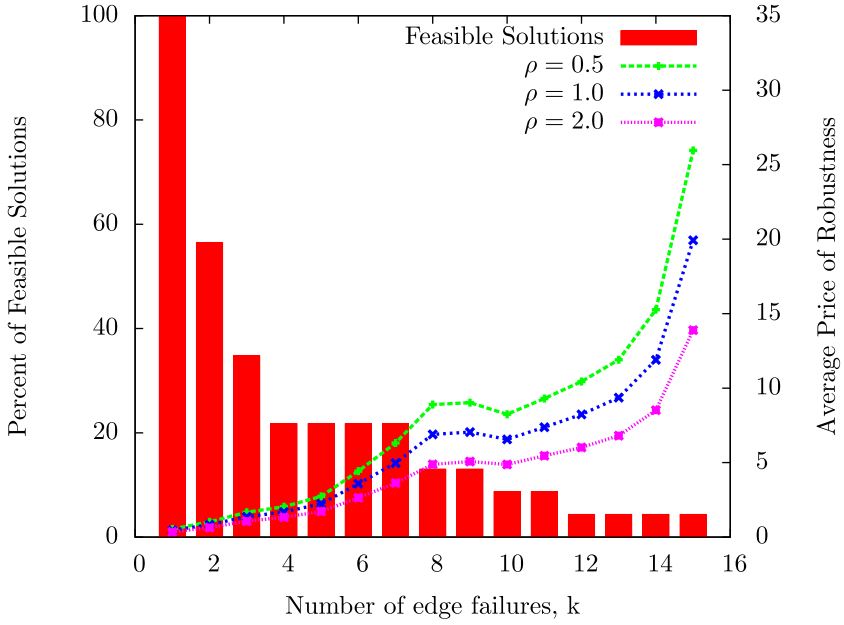

Fig. 1. Robust feasibility and average price of robustness over all networks under various ρ settings, as a function of the number of concurrent edge failures.

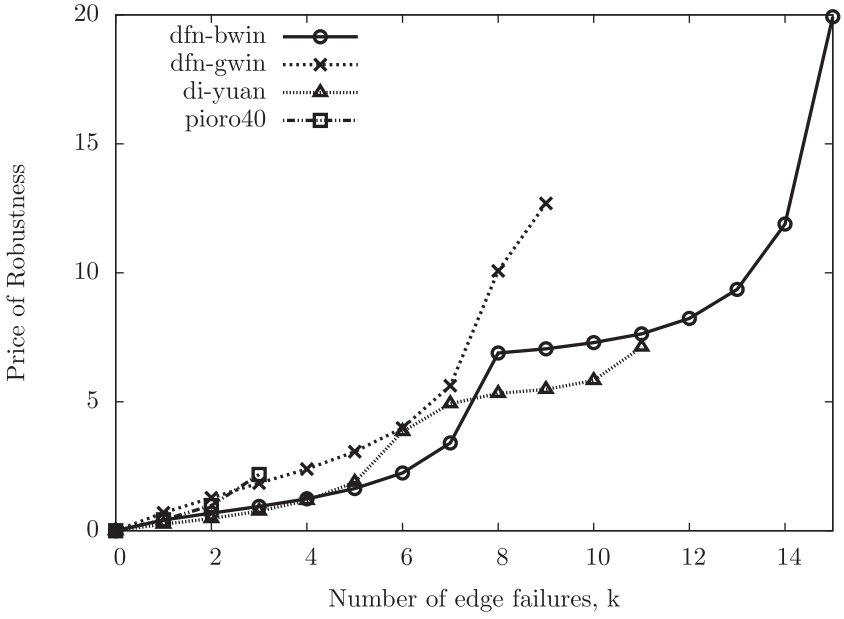

Fig. 2. Price of robustness for various representative networks from the SNDLib, as a function of number of concurrent edge failures.

As k increases, the optimal solutions become more robust at the expense of a more expensive network. The *price of robustness* (POR) can thus be calculated for each network at each value of k by comparing the total cost with the network's nominal cost (case $k = 0$, see Table 1); that is,

$$\text{POR}(k) = \frac{\text{Robust Optimal Network Cost at } k\text{- Deterministic Optimal Network Cost}}{\text{Deterministic Optimal Network Cost}}. \tag{25}$$

The percentage of networks that remain robust feasible, as well as the average price of robustness across all such networks, was calculated and plotted in Fig. 1 as a function of k. It is evident from this figure that the robust satisfaction of demand at all nodes in many networks is not possible for k as low as $k = 2$. In fact, only 13 out of the 23 networks considered in the study remain robust feasible at $k = 2$. Those networks that remain feasible incur, on average, a POR of 1.25 at $\rho = 0.5$ and a POR of 0.85 at $\rho = 2.0$. Only 5 networks remain robust feasible when $k = 5$, while only 2 of those remain feasible when $k = 10$. The network 'dfn-bwin' can handle the highest number of concurrent edge failures, remaining robust feasible up to $k = 15$.

While Fig. 1 indicates that the average POR is higher when the cost is more heavily capacity-dependent (i.e., when $\rho = 0.5$), the price of robustness is also very much dependent on the network itself. This can be inferred by the drop in average POR as we move from $k = 9$ to $k = 10$, which is due to the fact that network 'dfn-gwin' (a network with generally higher POR contribution to the overall average) became robust infeasible at that point.

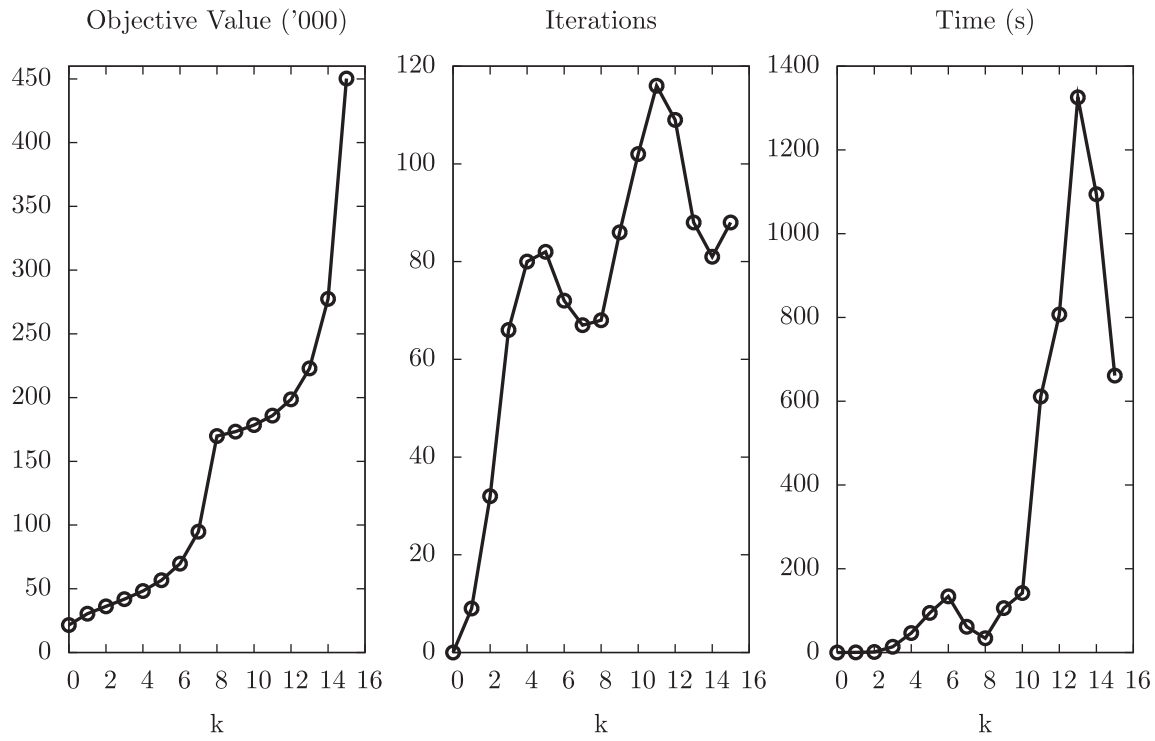

Fig. 3. Objective value, time to solve, and number of CCG iterations for network 'dfn-bwin', as a function of number of concurrent edge failures.

Network-specific POR values are plotted in Fig. 2 for a selection of representative networks. It can be observed that, while in all cases POR increases monotonically with k, the rate of increase shows high variability among different instances.

Focusing on network 'dfn-bwin' allows for a closer look at the performance of the algorithm over time and iterations, both in general terms as well as for specific k values. In particular, Fig. 3 highlights the optimal costs of 'dfn-bwin' as k increases, along with the number of iterations and time it took for the algorithm to converge. We observe that, while the runs required only a few minutes when $k \leq 10$, they got significantly longer as k increased beyond this point. The longest run required 1325 wall clock seconds, when $k = 13$, indicating that the runtime does not monotonically increase with k. This is also reflected in the average runtimes across all network instances, as shown in Table 2. Interestingly, while the number of iterations tends to increase initially in Fig. 3, there is not a significant trend or difference between the number of iterations over the final eight k values. The number of iterations also does not directly correlate to the time required for the run, indicating that much of the time requirement at larger k values may be due to specific subproblem calls rather than the increasing size of the master problem. Finally, Fig. 4 provides some insight into the behavior of the algorithm as the various iterations progress. As these plots suggest, many iterations are required before the first feasible upper bound is calculated, which can be attributed to the number of combinations of edge failures that can exist and that may cause the second stage to be infeasible. For the case when $k = 5$, about three quarters of the iterations were required before an initial upper bound was reached, and the gap was closed rather rapidly in the remaining iterations. This behavior was further amplified in the case when $k = 10$, where the gap was completely closed as soon as an upper bound was obtained after a total of 102 iterations. These observations point to the conclusion that the algorithm would greatly benefit from an early determination of a good robust feasible upper bound, which could, for example, be obtained via some heuristic.

Table 2
Average computational time (in seconds) required across each of the runs as a function of k and ρ.

k	$\rho = 0.5$	$\rho = 1.0$	$\rho = 2.0$
0	0.01	0.01	0.01
1	0.81	0.89	0.77
2	5.78	4.43	4.15
3	44.63	38.48	41.51
4	24.91	25.27	28.05
5	57.82	65.76	63.96
6	61.10	67.08	121.39
7	69.76	84.24	110.54
8	43.16	43.52	52.48
9	161.37	109.40	175.00
10	119.96	124.70	126.68
11	473.34	509.997	519.63
12	424.92	457.93	377.74
13	985.41	1325.45	1440.69
14	1151.11	1094.17	1324.14
15	689.55	661.21	653.93
16[a]	0.53	0.60	0.28

[a] The most resilient (and only still feasible at $k = 15$) network 'dfn-bwin' becomes infeasible at $k = 16$, which is fast to prove, and thus leads to a very short time entry in this row.

4.2. Comparison of affine decision rules and CCG with one edge failure

While we have already shown that the affine decision rules (ADR) approach is not applicable to the SNDlib problems when $k > 1$, its potential for the special case when $k = 1$ is of interest. Therefore, the ADR formulation was implemented for $k = 1$ and compared to the CCG algorithm. Table 3 demonstrates the relative effectiveness of both approaches in finding robust optimal solutions when $k = 1$ and for the intermediate $\rho = 1$ setting.

As expected, the optimal objective function values for both cases are exactly the same, confirming numerically that the ADR approach is exact for this case. Interestingly, however, in most

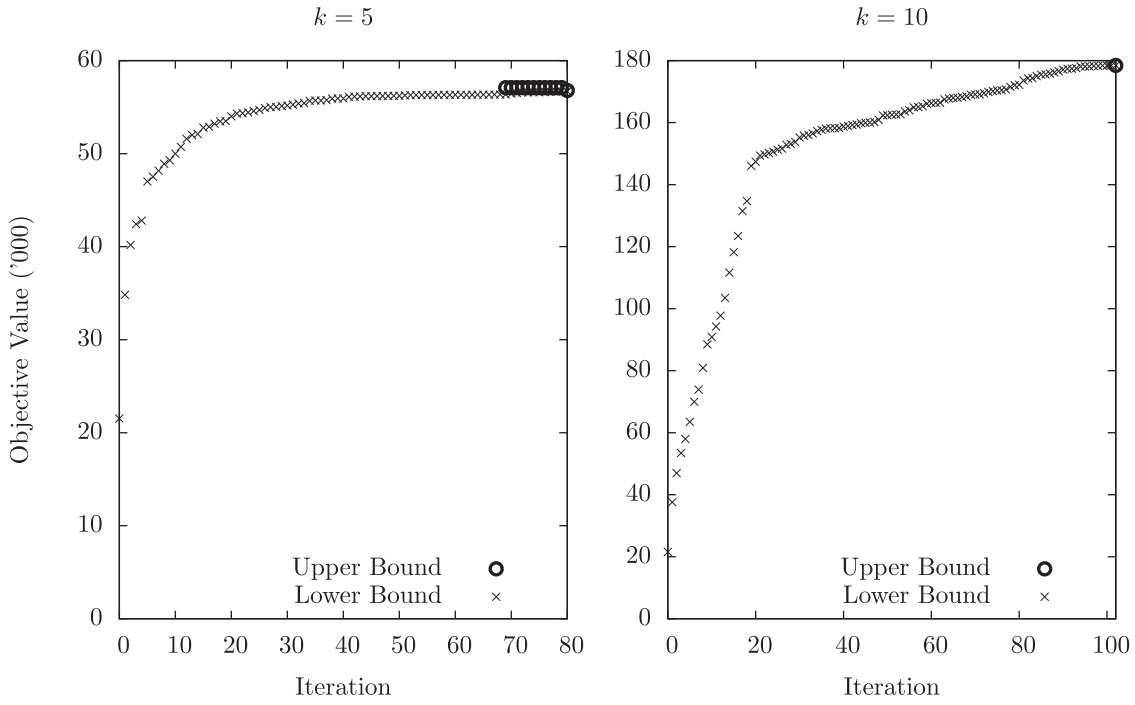

Fig. 4. Progress of upper and lower bounds for instance 'dfn-bwin' under two different settings for k.

Table 3
Comparison of computational times for the CCG algorithm versus the affine decision rules approach when $k = 1$ and $\rho = 1$.

Network	Optimal cost	CCG time (in seconds)	ADR time (in seconds)
abilene	120309.07	0.13	0.06
atlanta	18932785.00	0.08	0.23
brain	3134.30	0.28	0.74
dfn-bwin	30451.25	0.21	2.69
dfn-gwin	8099.44	0.26	4.35
di-yuan	5780085.68	0.26	2.20
france	8267.08	0.89	3.09
geant	144748.06	0.52	1.38
germany50	739454.44	3.21	21.45
giul39	53.27	3.25	68.51
india35	2278.54	1.94	36.11
newyork	24460.56	0.46	3.98
nobel-eu	1706129.61	0.53	1.98
nobel-germany	377555.62	0.23	0.51
nobel-us	4425847.75	0.12	0.26
norway	36088.09	0.54	5.79
pdh	25111336.36	0.21	1.13
pioro40	343107.85	2.87	58.82
polska	33120.88	0.09	0.06
sun	1124.84	0.68	7.43
ta1	10698220.03	0.73	7.22
ta2	16481127.50	2.03	61.19
zib54	625961.94	0.91	9.88
Average	N/A	0.89	13.00

cases the CCG algorithm finds the optimal solutions much faster than the latter. This is despite the fact that CCG requires multiple iterations, between 4 and 44 for these networks, with at least twice as many optimization calls during the course of the algorithm. It appears that the significant number of added flow adjustment variables ($\mathcal{O}(|E|^2)$) and additional constraints increases the complexity of the ADR formulations such that performing the many, albeit smaller, optimization calls in the CCG algorithm is more efficient than solving the large ADR models once. Evidently, the ability to focus only on the most important scenarios identified by the oracle is a strong benefit of the CCG approach.

5. Application on a defender-attacker paradigm

We now demonstrate how the CCG algorithm can be expanded to more complex applications of network design through the incorporation of a defender versus attacker setting (Alderson et al., 2015; Brown, Carlyle, Salmerón, & Wood, 2006; Lou & Zhang, 2011; Snyder et al., 2006; Yuan et al., 2016b; Yuan, Zhao, & Zeng, 2014), such as those that often arise in network interdiction (Avenhaus & Canty, 2009; Cormican, Morton, & Wood, 1998; Sullivan & Cole Smith, 2014; Washburn & Wood, 1995) or security (Baykal-Gürsoy, Duan, Poor, & Garnaev, 2014; Gueye, Walrand, & Anantharam, 2010) games. Rather than passively designing a network to account for potential edge failures, we can now actively seek to defend specific nodes up to a defense budget Γ_D, with $r_e \in \{0, 1\}$ representing the choice of an edge $e \in E$ being protected,

$$\sum_{e \in E} r_e \leq \Gamma_D. \tag{26}$$

The defender will here and now choose which edges cannot fail, wait and see which edges an attacker (be it a malevolent actor or a random event) causes to fail, and then make final defensive actions to reroute flows.

As a result, the uncertainty set must be extended to reflect the fact that, when $r_e = 1$, an edge $e \in E$ cannot fail. However, since r_e is explicitly in the control of the decision-maker, we warrant the use of a decision-dependent uncertainty set. Such sets have been recently proposed to allow robust optimization approaches handle uncertainty of *endogenous* nature (Lappas & Gounaris, 2018; Nohadani & Sharma, 2018; Poss, 2014). Here, we demonstrate that the column and constraint generation algorithm of Zeng and Zhao (2013) can also be applied in this context. More specifically, let a decision-dependent uncertainty set as follows:

$$\Xi_r(r) := \left\{ \xi \in \{0,1\}^{|E|} : \begin{array}{l} \sum_{e \in E} \xi_e \leq \Gamma_A \\ \xi_e \leq 1 - r_e \quad \forall e \in E \end{array} \right\}, \tag{27}$$

where Γ_A represents the attacker's budget, replacing k from the previous examples.

The random variable realization $\xi_e = 1$ still represents an edge failing, while the second constraint in the uncertainty set limits the admissible realizations for ξ such that protected edges cannot fail. We highlight that, since the defender now has a choice to immunize (at a cost) any edge, we have now lifted our original prohibition of simultaneous failure of all edges adjacent to a node, allowing such a scenario to be admitted by the uncertainty set. The new two-stage, defender-attacker-defender formulation appears below.

$$
\begin{aligned}
\min_{u,r} \quad & \sum_{e\in E} c_e^u u_e + c_e^r r_e + \max_{\xi\in\Xi_r(r)} \min_{f} \quad \sum_{e\in E} c_e^f \left(f_e^f + f_e^b\right) \\
\text{s.t.} \quad & \sum_{e\in E} r_e \le \Gamma_D \quad \text{s.t.} \quad f_e^b \le (1-\xi_e)(u_e + p_e) && \forall\, e \in E \\
& u_e \ge 0 \quad \forall\, e \in E \qquad f_e^f \le (1-\xi_e)(u_e + p_e) && \forall\, e \in E \\
& r_e \in \{0,1\} \quad \forall\, e \in E \quad \sum_{e:d(e)=i} \left(f_e^f - f_e^b\right) \\
& \qquad\qquad + \sum_{e:o(e)=i} \left(f_e^b - f_e^f\right) = b_i && \forall\, i \in N \\
& \qquad\qquad f_e^b \ge 0 && \forall\, e \in E \\
& \qquad\qquad f_e^f \ge 0 && \forall\, e \in E
\end{aligned}
\tag{28}
$$

5.1. Augmented master problem and subproblems

As in the previous examples, we shall seek to apply a column and constraint generation algorithm, which requires appropriate adaptations of the master problem (17) and the subproblems (21) and (23). When building the master problem, we must be especially careful to take into account that an edge cannot fail, if it is made resilient; that is, if the subproblem chooses a solution ξ^* causing an edge e to fail, there must be a way to disregard this edge failure in the master problem, if $r_e = 1$. We propose achieving this through two indicator constraints that define a new total capacity x_e^l for each edge in every scenario l, which considers edge failures only if the edge is not resilient:

$$
\begin{aligned}
& r_e = 1 \rightarrow x_e^l = u_e + p_e && \forall\, e \in E, \forall\, l \in L \\
& r_e = 0 \rightarrow \; x_e^l = (1 - \xi_e^{*,l})(u_e + p_e) && \forall\, e \in E, \forall\, l \in L
\end{aligned}
\tag{29}
$$

It is easy to see that whenever an edge is protected, its capacity will not be impacted by any possible edge failures. Using these new capacity variables, the master problem can be written in Formulation (30).

$$
\begin{aligned}
\min_{u,r,f^{b,l},f^{f,l}} \quad & \sum_{e\in E} (c_e^u u_e + c_e^r r_e) + \zeta \\
\text{s.t.} \quad & \sum_{e\in E} c_e^f \left(f_e^{f,l} + f_e^{b,l}\right) \le \zeta && \forall\, l \in L \\
& \sum_{e\in E} r_e \le \Gamma_D \\
& r_e = 1 \rightarrow x_e^l = u_e + p_e && \forall\, e \in E, \forall\, l \in L \\
& r_e = 0 \rightarrow x_e^l = (1 - \xi_e^{*,l})(u_e + p_e) && \forall\, e \in E, \forall\, l \in L \\
& 0 \le f_e^{b,l}, f_e^{f,l} \le x_e^l && \forall\, e \in E, \forall\, l \in L \\
& \sum_{e\in E:d(e)=i} f_e^{f,l} - f_e^{b,l} + \sum_{e\in E:o(e)=i} f_e^{b,l} - f_e^{f,l} = b_i && \forall\, i \in N, \forall\, l \in L \\
& x_e^l \ge 0 && \forall\, e \in E, \forall\, l \in L \\
& r_e \in \{0,1\} && \forall\, e \in E \\
& u_e \ge 0 && \forall\, e \in E
\end{aligned}
\tag{30}
$$

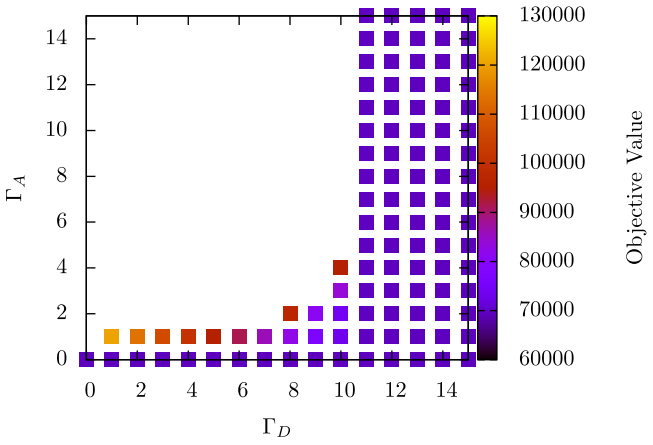

Fig. 5. Effect of the attacker and defender budgets on network feasibility for SNDLib network 'abilene'. Points do not appear when the combination of Γ_A and Γ_D is infeasible.

The general oracle is the defined as follows:

$$
\begin{aligned}
\max_{\xi\in\Xi_r^*} \quad \min_{f^b,f^f} \quad & \sum_{e\in E} c_e^f \left(f_e^f + f_e^b\right) \\
\text{s.t.} \quad & f_e^b \le (1-\xi_e)(u_e^* + p_e) && \forall\, e \in E \\
& f_e^f \le (1-\xi_e)(u_e^* + p_e) && \forall\, e \in E \\
& \sum_{e:d(e)=i} \left(f_e^f - f_e^b\right) + \sum_{e:o(e)=i} \left(f_e^b - f_e^f\right) = b_i && \forall\, i \in V \\
& f_e^b, f_e^f \ge 0 && \forall\, e \in E,
\end{aligned}
\tag{31}
$$

where Ξ_r^* represents the uncertainty set based on the optimal resilient edges r_e^* chosen in the master problem.

$$
\Xi_r^* := \left\{ \xi \in \{0,1\}^{|E|} : \begin{array}{l} \sum_{e\in E} \xi_e \le \Gamma_A \\ \xi_e \le 1 - r_e^* \quad \forall e \in E \end{array} \right\}. \tag{32}
$$

At this point, the derivation of the final subproblems follows that of Section 3, albeit with the modified uncertainty set. This is a straightforward exercise, which we shall not present in detail at the interest of brevity.

We are now ready to apply a column and constraint generation algorithm under the case of a decision dependent uncertainty set. In the master problem, the defender chooses what the capacity of each edge is and which edges should be resilient. Then via the subproblem, the attacker makes their best play to destroy the network, and the defender readjusts flows to account for this. If the feasibility subproblem reveals edge failures ξ_e^* that can make the network infeasible, then the master problem is solved again with these edge failures taken into account, adjusting which edges are made resilient and the overall edge capacities. If the problem is feasible but suboptimal, then (similarly to Section 3) the edge failures ξ_e^* are added back into the master problem to be accounted for by the defender. The algorithm iterates until the network is proven infeasible or a robust optimal solution is found.

5.2. Insights from the defender-attacker setting

For brevity, only a short discussion of results from this setting is provided. It should be noted that these problems result to an MILP master problem due to the binary decisions on edge resilience, which in general increase computational complexity and runtimes. However, it is still possible to gain interesting insights to the importance of the attacker and defender budgets, as seen in Fig. 5 for SNDlib network 'abilene'. In these runs, costs are defined such that $\rho = 1.0$ and $c_e^r := 100\, c_e^u$.

A key distinction in the defender-attacker formulation is that access to each node is no longer guaranteed by the uncertainty set. Access to each node must be ensured by the defender's actions, if their budget allows them to. Fig. 5 shows that when resilient edges are not utilized ($\Gamma_D = 0$), it is impossible to protect against an attacker; no solution exists when $\Gamma_A = 1$ and $\Gamma_D = 0$. Intuitively, this can be explained by the fact that (in the network 'abilene') there exists at least one node with a single connecting edge. When $\Gamma_D = 1$, a defense can be gained for up to one edge failure. However, a feasible network cannot be guaranteed for two simultaneous edge failures unless eight edges can be protected! Yet, by a defensive budget of 11, the entire 15 node network can be ensured to be feasible.

In this brief example, it is clear that the defender-attacker paradigm can utilize the same general column and constraint generation methodology to provide key insights into robust network design. We can now understand how many edges must be protected to guarantee robustness, and the amount at which the network can be attacked at each level of protection before the network fails. Furthermore, if costs are scaled similarly to the toy manner conducted in this section, it may also become evident that becoming robust is cheaper via making edges resilient rather than increasing edge capacity. While there is plenty more that could be discussed in the future regarding this paradigm, this section serves as one example of how the CCG methodology can be expanded to more advanced problems of robust network design.

6. Conclusions

The single-commodity network design problem under the possibility for multiple concurrent edge failures was formulated as a two-stage robust optimization problem and solved to optimality using a tailored column and constraint generation algorithm. It was shown that a column and constraint generation approach is more suited to this problem than an adjustable robust optimization formulation with affine decision rules, which is exact when only one edge failure occurs but cannot provide robust feasible solutions in most practical cases when two or more edges have failed at the same time. The column and constraint generation algorithm was tailored to this problem with two subproblems forming the oracle, providing random variable realizations that may cause infeasibility or that are most detrimental to the cost. Computational examples adapted from SNDLib demonstrate the effectiveness of this column and constraint generation algorithm. Optimal network topologies were calculated for each network up until the number of simultaneous edge failures for which no feasible topology exists for the network, with the price of robustness calculated in each of these cases. It is easy to see that this formulation can be readily extended to more complex settings, such as the defender-attacker-defender paradigm discussed briefly in this paper. The formulation can also be relaxed to allow for partial edge failures through continuous random variables, or to even allow demand infeasibility through the introduction of slack variables in the flow balance constraints. In conclusion, the results of this work demonstrate the suitability of two-stage robust optimization for designing resilient networks, as well as the potential of a well-implemented column and constraint generation algorithm for a variety of applications.

Acknowledgments

This research was conducted with Government support under and awarded by DoD, Air Force Office of Scientific Research, National Defense Science and Engineering Graduate (NDSEG) Fellowship, 32 CFR 168a, which LRM gratefully acknowledges. The work of IGK was partially supported by the National Institutes of Health (NBIB Project no. 1U01EB021956-01) and DARPA (Lagrange Program, Contract No. N66001-18-C-4031).

Appendix A. Affine decision rules formulation

The affine decision rules formulation in (5) can be addressed via standard reformulation techniques based on strong linear programming duality. More specifically, the requirement to enforce an inequality constraint for all uncertainty realizations admissible by the uncertainty set can be reformulated into a constraint-specific inner level linear optimization problem, enforcing that the constraint will be satisfied at the worst-case with respect to the uncertain parameters ξ. This inner level optimization problem is replaced with its dual problem, for which a feasible solution maintains that the original inequality constraint is satisfied. Thus, the original inequality constraints referencing parameters ξ are transformed into a set of regular linear constraints that constitute their robust counterparts. We remark that, while performing this step, we regard the random variables ξ as continuous in [0,1]. Since this is an effective relaxation of the uncertainty set, any feasible solution found will remain robust for the original uncertainty set (2). Notably, under mild assumptions on the problem input, including the setting $k = 1$, the uncertainty set is totally unimodular in ξ, leading the proposed continuous relaxation to be, in fact, exact.

In addition, the requirement to enforce an equality constraint for all uncertainty realizations admissible by the uncertainty set can be replaced with an equivalent set of linear conditions derived via coefficient matching. The conditions essentially enforce that the coefficient of every uncertain parameter referenced in the original equality constraint, as well as the latter's non ξ-dependent term, be zero at any feasible solution. We remark that these conditions constitute an exact reformulation of the original constraint for all settings of k, since for $k = 0$ they default to the nominal deterministic constraints, while for $k \geq 1$ the uncertainty set's construction ensures that the number of distinct scenarios admissible by the set is always greater than the number of uncertain parameters that may potentially obtain a nonzero value. After the reformulations described above, the final affine decision rules approximation of our original two-stage robust optimization problem is shown in Formulation (33). In this model, variables λ, μ and ν are appropriately defined dual variables that were introduced during the derivation process. Note how the final formulation constitutes a linear program that can be solved via standard linear optimization solvers.

$$\min_{u,f,\tau,\lambda,\mu,\nu\in\mathbb{R}} \quad \sum_{e\in E} c_e^u u_e + \tau$$

$$\begin{aligned}
\text{s.t. } & -\tau + \sum_{e\in E} c_e^f \left(f_e^{f0} + f_e^{b0}\right) + \lambda^1 k + \sum_{i\in V_T} (D_i - 1)\mu_i^1 + \sum_{e\in E} \nu_e^1 \le 0 \\
& \lambda^1 + \mu_{d(l)}^1 + \mu_{o(l)}^1 + \nu_l^1 - \sum_{e\in E} c_e^f \left(f_{e,l}^f + f_{e,l}^b\right) \ge 0 && \forall\ l \in E \\
& f_e^{b0} - u_e + \lambda^{e,2} k + \sum_{i\in V_T} (D_i - 1)\mu_i^{e,2} + \sum_{l\in E} \nu_l^{e,2} \le p_e && \forall\ e \in E \\
& \lambda^{e,2} + \mu_{d(l)}^{e,2} + \mu_{o(l)}^{e,2} + \nu_l^{e,2} - f_{e,l}^b \ge 0 && \forall\ l \in E, \forall\ e \in E, l \ne e \\
& \lambda^{e,2} + \mu_{d(e)}^{e,2} + \mu_{o(e)}^{e,2} + \nu_e^{e,2} - f_{e,e}^b - u_e \ge p_e && \forall\ e \in E \\
& f_e^{f0} - u_e + \lambda^{e,3} k + \sum_{i\in V_T} (D_i - 1)\mu_i^{e,3} + \sum_{l\in E} \nu_l^{e,3} \le p_e && \forall\ e \in E \\
& \lambda^{e,3} + \mu_{d(l)}^{e,3} + \mu_{o(l)}^{e,3} + \nu_l^{e,3} - f_{e,l}^f \ge 0 && \forall\ l \in E, \forall\ e \in E, l \ne e \\
& \lambda^{e,3} + \mu_{d(e)}^{e,3} + \mu_{o(e)}^{e,3} + \nu_e^{e,3} - f_{e,e}^f - u_e \ge p_e && \forall\ e \in E \\
& -f_e^{b0} + \lambda^{e,4} k + \sum_{i\in V_T} (D_i - 1)\mu_i^{e,4} + \sum_{l\in E} \nu_l^{e,4} \le 0 && \forall\ e \in E \\
& \lambda^{e,4} + \mu_{d(l)}^{e,4} + \mu_{o(l)}^{e,4} + \nu_l^{e,4} + f_{e,l}^b \ge 0 && \forall\ l \in E,\ \forall\ e \in E \\
& -f_e^{f0} + \lambda^{e,5} k + \sum_{i\in V_T} (D_i - 1)\mu_i^{e,5} + \sum_{l\in E} \nu_l^{e,5} \le 0 && \forall\ e \in E \\
& \lambda^{e,5} + \mu_{d(l)}^{e,5} + \mu_{o(l)}^{e,5} + \nu_l^{e,5} + f_{e,l}^f \ge 0 && \forall\ l \in E,\ \forall\ e \in E \\
& \sum_{e:d(e)=i} \left(f_e^{f0} - f_e^{b0}\right) + \sum_{e:o(e)=i} \left(f_e^{b0} - f_e^{f0}\right) = b_i && \forall\ i \in V \\
& \sum_{e:d(e)=i} \left(f_{e,l}^f - f_{e,l}^b\right) + \sum_{e:o(e)=i} \left(f_{e,l}^b - f_{e,l}^f\right) = 0 && \forall\ l \in E,\ \forall\ i \in V \\
& u_e \ge 0 && \forall\ e \in E \\
& \lambda^1 \ge 0 \\
& \lambda^{e,j} \ge 0 && \forall\ e \in E, \forall\ j \in \{2,3,4,5\} \\
& \mu_i^1 \ge 0 && \forall\ i \in V_T \\
& \mu_i^1 = 0 && \forall\ i \in V \setminus V_T \\
& \mu_i^{e,j} \ge 0 && \forall\ i \in V_T, \forall\ e \in E, \forall\ j \in \{2,3,4,5\} \\
& \mu_i^{e,j} = 0 && \forall\ i \in V \setminus V_T, \forall\ e \in E, \forall\ j \in \{2,3,4,5\} \\
& \nu_l^1 \ge 0 && \forall\ l \in E \\
& \nu_l^{e,j} \ge 0 && \forall\ l \in E, \forall\ e \in E, \forall\ j \in \{2,3,4,5\}
\end{aligned} \tag{33}$$

Appendix B. Preprocessing of initial graphs

Section 4 outlines how graphs provided from the Survivable Network Design Library are adapted from a telecommunications context to a single-commodity context for use with the formulations in this manuscript. After adapting the problem to this context, it became clear that in many cases, a variety of simplifications could be made to provide problem instances with fewer nodes or edges to consider, and consequently, formulations with fewer variables and constraints. Specifically, the following rules facilitate such preprocessing of the input data, and can be applied to simplify our network design instances. Note that application of this procedure is meant to be recursive; that is, if any simplifications are adopted, the whole procedure should be applied again on the simplified graph until no more simplifications are possible.

1. A node with degree one and demand zero may be removed from the graph, along with its adjacent edge, since flow to this node will never be required in an optimal network design, and since nature will never have an incentive to damage this edge.
2. If the graph contains a loop that involves a node i with degree no less than three, for which all nodes in the loop other than node i have degree two and demand zero, then flow will never enter this loop in an optimal solution, while nature will never have an incentive to damage any of its edges. Hence, each of the nodes and edges in the loop may be removed, with the exception of node i, which shall remain connected to the rest of the graph, albeit with a degree that is now reduced by two.

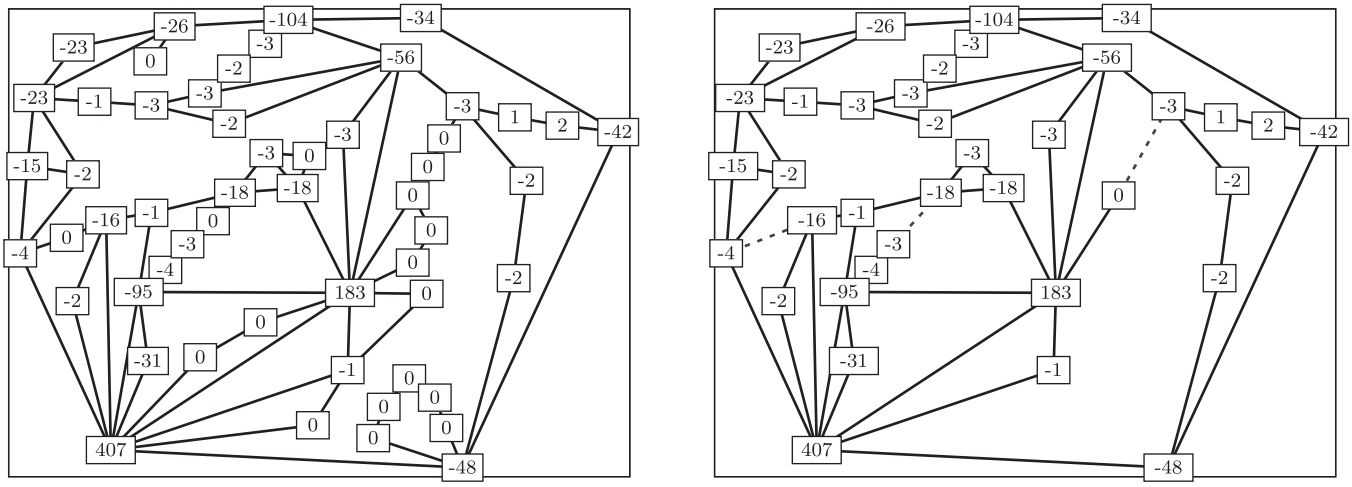

Fig. B1. Initial and simplified graphs of problem 'zib54'. Each node is labeled with its demand. The dashed lines indicate new edges gained through the removal of nodes with degree two and demand zero.

3. A node with degree two and demand zero may be removed from the graph, if its two adjacent edges feature the same pre-installed capacity. The two edges may then be replaced with one new edge, since routing flow through this transshipment node requires that the flows into and out of the node be equal. Furthermore, nature would have no incentive to damage both edges concurrently, while damaging one of the original edges is effectively equivalent to damaging the new combined edge. While the cost to install new capacity on the new edge shall be equal to the sum of costs applicable for the two replaced edges, the pre-installed capacity of the new edge shall be equal to their common original value.[8]

The preprocessing of the graphs can allow certain problems to be substantially simplified. For example, Fig. B.1 showcases the case of problem 'zib54', which featured a reduction in number of nodes from 54 to 37 and a reduction in total number of edges from 80 to 62. Note how a loop extending from the bottom right node of the initial graph was completely removed after this simplification.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.ejor.2019.06.021.

References

Alderson, D. L., Brown, G. G., & Carlyle, W. M. (2015). Operational models of infrastructure resilience. *Risk Analysis, 35*(4), 562–586. doi:10.1111/risa.12333.

Altın, A., Amaldi, E., Belotti, P., & Pınar, M. Ç. (2007). Provisioning virtual private networks under traffic uncertainty. *Networks, 49*(1), 100–115. doi:10.1002/net.20145.

Álvaro Lorca, Sun, X. A., Litvinov, E., & Zheng, T. (2016). Multistage adaptive robust optimization for the unit commitment problem. *Operations Research, 64*(1), 32–51. doi:10.1287/opre.2015.1456.

Álvarez Miranda, E., Cacchiani, V., Lodi, A., Parriani, T., & Schmidt, D. R. (2014). Single-commodity robust network design problem: Complexity, instances and heuristic solutions. *European Journal of Operational Research, 238*(3), 711–723. doi:10.1016/j.ejor.2014.04.023.

An, Y., & Zeng, B. (2015). Exploring the modeling capacity of two-stage robust optimization: Variants of robust unit commitment model. *IEEE Transactions on Power Systems, 30*(1), 109–122. doi:10.1109/TPWRS.2014.2320880.

Aneja, Y. P., Chandrasekaran, R., & Nair, K. P. K. (2001). Maximizing residual flow under an arc destruction. *Networks, 38*(4), 194–198. doi:10.1002/net.10001.

Atamtürk, A., & Zhang, M. (2007). Two-stage robust network flow and design under demand uncertainty. *Operations Research, 55*(4), 662–673. doi:10.1287/opre.1070.0428.

Avenhaus, R., & Canty, M. J. (2009). Inspection games. *Encyclopedia of complexity and systems science*, 4855–4868.

Babonneau, F., Vial, J.-P., Klopfenstein, O., & Ouorou, A. (2013). Robust capacity assignment solutions for telecommunications networks with uncertain demands. *Networks, 62*(4), 255–272. doi:10.1002/net.21515.

Baykal-Gürsoy, M., Duan, Z., Poor, H. V., & Garnaev, A. (2014). Infrastructure security games. *European Journal of Operational Research, 239*(2), 469–478. doi:10.1016/j.ejor.2014.04.033.

Ben-Tal, A., El Ghaoui, L., & Nemirovski, A. (2009). *Robust optimization*. Princeton University Press.

Ben-Tal, A., Goryashko, A., Guslitzer, E., & Nemirovski, A. (2003). Adjustable robust solutions of uncertain linear programs. *Mathematical Programming, 99*(2), 351–376.

Bertsimas, D., Iancu, D. A., & Parrilo, P. A. (2010). Optimality of affine policies in multistage robust optimization. *Mathematics of Operations Research, 35*(2), 363–394. doi:10.1287/moor.1100.0444.

Bertsimas, D., Nasrabadi, E., & Stiller, S. (2013). Robust and adaptive network flows. *Operations Research, 61*(5), 1218–1242. doi:10.1287/opre.2013.1200.

Bertsimas, D., & Sim, M. (2003). Robust discrete optimization and network flows. *Mathematical Programming, 98*(1), 49–71. doi:10.1007/s10107-003-0396-4.

Bertsimas, D., & Sim, M. (2004). The price of robustness. *Operations research, 52*(1), 35–53.

Boginski, V. L., Commander, C. W., & Turko, T. (2009). Polynomial-time identification of robust network flows under uncertain arc failures. *Optimization Letters, 3*(3), 461–473. doi:10.1007/s11590-009-0125-x.

Brown, G., Carlyle, M., Salmerón, J., & Wood, K. (2006). Defending critical infrastructure. *Interfaces, 36*(6), 530–544.

Cacchiani, V., Jünger, M., Liers, F., Lodi, A., & Schmidt, D. R. (2016). Single-commodity robust network design with finite and hose demand sets. *Mathematical Programming, 157*(1), 297–342. doi:10.1007/s10107-016-0991-9.

Chen, R. L.-Y., & Phillips, C. A. (2013). k-edge failure resilient network design. *Electronic Notes in Discrete Mathematics, 41*(Supplement C), 375–382. doi:10.1016/j.endm.2013.05.115.

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

Gong, J., Garcia, D. J., & You, F. (2016). Unraveling optimal biomass processing routes from bioconversion product and process networks under uncertainty: An adaptive robust optimization approach. *ACS Sustainable Chemistry & Engineering, 4*(6), 3160–3173. doi:10.1021/acssuschemeng.6b00188.

Gong, J., & You, F. (2017). Optimal processing network design under uncertainty for producing fuels and value-added bioproducts from microalgae: Two-stage adaptive robust mixed integer fractional programming model and computationally efficient solution algorithm. *AIChE Journal, 63*(2), 582–600. doi:10.1002/aic.15370.

Gounaris, C., & Schmidt, D. (2019). Generalized hose uncertainty in single-commodity robust network design. *Optimization Letters*. Articles In Advance. doi: 10.1007/s11590-019-01427-8.

Gounaris, C. E., Rajendran, K., Kevrekidis, I. G., & Floudas, C. A. (2011). Generation of networks with prescribed degree-dependent clustering. *Optimization Letters, 5*(3), 435–451. doi:10.1007/s11590-011-0319-x.

Gounaris, C. E., Rajendran, K., Kevrekidis, I. G., & Floudas, C. A. (2016). Designing networks: A mixed-integer linear optimization approach. *Networks, 68*(4), 283–301. doi:10.1002/net.21699.

Gounaris, C. E., Wiesemann, W., & Floudas, C. A. (2013). The robust capacitated vehicle routing problem under demand uncertainty. *Operations Research, 61*(3), 677–693. doi:10.1287/opre.1120.1136.

[8] If the pre-installed capacities on the two edges are not the same, the replacement dictated by this rule is not allowed, as the new installed capacity for one edge may be larger than the other.

Gueye, A., Walrand, J. C., & Anantharam, V. (2010). Design of network topology in an adversarial environment. In T. Alpcan, L. Buttyán, & J. S. Baras (Eds.), *Decision and game theory for security* (pp. 1–20). Berlin, Heidelberg: Springer Berlin Heidelberg.

IBM (2016). *IBM ILOG CPLEX optimization studio CPLEX user's manual: Version 12 release 7*. https://www.ibm.com/support/knowledgecenter/SSSA5P_12.7.0/ilog.odms.studio.help/pdf/usrcplex.pdf.

Jabr, R. A., Džafić, I., & Pal, B. C. (2015). Robust optimization of storage investment on transmission networks. *IEEE Transactions on Power Systems, 30*(1), 531–539. doi:10.1109/TPWRS.2014.2326557.

Koster, A. M. C. A., & Kutschka, M. (2011). An integrated model for survivable network design under demand uncertainty. In *Proceedings of the 8th international workshop on the design of reliable communication networks (DRCN)* (pp. 54–61). doi:10.1109/DRCN.2011.6076885.

Lappas, N. H., & Gounaris, C. E. (2016). Multi-stage adjustable robust optimization for process scheduling under uncertainty. *AIChE Journal, 62*(5), 1646–1667. doi:10.1002/aic.15183.

Lappas, N. H., & Gounaris, C. E. (2018). Robust optimization for decision-making under endogenous uncertainty. *Computers & Chemical Engineering, 111*, 252–266.

Lee, C., Lee, K., & Park, S. (2013). Benders decomposition approach for the robust network design problem with flow bifurcations. *Networks, 62*(1), 1–16. doi:10.1002/net.21486.

Lee, C., Liu, C., Mehrotra, S., & Bie, Z. (2015). Robust distribution network reconfiguration. *IEEE Transactions on Smart Grid, 6*(2), 836–842. doi:10.1109/TSG.2014.2375160.

Lee, C., Liu, C., Mehrotra, S., & Shahidehpour, M. (2014). Modeling transmission line constraints in two-stage robust unit commitment problem. *IEEE Transactions on Power Systems, 29*(3), 1221–1231. doi:10.1109/TPWRS.2013.2291498.

Lou, Y., & Zhang, L. (2011). Defending transportation networks against random and targeted attacks. *Transportation Research Record: Journal of the Transportation Research Board, 2234*, 31–40. doi:10.3141/2234-04.

Mudchanatongsuk, S., Ordóñez, F., & Liu, J. (2008). Robust solutions for network design under transportation cost and demand uncertainty. *The Journal of the Operational Research Society, 59*(5), 652–662.

Nohadani, O., & Sharma, K. (2018). Optimization under decision-dependent uncertainty. *SIAM Journal on Optimization, 28*(2), 1773–1795. doi:10.1137/17M1110560.

Ordóñez, F., & Zhao, J. (2007). Robust capacity expansion of network flows. *Networks, 50*(2), 136–145. doi:10.1002/net.20183.

Orlowski, S., Wessäly, R., Pióro, M., & Tomaszewski, A. (2010). Sndlib 1.0–Survivable Network Design Library. *Networks, 55*(3), 276–286. doi:10.1002/net.20371.

Poss, M. (2014). Robust combinatorial optimization with variable cost uncertainty. *European Journal of Operational Research, 237*(3), 836–845.

Poss, M., & Raack, C. (2013). Affine recourse for the robust network design problem: Between static and dynamic routing. *Networks, 61*(2), 180–198. doi:10.1002/net.21482.

Simchi-Levi, D., Wang, H., & Wei, Y. (2017). Constraint generation for two-stage robust network flow problem. *INFORMS Journal on Optimization, 1*(1), 49–70. doi:10.1287/ijoo.2018.0003.

Snyder, L. V., Scaparra, M. P., Daskin, M. S., & Church, R. L. (2006). Planning for disruptions in supply chain networks. *Models, methods, and applications for innovative decision making* (pp. 234–257). doi:10.1287/educ.1063.0025.

Sullivan, K. M., & Cole Smith, J. (2014). Exact algorithms for solving a euclidean maximum flow network interdiction problem. *Networks, 64*(2), 109–124.

Thiele, A., Terry, T., & Epelman, M. (2009). Robust linear optimization with recourse. *Optimization Online*. http://www.optimization-online.org/DB_FILE/2009/03/2263.pdf.

Tomaszewski, A., Pióro, M., & Żotkiewicz, M. (2010). On the complexity of resilient network design. *Networks, 55*(2), 108–118. doi:10.1002/net.20321.

Wang, X., & Pardalos, P. M. (2017). A modified active set algorithm for transportation discrete network design bi-level problem. *Journal of Global Optimization, 67*(1), 325–342. doi:10.1007/s10898-015-0396-y.

Washburn, A., & Wood, K. (1995). Two-person zero-sum games for network interdiction. *Operations Research, 43*(2), 243–251.

Ye, H., & Li, Z. (2016). Robust security-constrained unit commitment and dispatch with recourse cost requirement. *IEEE Transactions on Power Systems, 31*(5), 3527–3536. doi:10.1109/TPWRS.2015.2493162.

Yuan, W., Wang, J., Qiu, F., Chen, C., Kang, C., & Zeng, B. (2016a). Robust optimization-based resilient distribution network planning against natural disasters. *IEEE Transactions on Smart Grid, 7*(6), 2817–2826. doi:10.1109/TSG.2015.2513048.

Yuan, W., Wang, J., Qiu, F., Chen, C., Kang, C., & Zeng, B. (2016b). Robust optimization-based resilient distribution network planning against natural disasters. *IEEE Transactions on Smart Grid, 7*(6), 2817–2826.

Yuan, W., Zhao, L., & Zeng, B. (2014). Optimal power grid protection through a defender–attacker–defender model. *Reliability Engineering & System Safety, 121*, 83–89.

Zeng, B., & Zhao, L. (2013). Solving two-stage robust optimization problems using a column-and-constraint generation method. *Operations Research Letters, 41*(5), 457–461. doi:10.1016/j.orl.2013.05.003.

Zhang, L., Lawphongpanich, S., & Yin, Y. (2009). An active-set algorithm for discrete network design problems. In W. H. K. Lam, S. C. Wong, & H. K. Lo (Eds.), *Transportation and Traffic Theory 2009: Golden Jubilee: Papers selected for presentation at ISTTT18, a peer reviewed series since 1959* (pp. 283–300)). Boston, MA: Springer US. doi:10.1007/978-1-4419-0820-9_14.

Zhao, L., & Zeng, B. (2012). An exact algorithm for two-stage robust optimization with mixed integer recourse problems. *Optimization Online* http://www.optimization-online.org/DB_HTML/2012/01/3310.html.