Robust capacitated Steiner trees and networks with uniform demands

Cédric Bentz, Marie-Christine Costa, Pierre-louis Poirion, Thomas Ridremont

Received: 8 December 2021 | Revised: 9 August 2022 | Accepted: 11 December 2022

DOI: 10.1002/net.22143

RESEARCH ARTICLE

WILEY

Robust capacitated Steiner trees and networks with uniform demands

Cédric Bentz[1] | **Marie-Christine Costa**[2] | **Pierre-Louis Poirion**[3] | **Thomas Ridremont**[4]

[1]CEDRIC, CNAM, Paris, France
[2]CEDRIC, CNAM (and ENSTA IP-Paris), Paris, France
[3]Center for Advanced Intelligence Project, RIKEN, Tokyo, Japan
[4]Artelys, Paris, France

Correspondence
Cédric Bentz, CEDRIC, CNAM, Paris, France.
Email: cedric.bentz@cnam.fr

Funding information
Gaspard Monge Program for Optimization, Operations Research and Their Interactions with Data Sciences (PGMO)

Abstract

We are interested in the design of robust (or resilient) capacitated rooted Steiner networks in the case of terminals with uniform demands. Formally, we are given a graph, capacity, and cost functions on the edges, a root, a subset of vertices called *terminals*, and a bound k on the number of possible edge failures. We first study the problem where $k = 1$ and the network that we want to design must be a tree covering the root and the terminals: we give complexity results and propose models to optimize both the cost of the tree and the number of terminals disconnected from the root in the worst case of an edge failure, while respecting the capacity constraints on the edges. Secondly, we consider the problem of computing a minimum-cost survivable network, that is, a network that covers the root and terminals even after the removal of any k edges, while still respecting the capacity constraints on the edges. We also consider the possibility of protecting a given number of edges. We propose three different formulations: a bilevel formulation (with an attacker and a defender), a cutset-based formulation and a flow-based one. We compare the formulations from a theoretical point of view, and we propose algorithms to solve them and compare their efficiency in practice.

KEYWORDS

bilevel programs, complexity, discrete optimization, interdiction problems, mathematical programming, network design, robust networks, Steiner trees

1 | INTRODUCTION

1.1 | Presentation of the problems

In this paper, we focus on the design of robust (or resilient) trees or networks able to route a given flow from a root to a set of terminals while respecting capacity constraints on the edges [12,23,27]. Breakdowns can occur in the network, and we aim to minimize the cost of the network to be built while respecting robustness constraints intended to limit the damaging repercussions in case of a breakdown on one or several edges in the network. Notice that, in directed graphs, breakdowns on vertices can be reduced to breakdowns on arcs [38]. We study several notions of robustness, which will be described later. In one variant, we also consider the possibility of "protecting" some edges: a protected edge cannot fail.

The problem originates from a study on wiring networks in wind farms, designed to route the energy produced by wind turbines (i.e., terminals) towards the substation (i.e., the root) linking the wind farm to the electrical network, with respect to some technical constraints (cable capacities, nonsplitting constraints, etc.) [19,24]. In that case, the flow is routed from the terminals to the root. We know the location of each wind turbine, and the one of the substation. We also know the location of

possible cables that can be built and of optional interconnection vertices which make it possible to connect cables between them. Each wind turbine produces a known maximum quantity of energy, and with each cable are associated a cost and a capacity (the maximum amount of energy that can be routed through this cable). Generally in offshore environments, and often in onshore environments, each wind turbine produces about the same quantity of electricity [34], so we can make the assumption that the energy produced by the wind turbines is uniform. Furthermore, those networks must be resilient to cable failures but some links can be protected, for instance with thicker or additional parallel cables.

Informally, we consider a support graph with a root vertex and a set of terminals, and we want to select vertices and edges of the graph in order to build a minimum-cost network (i.e., subgraph of the support graph which must respect some known additional constraints) linking the root to the terminals, while respecting the capacity constraints on the edges. The given graph can be directed or undirected, but the Steiner tree or network to be built is always directed from the root towards the terminals. Nevertheless, considering that it is easy to adapt solutions given for directed graphs (or digraphs) to cases where the graph is undirected, either by replacing each edge of the undirected graph by two opposite arcs with the same cost and capacity or by slightly modifying the models and the methods proposed to solve the problem, we only consider directed graphs (digraphs) in our models while the problems can be defined in undirected graphs (graphs).

Formally, we are given a digraph $G = (V, A)$, with a subset of vertices $T \subset V$ called *terminals*, a root $r \in V \setminus T$, and cost and capacity functions on the set of arcs A, denoted, respectively, by c and u. The vertices in $V \setminus \{T \cup \{r\}\}$, that is, the optional interconnection vertices, are called *Steiner vertices*. We will refer to $\Gamma^+_G(v)$ and $\Gamma^-_G(v)$ as the set of successors and predecessors of a vertex $v \in V$, respectively. We assume that some arcs can break down, and that there is a bound k on the number of possible arc failures. We aim to protect the network to be built against the worst case of breakdowns, that is, the one that makes the maximum number of terminals unreachable from the root. Note that, in the case of our main problem, we want this maximum number to be equal to 0. Also note that, in one variant, we consider a given budget of protection k', meaning that we can protect at most k' arcs against failures.

We first study the case where the network we want to build is an arborescence (rooted tree). We assume that one arc can break down, and we aim at generating a robust arborescence, that is, an arborescence that minimizes the maximum number of terminals that can be disconnected from the root in case of a breakdown, which is called the worst case in what follows. Then, we study the so-called Capacitated Rooted k-Arc Connected Steiner Network problem, which is our main problem: we aim to design a minimum-cost network in which, after the failure of any k arcs, we can still route one unit of flow from the root to each terminal. Indeed, we deal with the special case where the demand is identical at each terminal, as it is generally assumed for wind farms: the flow from the root to each terminal is a constant, and hence can be set to 1 without loss of generality. The capacity u_{ij} of an arc (i,j) is then the maximum number of terminals linked to the root through this arc. For trees, this is equivalent to the maximum number of terminals in the subtree rooted at j. We can hence assume w.l.o.g. that u_a is a positive integer for each $a \in A$. Eventually, we also study the possibility of protecting a subset of arcs of a given size: a protected arc cannot break down.

1.2 | Overview of related work

When the network to be built is a rooted tree, the problem without breakdown corresponds to the capacitated Steiner tree (or arborescence) problem, which has been studied for instance in [6,13,33]. This problem, in turn, generalizes the well-known Steiner tree problem [16,22,25,30].

Obviously, our main problem is similar to the Survivable or Robust Network Design problem. For a general presentation of robustness, the reader is referred, for instance, to the classic article of Bertsimas and Sim [8], and to the book of Ben-Tal et al. [5]. For the design of survivable networks, see for instance [23].

However, the authors of [23] do not take into account the arc capacities, and consider a given number of arc deletions between each pair of vertices. A survey and other work on this problem are available in [21,27]. In [12], a method based on Benders decomposition is proposed for a problem with hop-constraints (the number of arcs in a path from the root vertex to any of the terminal vertices is limited). In [9,26,36], the authors take capacities into account, but they allocate them whereas, in our main problem, the capacities are fixed and the flow is adaptive, as in [7], where the aim is to find a maximum, robust (for the authors, that means respecting a weak flow conservation constraint at each vertex), and nonintegral flow. In [31], a two-stage model is proposed for the design of resilient single-commodity flow networks: the aim is to allocate (nonintegral) capacities to edges in order to minimize the cost of building the network plus the cost of the flow circulation (with a linear cost on each edge) in order to serve a given demand at each vertex, using nonintegral flows. The authors also consider the possibility of protecting some edges against failures.

Such problems often fall under the two-level optimization framework. This means that "there are two decision makers, commonly denoted as the leader and the follower, and decisions are made in a hierarchical manner: first, the leader makes a

decision, and then the follower optimizes its objective, affected by the decisions of the leader. It is assumed that the leader can anticipate the decisions of the follower" (see [15,18]).

Actually, our main problem also falls under this framework (see also Section 3.2.1): the leader builds a network at minimum cost, and then the follower deletes some arcs. The goal of the leader is to ensure that, whatever the deleted arcs are, one can still route one unit of flow from the root to each terminal, while respecting the arc capacities.

The problem associated with the follower is then similar to the one of finding a subset of k arcs such that the deletion of these k arcs results in the maximum decrease of the value of the maximum flow between two vertices, which is referred to as the k most vital arcs problem, and which has been studied by Ratliff et al. [37]. Later, this problem was generalized into the network interdiction problem by Wood [42], who considers a deletion cost associated with each arc and a maximum budget of deletions: he shows that the problem is strongly $\mathcal{NP}$-hard, and proposes a mixed-integer formulation to solve it. See also [39] for a recent survey on this subject. Note that some authors have considered the k most vital arcs for other related graph problems, such as spanning trees [4].

Finally, studies on multicommodity versions of network design problems are available in [14,41], and polyhedral studies have been conducted on problems corresponding to the uncapacitated [3] or unrooted [10] version of our main problem. However, none of these papers has considered the case where a small subset of arcs can be protected from failure, and thus cannot break down.

1.3 | Contributions and outline of the paper

In Section 2, we study the problem of finding a Steiner or spanning arborescence taking into account both the cost and the number of terminals disconnected from the root in the worst case of an arc breakdown. We prove that deciding whether there exists a spanning arborescence respecting the capacity constraints is an NP-Complete problem (this is the special case of our problem with a demand equal to 1 at each vertex except the root), which extends a complexity result of Papadimitriou [33]. We also propose different formulations, considering several criteria (cost, maximum number of disconnected terminals, etc.), either as objectives or as constraints with given bounds.

In Section 3, we present the Capacitated Rooted k-Arc Connected Steiner Network problem, which amounts to searching for a minimum-cost network that, in the worst case of k arcs failures, can still route one unit of flow from the root to each terminal while respecting the capacity constraints.

We give a bilevel formulation, with a leader and a follower, where the second level is a min-max problem corresponding to optimizing the worst case for the follower [2], and two formulations based on cutsets and flows, respectively. We study the relations between the three formulations from a theoretical point of view, and show the equivalence between a linearization of the bilevel formulation and both the cutset and the flow formulations, except in the case of uniform capacities on the arcs. Then, we propose algorithms based on integer linear programming and cutting plane methods to solve the problem.

In Section 4, as in [31], we consider the case where a set of arcs can be protected (and thus cannot break down): in this case, the bilevel and cutset formulations are no longer equivalent. Then, in Section 5, we propose strengthening inequalities to improve the efficiency of the algorithms, with and without arc protection.

Finally, in Section 6, we first compare the formulations given in Section 2 by testing them on real wind farm data. Then, we compare the efficiency of the models and methods proposed in Section 3, by testing them on a set of randomly generated but similar to real-life (wind farm) data, and on a set of instances from SNDLib [40], before concluding.

2 | ROBUST ARBORESCENCES

In this section, we focus on finding a robust Steiner arborescence covering the root and the terminals of G. Here, the robust optimization approach consists in finding a feasible solution that minimizes the number of terminals disconnected from the root in the worst case of an arc failure.

This setting arises in some wind farm cabling problems (see Section 1), when technical constraints impose that all electrical flows arriving at any device except the substation must leave it through one and only one cable: an inclusion-wise minimal subnetwork of G respecting those constraints then corresponds to a Steiner anti-arborescence.

2.1 | Definition of the problems and complexity results

To obtain stronger complexity results, we assume in this section that the graph $G = (V, E)$ is undirected and, when considering a subtree $G' = (V', A')$ of G, rooted at some vertex, to which we give an orientation, we have $V' \subseteq V$ and A' is the set of arcs of G' corresponding to a subset of edges of G denoted by $\bar{A}' \subseteq E$. Since the problem in a graph is a special case of the problem in

a digraph (see Section 1), our complexity results can be extended to digraphs. We define the robust problem without capacity constraints as follows:

2.1.1 | Robust Steiner arborescence problem (RStA)

INSTANCE: A connected graph $G = (V, E)$ with a root vertex $r \in V$ and a set of terminals $T \subseteq V \setminus \{r\}$.

PROBLEM: Find an arborescence $S = (V_S, A_S)$ such that $V_S \subseteq V$, $\bar{A}_S \subseteq E \cap (V_S \times V_S)$, and $T \subset V_S$, which is rooted at r and minimizes the number of terminals disconnected from r when an arc is removed from A_S, in the worst case.

We also consider the spanning version of the problem (i.e., $T = V \setminus \{r\}$). In this case, the problem is to minimize the number of vertices in the largest (regarding the number of vertices) subarborescence not containing r. We define it as follows:

2.1.2 | Robust spanning arborescence problem

INSTANCE: A connected graph $G = (V, E)$ and a root vertex $r \in V$.

PROBLEM: Find a spanning arborescence S of G, rooted at r, which minimizes the size of the largest subarborescence of S not containing r.

Obviously, the largest subarborescence not containing r is rooted at a vertex $v \in \Gamma_G(r)$, where $\Gamma_G(u)$ is the set of vertices adjacent to a vertex u in G, and the worst case is the failure of an arc incident to the root. We have the following properties:

Lemma 2.1. *(a) There is an optimal solution $S^* = (V, A^*)$ of robust spanning arborescence problem (RSpA) containing (r, v) for all $v \in \Gamma_G(r)$ (i.e., $\Gamma_G(r) = \Gamma^+_{S^*}(r)$).*

(b) There is an optimal solution $S^* = (V^*_S, A^*_S)$ of RStA containing (r, v) for all $v \in V^*_S \cap \Gamma_G(r)$.

Proof. Let $S = (V, A_S)$ be an optimal solution of RSpA such that there is $v \in \Gamma_G(r)$ with $(r, v) \notin A_S$, and let w be the predecessor of v in the path from r to v in S. If we remove (w, v) from A_S and add (r, v), we obtain a new spanning arborescence at least as good as S, since we have replaced a subarborescence by two subarborescences of smaller sizes. Doing so for each $v \in \Gamma_G(r)$ with $(r, v) \notin A_S$ yields a solution S^* satisfying the property.

The proof is similar for RStA, by replacing $\Gamma_G(r)$ by $V^*_S \cap \Gamma_G(r)$: if we remove (w, v) from A_S and add (r, v), we obtain a new Steiner arborescence at least as good as S, since we have replaced a subarborescence by two subarborescences spanning at most the same number of terminals. ∎

Notice that these properties do not necessarily hold if we have capacity constraints, because the capacity of (r, v) can be smaller than the one of (w, v) in the proof above. Let us now introduce the feasibility problem associated with **RSpA**.

2.1.3 | Robust spanning arborescence feasibility problem

INSTANCE: A connected graph $G = (V, E)$ with a root vertex $r \in V$ and an integer β with $1 \leq \beta \leq |V| - 1$.

QUESTION: Is there a spanning arborescence $S = (V, A_S)$ of $G, \bar{A}_S \subseteq E$, rooted at r, such that the size of any subarborescence of S not containing r is at most β?

Theorem 2.2. *Robust spanning arborescence feasibility problem (RSpAF) is strongly NP-Complete.*

Proof. We introduce the NP-complete 3-Partition problem [20] in order to transform any instance of this problem into an **RSpAF** one. ∎

2.1.4 | 3-Partition problem

INSTANCE: A finite set D of $3m$ positive integers d_i, $i = 1, .., 3m$, and a positive integer B such that $\sum_{i=1,...,3m} d_i = mB$ and $B/4 < d_i < B/2 \; \forall i = 1, ..., 3m$.

QUESTION: Can D be partitioned into m disjoint subsets $M_1, M_2, ..., M_m$ of three elements each such that the sum of the numbers in each subset is equal to B?

To obtain an instance of **RSpAF** from an instance of 3-Partition, we set $\beta = B + 1$ and we construct the following graph $G = (V, E)$: we define a root r and m vertices v_j with an edge $[r, v_j]$ for $j = 1, ..., m$, each vertex v_j corresponding to a set M_j. We add $3m$ vertices w_i and the edges $[v_j, w_i]$ for all $j = 1, .., m$ and all $i = 1, .., 3m$, each vertex w_i corresponding to the element d_i of D (the subgraph induced by the vertices v_j and w_i is complete bipartite). Finally, for each $i = 1, .., 3m$, we add $d_i - 1$ vertices only adjacent to w_i: the subgraph induced by those vertices and the vertices w_i is made of $3m$ stars. See Figure 1 for a graph representation of a 3-Partition instance with $m = 2$, $B = 11$ and $D = \{5, 3, 4, 3, 4, 3\}$. Notice that $|V| = 1 + m + mB$.

Solving **RSpAF** on G with $\beta = B+1$ amounts to finding an arborescence where the size of the subarborescence rooted at each v_j is smaller than or equal to $B + 1$. If there is a solution to **RSpAF** on G, then, from the proof of Lemma 2.1, there is a solution

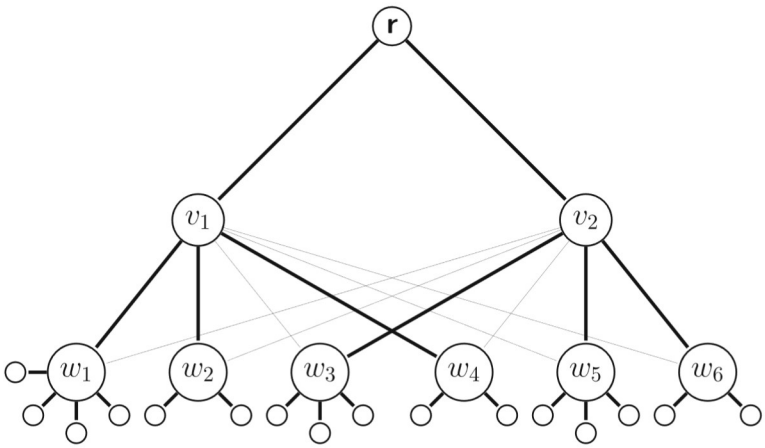

FIGURE 1 Graph and **RSpAF** solution resulting from the 3-Partition instance in which $m = 2$, $B = 11$, $D = \{5, 3, 4, 3, 4, 3\}$.

S such that $(r, v_j) \in S\ \forall j = 1, ..., m$, and each w_i is connected to exactly one v_j, otherwise there is a cycle. Given a vertex $v \in S$, let $S(v)$ be the subarborescence of S rooted at v: $\forall j = 1, ..., m$, we have $|S(v_j)| \leq B+1$ and $\sum_{j=1,...,m} |S(v_j)| = |V \setminus \{r\}| = mB+m$. Thus, for each $j = 1, ..., m$, $|S(v_j)| = B + 1$ and $S(v_j)$ contains v_j and several vertices w_i, each having $d_i - 1$ successors in S. Finally, the constraints $B/4 < d_i < B/2$ imply that, for each $j = 1, ..., m$, v_j is adjacent in S to exactly three vertices w_i denoted in the following by w_{j_1}, w_{j_2} and w_{j_3}, and such that $|S(w_{j_1})| + |S(w_{j_2})| + |S(w_{j_3})| = |S(v_j)| - 1 = B$.

Then, it is easy to obtain a solution to the 3-Partition instance. For each $j = 1, .., m$, we set $M_j = \{|S(w_{j_1})|, |S(w_{j_2})|, |S(w_{j_3})|\} = \{d_{j_1}, d_{j_2}, d_{j_3}\}$. We have m disjoint sets, each of size B, which cover exactly D. For the instance given in Figure 1, a solution to 3-Partition can be associated with the arborescence given in thick: $M_1 = \{5, 3, 3\}$ and $M_2 = \{4, 4, 3\}$.

Similarly, from a solution to the 3-Partition instance, it is easy to obtain a solution S to **RSpAF** for the associated graph G.

The 3-Partition problem is NP-Complete in the strong sense, meaning that it remains NP-Complete even if the integers in D are bounded above by a polynomial in m [20]. Thus, the reduction can be done in polynomial time, and **RSpAF**, which is clearly in NP, is strongly NP-Complete.

RSpAF being NP-Complete, **RSpA** is NP-Hard, and so is **RStA** because it is a generalization of **RSpA**. Let us now consider capacity constraints on the edges. **RSpAF** can be seen as a special case of the general capacitated Steiner arborescence problem where the demand at each vertex, except r, is an integer (our demands are all equal to 1, and the capacities are all equal to β), and hence from Theorem 2.2 we obtain the following corollary:

> **Corollary 2.3.** *Given a graph $G = (V, E)$ with a vertex $r \in V$ and two functions d, from $V \setminus \{r\}$ to $\mathbb{N}$, and u, from E to $\mathbb{N}$, representing respectively the demands at each vertex except r and the capacities of the edges, the problem of deciding whether there exists a Steiner arborescence of G, rooted at r and respecting the capacities, is strongly NP-Complete (even if u is a uniform function and all demands are equal to 1, i.e., we look for a spanning arborescence).*

This strengthens the following result due to Papadimitriou [33]: given two positive values C and K and a graph $G = (V, E, r, c)$ where c is a cost function on the edges, the problem of deciding whether there exists a spanning arborescence S of G rooted at r, such that each subarborescence of S not containing r contains at most K vertices, and with total cost at most C, is NP-Complete.

In the following, we study the more general problem, which is hence also NP-Hard (note that, for a digraph, "connected" means that the underlying undirected graph is connected):

2.1.5 | Robust capacitated Steiner arborescence problem

INSTANCE: A connected digraph $G = (V, A)$, a root vertex $r \in V$, a set of terminals $T \subseteq V \setminus \{r\}$ and u a positive integral function on A representing the arc capacities.

PROBLEM: Find an arborescence $S = (V_S, A_S)$ with $V_S \subseteq V$ and $A_S \subseteq A \cap (V_S \times V_S)$, rooted at r and spanning the terminals of T, which respects the arc capacities and minimizes the number of terminals disconnected from r when an arc a is removed from A_S, in the worst case.

2.2 | Mathematical formulations

In this section we propose formulations for robust Steiner problems where robustness is viewed either as a constraint with the objective of minimizing the cost, or as an objective with or without constraints on the cost. Moreover, we study two kinds of robustness, by considering the worst case of arc breakdowns and a kind of "average" arc breakdowns, which is defined later. We will denote by *R-robustness* the first type of robustness, and by *B-robustness* the second type of robustness. Here, our approach somehow falls under the multicriteria optimization framework [17]: however, our goal is not to compute the whole Pareto front, which would require a full computational study, but only to show that there exist efficient solutions (i.e., solutions lying on the Pareto front) that yield a good compromise between all the criteria from a practical point of view.

Let $G = (V, A)$ be a digraph. To formulate the different problems, for each arc $(i, j) \in A$ we introduce the 0-1 variable y_{ij} and the integral variable x_{ij}, where $y_{ij} = 1$ if and only if the arc (i, j) is selected in the considered solution, and x_{ij} represents the number of terminals connected to the root through the arc (i, j), or equivalently the number of terminals in the subarborescence rooted at j. We introduce the following feasible set $\mathcal{P}$:

$$\mathcal{P} = \left\{ x \in \mathbb{N}^{|A|},\ y \in \{0,1\}^{|A|} \;\middle|\; \begin{array}{ll} \displaystyle\sum_{i \in \Gamma_G^-(j)} x_{ij} - \sum_{k \in \Gamma_G^+(j)} x_{jk} = \begin{cases} -|T| & \text{if } j = r \\ 1 & \text{if } j \in T \\ 0 & \text{otherwise} \end{cases} & \forall j \in V \\ \displaystyle\sum_{i \in \Gamma_G^-(j)} y_{ij} \le 1 & \forall j \in V \setminus \{r\} \\ x_{ij} \le u_{ij} y_{ij} & \forall (i,j) \in A \end{array} \right\}.$$

In the following, we write $(x, y) \in \mathcal{P}$ when we consider a couple of variable vectors satisfying the constraints of $\mathcal{P}$. The first set of constraints in $\mathcal{P}$ ensures both the conservation of the number of terminals connected through each Steiner vertex $j \in V$ (flow conservation) and the connection of the root to all terminals. The second set of constraints ensures that the solution is an arborescence, that is, that each vertex has at most one predecessor in it. Finally, the third set ensures that there is no flow on a nonselected arc, and that the number of terminals connected through an arc $(i, j) \in A$ does not exceed its capacity. In the following, the relative gap between two costs will be denoted by Δ. The well-known problem of the capacitated Steiner arborescence (**CStA**) can be formulated as follows [13]:

$$\text{(CStA)} \quad \left| \quad \min_{(x,y) \in \mathcal{P}} \ \sum_{(i,j) \in A} c_{ij} y_{ij} \right.$$

As explained previously, we evaluate the first type of robustness, called *R*-robustness, by considering the number of terminals disconnected from the root in a worst-case scenario, that is, the maximum number of terminals connected through an arc incident to the root, which is equal to $\max_{j \in \Gamma_G^+(r)} x_{rj}$. Let R be a fixed bound on this value: we must not disconnect more than R terminals from the root by deleting any arc. We propose the following formulation for the Capacitated Steiner Arborescence problem with bounded *R*-robustness ($\text{CStA}_{\text{bounded}-R\_\text{robust}}$):

$$(\text{CStA}_{\text{bounded}-R\_\text{robust}}) \quad \left| \quad \begin{array}{ll} \min\limits_{(x,y) \in \mathcal{P}} & \sum\limits_{(i,j) \in A} c_{ij} y_{ij} \\ \text{s.t.} & \max\limits_{j \in \Gamma_G^+(r)} x_{rj} \le R \end{array} \right.$$

Let us now look at robustness as an objective. Note that we implicitly assume that the default objective function is to minimize the cost of the solution. If a model uses another objective function, then its name will start by a given letter, for example, R if we want to optimize *R*-robustness. We propose the following formulation for Robust capacitated Steiner arborescence problem (**RCStA**):

$$\text{(RCStA)} \quad \left| \quad \min_{(x,y) \in \mathcal{P}} \ \max_{j \in \Gamma_G^+(r)} x_{rj}. \right.$$

Since this formulation does not take the cost into account, we also propose a new formulation where we bound the cost of a solution by a given value C:

$$(\text{RCStA}_{\text{bounded}-cost}) \quad \left| \quad \begin{array}{ll} \min\limits_{(x,y) \in \mathcal{P}} & \max\limits_{j \in \Gamma_G^+(r)} x_{rj} \\ \text{s.t.} & \sum\limits_{(i,j) \in A} c_{ij} y_{ij} \le C \end{array} \right.$$

However, the previous models only consider the worst case of arc breakdowns. It appears that it can also be interesting to "balance" the tree in order to reduce the loss due to an "average" breakdown. To this end, we consider arc failures at each vertex

and not only at the root, that is, for each $i \in V$, we consider the worst case of a breakdown of an arc leaving i. This corresponds, for each $i \in V$, to the maximum number of terminals that cannot be reached from the root in case of a breakdown of an arc (i,j), $j \in \Gamma^+_G(i)$, or equivalently to the maximum flow on an arc (i,j), $j \in \Gamma^+_G(i)$. We define B-robustness, for "balanced" robustness, as the sum of these values: $\sum_{i\in V} \max_{j\in\Gamma^+_G(i)} x_{ij}$.

We will use the letters BR to refer to models where one wants to optimize B-robustness. We propose formulations similar to the previous ones for the Capacitated Steiner Arborescence with bounded B-robustness, where we bound the B-robustness of a solution by a given value BR:

$$(\text{CStA}_{\text{bounded-balanced\_robust}}) \quad \left| \begin{array}{ll} \min\limits_{(x,y)\in\mathcal{P}} & \sum\limits_{(i,j)\in A} c_{ij}y_{ij} \\ \text{s.t.} & \sum\limits_{i\in V} \max\limits_{j\in\Gamma^+_G(i)} x_{ij} \leq \text{BR} \end{array} \right. .$$

The following formulation aims at computing the best B-robustness:

$$(\text{BRCStA}) \quad \left| \min_{(x,y)\in\mathcal{P}} \sum_{i\in V} \max_{j\in\Gamma^+_G(i)} x_{ij}. \right.$$

Moreover, we can keep this latter objective while bounding both the R-robustness (by R) and the cost of the solution (by C). We obtain:

$$(\text{BRCStA}_{\text{bounded-}R\text{\_robust-}cost}) \quad \left| \begin{array}{ll} \min\limits_{(x,y)\in\mathcal{P}} & \sum\limits_{i\in V} \max\limits_{j\in\Gamma^+_G(i)} x_{ij} \\ \text{s.t.} & \max\limits_{j\in\Gamma^+_G(r)} x_{rj} \leq R \\ & \sum\limits_{(i,j)\in A} c_{ij}y_{ij} \leq C \end{array} \right. .$$

Notice that all these formulations can easily be linearized: each term of the form $\max_{x\in D} f(x)$ is replaced by z, and the constraints "$z \geq f(x)$ for all $x \in \mathcal{D}$" are added to the model. For instance, $(\text{BRCStA}_{\text{bounded-}R\text{\_robust-}cost})$ can be rewritten as:

$$(\text{LBRCStA}_{\text{bounded-}R\text{\_robust-}cost}) \quad \left| \begin{array}{lll} \min\limits_{(x,y)\in\mathcal{P},z} & \sum\limits_{i\in V} z_i & \\ \text{s.t.} & z_i \geq x_{ij} \quad \forall i \in V & \forall j \in \Gamma^+_G(i) \\ & z_r \leq R & \\ & \sum\limits_{(i,j)\in A} c_{ij}y_{ij} \leq C & \end{array} \right. .$$

The results of our experiments comparing the effects of the formulations on the solutions are given in Section 6.1.

3 | CACPACITATED ROOTED K-ARC CONNECTED STEINER NETWORK PROBLEM

3.1 | Definitions and notations

In this section, we study the problem of designing networks that are resilient to a given number of arc failures. A feasible solution to the problem we consider is a network rooted at a given root and covering a given set of terminals, and such that, after deleting any k arcs, it is still possible to route a unit of flow from the root to each terminal, while respecting given capacities on the arcs. Note that, if $k > |A| - |T|$, then at least one terminal will be disconnected from the root, and hence there exists no feasible solution. Formally, we define the following problem (note once again that, for a digraph, "connected" means that the underlying undirected graph is connected):

3.1.1 | Capacitated rooted k-arc connected Steiner network problem (CRkACSN)

INSTANCE: A connected digraph $G = (V, A)$ with a set of vertices V, a set of arcs A, a root vertex $r \in V$, a set of terminals $T \subseteq V \setminus \{r\}$, an integral capacity function u on A, a cost function c on A and an integer k with $1 \leq k \leq |A| - |T|$.

QUESTION: Find a subset $A' \subseteq A$ of minimum cost such that there is a feasible flow (i.e., respecting the arc capacities) routing a unit of flow from r to each vertex of T in the subgraph of G induced by A', even if any subset of k arcs in A' is deleted.

Property 3.1. A necessary condition for a solution A' to be feasible is that there are at least $k+1$ arc-disjoint paths between the root and each terminal in $G' = (V, A')$. Furthermore, any inclusion-wise minimal feasible solution induces at least a 2-edge-connected graph in the underlying undirected graph.

Proof. The first part of the property is a direct consequence of Menger's well-known Theorem [32]. Now, let G' be an inclusion-wise minimal feasible solution, and assume that G' is not 2-edge-connected in the underlying undirected graph. Then, there exists at least one edge e whose removal cuts G' into two parts. If the part that does not include the root contains terminals, then G' is clearly not a feasible solution because, if we remove e, then at least one terminal cannot be reached from the root. Otherwise, G' is not inclusion-wise minimal because, if we remove e, then the resulting graph is still a feasible solution. Hence, any inclusion-wise minimal feasible solution induces at least a 2-edge-connected graph. ∎

Remark 3.1. We assume without loss of generality that there are at least $k+1$ arc-disjoint paths between the root and each terminal in the digraph G, and that the underlying undirected graph is 2-edge-connected, otherwise there is no feasible solution. Moreover, Property 3.1 implies that $|A'| \geq k$, and so deleting k arcs from A' is always possible.

In order to simplify the formulations proposed in the next sections, we add to the input digraph a vertex s (which corresponds to a fictive sink) connected to every terminal $t \in T$ by a fictive arc (t,s) with $c_{ts} = 0$ and $u_{ts} = 1$. Then, s is added to V and the fictive arcs are added to A. Obviously, finding a flow which routes one unit of flow between r and each terminal in the input digraph is equivalent to finding a flow of value $|T|$ from r to s in the transformed digraph.

In the following, A_I denotes the set of initial arcs, A the set of arcs of the transformed support digraph (i.e., $A = A_I \cup \{(t,s), \forall t \in T\}$), and A' the set of arcs of the selected network (including the arcs (t,s), for each $t \in T$).

3.2 | Three formulations for CRkACSN

We define the following variables, for each $(i,j) \in A$:

y_{ij} is a binary variable: $y_{ij} = 1$ if and only if the arc (i,j) is selected in A'; in particular, $y_{ts} = 1$ for each $t \in T$: the fictive arcs are always selected in the final network, and they cannot fail;

b_{ij} is a binary variable: $b_{ij} = 1$ if and only if the arc (i,j) is deleted (i.e., breaks down);

x_{ij} is a positive variable: x_{ij} = amount of flow routed through the arc (i,j) = number of terminals linked to the root through the arc (i,j).

We introduce the following set:

$$\mathcal{B} = \left\{ b \in \{0,1\}^{|A|} \mid \sum_{(i,j)\in A} b_{ij} = k \; ; \; b_{ts} = 0 \quad \forall t \in T \right\},$$

which defines, in a way similar to [7], the set of possible scenarios of arc failures, ensuring that no fictive arc (t,s), $t \in T$, is deleted.

Note that, whatever the formulation, the objective is to minimize the cost of the selected arcs ($\sum_{(i,j)\in A} c_{ij} y_{ij}$). Moreover, since it does not make sense to delete a nonselected arc, we should have the constraints $b_{ij} \leq y_{ij}$ $\forall (i,j) \in A$ included in the definition of $\mathcal{B}$. Nevertheless, for any optimal solution with $y_{ij} = 0$ and $b_{ij} = 1$, it will be easy to check that, for all our formulations, there is a feasible solution having the same cost with $y_{ij} = 0$ and $b_{ij} = 0$: thus, these constraints are not considered in $\mathcal{B}$.

Remark 3.2. Note that, in a selected network defined by y, if a worst case of breakdowns is obtained with $\sum_{(i,j)\in A} b_{ij} < k$, then, from Remark 3.1, an equivalent worst case can be obtained by deleting more arcs in order to have $\sum_{(i,j)\in A} b_{ij} = k$.

3.2.1 | Bilevel formulation

The bilevel formulation proposed here is particular in that the second level is a min max problem. We assume w.l.o.g. that there is no arc entering the root r. Given two vectors y and b, each of size $|A|$, we define the following feasible set:

$$\mathcal{X}(y,b) = \left\{ \begin{array}{rcll} \displaystyle\sum_{i\in\Gamma^-(j)} x_{ij} - \sum_{i\in\Gamma^+(j)} x_{ji} = 0 & & \forall j \in V \setminus \{r,s\} & (3.1a) \\ x_{ij} & \leq & u_{ij} y_{ij} & \forall (i,j) \in A \quad (3.1b) \\ x_{ij} & \leq & u_{ij}(1 - b_{ij}) & \forall (i,j) \in A \quad (3.1c) \\ x_{ij} & \in & \mathbb{N} & \forall (i,j) \in A \end{array} \right\}$$

The value of each variable x_{ij} represents a number of terminals, and so is a positive integer. For given vectors y and b, the set $\mathcal{X}(y,b)$ corresponds to the set of possible flows in the subgraph of G induced by the arcs (i,j) such that $y_{ij} = 1$, provided that they have not been deleted (i.e., that $b_{ij} = 0$). Constraints (3.1a) are the flow conservation constraints, Constraints (3.1b) are

the capacity constraints (on the selected arcs), and Constraints (3.1c) impose a flow equal to 0 on any deleted arc. We finally propose the following bilevel program:

$$\text{(BILEVEL)} \quad \left| \begin{array}{lll} \min\limits_{y\in\{0,1\}^{|A|}} & \sum\limits_{(i,j)\in A} c_{ij}y_{ij} & \\ \text{s.t.} & f(y) \geq |T| & (3.2a) \\ & \text{where } f(y) = \min\limits_{b\in\mathcal{B}} \max\limits_{x\in\mathcal{X}(y,b)} \sum\limits_{j\in\Gamma^+(r)} x_{rj} & (3.2b) \end{array} \right.$$

This can be seen as a game with a defender and an attacker (corresponding, respectively, to the leader and the follower). At the upper level, the defender selects the set of arcs of the network to be built, by choosing a value of y in $\{0,1\}^{|A|}$. The attacker then deletes some arcs by computing $b \in \mathcal{B}$ in order to minimize the value of the maximum flow that the defender can obtain by choosing x in $\mathcal{X}(y,b)$. The aim of the defender is to ensure that the value of this flow is larger than or equal to $|T|$ (Constraint (3.2a)).

Note that, if there are no breakdowns, the problem reduces to the NP-Hard Minimum Arc-Cost Flow problem [20] with a unit capacity on each arc linking a terminal to the sink.

3.2.2 | Cutset formulation

We consider now the $r-s$ cuts $[V \setminus V_S, V_S]$ with $V_S \subset V$, $r \in V \setminus V_S$ and $s \in V_S$. Let $\mathcal{S}$ be the set of all the associated cutsets S in A where S is the set of arcs entering V_S (i.e., $S = \{(i,j) \in A \mid i \in V \setminus V_S,\ j \in V_S\}$). Note that if $S \in \mathcal{S}$ then $S \cap A'$ is a cutset in the selected network. For any set $S \in \mathcal{S}$, let C_k^S be the set of subsets of S of k nonfictive arcs. Except for the cutset containing only the fictive arcs, there is at least one terminal in V_S and, from Remark 3.1, at least $k+1$ nonfictive arcs in S, and hence C_k^S cannot be empty. For the cutset S^0 consisting of k fictive arcs, we have, by convention: $\{S^0 \setminus C,\ \forall C \in C_k^{S^0}\} = S^0$. We propose the following cutset formulation:

$$\text{(CUT)} \quad \left| \begin{array}{lll} \min\limits_{y} & \sum\limits_{(i,j)\in A} c_{ij}y_{ij} & \\ \text{s.t.} & \sum\limits_{(i,j)\in S\setminus C} u_{ij}y_{ij} \geq |T| & \forall S \in \mathcal{S},\ \forall C \in C_k^S \quad (3.3) \\ & y_{ij} \in \{0,1\} & \forall (i,j) \in A \end{array} \right.$$

Constraints (3.3) ensure that, in the digraph induced by the arcs (i,j) such that $y_{ij} = 1$, the capacity of each cutset (and consequently of a min-cut) after the deletion of any k arcs of this cutset is at least equal to the number of terminals, which is a necessary and sufficient condition for the existence of a flow of value $|T|$ from r to s [1]. For S^0, Constraint (3.3) is $\sum_{(i,j)\in S^0\setminus C} u_{ij}y_{ij} = \sum_{t\in T} y_{ts} \geq |T|$, which implies that $y_{ts} = 1$ for each $t \in T$.

3.2.3 | Flow formulation

In this section, we introduce a formulation based on flow variables. We define $\mathcal{F}$ as the set of all possible arc-failure scenarios: it corresponds to the set of all k-combinations in A_I. We introduce the variable x_{ij}^F which represents the amount of flow routed through the arc $(i,j) \in A$ when the scenario $F \in \mathcal{F}$ occurs. We propose the following flow formulation:

$$\text{(FLOW)} \quad \left| \begin{array}{lll} \min\limits_{x,y} & \sum\limits_{(i,j)\in A} c_{ij}y_{ij} & \\ \text{s.t.} & \sum\limits_{i\in\Gamma^-(j)} x_{ij}^F - \sum\limits_{k\in\Gamma^+(j)} x_{jk}^F = 0 & \forall j \in V\setminus\{r,s\},\ \forall F \in \mathcal{F} \quad (3.4a) \\ & \sum\limits_{t\in\Gamma^-(s)} x_{ts}^F = |T| & \forall F \in \mathcal{F} \quad (3.4b) \\ & x_{ij}^F \leq u_{ij}y_{ij} & \forall (i,j) \in A,\ \forall F \in \mathcal{F} \quad (3.4c) \\ & x_{ij}^F = 0 & \forall F \in \mathcal{F},\ \forall (i,j) \in F \quad (3.4d) \\ & x \in \mathbb{R}_+^{|A|\times|\mathcal{F}|},\quad y \in \{0,1\}^{|A|} & \end{array} \right.$$

For each arc-failure scenario $F \in \mathcal{F}$, Constraints (3.4a) and (3.4b) ensure that there is a flow of value $|T|$ (routing a unit of flow to each terminal), Constraints (3.4c) ensure that the arc capacities are satisfied, and Constraints (3.4d) ensure that no flow is routed through deleted arcs. One can notice that the value of any variable x_{ij}^F must be an integer (because it corresponds to a number of terminals). However, we relax this integrality constraint. Indeed, for any value of $y \in \{0,1\}^{|A|}$, setting the value of x corresponds to routing a set of flows of value $|T|$ on $|\mathcal{F}|$ different networks with integral capacities. Then, for any given value

of $y \in \{0,1\}^{|A|}$, all the components of x are integers in any basic solution [1], and hence there exists an optimal solution where x is integral.

3.3 | Theoretical comparison between formulations and related properties

3.3.1 | Properties and rewriting of the bilevel formulation

Let us consider the bilevel formulation, and let (F-MAX) denote the maximum flow problem in the second level:

$$(\mathrm{F}-MAX): \quad \max_{x \in \mathcal{X}(y,b)} \quad z_1(x) = \sum_{j \in \Gamma^+(r)} x_{rj}.$$

At this stage, y and b are already fixed; we refer to their values as $\hat{y}$ and $\hat{b}$, respectively. We propose the new following program:

$$(\mathrm{MAX}) \left| \begin{array}{lll} \max & z_2(x) = \sum_{j \in \Gamma^+(r)} x_{rj} - \sum_{(i,j) \in A} \hat{b}_{ij} x_{ij} & \\ \text{s.t.} & \sum_{i \in \Gamma^-(j)} x_{ij} - \sum_{i \in \Gamma^+(j)} x_{ji} = 0 & \forall j \in V \setminus \{r,s\} \quad (3.5a) \\ & x_{ij} \leq u_{ij} \hat{y}_{ij} & \forall (i,j) \in A \quad (3.5b) \\ & x_{ij} \geq 0 & \forall (i,j) \in A. \quad (3.5c) \end{array} \right.$$

Remark 3.3. For any value of y in $\{0,1\}^{|A|}$, there is an integral optimal solution of (MAX), and an integral optimal solution of (F-MAX) when the integrality constraint is relaxed in $\mathcal{X}(y,b)$.

Indeed, the coefficient matrix M of (MAX) is the vertex-arc incidence matrix of a directed network to which an identity matrix is appended: it is well known that such a matrix is totally unimodular [1]. Since the capacities are integers, this ensures that the extreme points of the associated polytope have integral coordinates even if the integrality constraints on x are relaxed. The integrality constraint can also be relaxed in $\mathcal{X}(y,b)$, because the coefficient matrix of $\mathcal{X}(y,b)$ is equal to M to which an identity matrix is appended, and thus it is also totally unimodular.

We have the following result:

Proposition 3.1. *There is an optimal solution x^* of (MAX) such that $\sum_{(i,j) \in A} \hat{b}_{ij} x^*_{ij} = 0$, and this solution is an optimal solution of (F-MAX). Moreover, this is true whether y and/or x are integral or not.*

Proof. We first show that there is an optimal solution x^* of (MAX) such that $\hat{b}_{ij} x^*_{ij} = 0$ for all $(i,j) \in A$. Any feasible solution of (MAX) defines a feasible flow from r to s in the digraph defined, from G, in the following way: there is an arc (i,j) if and only if $\hat{y}_{ij} > 0$. Assume that x^1 is an optimal solution of (MAX) with $\hat{b}_{vw} x^1_{vw} > 0$ for an arc $(v,w) \in A$ (notice that $\hat{y}_{vw} > 0$). There is a path π from r to s containing (v,w) and a flow of value $f > 0$ routed along π. Let x^2 be a new solution of (MAX) obtained by decreasing the flow by f along π. Then, $z_2(x^2) = (\sum_{j \in \Gamma^+(r)} x^1_{rj} - f) - (\sum_{(i,j) \in A} \hat{b}_{ij} x^1_{ij}) + \sum_{(i,j) \in \pi} \hat{b}_{ij} f$. Since $\sum_{(i,j) \in \pi} \hat{b}_{ij} f \geq f$ (because at least $(v,w) \in \pi$ is such that $\hat{b}_{ij} = 1$), we have $z_2(x^2) \geq z_2(x^1)$ and, since x^1 is optimal, $z_2(x^2) = z_2(x^1)$. Doing similar modifications while there are arcs (i,j) with $\hat{b}_{ij} x_{ij} > 0$, we obtain a solution x^* of (MAX) such that $\sum_{(i,j) \in A} \hat{b}_{ij} x^*_{ij} = 0$ and $z_2(x^*) = z_2(x^1)$. The modifications were made along paths from r to s, so Constraints (3.5a) are satisfied for x^* and, since we have decreased the values of x^1, we know that $x^*_{ij} \leq x^1_{ij}$ for all (i,j), and hence Constraints (3.5b) are satisfied too.

Let x^* be an optimal solution of (MAX) such that $\hat{b}x = 0$. Constraints (3.5a) and (3.5b) are the same as Constraints (3.1a) and (3.1b), and are satisfied for x^* in (F-MAX); moreover, $\hat{b}_{ij} x^*_{ij} = 0$ for all $(i,j) \in A$, thus $x^*_{ij} > 0$ implies that $\hat{b}_{ij} = 0$, and Constraints (3.1c) of (F-MAX) are also satisfied. If x^* is integral, x^* is an optimal solution of (MAX) that is feasible for (F-MAX), and such that $z_1(x^*) = z_2(x^*)$.

Finally, let x^* be an optimal solution of (F-MAX): x^* is clearly a feasible solution of (MAX). From Constraints (3.1c), $\hat{b}_{ij} x^*_{ij} = 0$ for all $(i,j) \in A$: thus, $\sum_{(i,j) \in A} \hat{b}_{ij} x_{ij} = 0$ and $z_1(x^*) = z_2(x^*)$.

From Remark 3.3, if $\hat{y}$ is integral, then there is an integral solution x^*, but the proposition is true even if $\hat{y}$ is not integral and if we replace "$x_{ij} \in \mathbb{N}$" by "$x_{ij} \geq 0$" in $\mathcal{X}(y,b)$. Then, x may be integral or not, but the proof remains the same. ∎

There always exists a feasible flow of value 0 in (MAX), and the problem is upper bounded by $|T|$ (because of the cutset with only the fictive arcs). Hence, the strong duality in linear programming holds, and we can introduce the dual of (MAX),

where μ is the vector of dual variables associated with Constraints (3.5a), and λ is associated with Constraints (3.5b). After a slight reformulation due to the addition of μ_r and μ_s, given $\hat{y}$ and $\hat{b}$, the dual problem can be written as follows (see [1]):

$$
(\text{DMAX}) \left| \begin{array}{lll}
\min\limits_{\lambda,\mu} & \sum\limits_{(i,j)\in A} u_{ij}\hat{y}_{ij}\lambda_{ij} & \\
\text{s.t.} & \lambda_{ij} + \hat{b}_{ij} - \mu_i + \mu_j \geq 0 & \forall (i,j) \in A \quad (3.6a)\\
& \mu_r = 1 & (3.6b)\\
& \mu_s = 0 & (3.6c)\\
& \lambda \in [0,1]^{|A|}, \quad \mu \in [0,1]^{|V|} & (3.6d)
\end{array} \right.
$$

For a given vector $\hat{b}$, we denote by $\mathcal{D}(\hat{b})$ the polytope defined by dual Constraints (3.6a)–(3.6d). The associated matrix is totally unimodular because it is the transpose of a totally unimodular matrix, and the right-hand side is an integral vector: thus, there is an optimal solution of (*DMAX*) with λ and μ integral.

Remark 3.4. (DMAX) is a special formulation of a min-cut problem where the variables (λ, μ) and $\hat{b}$ define a cut in the input digraph, whether $\hat{y}$ is integral or not. The capacity is equal to $u_{ij}\hat{y}_{ij}$ on each arc $(i,j) \in A$. μ defines the two parts of the cut (if $\mu_i = 1$, then the vertex i is in the same part as r, otherwise i is in the same part as s), while λ and $\hat{b}$ define the arcs in the cutset ((i,j) is in the cutset if and only if $\lambda_{ij} = 1$ or $\hat{b}_{ij} = 1$); in particular, $\hat{b}$ defines the (nonfictive) arcs in the cutset that are deleted. Moreover, because of the minimization of the objective function, if $\hat{b}_{ij} = 1$ then $\lambda_{ij} = 0$ in an optimal solution, and so $\lambda_{ij} + \hat{b}_{ij} = 1$ for each arc (i,j) in the cutset.

Since the second level of (BILEVEL) can be reformulated as a min-min function by using the dual described above, given $\hat{y}$, it can be rewritten as the following integer linear program:

$$
(SEC - \text{BILEVEL}) \left| \begin{array}{lll}
\min\limits_{b,\lambda,\mu} & \sum\limits_{(i,j)\in A} u_{ij}\hat{y}_{ij}\lambda_{ij} & \\
\text{s.t.} & b \in \mathcal{B} & \cdot\\
& (\lambda,\mu) \in \mathcal{D}(b) &
\end{array} \right.
$$

We can then rewrite the bilevel program as:

$$
(\text{BILEVEL} - \text{RW}) \left| \begin{array}{llll}
\min\limits_{y\in\{0,1\}^{|A|}} & \sum\limits_{(i,j)\in A} c_{ij}y_{ij} & & \\
\text{s.t.} & f(y) \geq |T| & & \\
& \text{where} \quad f(y) = & \min\limits_{b,\lambda,\mu} & \sum\limits_{(i,j)\in A} u_{ij}y_{ij}\lambda_{ij} \cdot\\
& & \text{s.t.} & b \in \mathcal{B}\\
& & & (\lambda,\mu) \in \mathcal{D}(b)
\end{array} \right.
$$

Let us consider the convex hull of the feasible points of the second-level program (defined by $\mathcal{B}$ and $\mathcal{D}$), and denote by $\mathcal{H}$ the set of its extreme points. One can notice that this convex hull does not depend on y: the set of extreme points $\mathcal{H}$ remains the same for every value of y. We denote by λ^h the value of λ at the extreme point $h \in \mathcal{H}$; from above, these points have integral coordinates. We can then reformulate the bilevel formulation as a single-level integer linear program as follows:

$$
(\text{LIN} - \text{BILEVEL}) \left| \begin{array}{lll}
\min\limits_{y} & \sum\limits_{(i,j)\in A} c_{ij}y_{ij} & \\
\text{s.t.} & \sum\limits_{(i,j)\in A} u_{ij}y_{ij}\lambda^h_{ij} \geq |T| & \forall h \in \mathcal{H} \quad (3.7)\\
& y \in \{0,1\}^{|A|} &
\end{array} \right.
$$

Constraints (3.7) ensure that, for each extreme point in $\mathcal{H}$, $\sum_{(i,j)\in A} u_{ij}y_{ij}\lambda^h_{ij}$ is greater than $|T|$, and so is the minimum value $f(y)$, meaning that the value of a maximum flow cannot become smaller than $|T|$, even after any k breakdowns.

3.3.2 | Comparison between the formulations

We have the following first result:

Theorem 3.1. *The optimal values of the continuous relaxations of the bilevel formulation (RLIN-BILEVEL) and of the cutset formulation (RCUT) are equal.*

Proof. We prove this theorem by proving that Constraints (3.7) and (3.3) are equivalent for any value $\hat{y}$ of y (integral or continuous). In both cases, the digraph is defined by $\hat{y}$.

Assume first that Constraints (3.7) are not satisfied for some $(\hat{b}, \hat{\lambda}, \hat{\mu})$ with $\hat{b} \in \mathcal{B}$ and $(\hat{\lambda}, \hat{\mu})$ in $\mathcal{D}(\hat{b})$: $(\hat{\lambda}, \hat{\mu})$ is a feasible solution of (DMAX) (for $\hat{b}$) and $\sum_{(i,j)\in A} u_{ij}\hat{y}_{ij}\hat{\lambda}_{ij} < |T|$. From Remark 3.4, we can consider a triplet $(\hat{b}, \tilde{\lambda}, \tilde{\mu})$ defining an optimal integral solution of (DMAX) (for $\hat{b}$) such that $\tilde{\lambda}_{ij} + \hat{b}_{ij} \leq 1$; we have $\sum_{(i,j)\in A} u_{ij}\hat{y}_{ij}\tilde{\lambda}_{ij} \leq \sum_{(i,j)\in A} u_{ij}\hat{y}_{ij}\hat{\lambda}_{ij}$. From Remark 3.4 again, $(\hat{b}, \tilde{\lambda}, \tilde{\mu})$ defines a cutset $\tilde{S}$ from which a set $\tilde{C}$ of k nonfictive arcs is removed: $\tilde{S} = \{(i,j)$ such that $\tilde{\lambda}_{ij} = 1$ or $\hat{b}_{ij} = 1\}$ and $\tilde{C} = \{(i,j)$ such that $\hat{b}_{ij} = 1\}$. Then $\tilde{S} \setminus \tilde{C} = \{(i,j)$ such that $\tilde{\lambda}_{ij} = 1\}$ and $\sum_{(i,j)\in\tilde{S}\setminus\tilde{C}} u_{ij}\hat{y}_{ij} = \sum_{(i,j)\in A} u_{ij}\hat{y}_{ij}\tilde{\lambda}_{ij} \leq \sum_{(i,j)\in A} u_{ij}\hat{y}_{ij}\hat{\lambda}_{ij} < |T|$. Thus, Constraints (3.3) are not satisfied for $(\tilde{S}, \tilde{C})$.

Assume now that Constraints (3.3) are not satisfied for some $\hat{S} \in S$ and $\hat{C} \in C_k^{\hat{S}}$: $\sum_{(i,j)\in\hat{S}\setminus\hat{C}} u_{ij}\hat{y}_{ij} < |T|$. Set $\hat{b}_{ij} = 1$ if and only if $(i,j) \in \hat{C}$; since $\hat{C} \in C_k^S$, $\hat{b} \in \mathcal{B}$. Then, we set $\hat{\mu}_i = 1$ if the vertex i is in the same part of the cut defined by $\hat{S}$ as r, and $\hat{\mu}_i = 0$ otherwise. This implies that the arcs of $\hat{S}$ are the arcs (i,j) such that $\hat{\mu}_i = 1$ and $\hat{\mu}_j = 0$. Finally, if $\hat{\mu}_i = 1$, $\hat{\mu}_j = 0$ and $\hat{b}_{ij} = 0$, then $\hat{\lambda}_{ij} = 1$, otherwise $\hat{\lambda}_{ij} = 0$. In this way, $\hat{\lambda}_{ij} + \hat{b}_{ij} \leq 1$, and the arcs (i,j) of $\hat{S}$ are such that $\hat{\lambda}_{ij} + \hat{b}_{ij} = 1$. We have $\sum_{(i,j)\in A} u_{ij}\hat{y}_{ij}\hat{\lambda}_{ij} = \sum_{(i,j)\in\hat{S} \text{ s.t. } \hat{b}_{ij}=0} u_{ij}\hat{y}_{ij}\hat{\lambda}_{ij} = \sum_{(i,j)\in\hat{S}\setminus\hat{C}} u_{ij}\hat{y}_{ij} < |T|$.

It is easy to check that $(\hat{\lambda}, \hat{\mu}) \in \mathcal{D}(\hat{b})$, and so $(\hat{\lambda}, \hat{\mu})$ is a feasible solution of (DMAX) (for $\hat{b}$). Then, there is an extreme point of (DMAX), $(\tilde{\lambda}, \tilde{\mu})$ such that $\sum_{(i,j)\in A} u_{ij}\hat{y}_{ij}\tilde{\lambda}_{ij} \leq \sum_{(i,j)\in A} u_{ij}\hat{y}_{ij}\hat{\lambda}_{ij} = \sum_{(i,j)\in\hat{S}\setminus\hat{C}} u_{ij}\hat{y}_{ij} < |T|$, and Constraints (3.7) are not satisfied. ∎

As a consequence, we made our tests only for the bilevel formulation, as described in Section 3.2.1 (see Section 6). We now give a second result:

Theorem 3.2. *The optimal values of the continuous relaxations of the flow formulation (RFLOW) and of the cutset formulation (RCUT) are equal.*

Proof. We first recall the continuous relaxations of the cutset and flow formulations:

$$\text{(RCUT)} \quad \left| \begin{array}{lll} \min\limits_{y} & \sum\limits_{(i,j)\in A} c_{ij}y_{ij} & \\ \text{s.t.} & \sum\limits_{(i,j)\in S\setminus C} u_{ij}y_{ij} \geq |T| & \forall S \in \mathcal{S}, \forall C \in C_k^S. \\ & 0 \leq y_{ij} \leq 1 & \forall (i,j) \in A \end{array} \right.$$

$$\text{(RFLOW)} \quad \left| \begin{array}{lll} \min\limits_{x,y} & \sum\limits_{(i,j)\in A} c_{ij}y_{ij} & \\ \text{s.t.} & \sum\limits_{i\in\Gamma^-(j)} x_{ij}^F - \sum\limits_{k\in\Gamma^+(j)} x_{jk}^F = 0 & \forall j \in V \setminus \{r,s\},\ \forall F \in \mathcal{F} \\ & \sum\limits_{t\in\Gamma^-(s)} x_{ts}^F = |T| & \forall F \in \mathcal{F} \\ & x_{ij}^F \leq u_{ij}y_{ij} & \forall (i,j) \in A,\ \forall F \in \mathcal{F} \\ & x_{ij}^F = 0 & \forall F \in \mathcal{F},\ \forall (i,j) \in F \\ & x \in \mathbb{R}_+^{|A|\times|\mathcal{F}|}, \quad y \in [0,1]^{|A|} & \end{array} \right. ,$$

Let (y^1, x^1) be a feasible solution of (RFLOW). Setting $y = y^1$ in (RCUT) would give a solution with the same value. Let us check that such a solution is feasible for (RCUT). Since (y^1, x^1) is feasible for (RFLOW), there exists a flow of value $|T|$ for each scenario of breakdown of k arcs, considering the capacity $u_{ij}y_{ij}^1$ on each arc (i,j), except obviously the arcs which are attacked in this scenario, which have a capacity equal to 0. Let us assume that y^1 is not a feasible solution for (RCUT). Then there is at least one constraint of type $\sum_{(i,j)\in S\setminus C} u_{ij}y_{ij}^1 \geq |T|$ which is violated for some $S \in \mathcal{S}$ and $C \in C_k^s$. It means that there is a $r-s$ cutset for which the capacity is smaller than $|T|$ if we delete k arcs. Then, the scenario of breakdown $F \in \mathcal{F}$ in which those k arcs are deleted would not admit a feasible flow of value $|T|$: a contradiction. So y^1 is a feasible solution of (RCUT) with the same value as (y^1, x^1) for (RFLOW).

Now let y^2 be a feasible solution of (RCUT). Setting $y = y^2$ in (RFLOW) would give a solution with the same value. Assume there does not exist any x^2 such that (y^2, x^2) is a feasible solution of (RFLOW). It means that there exists at least a scenario of breakdown $F \in \mathcal{F}$ such that there is no feasible flow of value $|T|$ on the residual network (where we delete the arcs in F) with capacities equal to $u_{ij}y_{ij}^2$ for each arc (i,j). As before, this is impossible since

this would involve a residual $r - s$ cutset with a capacity smaller than $|T|$. So, there always exists at least one x^2 such that (y^2, x^2) is a feasible solution of (RFLOW).

Hence, we can transform any feasible solution of (RFLOW) into a feasible solution of (RCUT) having the same value, and vice versa. ∎

Remark 3.5. When there are no breakdowns, the optimal value of (RBILEVEL), the continuous relaxation of (BILEVEL), is equal to the one of (RLIN-BILEVEL), (RFLOW) and (RCUT) (we shall see later, namely in Proposition 3.6, that this is no longer the case when there are arc failures, i.e., when the objective of the second level is a min-max function). Indeed, without any breakdown, (RBILEVEL) becomes $\min_{y\in[0,1]^{|A|}} \sum_{(i,j)\in A} c_{ij}y_{ij}$ such that $f(y) \geq |T|$, where $f(y) = \max_x \sum_{j\in\Gamma^+(r)} x_{rj}$, with $x \in \mathbb{N}^{|A|}$, x satisfies the flow conservation constraints, and $x_{ij} \leq u_{ij}y_{ij}$, $\forall (i,j) \in A$. The objective function is minimized and the coefficients c_{ij} are positive, so, in any optimal solution, we will have $y^*_{ij} = x^*_{ij}/u_{ij}$ and $\sum_{j\in\Gamma^+(r)} x^*_{rj} = |T|$. Then, (RBILEVEL) becomes $\min_{x\in\mathbb{N}} \sum_{(i,j)\in A}(c_{ij}/u_{ij})x^*_{ij}$ such that x satisfies the flow conservation constraints, $\sum_{j\in\Gamma^+(r)} x^*_{rj} = |T|$, and $x_{ij} \leq u_{ij}$, $\forall (i,j) \in A$ (which makes it possible to satisfy $y_{ij} \leq 1$, $\forall (i,j) \in A$). The problem becomes a minimum cost (integral) flow problem, solvable in polynomial time, and it is equivalent to the flow model without any breakdown (with $x \in \mathbb{N}$ and $y \in [0,1]^{|A|}$).

3.3.3 | The special case of uniform capacities

In the case studied in this subsection, there is a uniform capacity U on each arc $a \in A_I$, and hence the capacity of any set of k arcs is a constant equal to kU. In the bilevel formulation, this makes it possible to remove variables b: Constraints (3.6a) become $\lambda_{ij} - \mu_i + \mu_j \geq 0$ $\forall (i,j) \in A$, and the objective function of (DMAX) becomes $\sum_{(i,j)\in A} U\hat{y}_{ij}\lambda_{ij} - kU$. This reduces the dimension of $\mathcal{H}$ and the number of its extreme points. Nevertheless, the cutset formulation becomes even more interesting in this case. Indeed, it can be rewritten as follows:

$$\text{(CUT - UNIF)} \quad \left| \begin{array}{lll} \min\limits_{y} & \sum\limits_{(i,j)\in A} c_{ij}y_{ij} & \\ \text{s.t.} & \sum\limits_{(i,j)\in S} Uy_{ij} \geq |T| + kU & \forall S \in S \quad (3.8a) \\ & y_{ts} = 1 & \forall t \in T \quad (3.8b) \\ & y_{ij} \in \{0,1\} & \forall (i,j) \in A \end{array} \right.$$

The number of constraints is still exponential, but highly reduced compared to the nonuniform case, and we used this formulation to solve the problem in this special case (see the results in Section 6).

3.4 | Problem solving method

3.4.1 | Basic cutting plane approaches

We first consider the bilevel formulation (LIN-BILEVEL), in which there are an exponential number of Constraints (3.7). To tackle this issue, we chose to use a cutting plane algorithm.

We relax Constraints (3.7) by considering a subset of points in $\mathcal{H}$, and we use (SEC-BILEVEL) as the separation problem: while the optimum value of (SEC-BILEVEL) is smaller than $|T|$ for the current solution $\hat{y}$, we generate the constraint (3.7) associated with the extreme point whose coordinates are the optimal values of (b, λ, μ) in (SEC-BILEVEL). An initial subset of Constraints (3.9) is obtained by considering the cutsets formed by all the arcs incident to s or r. This provides the values of λ^1 and λ^2: we have $\lambda^1_{ri} = 1$ and $\lambda^2_{ri} = 0$ for all i adjacent to r, $\lambda^1_{js} = 0$ and $\lambda^2_{js} = 1$ for all j adjacent to s, as well as $\lambda^1_{ij} = \lambda^2_{ij} = 0$ for all $i \neq r$ and $j \neq s$.

For small values of k, one straightforward method to solve (SEC-BILEVEL) is the following: for each combination C of k arcs in A_I which are selected in the current solution, that is, such that $\hat{y}_{ij} = 1$, we compute a minimum cut in the digraph where the capacity of each arc (i,j) is defined as u_{ij}, except for the k arcs of C whose capacities are set to 0. Otherwise, that is, if k is too big, we solve a MIP.

Since, from Theorem 3.1, in the case of nonuniform capacities, (LIN-BILEVEL) and (CUT) are equivalent formulations and share the same variable space, we no longer consider the cut formulation. Concerning the flow formulation, the number of variables and constraints is exponential for arbitrary values of k, and hence we propose again a cutting plane algorithm to solve the problem. We begin with a small subset of $\mathcal{F}$. The separation problem is the problem of the k most vital links in a flow network, which is $\mathcal{NP}$-hard [37]: we search for k arcs which, once simultaneously deleted, reduce the most the value of a maximum $s - t$ flow. For small values of k, we compute a maximum $s - t$ flow for each combination of k selected arcs of A_I. If

there is a combination of arcs whose deletion results in a maximum $s-t$ flow of value smaller than $|T|$, we add this arc-failure scenario, otherwise the solution is feasible. If k is too big, we use the auxiliary MIP (SEC-BILEVEL).

3.4.2 | Improving the resolution of (LIN-BILEVEL)

We now propose a variant to generate the constraint added at each step of the cutting plane algorithm used to solve (LIN-BILEVEL), which has revealed to be more effective in solving the problem during the tests.

We explain how to get a new value of y in order to deduce a new value λ^h (i.e., a new point h), which generates a new constraint (3.7).

Given a current solution $\hat{y}$, we search for a cutset S with a minimum number of arcs in the initial support digraph G, such that S is nonvalid in the network induced by $\hat{A}$ (i.e., by the arcs (i,j) with $\hat{y}_{ij}=1$), which means that, if we remove k well-chosen arcs of $S\cap\hat{A}$, the remaining capacity of S is smaller than $|T|$. Given $\hat{y}$, this can be modeled as follows:

$$(\text{AUX}-\text{BILEVEL})\left|\begin{array}{ll}\min\limits_{b,\lambda,\mu} & \sum_{(i,j)\in A}\lambda_{ij}\\ \text{s.t.} & \sum\limits_{(i,j)\in A}u_{ij}\hat{y}_{ij}\lambda_{ij}\leq \quad |T|-1\\ & b\in\mathcal{B}\\ & (\lambda,\mu)\in\mathcal{D}(b),\ \mu\in\{0,1\}^{|V|}\end{array}\right. \tag{3.9}$$

Notice that the constraint matrix is no longer totally modular (because of Constraint (3.9)); that is why we set $\mu\in\{0,1\}^{|V|}$. Then, there always exists an optimal solution with $\lambda\in\{0,1\}^{|A|}$: for an arc (i,j), because of the minimization of the objective function, if $\mu_i=\mu_j$ or $\mu_i=0$ and $\mu_j=1$, there is an optimal solution with $\lambda_{ij}=0$; if $\mu_i=1$ and $\mu_j=0$, we must have $\lambda_{ij}+b_{ij}\geq 1$, and there always exists an optimal solution such that exactly one among λ_{ij} and b_{ij} is equal to 1.

From Remark 3.4, (AUX-BILEVEL) without Constraint (3.9) is the formulation of a cutset problem: the objective is to minimize the number of arcs of A in the cutset, provided these arcs are not deleted (since $\lambda_{ij}=0$ if $b_{ij}=1$), while Constraint (3.9) ensures that the solution is a nonvalid cutset (i.e., one with an insufficient capacity).

If (AUX-BILEVEL) has no solution, then the current solution $\hat{y}$ is a feasible solution.

If (AUX-BILEVEL) admits a solution, then this solution gives new values, $\hat{\lambda}$, $\hat{\mu}$, and $\hat{b}$ of λ, μ and b. A new value $\tilde{y}$ of y is then computed as follows: we set $\tilde{y}_{ij}=1$ for all (i,j) with $\hat{b}_{ij}=\hat{\lambda}_{ij}=0$, and leave the others at their current value ($\tilde{y}_{ij}=\hat{y}_{ij}$). Doing so, we add as many arcs as possible to the solution defined by $\hat{y}$.

Proposition 3.2. *If (AUX-BILEVEL) admits a solution, then there exists a cutset with at most $|T|-1$ arcs in the network defined by $\tilde{y}$.*

Proof. Let $g(y,\lambda)=\sum_{(i,j)\in A}u_{ij}y_{ij}\lambda_{ij}$. For any (i,j) such that $\tilde{y}_{ij}=1$ and $\hat{y}_{ij}=0$, we have $\hat{b}_{ij}=\hat{\lambda}_{ij}=0$. Thus $g(\tilde{y},\hat{\lambda})=g(\hat{y},\hat{\lambda})\leq|T|-1$, and $(\hat{\lambda},\hat{\mu},\hat{b})$ defines a cutset with at most $|T|-1$ arcs in the network induced by $\tilde{y}$. ∎

Proposition 3.3. *For any value of λ, if adding the constraint $g(\tilde{y},\lambda)\geq|T|$ to the set of constraints (3.7) of (LIN-BILEVEL) makes the solution $y=\tilde{y}$ infeasible, then it also makes the solution $y=\hat{y}$ infeasible.*

Proof. Indeed, we have $g(\tilde{y},\lambda)\geq g(\hat{y},\lambda)$ for each λ, since $\tilde{y}_{ij}\geq\hat{y}_{ij}$ for each (i,j) (recall that u and λ are nonnegative). Hence, if $g(\tilde{y},\lambda)\leq|T|-1$, then $g(\hat{y},\lambda)\leq|T|-1$. ∎

The reverse is not true as will be seen later (see Proposition 3.5).

Let us denote by $(SEC-\text{BILEVEL})(\bar{y})$ (resp. $(\text{AUX}-\text{BILEVEL})(\bar{y})$) the program $(SEC-\text{BILEVEL})$ (resp. $(\text{AUX}-\text{BILEVEL})$) where $\hat{y}$ is replaced by $\bar{y}\in\{\hat{y},\tilde{y}\}$.

Proposition 3.4. *Solving $(AUX-BILEVEL)(\tilde{y})$ is equivalent to solving $(AUX-BILEVEL)(\hat{y})$.*

Proof. Let $(\hat{\lambda},\hat{\mu},\hat{b})$ be an optimal solution of $(\text{AUX}-\text{BILEVEL})(\hat{y})$. From the proof of Proposition 3.2, $(\hat{\lambda},\hat{\mu},\hat{b})$ is a feasible solution of $(\text{AUX}-\text{BILEVEL})(\tilde{y})$.

Now, let $(\tilde{\lambda},\tilde{\mu},\tilde{b})$ be an optimal solution of $(\text{AUX}-\text{BILEVEL})(\tilde{y})$. From the proof of Proposition 3.3, $(\tilde{\lambda},\tilde{\mu},\tilde{b})$ is a feasible solution of $(\text{AUX}-\text{BILEVEL})(\hat{y})$.

Since any solution gives the same value to both programs, we get the equivalence. ∎

We have theoretically compared the use of $(SEC-\text{BILEVEL})(\hat{y})$, $(SEC-\text{BILEVEL})(\tilde{y})$, and $(\text{AUX}-\text{BILEVEL})(\hat{y})$, to solve (LIN-BILEVEL), without being able to come up with a theoretical conclusion on their respective effectiveness. More precisely, we prove:

Proposition 3.5. *Given any of the three feasible sets of the continuous relaxations of (LIN-BILEVEL) obtained after adding a new constraint generated by solving* $(SEC-BILEVEL)(\hat{y})$, $(SEC-BILEVEL)(\tilde{y})$, *or* $(AUX-BILEVEL)(\hat{y})$, *this feasible set is not always included in one of the two others, even without any breakdown.*

In order to ease the reading, corollaries and proofs related to this proposition have been moved to Appendix A. We discuss in Section 6 the use of (AUX − BILEVEL) and (*SEC* − BILEVEL) for generating the constraints added in the cutting plane algorithm, based on preliminary empirical results.

Remark 3.6. Another possible approach would be to bundle Constraints (3.1b) and (3.1c) into $x_{ij} \leq u_{ij}(y_{ij} - b_{ij})$. But, if, for a given (i,j), we have $b_{ij} = 1$ and $y_{ij} = 0$, then $\mathcal{X}(y,b)$ is empty, and the max problem of the second level has no feasible solution, which leads the optimal value of (DMAX) to be unbounded in this case. Moreover, if we add the set of constraints $b_{ij} \leq y_{ij}$ in the feasible set $\mathcal{B}$ to tackle this issue, there are constraints depending on y in the second level, and our way of solving the bilevel program is not suitable anymore.

The cutting plane algorithm used to solve the problem, as reported in Section 6, is based on the model (*LIN* − *BILEVEL*). Nevertheless, we have the following result:

Proposition 3.6. *Let opt(RBILEVEL) (resp. opt(RLIN* − *BILEVEL)) be the optimal value of (BILEVEL) (resp. (LIN* − *BILEVEL)) when we relax the integrality constraints on y. Then, we have opt(RBILEVEL)* ≥ *opt(RLIN* − *BILEVEL).*

Proof. If y is not required to be integral, the right-hand side of each constraint of the program (MAX) is possibly not integral and, although the constraint matrix is totally unimodular, x may not be integral. The formulation (RLIN-BILEVEL) is based on the dualization of (MAX), and, thus, it requires us to relax the integrality of the variable x in the second level of (BILEVEL), in addition to the one of y, while that is not required to formulate (RBILEVEL).

More formally, from Proposition 3.1, with $y \in [0,1]^{|A|}$ and $x \in \mathbb{N}^{|A|}$, the continuous relaxation of (BILEVEL), denoted by (RBILEVEL), can be written as:

$$\min_{y\in[0,1]^{|A|}} \sum_{(i,j)\in A} c_{ij}y_{ij} \text{ such that } f(y) \geq |T|, \text{ with } f(y) = \min_{b\in\mathcal{B}} \max_{x\in\mathcal{X}(y,b)} \sum_{j\in\Gamma^+(r)} x_{rj} - \sum_{(i,j)\in A} b_{ij}x_{ij}.$$

From the rewriting given before, (RLIN-BILEVEL), the continuous relaxation of (LIN-BILEVEL), is equivalent to:

$$\min_{y\in[0,1]^{|A|}} \sum_{(i,j)\in A} c_{ij}y_{ij} \text{ such that } g(y) \geq |T| \text{ with } g(y) = \min_{b\in\mathcal{B}} \min_{\{\lambda,\mu\}\in D(b)} \sum_{(i,j)\in A} u_{ij}y_{ij}\lambda_{ij}.$$

Moreover, by duality, we have:

$$\min_{\{\lambda,\mu\}\in D(b)} \sum_{(i,j)\in A} u_{ij}y_{ij}\lambda_{ij} = \max_{x\in R\mathcal{X}(y,b)} \sum_{j\in\Gamma^+(r)} x_{rj} - \sum_{(i,j)\in A} b_{ij}x_{ij},$$

where, for any given vectors y and b of size $|A|$, $R\mathcal{X}(y,b)$ is equal to $\mathcal{X}(y,b)$ with "$x_{ij} \in \mathbb{N},\ \forall (i,j) \in A$" replaced by "$x_{ij} \geq 0,\ \forall (i,j) \in A$". Thus, $\min_{\{\lambda,\mu\}\in D(b)} \sum_{(i,j)\in A} u_{ij}y_{ij}\lambda_{ij} \geq \max_{x\in\mathcal{X}(y,b)} \sum_{j\in\Gamma^+(r)} x_{rj} - \sum_{(i,j)\in A} b_{ij}x_{ij},\ \forall b \in \mathcal{B}$.

For any value of y, we have $g(y) \geq f(y)$, and hence any feasible solution of (RBILEVEL) (i.e., satisfying $f(y) \geq |T|$) is a feasible solution of (RLIN-BILEVEL) (i.e., satisfying $g(y) \geq |T|$), which implies that opt(RBILEVEL) ≥ opt(RLIN − BILEVEL). The opposite is not always true: indeed, let y^* be an optimal nonintegral solution of (RLIN-BILEVEL); then, the capacities in $\mathcal{X}(y,b)$ are not integers, and the value of a maximum continuous flow can be greater than the one of a maximum integer flow. Hence, we can have $g(y^*) \geq |T|$ while $f(y^*) < |T|$, which makes the solution y^* infeasible for (RBILEVEL). ∎

4 | ADDITION OF PROTECTED ARCS

4.1 | Problem definition

Let us now define an even more general version of the problem, where we add the possibility of protecting $k' \geq 0$ arcs. In this version, in addition to A', we also select a subset $A'_p \subseteq A'$ with $|A'_p| = k' \leq |A|$; those arcs are called *protected arcs* and cannot fail or be deleted by the attacker. In the wind farm application, protecting arcs can be seen as doubling a set of cables or protecting cables from a difficult environment (like extreme cold).

With the addition of protected arcs, Property 3.1 in Section 3.1 does not hold anymore: if some arcs are protected, then the existence of a feasible solution does not necessarily imply that there are $k+1$ arc-disjoint paths between the root and each terminal. For example, $k+1$ paths which are pairwise arc-disjoint except for the fact that they share a common arc (u, t_1) can be

sufficient to ensure that the terminal t_1 can be reached from the root even after any k arc deletions, if the arc (u, t_1) is protected and capacities are sufficient. Hence, Property 3.1 and Remark 3.1 can be ignored here.

4.2 | Formulations in the case of protected arcs

In that case, it could become impossible to delete k arcs as in Section 3.2, and we need to slightly modify the definition of $\mathcal{B}$ by replacing the equality with an inequality:

$$\mathcal{B} = \left\{ b \in \{0,1\}^{|A|} \mid \sum_{(i,j)\in A} b_{ij} \leq k \; ; \; \sum_{t\in T} b_{ts} = 0 \right\}.$$

We also define the binary variables $p_{ij} = 1$ if and only if (i,j) is protected, for each arc $(i,j) \in A$, as well as the following set, for a given vector y of size $|A|$:

$$\mathcal{P}(y) = \left\{ p \in \{0,1\}^{|A|} \mid \sum_{(i,j)\in A} p_{ij} \leq k' \; ; \sum_{t\in T} p_{ts} = 0 \; ; \; p_{ij} \leq y_{ij} \quad \forall (i,j) \in A \right\}.$$

This set ensures that there are at most k' protected arcs, and that we do not protect arcs which are not selected in the final network or which cannot be deleted.

Remark 4.1. Since the budget of protection is fixed, there exists an optimal solution such that $\sum_{(i,j)\in A} p_{ij} = \min\{k', \sum_{(i,j)\in A_I} y_{ij}\} = \min(k', |A'|)$. Hence, if $\sum_{(i,j)\in A_I} y_{ij} \leq k'$, then $p_{ij} = y_{ij}$ for each $(i,j) \in A_I$.

In the following, we propose modifications for each one of the previous formulations.

4.2.1 | Bilevel formulation

In order to include the possibility of protecting arcs, we introduce, for a given vector p of size $|A|$, the set $\mathcal{B}(p)$, defined as

$$\mathcal{B}(p) = \left\{ b \in \mathcal{B} \text{ such that } \quad b_{ij} \leq 1 - p_{ij} \quad \forall (i,j) \in A \right\},$$

which forbids the attacker to delete a protected arc.

The bilevel formulation introduced in Section 3.2.1 can be adapted to the protected case as follows:

$$\text{(PBILEVEL)} \left| \begin{array}{lll} \min\limits_{y\in\{0,1\}^{|A|}, p\in\mathcal{P}(y)} & \sum\limits_{(i,j)\in A} c_{ij} y_{ij} & \\ \text{s.t.} & f(y,p) \geq |T| & (4.1a) \\ & \text{where } f(y,p) = \min\limits_{b\in\mathcal{B}(p)} \max\limits_{x\in\mathcal{X}(y,b)} \sum\limits_{j\in\Gamma^+(r)} x_{rj} & (4.1b) \end{array} \right.$$

The max problem of the second level is exactly the same as in the unprotected case. By dualizing this max problem as in Section 3.2.1, for given values $\hat{y}$ and $\hat{p}$ of y and p, the reformulated second level problem is the following:

$$\text{(PSEC} - \text{BILEVEL)} \left| \begin{array}{lll} \min\limits_{b,\lambda,\mu} & \text{fbil}(\lambda) = \sum\limits_{(i,j)\in A} u_{ij} \hat{y}_{ij} \lambda_{ij} & \\ \text{s.t.} & b \in \mathcal{B}(\hat{p}) & (4.2a) \\ & \lambda_{ij} + b_{ij} - \mu_i + \mu_j \geq 0 \qquad \forall (i,j) \in A & (4.2b) \\ & \mu_r = 1, \quad \mu_s = 0 & (4.2c) \\ & \lambda \in [0,1]^{|A|}, \quad \mu \in [0,1]^{|V|} & (4.2d) \end{array} \right.$$

For an optimal value λ^* of λ in (PSEC-BILEVEL), we have $fbil(\lambda^*) = f(\hat{y}, \hat{p})$. There is a crucial difference between (PSEC-BILEVEL) and (SEC-BILEVEL) defined in Section 3.2.1: now, the convex hull of the second-level feasible set depends on the first-level variables p (see Constraints (4.2a)).

4.2.2 | Cutset formulation

The cutset formulation proposed in Section 3.2.2 can be adapted to the protected case by replacing Constraints (3.3) by the following constraints: $\sum_{(i,j)\in S} u_{ij} y_{ij} - \sum_{(i,j)\in C} u_{ij} y_{ij} (1 - p_{ij}) \geq |T| \quad \forall S \in \mathcal{S}, \; \forall C \in C^S_{\leq k}$, where $\mathcal{S}$ is defined as in Section 3.2.2, and $C^S_{\leq k}$ is the set of subsets of S containing no more than k nonfictive arcs of S. These constraints ensure that, in the selected digraph, the capacity of each cutset minus the capacity of at most k arcs of this cutset, provided that they are unprotected, is

always larger than $|T|$. They are not linear, but we also add the constraint $p \in \mathcal{P}(y)$, which implies that $p_{ij} \leq y_{ij}$ and, y and p being 0-1 variables, that $p_{ij}y_{ij} = p_{ij}$ for all $(i,j) \in A$. We obtain the following linear cutset formulation:

$$\text{(PCUT)} \quad \left| \begin{array}{lll} \min\limits_{y\in\{0,1\}^{|A|}, p\in\mathcal{P}(y)} & \sum\limits_{(i,j)\in A} c_{ij}y_{ij} & \\ \text{s.t.} \sum\limits_{(i,j)\in S} u_{ij}y_{ij} - \sum\limits_{(i,j)\in C} u_{ij}(y_{ij} - p_{ij}) & \geq \quad |T| & \forall S \in \mathcal{S},\ \forall C \in C^S_{\leq k} \end{array} \right. \tag{4.3}$$

In order to solve the resulting MIP using a constraint generation algorithm, we search for a cutset S of minimum residual capacity once we delete its most capacitated arcs, taking into account the fact that the capacities of the protected arcs cannot be removed and that the maximum number of deleted arcs is k. If the capacity of this minimum cutset is smaller than $|T|$, then we add the constraint associated with S. Otherwise, the algorithm stops for the current node of the branch-and-bound.

Let $\hat{y} \in \{0,1\}^{|A|}$ and $\hat{p} \in \mathcal{P}(\hat{y})$ be the current solution; a minimum cutset is obtained by solving the following MIP (PCUTSEP):

$$\text{(PCUTSEP)} \quad \left| \begin{array}{lll} \min\limits_{b,\lambda,\mu} & \text{fcut}(b,\lambda) = \sum\limits_{(i,j)\in A} u_{ij}(\hat{y}_{ij}\lambda_{ij} + \hat{p}_{ij}b_{ij}) & \\ \text{s.t.} & b \in \mathcal{B} & \text{(4.4a)} \\ & \lambda_{ij} + b_{ij} - \mu_i + \mu_j \geq 0 \qquad \forall (i,j) \in A & \text{(4.4b)} \\ & \mu_r = 1 \quad \mu_s = 0 & \text{(4.4c)} \\ & \lambda \in [0,1]^{|A|}, \quad \mu \in [0,1]^{|V|} & \text{(4.4d)} \end{array} \right.$$

Constraints (4.4b)–(4.4d) ensure that b, λ, and μ provide a cutset in the current network: $\hat{S} = \{(i,j) \text{ s.t. } \lambda_{ij} + b_{ij} = 1\}$ defines the arcs of the cutset (as in Remark 3.4, if $b_{ij} = 1$, then we have $\lambda_{ij} = 0$ in any optimal solution). Constraints (4.4a) ensure that at most k nonfictive arcs are "deleted": here, "deleted" means "selected to be deleted," but actually deleted only if nonprotected, that is, the capacity of an arc (i,j) such that $b_{ij} = 1$ is removed only if it is not protected. Recall that $y_{ij}p_{ij} = p_{ij}$. Thus, the remaining capacity of a cutset is $\sum_{(i,j)\in A}(u_{ij}\hat{y}_{ij}(\lambda_{ij} + b_{ij}) - u_{ij}\hat{y}_{ij}b_{ij}(1 - \hat{p}_{ij})) = \sum_{(i,j)\in A} u_{ij}(\hat{y}_{ij}\lambda_{ij} + \hat{p}_{ij}b_{ij})$.

We have the following property:

Property 4.1. For a given $(\hat{y}, \hat{p})$, there is a bijection between the sets (S, C) in Constraints (4.3) and feasible solutions (b, λ) in (*PCUTSEP*). Moreover, if (S^1, C^1) corresponds to (b^1, λ^1), then $\sum_{(i,j)\in S^1} u_{ij}\hat{y}_{ij} - \sum_{(i,j)\in C^1} u_{ij}(\hat{y}_{ij} - \hat{p}_{ij}) = \sum_{(i,j)\in A} u_{ij}(\hat{y}_{ij}\lambda^1_{ij} + \hat{p}_{ij}b^1_{ij})$.

From the definition of $\mathcal{B}$ given at the beginning of Section 4, we make the following remark:

Remark 4.2. When arcs can be protected, the case of uniform capacities does not admit a reformulation similar to (CUT-UNIF) (see Section 3.2.2), because in this case $\sum_{(i,j)\in A} b_{ij} = k$ does not necessarily hold, and so the capacity of a cut after considering the worst breakdowns is not always decreased by kU.

4.2.3 | Flow formulation

For the flow formulation, we define $\mathcal{F}$ as the set of all combinations of *at most* k arcs of A, we add to the MIP (*FLOW*) of Section 3.2.3 the constraint $p \in \mathcal{P}(y)$, and we replace Constraints (3.4d) by:

$$x^F_{ij} \leq u_{ij}p_{ij} \qquad \forall F \in \mathcal{F},\ \forall (i,j) \in F.$$

Those constraints ensure that, in a scenario F with $(i,j) \in F$, we can route some flow through arc (i,j) only if this arc is protected. Again, we can use the same columns-and-constraints generation algorithm as in Section 3.2.3, in order to find the most vital arcs in the separation problem among the nonfictive and nonprotected arcs (i.e., we consider only combinations of selected but nonprotected arcs when computing the set of maximum flows).

4.3 | Theoretical comparison between formulations

We first compare the continuous relaxations (relative to y and p) of the bilevel and cutset formulations from a theoretical point of view.

We denote by $\mathcal{RP}(y)$ the polytope defined by $\mathcal{P}(y)$ where $p \in \{0,1\}^{|A|}$ is replaced by $p \in [0,1]^{|A|}$, and we consider the continuous relaxation (RPBILEVEL) of (PBILEVEL), and (RPCUT) of (PCUT), with $y \in [0,1]^{|A|}$ and $p \in \mathcal{RP}(y)$.

Let us take the defender's point of view in (RPBILEVEL): for a given value of an optimal solution y^*, an optimal value of p^* will be such that $f(y^*, p^*)$ takes the highest possible value. This is obtained, for instance, by setting $p^*_{ij} = \min\left(\frac{k'}{|A^*|}, y^*_{ij}\right)$ for all $(i,j) \in A$, where $A^* = \{(i,j) \in A_I \text{ such that } y^*_{ij} > 0)\}$, which gives $\sum_{(i,j)\in A} p_{ij} \leq k'$ and $p \in \mathcal{RP}(y)$. In this way, $b \in \mathcal{B}(p^*)$ implies that $b_{ij} \leq 1 - p^*_{ij} < 1$, and thus $b_{ij} = 0$, for each $(i,j) \in A^*$. Hence, there is an optimal solution of (PSEC-BILEVEL) level such that $b_{ij} = 0$ for each $(i,j) \in A$: (PSEC-BILEVEL) becomes a simple min-cut problem in G, with a capacity equal to $u_{ij}y^*_{ij}$ on each arc (i,j). The relaxed bilevel program with protected arcs can be written as follows:

$$\text{(RPBILEVEL)} \left| \begin{array}{ll} \min\limits_{y\in[0,1]^{|A|}, p\in\mathcal{RP}(y)} & \sum\limits_{(i,j)\in A} c_{ij}y_{ij} \\ \text{s.t.} \sum\limits_{(i,j)\in A} u_{ij}y_{ij}\lambda_{ij} & \geq |T| \qquad \forall(\lambda,\mu) \text{ defining a } r\text{-}s \text{ cut in A} \end{array} \right. \tag{4.5}$$

or, equivalently,

$$\text{(RPBILEVEL)} \left| \begin{array}{lll} \min\limits_{y\in[0,1]^{|A|}, p\in\mathcal{RP}(y)} & \sum\limits_{(i,j)\in A} c_{ij}y_{ij} & \\ \text{s.t.} \sum\limits_{(i,j)\in S} u_{ij}y_{ij} & \geq \quad |T| & \forall S \in \mathcal{S} \end{array} \right. \tag{4.6}$$

If we compare this formulation with the bilevel formulation without protected arcs (see the programs (LIN-BILEVEL) and (AUX-BILEVEL) in Section 3.2.1), the continuous relaxation of the problem with breakdowns and protected arcs is easier to solve than the problem with breakdowns only, but we shall see that it does not give a good bound in the branch-and-bound algorithm. Indeed, we have the following theorem, where opt(PROG) denotes the optimal value of the program (PROG):

Theorem 4.1. *Let (RPBILEVEL) and (RPCUT) be the continuous relaxations of (PBILEVEL) and (PCUT), respectively, with $y \in [0,1]^{|A|}$ and $p \in \mathcal{RP}(y)$. Then, opt(RPCUT) $\geq$ opt(RPBILEVEL), and there are some instances for which opt(RPCUT) > opt(RPBILEVEL).*

Proof. Let us consider the continuous relaxation of the cutset formulation:

$$\text{(RPCUT)} \left| \begin{array}{ll} \min\limits_{y\in[0,1]^{|A|}, p\in\mathcal{RP}(y)} & \sum\limits_{(i,j)\in A} c_{ij}y_{ij} \\ \text{s.t.} \sum\limits_{(i,j)\in S} u_{ij}y_{ij} - \sum\limits_{(i,j)\in C} u_{ij}(y_{ij} - p_{ij}) & \geq \quad |T| \qquad \forall S \in \mathcal{S},\ \forall C \in C^S_{\leq k} \end{array} \right. \tag{4.7}$$

We have $p_{ij} \leq y_{ij}$ and so $\sum_{(i,j)\in C} u_{ij}(y_{ij} - p_{ij}) \geq 0$, which implies that, if Constraints (4.8) are satisfied, then Constraints (4.7) are satisfied too, and thus $opt(RPCUT) \geq opt(RPBILEVEL)$.

The opposite is not always true. Let us consider the following instance: a digraph $G = (V, A)$ with two terminal vertices t_1 and t_2, a root r, four Steiner vertices v_1, v_2, v_3, v_4, four paths rv_1t_1, rv_2t_1, rv_3t_2, rv_4t_2, and a sink s with the arcs (t_1, s) and (t_2, s). We set a capacity $u_{ij} = 1$ on each arc, a cost $c_{ij} = 1$ for each arc in A_I ($c_{t_1s} = c_{t_2s} = 0$), and $k = k' = 1$ (i.e., at most one breakdown and one protected arc). If there is an arc of G missing in $G' = (V, A')$ (i.e., $\exists\ (i,j) \in A_I$ s.t. $y_{ij} = 0$), then w.l.o.g., between r and t_1, there is only one remaining path from r to t_1 and, since we can protect only one arc, an arc of this path can always be deleted. Thus, an optimal integer solution requires $y_{ij} = 1$ for all $(i,j) \in A$, and its cost is 8.

Now, let us consider the following nonintegral solution: $\hat{y}_{ij} = \frac{1}{2}$ and $\hat{p}_{ij} = \frac{1}{8}$ on each arc $(i,j) \in A_I$, $\hat{y}_{t_1s} = \hat{y}_{t_2s} = 1$, and $\hat{p}_{t_1s} = \hat{p}_{t_2s} = 0$.

For the bilevel formulation, we have $b_{ij} = 0$ for all $(i,j) \in A$ (since b is integer and $b \leq p < 1$) and $(\hat{y}, \hat{p})$ is a feasible solution of (RPBILEVEL) of cost 4. In fact, this is an optimal solution: considering two of the cutsets, we must have $y_{rv_1} + y_{rv_2} + y_{rv_3} + y_{rv_4} \geq |T| = 2$ and $y_{v_1t_1} + y_{v_2t_1} + y_{v_3t_2} + y_{v_4t_2} \geq |T| = 2$, which implies, by summing these two inequalities, that $\sum_{(i,j)\in A_I} y_{ij} \geq 4$ and so $\sum_{(i,j)\in A} c_{ij}y_{ij} \geq 4$.

We remark that this solution is infeasible for (RPCUT): indeed, the solution given by $b_{rv_1} = 1$, $b_{ij} = 0$ for all $(i,j) \neq (r, v_1)$, $\mu_i = 1$ if $i \in \{r, v_3, v_4, t_2\}$ and $\mu_i = 0$ otherwise, $\lambda_{rv_2} = \lambda_{t_2s} = 1$ and $\lambda_{ij} = 0$ for all $(i,j) \notin \{(r, v_2), (t_2, s)\}$, is a feasible solution of (PCUTSEP) of value $1 + \frac{1}{2} + \frac{1}{8} < 2 = |T|$.

In fact, we show that setting $\hat{y}_{ij} = \frac{7}{8}$, $\hat{p}_{ij} = \frac{1}{8}$ on each arc $(i,j) \in A_I$, $\hat{y}_{t_1s} = \hat{y}_{t_2s} = 1$ and $\hat{p}_{t_1s} = \hat{p}_{t_2s} = 0$, yields an optimal solution of (RPCUT). When solving (PCUTSEP), all the arcs in A_I are equivalent, and each cutset with three or four arcs is such that $b_{ij} = 1$ for one arc of the cutset and $\lambda_{ij} = 1$ for all the others. Any cutset of three arcs contains (t_1, s) (or (t_2, s)) and two arcs of A_I, one on each path from r to t_2 (or to t_1), which gives $\text{fcut} = 1 + \frac{7}{8} + \frac{1}{8} = 2$.

Any cutset of four arcs gives fcut $= 3\frac{7}{8} + \frac{1}{8} > 2$. Finally, there is one cutset of two arcs $\{(t_1, s), (t_2, s)\}$, which gives fcut $= 2$. Thus, this solution is a feasible solution of (RPCUT). Let us check that it is optimal. Let us consider a cutset of three arcs, for instance $S = \{(t_1, s), (r, v_3), (r, v_4)\}$: from Constraint (4.3) and Property 4.1 with $C_1 = \{(r, v_3)\}$ and $C_2 = \{(r, v_4)\}$, we must have, respectively, $y_{t_1 s} + y_{rv_4} + p_{rv_3} \geq |T| = 2$ and $y_{t_1 s} + y_{rv_3} + p_{rv_4} \geq 2$, and hence, since $y_{t_1 s}$ must be equal to 1, $y_{rv_4} + p_{rv_3} \geq 1$ and $y_{rv_3} + p_{rv_4} \geq 1$. Writing similar inequalities for each one of the eight cutsets of three arcs, and summing all these inequalities, we obtain $2(\sum_{(i,j)\in A_I} y_{ij} + \sum_{(i,j)\in A_I} p_{ij}) \geq 16$ and, since $\sum_{(i,j)\in A_I} p_{ij} \leq k' = 1$ (see Remark 4.1), we get $\sum_{(i,j)\in A_I} y_{ij} \geq 7$, and so any feasible solution has a cost at least equal to 7. Thus, the solution $\hat{y}_{ij} = \frac{7}{8}$ and $\hat{p}_{ij} = \frac{1}{8}$ on each arc $(i, j) \in A_I$ is an optimal continuous solution, since its cost is $8.\frac{7}{8} = 7$, close to the cost of the optimal integer solution (8). ∎

Finally, using a proof very similar to the one given for Theorem 3.2, we have the following theorem:

Theorem 4.2. *The optimal values of the continuous relaxations of the flow formulation (RPFLOW) and of the cutset formulation (RPCUT) are equal.*

5 | PREPROCESSING AND ADDITION OF VALID AND STRENGTHENING INEQUALITIES

In this section, we propose some preprocessing and strengthening inequalities for both formulations, in order to enhance the quality of the lower bound obtained by solving the continuous relaxation. We first consider the case where we are not allowed to protect some arcs of the network. Secondly, we propose modifications of those inequalities to take the possibility of protecting arcs into account.

5.1 | Preprocessing

The preprocessing we apply is adapted from Koch and Martin's for Steiner trees without capacities [29]. First, any Steiner vertex of degree one is removed. Secondly, if a Steiner vertex v has degree two, the two incident arcs (u, v) and (v, w), $u \neq w$, can be replaced by an arc (u, w) of cost $c_{uw} = c_{uv} + c_{vw}$ and capacity $\min(u_{uv}, u_{vw})$. This reduction can be done because, in a worst case of failure, there is no interest in selecting the two arcs (u, v) and (v, w). Thirdly, any arc entering a terminal vertex of degree one is always in an optimal solution. We also suppress any arc entering the root.

5.2 | Case without the possibility of protecting arcs

Inequalities (5.1a) ensure that there are at least $k + 1$ arcs entering each terminal. Indeed, if there are at most k arcs entering it, then it is possible to delete all of them and thus prevent one unit of flow from reaching the sink. Inequality (5.1b) states the same constraint for the arcs leaving the root.

$$\sum_{(i,t)\in A} y_{it} \geq k + 1 \qquad \forall t \in T \tag{5.1a}$$

$$\sum_{(r,i)\in A} y_{ri} \geq k + 1. \tag{5.1b}$$

Both Inequalities (5.1a) and (5.1b) are valid and cut off some nonintegral solutions. In Figure 2, we have $T = \{t_1\}$, and let u_i be the capacity of a_i for $i = 1, .., 5$, and y_i the variable associated with the selection of a_i. The constraints associated with this digraph for the cutset formulation with $k = 1$ are $u_1 y_1 \geq 1$, $u_2 y_2 \geq 1$, $u_3 y_3 \geq 1$, $u_4 y_4 \geq 1$ and $y_5 = 1$. If we set $u_i = 2$ for all $i = 1, .., 4$ ($u_5 = 1$ because a_5 is a fictive arc), then we have that the solution in which $y_1 = y_2 = y_3 = y_4 = 0.5$ and $y_5 = 1$

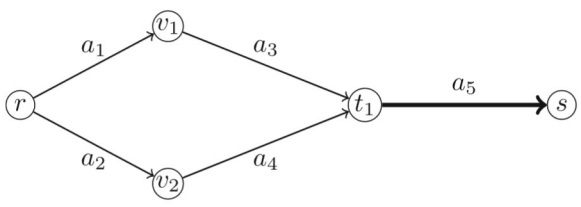

FIGURE 2 A digraph where Inequalities (5.1a) and (5.1b) cut off some nonintegral solutions.

is optimal for the continuous relaxation, since we consider positive costs in the objective function. Inequalities (5.1a) are then violated and impose that $y_3 + y_4 \geq 2$ (i.e., $y_3 = y_4 = 1$). Similarly, Inequalities (5.1b) impose that $y_1 + y_2 \geq 2$ (i.e., $y_1 = y_2 = 1$). In this case, the addition of both inequalities results in cutting off nonintegral solutions (here, it even results in an optimal value of the integer problem equal to the optimal value of the obtained continuous relaxation).

Inequalities (5.2a) state that, for each Steiner vertex j, if an arc entering j is selected, then at least one arc leaving j must be selected, since all arc costs are assumed to be positive. Inequalities (5.2b) state the same for arcs leaving a Steiner vertex j [29]. Notice that these inequalities cut off some integral but nonoptimal feasible solutions.

$$y_{ij} \leq \sum_{k \in \Gamma^+(j) \setminus \{i\}} y_{jk} \qquad \forall j \in V \setminus \{T \cup \{r\}\},\ \forall i \in \Gamma^-(j) \tag{5.2a}$$

$$y_{jk} \leq \sum_{i \in \Gamma^-(j) \setminus \{k\}} y_{ij} \qquad \forall j \in V \setminus \{T \cup \{r\}\},\ \forall k \in \Gamma^+(j). \tag{5.2b}$$

Finally, we have the inequalities (5.3a), since all the costs are positive, and any solution (x, y) with $y_{ij} = y_{ji} = 1$ can be transformed into a solution (x', y') with either $y'_{ij} = 1, y'_{ji} = 0, x'_{ij} = x_{ij} - x_{ji}, x'_{ji} = 0$ if $x_{ij} \geq x_{ji}$ or $y'_{ij} = 0, y'_{ji} = 1, x'_{ij} = 0, x'_{ji} = x_{ji} - x_{ij}$ if $x_{ij} \leq x_{ji}$. These inequalities cut off some integral but nonoptimal feasible solutions.

$$y_{ij} + y_{ji} \leq 1 \qquad \forall \{(i,j),(j,i)\} \subset A^2. \tag{5.3a}$$

5.3 | Case with the possibility of protecting arcs

The two families of Inequalities (5.1a) and (5.1b) are only true for the case without protection ($k' = 0$). Since one arc may be sufficient to ensure that one of the terminals is not isolated if it is protected, we can replace Inequalities (5.1a) and (5.1b) by (5.4a-5.4b) and (5.4c-5.4d) in this case. Inequalities (5.4a-5.4b) state that, for any terminal t, if there are no protected arcs entering t, there must be at least $k + 1$ arcs entering t, and otherwise there must be at least one. Inequalities (5.4c) and (5.4d) state the same constraint for the arcs leaving the root.

$$\sum_{i \in \Gamma^-_G(t)} y_{it} \geq 1 + (k(1 - \sum_{i \in \Gamma^-_G(t)} p_{it})) \qquad \forall t \in T \tag{5.4a}$$

$$\sum_{i \in \Gamma^-_G(t)} y_{it} \geq 1 \qquad \forall t \in T \tag{5.4b}$$

$$\sum_{i \in \Gamma^+_G(r)} y_{ri} \geq 1 + \left(k \left(1 - \sum_{i \in \Gamma^+_G(r)} p_{ri} \right) \right) \tag{5.4c}$$

$$\sum_{i \in \Gamma^+_G(r)} y_{ri} \geq 1. \tag{5.4d}$$

Inequalities (5.5) state that at least one arc entering a terminal t must be protected if there are less than $k + 1$ arcs entering t.

$$\sum_{i \in \Gamma^-_G(t)} p_{it} \geq 1 \qquad \forall t \in T \text{ with } |\Gamma^-(t)| \leq k. \tag{5.5}$$

Inequalities (5.2a), (5.2b), and (5.3a) are still valid inequalities for the case with the possibility of protecting arcs.

6 | RESULTS ANALYSIS

In this section, we present the results of the tests for the formulations proposed previously.

6.1 | Experimental results for robust arborescences

We tested the formulations proposed in Section 2 on real wind farm datasets. All experiments were performed on a computer with a 2.40 GHz Intel(R) Core(TM) i7-5500U CPU and 16 GB RAM, using CPLEX version 12.6.1 as MILP-solver, interfaced with Julia 0.6.0. Even if the number of instances is small, the results are interesting to analyze, and we can compare the robustness, costs and structures of the solutions. Data parameters and results are available, respectively, in Table 1A,B. Figure 3 allows one to visually compare the arborescences obtained according to the different models for the fourth data set (the filled circles correspond to terminals).

Figure 3A gives an optimal (nonrobust) capacitated Steiner arborescence (optimal solution of (CStA)); let us denote its cost by C^*. This arborescence cannot be qualified as robust with respect to R-robustness since, in the worst case, all terminals can

TABLE 1 Data parameters and results on robust arborescences.

(A) Data parameters				(B) Results on robust arborescences						
Set	$\|V\|$	$\|E\|$	$\|T\|$	Set	R^*	R_8	R_{12}	Δ_{Crob}	Δ_{Cbrob}	R_{BR^*}
1	91	220	42	1	21	35	29	0.18	0.56	21
2	143	382	40	2	20	21	20	0.09	0.64	20
3	220	510	88	3	22	32	30	0.24	0.33	22
4	255	662	73	4	37	41	38	0.19	0.37	37

be disconnected by deleting the only arc incident to the root. Furthermore, the tree has a large depth, and hence its B-robustness is not good either. This proves the importance of searching for a more robust solution. We consider first R-robustness (i.e., (RCStA)), and we denote by R^* the best possible R-robustness, that is, the minimum value of the loss of terminals in the worst case of a single arc deletion. See Figure 3B for the associated solution on the test instance. Then, to obtain the minimum cost of a most robust solution, denoted by $C^*_{R^*}$, we solve ($\text{CStA}_{\text{bounded}-R\_\text{robust}}$) with $R = R^*$: notice that the constraint is saturated in any optimal solution. Then, $\Delta_{\text{Crob}} = (C^*_{R^*} - C^*)/C^*$ represents the "cost of robustness," that is, the percentage of increase of the cost to get a R-robust solution.

In the same way, let BR^* be the best possible B-robustness (optimal value of (BRCStA), not given in Table 1); see Figure 3C for the associated solution on the test instance. The cost of a solution with the best B-robustness, denoted by $C^*_{\text{BR}^*}$, is obtained by solving ($\text{CStA}_{\text{bounded}-\text{balanced\_robust}}$) with $\text{BR} = \text{BR}^*$, and $\Delta_{\text{Cbrob}} = (C^*_{\text{BR}^*} - C^*)/C^*$ represents the "cost of B-robustness," that is, the percentage of increase of the cost of a nonrobust arborescence to get a balanced robust solution.

We also study the behavior of R-robustness when we bound the cost to a value close to the one of an optimal nonrobust arborescence: R_8 (resp. R_{12}) corresponds to the optimal value of ($\text{RCStA}_{\text{bounded}-cost}$) with a bound $C = 1.08C^*$ (resp. $C = 1.12C^*$). Such values of C were fixed after preliminary experiments, in order to obtain a good trade-off between a substantial improvement for R-robustness and an acceptable increase in the cost.

We now analyze the results. The cost of R-robustness is quite variable on those instances (from 9% to 24%) but remains rather low. On the contrary, we can see that the optimization of the B-robustness is considerably more expensive (increase from 33% to 64% of the cost), because it involves significantly more arcs (see Figure 3C).

As we can see in Table 1B, a cost increase of 8% or 12% on the optimal cost can result in a solution with a good value of R-robustness for some instances: instances 2 and 4 present an excellent value of such robustness with only a cost increase of 8%, while instances 1 and 3 have a rather good one with a cost increase of 12%.

Finally, we compare the optimal R-robustness R^* to the R-robustness value of the balanced arborescence S_b obtained by solving (BRCStA), that is, we compute in S_b (see Figure 3C) the maximum number of terminals which are disconnected after the deletion of an arc incident to the root. Let R_{BR^*} be this number, shown in the last column of Table 1B. For the test instances, the values of R^* and R_{BR^*} are the same, which means that S_b is a good solution for both R-robustness and B-robustness, but we have seen before that its cost is high. Indeed, for these instances, we see that forcing a solution with $R = R^*$ to be optimally balanced increases the cost by at least 33 %. Nevertheless, there is no guarantee in the general case that the best balanced solution also has the best possible R-robustness, although the arcs incident to the root are involved in the computation of B-robustness.

When trying to obtain the minimum value for R-robustness (see (RCStA) in Figure 3B), we saw that the associated solutions have a reasonable cost, but their B-robustness is not good, since the tree remains too deep. When finding the Balanced Steiner arborescence (see (BRCStA) in Figure 3C), we observed that its value for B-robustness is optimal and its value for R-robustness is good enough, but the cost can be really high (an increase of the optimal cost to 64% on those data sets). Adding bounds on both cost and R-robustness, while minimizing B-robustness (see ($\text{BRCStA}_{\text{bounded}-R\_\text{robust}-cost}$) in Figure 3D), yields solutions which have both a reasonable cost and really good values for R-robustness and B-robustness. Hence, it seems that this approach actually yields the best compromise between the three optimization criteria (the cost and the two types of robustness considered here).

6.2 | Experimental results for (CRkACSN)

We tested the branch-and-cut framework for mixed-integer bilevel linear programs proposed in [18] to solve directly the bilevel formulation (BILEVEL-RW), but the results were rather disappointing. This is probably due to several reasons. Firstly, this method needs a linear second level, which requires the linearization of the quadratic terms $y_{ij}\lambda_{ij}$, and results in a huge formulation. Secondly, the optimal value of the continuous relaxation obtained after linearizing the quadratic terms (using the McCormick linearization) and relaxing the integrality constraints is known to be very bad (compared to the optimal value).

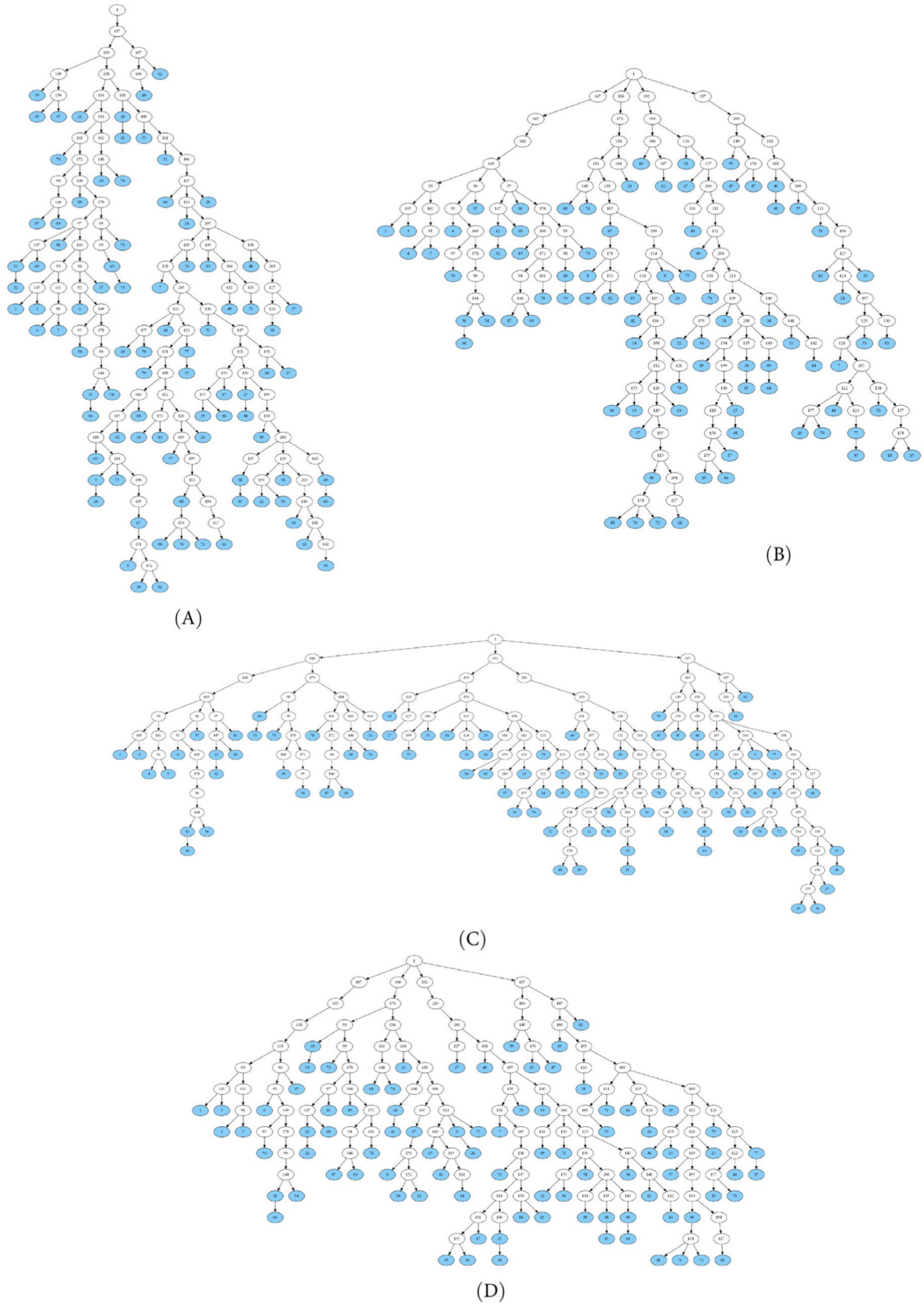

FIGURE 3 Resulting arborescences. (A) (CStA); (B) (RCStA); (C) (BRCStA); (D) ($\text{BRCStA}_{\text{bounded}-R\_\text{robust}-cost}$)

Thirdly, the follower's preprocessing has a large impact on the performance of this method, but it has probably little effect in our case. Thus, we built a specific algorithm, based on a branch-and-cut scheme as in [18].

We give the results for sparse networks corresponding to small wind farms, that is, $30 \leq |V| \leq 90$ and $|E| \approx 3|V|$. The problem is very hard and digraphs with more vertices or a higher density would be too difficult to solve. All instances have been generated in the following way: the vertices have been randomly generated in the plane, and the capacity of an arc is more likely to be high if this arc is close to the root. The arc capacities are high enough to ensure that there is at least one feasible solution to the problem with one breakdown, but low enough to keep the problem difficult to solve, the problem without capacity constraints being much easier. More precisely, the capacities are chosen randomly among four values: $\lfloor 0.8|T| \rfloor$, $\lfloor 0.6|T| \rfloor$ and, except for the arcs with endpoints at distance 1 or 2 from the root, $\lfloor 0.4|T| \rfloor$ and $\lfloor 0.2|T| \rfloor$. Furthermore, the cost of an arc depends on both its length and its capacity (and hence is not necessarily integral), and any Steiner vertex has degree at least 3 (see Section 5.1).

All experiments were performed using the same computer and software as the ones described in the previous subsection. We also used the package *JuMP*, a tool allowing mathematical modeling. For each test, the algorithm was stopped after 3000 s if it still had not terminated.

We tested the flow formulation on all the instances. In spite of the fact that the optimal values of the continuous relaxations of the flow and cutset formulations are equal (see Theorem 3.2), the flow formulation appeared to be much less efficient. It did not allow us to solve instances with more than 60 vertices for $k = 1$; moreover, the solution time increases a lot for $k = 2$ or 3. Thus, we only present the results for the bilevel and cutset formulations.

We first present the results for digraphs with nonuniform capacities and no protection allowed. In that case, the bilevel and cutset formulations are equivalent (see Theorem 3.1), and we use the formulations given in Section 3.3.1. Since no systematic theoretical comparison is possible between ($SEC-$BILEVEL) and (AUX$-$BILEVEL) for solving the separation problem (see Proposition 3.5), we tested and compared different approaches for solving the problem. The best results, presented in the tables given below, were obtained as follows:

1. CPLEX starts the resolution of (LIN-BILEVEL) with the initial subset of two Constraints (3.7) obtained by considering the cutsets formed by all the arcs incident to s or r, which provides the values of λ^1 and λ^2: we have $\lambda^1_{ri} = 1$ and $\lambda^2_{ri} = 0$ for all i adjacent to r, $\lambda^1_{js} = 0$ and $\lambda^2_{js} = 1$ for all j adjacent to s, as well as $\lambda^1_{ij} = \lambda^2_{ij} = 0$ for all $i \neq r$ and $j \neq s$.
2. For each integer solution $\hat{y}$ found by CPLEX:
   – Solve (AUX$-$BILEVEL)($\hat{y}$). If there is no feasible solution, $\hat{y}$ is a feasible solution of the complete problem (CPLEX continues with a new active node or terminates). Otherwise, we obtain new values of $\hat{\lambda}$, and then of $\tilde{y}$.
   – Solve ($SEC-$BILEVEL)($\tilde{y}$) to obtain a new point h, that is, a new constraint to add to the subset of Constraints (3.7).

Notice that the constraints are added as *lazy constraints* in the branch-and-bound of CPLEX.

Table 2 presents the results for the bilevel (or cutset) formulation for one to three arc deletions (i.e., $k = 1, 2$, or 3, and $k' = 0$). The column $|V|$ gives the number of vertices of the digraphs, and, for each value of k, the column gap_{LR} gives the gap in percentage between the optimal integer value and the optimal value of the initial continuous relaxation, taking into account the valid inequalities given in Section 5. The column gap_f gives the final gap in percentage between the best lower bound best_{LB} and the value of the best feasible integral solution found best_I: if the gap is equal to 0, then we have found an optimal solution; otherwise, we give the gap obtained after 3000 s of computation (formally, we have $\text{gap}_f = 100(\text{best}_I - \text{best}_{\text{LB}})/\text{best}_I$) . The column time gives the time in seconds needed to find an optimal solution, or 3000 if the algorithm has not terminated yet.

We consider five instances for each different value of $|V|$ with five different values of $|T|$, from $\frac{1}{10}|V|$ to $|V| - 1$ (which corresponds to a spanning network). It appeared that the results vary little depending on the number of terminals, which essentially means that, in digraphs with a root, the minimum-cost resilient capacitated network problem is as difficult to solve for

TABLE 2 Wind farm networks: results with the bilevel/cutset formulation for digraphs with nonuniform capacities, when $k' = 0$.

	$k = 1$			$k = 2$			$k = 3$		
$\|V\|$	gap_{LR}	gap_f	Time (s)	gap_{LR}	gap_f	Time (s)	gap_{LR}	gap_f	Time (s)
30	18.6	0.0	8.32	16.7	0.0	8.03	9.3	0.0	32.53
40	15.2	0.0	15.94	16.0	0.0	62.63	21.0	0.0	3.50
50	20.2	0.0	703.42	20.3	0.0	407.90	8.0	0.0	10.20
60	21.2	0.4	1038.00	19.0	2.3	1093.90	27.0 *	12.0	3000.00
70	20.8	6.4	2449.10	21.5 *	0.0	579.50	22.5 *	1.5	2097.00
80	21.0	9.0	3000.00	22.0	4.0	3000.00	—	—	—
90	19.2	5.5	3000.00	16.2	7.8	2407.50	—	—	—

spanning networks as it is for Steiner networks. That allows us to give the mean values of each set of five instances for $k = 1$. For $k = 2$ or 3, several instances have no feasible solution because having instances difficult enough to solve for $k = 1$ makes several instances infeasible for $k = 2$ or 3. So, the mean value is computed only over the instances admitting a feasible solution: generally three or more instances, but only two instances (marked with a star) for $|V| = 70$ and $k = 2$ or 3, only one (marked with a star) for $|V| = 60$, and no instance for $k = 3$ and $|V| = 80$ or 90.

We can see that we have an optimal solution for all the instances with at most 50 vertices, and that the final gap remains small enough, since it is smaller than 9% for all the instances with $k = 1$ or 2, and smaller than 12 % for $k = 3$. Moreover, the results do not seem to be very sensitive to an increase of k.

We also ran the tests without the addition of the valid inequalities proposed in Section 5. Adding these valid inequalities has a huge impact: on average, the total solution time is divided by 3.24, and the optimal value of the continuous relaxation is multiplied by 1.28.

We also applied our algorithm to denser networks from the SND Library [40]. These networks are given with demand matrices and capacities on the edges. In our problem, the demands are equal to 1 (terminal vertices) or 0 (Steiner vertices). We have adjusted the capacities to make instances difficult to solve in the case of a single arc failure, as for wind farm networks, but as a consequence there is not always a feasible solution for these instances in the case of two or three arc failures. The root and terminals have been placed randomly. The results given in Table 3 correspond to the mean value for four instances on each network, with $|T| = \lfloor 0.2|V| \rfloor, \lfloor 0.4|V| \rfloor, \lfloor 0.6|V| \rfloor$ and $|V| - 1$, except in cases marked with *, where only one instance could be processed (the other instances had no feasible solutions).

Table 4 presents the results for wind farms when the arcs have a uniform capacity. We ran the tests for the *Regular-Cutset* formulation, which corresponds to the classic cutset formulation equivalent to the bilevel one, proposed in Section 3.2.2 (CUT), and the specific *Uniform-Cutset* formulation (CUT-UNIF), which reduces the number of constraints in the case of uniform capacities, given at the end of Section 3.2.2. The tests were made on the same instances as before with a capacity equal to $\lfloor 0.4|T| \rfloor$, $\lfloor 0.6|T| \rfloor$ or $\lfloor 0.8|T| \rfloor$, since for a capacity equal to $\lfloor 0.2|T| \rfloor$ we had no feasible solutions in any of the considered instances. Again, the results are not sensitive to an increase of the number of terminals, so we give the mean values on five instances.

We do not give the final gap since each instance has been optimally solved by both formulations within the 3000 s. We neither give the results for $k = 3$, because too many instances have no feasible solution in that case. Column $|V|$ gives the size of the instances, and the next columns give the solution times in seconds for each formulation and for $k = 1$ or 2.

There are two main observations about these results. Firstly, the case with uniform capacities is much easier to solve than the nonuniform case, since we always obtain optimal solutions. Secondly, the specific Uniform-Cutset formulation is much more

TABLE 3 SNDlib networks: results with the bilevel/cutset formulation for digraphs with nonuniform capacities, when $k' = 0$.

	$k = 1$			$k = 2$			$k = 3$		
Network-$\|V\|$	**gap_{LR}**	**gap_f**	**Time (s)**	**gap_{LR}**	**gap_f**	**Time (s)**	**gap_{LR}**	**gap_f**	**Time (s)**
France-25	9.9	0	158.90	9.9 *	0	80.11	–	–	–
Germany-50	11.9	0	1180.22	–	–	–	–	–	–
Giul-39	17.4	0	639.00	14.2	0	705.14	–	–	–
Janos-us-50	7.6	0	1223.20	13.3 *	0	396.40	10.2 *	0	187.55
Janos-ca-50	6.4	0	548.94	21.5 *	0	2044.95	59.7 *	28	3000.00
Newyork-39	10.1	0	667.83	9.7	0	341.75	25.5 *	0	1406.12
Pioro-40	18.8	0	33.22	14.4	0	30.77	–	–	–
Sun-27	13.0	0	5.20	12.4 *	0	1.22	–	–	–

TABLE 4 Solution time (in s) for digraphs with uniform capacities, when $k = 1, 2$ and $k' = 0$.

	$k = 1$		$k = 2$	
$\|V\|$	**Uniform-Cutset**	**Regular-Cutset**	**Uniform-Cutset**	**Regular-Cutset**
30	0.46	0.91	0.85	3.32
40	0.67	1.36	2.04	6.10
50	1.57	3.74	3.53	22.08
60	2.83	6.37	10.81	696.66
70	2.87	7.32	7.57	57.29
80	6.96	19.81	50.38	2656.92
90	12.68	45.08	45.92	1183.60

TABLE 5 Results obtained with the cutset formulation for digraphs with nonuniform capacities, when $k = 1$ and $k' = 1, 2$, or 3.

	$k' = 1$			$k' = 2$			$k' = 3$		
$\|V\|$	gap_{LR}	gap_f	Time (s)	gap_{LR}	gap_f	Time (s)	gap_{LR}	gap_f	Time (s)
30	23.5	0.0	8.5	26.0	0.0	12	29.0	0	17.5
40	23.5	0.5	369.5	24.5	1.0	12	29.0	0	484.6
50	24.0	1.0	839.0	26.0	1.0	1072	28.0	2	1133.0
60	26.0	1.0	952.0	28.0	1.0	1207	30.0	2	1321.0
70	25.0	4.5	1564.0	27.5	5.5	1898	29.5	8	1938.0
80	24.0	8.0	2176.0	27.0	10.0	2590	29.0	13	2755.0
90	20.0	10.0	3000.0	23.0	14.0	3000	24.0	15	3000.0

efficient than the general Regular-Cutset formulation, and makes it possible to solve all the instances in less than 50 s: recall that the number of constraints of the Uniform-Cutset formulation is highly reduced, and that the number of variables remains the same. For $k = 2$, the Uniform-Cutset formulation appears to be even more efficient than the Regular-Cutset formulation: with the former one, the mean solution time has only slightly increased compared to the case $k = 1$, unlike what happens with the latter one.

Figure illustrates the cost of designing failure-resilient wind farm networks without protected arcs (i.e., when $k' = 0$). Here, we give the cost per instance, and the number of the corresponding test instance is displayed on the x-axis: for each size of instance ($|V|$), we give the cost for an increasing number of terminals from $\frac{1}{10}|V|$ to $|V| - 1$. Each time the curve falls, this indicates a change in the number of vertices (next value of $|V|$) and a decrease in the number of terminals which is reinitialized to $\frac{1}{10}|V|$. Note that we have selected a subset of instances to make the figure clearer. This figure gives the cost of an optimal solution for the cases where k is equal to 0 (no arc deleted), 1, 2, and 3. As could have been expected, for a given number of vertices, the optimal cost increases with the number of terminals. The figure also shows that designing a network resilient to even a small number of arc-failures can be costly (the cost increases greatly with the value of k).

Finally, we obtain results for the case with protected arcs for all the instances with $k = 1$. As could be expected from Theorem 4.1, the cutset formulation gave a better initial gap and better solution times than the bilevel formulation. Concerning the flow formulation, it behaves even worse than in the case without protected arcs. So, we only give results for the cutset formulation in Table 5. Notice that now, thanks to the protected arcs, we can obtain feasible solutions for some instances with three arc failures, even when $|V| = 80$ or 90. Nevertheless, the solution time being too big for most of the instances with $k \geq 2$, we only give the results for the case $k = 1$.

The addition of protected arcs deteriorates the optimal value of the continuous relaxation, but it appeared that the final gaps are very similar to the final gaps for the problem without protection. On the contrary, the solution time increases when k' becomes bigger.

7 | CONCLUSION

Our study focused on networks with a root and several terminals having a uniform demand, taking into account both capacities on the arcs and resilience to failures. We have first designed arborescences minimizing the losses if an arc failure occurs. We have shown that a restriction of the problem to the search of a spanning arborescence such that the failure of one arc disconnects at most a given number β of terminals is strongly NP-complete, which strengthens a complexity result due to Papadimitriou, who in addition considered costs on the arcs. Then, we have derived some new mathematical formulations to tackle the problem, and compared the kind of arborescences obtained (i.e., their heights, widths, etc.) according to the selected objective function.

Then, we have designed networks able to route a flow of a given value even after the deletion of any k arcs. We have given a bilevel formulation where the second level is a min-max problem, and we have reformulated the problem as a linear integer program with an exponential number of constraints. We have used a cutting plane method for the main problem, with additional valid inequalities, and we have compared different approaches to solve the separation problem: we have proved that they cannot be theoretically compared, so we have compared their practical efficiency, and selected the best one. We have also proved that the linear formulation of the bilevel program is theoretically equivalent to a cutset-based formulation, except in the case of uniform capacities where the cutset formulation is better suited. Finally, we have shown that a third formulation based on a flow model is not computationally efficient compared to the previous ones, despite the fact that they all yield the same optimal value when the integrality constraints are relaxed. Nevertheless, if we only relax the integrality constraints on the first level decision variables, keeping the integrality constraints on the flow variables, the bilevel formulation could provide a better optimal value than the three others.

In the last study, we have introduced the possibility of protecting some arcs against failures and proved that, in that case, the bilevel and cutset formulations are no longer equivalent, a specific cutset formulation being better. For the problem with nonuniform capacities and without protected arcs, we have been able to solve, in less than 1 h, instances in sparse digraphs with up to 90 vertices and three arc failures, most of them being solved optimally, and the others with an optimality gap smaller than 10%. For larger instances, a heuristic approach would be more appropriate, our exact method being able to guarantee its efficiency at least for small instances. Notice that all our models and solutions exploit the fact that the flow is routed from a given root to a set of terminals with a uniform demand: they could not be adapted to the case of multicommodity flows, since the constraint matrix of the multicommodity flow problem with more than one commodity is no longer totally unimodular.

In future works, it could also be interesting to develop a primal heuristic to solve the problem, and see if a Benders decomposition approach would be more efficient on some instances than the ones presented in this paper [11,28]. Notice, in particular, that designing a good and fast primal heuristic could also enable us to apply additional preprocessing rules on the instances to be solved, such as the domain propagation technique based on dual ascent and reduced-cost fixing proposed in [35].

FUNDING INFORMATION

This work was partially supported by PGMO, Gaspard Monge program for optimization and operations research, of Fondation Mathématique Jacques Hadamard and their Interactions with data sciences.

DATA AVAILABILITY STATEMENT

Data available on request from the authors.

ORCID

Marie-Christine Costa https://orcid.org/0000-0002-2250-0538

REFERENCES

[1] R. Ahuja, T. Magnanti, and J. Orlin, *Networks flows, theory, algorithms, applications*, Prentice Hall, Upper Saddle River, NJ, 1993.
[2] H. Aissi, C. Bazgan, and D. Vanderpooten, *Min–max and min–max regret versions of combinatorial optimization problems: a survey*, Eur. J. Oper. Res. **197-2** (2009), 427–438.
[3] M. Baïou and A. R. Mahjoub, *Steiner 2-edge connected subgraph polytopes on series-parallel graphs*, SIAM J. Discret. Math. **10** (1997), no. 3, 505–514.
[4] C. Bazgan, S. Toubaline, and D. Vanderpooten, *Efficient determination of the k most vital edges for the minimum spanning tree problem*, Comput. Oper. Res. **39-11** (2012), 2888–2898.
[5] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust optimization*, Princeton University Press, Princeton, NJ, 2009.
[6] C. Bentz, M.-C. Costa, and A. Hertz, *On the edge capacitated Steiner tree problem*, Discret. Optim. **38** (2020), 100607.
[7] D. Bertsimas and M. Sim, *The price of robustness*, Oper. Res. **52** (2004), no. 1, 35–53.
[8] D. Bertsimas, E. Nasrabadi, and S. Stiller, *Robust and adaptive network flows*, Oper. Res. **61** (2013), no. 5, 1218–1242.
[9] D. Bienstock and G. Muratore, *Strong inequalities for capacitated survivable network design problems*, Math. Program. **89-1** (2000), 127–147.
[10] M. D. Biha and A. R. Mahjoub, *Steiner k-edge connected subgraph polyhedra*, J. Comb. Optim. **4-1** (2000), 131–144.
[11] S. Bolusani, S. Coniglio, T. K. Ralphs, and S. Tahernejad, "*A unified framework for multistage discrete optimization*," *In "Bilevel optimization: Advances and next challenges"*, S. Dempe and A. Zemkoho (eds.), Springer, Cham, Switzerland, 2020.
[12] Q. Botton, B. Fortz, L. Gouveia, and M. Poss, *Benders decomposition for the hop-constrained survivable network design problem*, INFORMS J. Comput. **25-1** (2013), 13–26.
[13] C. Bousba and L. Wolsey, *Finding minimum cost directed trees with demands and capacities*, Ann. Oper. Res. **33-4** (1991), 285–303.
[14] G. Dahl and M. Stoer, *A cutting plane algorithm for multicommodity survivable network design problems*, INFORMS J. Comput. **10-1** (1998), 1–11.
[15] S. Dempe, *Foundations of bilevel programming*, Springer Science and Business Media, New York, NY, 2002.
[16] D. Z. Du, J. M. Smith, and J. H. Rubinstein, *Advances in Steiner trees*, Vol **6**, Springer Science & Business Media, New York, NY, 2013.
[17] M. Ehrgott, *Multicriteria optimization*, Lecture notes in economics and mathematical systems, Vol **491**, Springer, Berlin, Germany, 2000.
[18] M. Fischetti and D. Pisinger, *Optimizing wind farm cable routing considering power losses*, Eur. J. Oper. Res. **270-3** (2018), 917–930.
[19] M. Fischetti, I. Ljubić, M. Monaci, and M. Sinnl, *A new general-purpose algorithm for mixed-integer bilevel linear programs*, Oper. Res. **65-6** (2017), 1429–1731.
[20] M. R. Garey and D. S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness*, W. H. Freeman, New York, NY, 1979.
[21] M. Goemans and D. Bertsimas, *Survivable networks, linear programming relaxations and the parsimonious property*, Math. Program. **60-1-3** (1993), 145–166.
[22] M. Goemans and Y. S. Myung, *A catalog of Steiner tree formulations*, Networks **23-1** (1993), 19–28.
[23] M. Grotschel, C. L. Monma, and M. Stoer, *Design of survivable networks, handbooks in operations research and management*, Science **7** (1995), 617–672.
[24] A. Hertz, O. Marcotte, A. Mdimagh, M. Carreau, and F. Welt, *Optimizing the design of a wind farm collection network*, Inform. Syst. Operat. Res. **50** (2012), no. 2, 95–104.
[25] F. K. Hwang, D. S. Richards, and P. Winter, *The Steiner tree problem*, Elsevier, Amsterdam, Netherlands, 1992.
[26] H. Kerivin and A. R. Mahjoub, *Design of survivable networks: a survey*, Networks **46-1** (2005), 1–21.

[27] H. Kerivin, D. Nace, and J. Geffard, *Design of survivable networks with a single facility*, 2nd Eur. Conf. Univ. Multiserv. Netw., IEEE, 2002, pp. 208–218.
[28] T. Kleinert, M. Labbé, I. Ljubić, and M. Schmidt, A survey on mixed-integer programming techniques in bilevel optimization *EURO*, J. Comput. Optimiz. **9** (2021), 100007.
[29] T. Koch and A. Martin, *Solving Steiner tree problems in graphs to optimality*, Networks **32-3** (1998), 207–232.
[30] I. Ljubic, *Solving Steiner trees: Recent advances, challenges, and perspectives*, Networks **77-2** (2021), 177–204.
[31] L. R. Matthews, C. E. Gounaris, and I. G. Kevrekidis, *Designing networks with resiliency to edge failures using two-stage robust optimization*, Eur. J. Oper. Res. **279-3** (2019), 704–720.
[32] K. Menger, *Zur allgemeinen kurventheorie*, Fund. Math. Inst. Math. Polish Acad. Sci. **10** (1927), no. 1, 96–115.
[33] C. H. Papadimitriou, *The complexity of the capacitated tree problem*, Networks **8** (1978), no. 3, 217–230.
[34] A. C. Pillai, J. Chick, L. Johanning, M. Khorasanchi, and V. de Laleu, *Offshore wind farm electrical cable layout optimization*, Eng. Optim. **47** (2015), no. 12, 1–20.
[35] T. Polzin, *Algorithms for the Steiner problem in networks*, Ph.D. thesis, Saarland University, 2003.
[36] D. Rajan and A. Atamtürk, *A directed cycle-based column-and-cut generation method for capacitated survivable network design*, Networks **43-4** (2004), 201–211.
[37] H. D. Ratliff, G. T. Sicilia, and S. H. Lubore, *Finding the n most vital links in flow networks*, Manag. Sci. **21-5** (1975), 531–539.
[38] A. Schrijver, *Combinatorial optimization - polyhedra and efficiency*, Springer, Berlin, Germany, 2003.
[39] J. C. Smith and Y. Song, *A survey of network interdiction models and algorithms*, Eur. J. Oper. Res. **283-3** (2020), 797–811.
[40] SNDlib. *SNDlib: library of test instances for survivable fixed telecommunication network design*. http://sndlib.zib.de/
[41] M. Stoer and G. Dahl, *A polyhedral approach to multicommodity survivable network design*, Numer. Math. **68-1** (1994), 149–167.
[42] K. Wood, *Deterministic network interdiction*, Math. Comput. Model. **17-2** (1993), 1–18.

How to cite this article: C. Bentz, M. -C. Costa, P. -L. Poirion, and T. Ridremont, *Robust capacitated Steiner trees and networks with uniform demands*, Networks.. (2023), 1–29. https://doi.org/10.1002/net.22143

APPENDIX

Comparing the use of (SEC-BILEVEL) ($\hat{y}$), (SEC-BILEVEL)($\tilde{y}$) and (AUX-BILEVEL)($\hat{y}$) to solve (LIN-BILEVEL): The proof of Proposition 3.5

Proof. Let us consider an instance without breakdown (i.e., we have $k = 0$) represented by the digraph $G = (V, A, U)$ given in Figure A1, where $T = \{t_1 = e, t_2 = f, t_3 = g, t_4 = h\}$ (and hence $|T| = 4$), U is the capacity function (values in square brackets), and all the arc costs are positive. (Note that the only feasible, and hence optimal, solution here consists in selecting all the arcs.) We start the resolution of (LIN-BILEVEL) with only the constraints associated with the two cutsets incident to r and s, which impose that $\sum_{i\in\Gamma^+_G(r)} u_{ri}y_{ri} \geq |T| = 4$ and $\sum_{i\in\Gamma^-_G(s)} u_{is}y_{is} \geq |T| = 4$: this immediately yields a first integral (partial) solution $\hat{y}$ containing all the arcs incident to r or s and represented in G by the double arcs, since all the arc costs are positive (and, in fact, all these arcs are mandatory). The (only) optimal solution of $(\text{AUX} - \text{BILEVEL})(\hat{y})$ is given by the cutset $C_1 = \{(r,a),(b,d)\}$, that is, $\hat{\lambda}^{\text{aux}}_{ra} = \hat{\lambda}^{\text{aux}}_{bd} = 1$ and $\hat{\lambda}^{\text{aux}}_{ij} = 0$ for each arc $(i,j) \notin \{(r,a),(b,d)\}$. Then, $\tilde{y}$ is given by the double arcs plus the arcs in bold: $\{(i,j) \in A \text{ s.t. } \tilde{y}_{ij} = 1\} = A \setminus \{(b,d)\}$. For (SEC-BILEVEL) (with either $\tilde{y}$ or $\hat{y}$), on the contrary, there exist several optimal solutions. On the one hand, an optimal solution of $(SEC - \text{BILEVEL})(\tilde{y})$ is given, for instance, by the cutset $C_2 = \{(a,e),(a,f),(b,d)\}$, i.e., $\tilde{\lambda}^{\text{sec}}_{ae} = \tilde{\lambda}^{\text{sec}}_{af} = \tilde{\lambda}^{\text{sec}}_{bd} = 1$ and $\tilde{\lambda}^{\text{sec}}_{ij} = 0$ for each arc $(i,j) \notin \{(a,e),(a,f),(b,d)\}$. On the other hand, an optimal solution of $(SEC - \text{BILEVEL})(\hat{y})$ is given, for instance, by the cutset $C_3 = \{(a,e),(a,f),(d,g),(d,h)\}$, that is, $\hat{\lambda}^{\text{sec}}_{ae} = \hat{\lambda}^{\text{sec}}_{af} = \hat{\lambda}^{\text{sec}}_{dg} = \hat{\lambda}^{\text{sec}}_{dh} = 1$ and $\hat{\lambda}^{\text{sec}}_{ij} = 0$ for each arc $(i,j) \notin \{(a,e),(a,f),(d,g),(d,h)\}$.

Now, consider the two (partial) solutions y^1 and y^2 given in Figure A2: whatever the (positive) arc costs are, y^1 (resp. y^2) is an optimal solution obtained by solving (LIN-BILEVEL) again, after adding the one Constraint (3.7) obtained by solving $(\text{AUX} - \text{BILEVEL})(\hat{y})$ (resp. $(SEC - \text{BILEVEL})(\hat{y})$), that is, the constraint associated with the cutset C_1 (resp. with the cutset C_3).

Firstly, we have $\sum_{(i,j)\in A} u_{ij}y^1_{ij}\hat{\lambda}^{\text{sec}}_{ij} = 0 \leq \sum_{(i,j)\in A} u_{ij}y^1_{ij}\tilde{\lambda}^{\text{sec}}_{ij} = 2 \leq 3 = |T| - 1$, while obviously $\sum_{(i,j)\in A} u_{ij}y^1_{ij}\hat{\lambda}^{\text{aux}}_{ij} = 4 = |T|$. This implies that the solution y^1 would not have been feasible if, starting from $\hat{y}$, we had solved (LIN-BILEVEL) again, after adding the one Constraint (3.7) obtained by solving either $(SEC - \text{BILEVEL})(\tilde{y})$ (i.e., the constraint associated with the cutset C_2) or $(SEC - \text{BILEVEL})(\hat{y})$ (i.e., the constraint associated with the cutset C_3), instead of $(\text{AUX} - \text{BILEVEL})(\hat{y})$.

Secondly, we have $\sum_{(i,j)\in A} u_{ij}y^2_{ij}\hat{\lambda}^{\text{aux}}_{ij} = \sum_{(i,j)\in A} u_{ij}y^2_{ij}\tilde{\lambda}^{\text{sec}}_{ij} = 2 \leq 3 = |T| - 1$, while obviously $\sum_{(i,j)\in A} u_{ij}y^2_{ij}\hat{\lambda}^{\text{sec}}_{ij} = 4 = |T|$. This implies that the solution y^2 would not have been feasible if, starting from $\hat{y}$, we had solved (LIN-BILEVEL) again, after adding the one Constraint (3.7) obtained by solving either $(\text{AUX} - \text{BILEVEL})(\hat{y})$ (i.e., the constraint associated with the cutset C_1) or $(SEC - \text{BILEVEL})(\tilde{y})$ (i.e., the constraint associated with the cutset C_2), instead of $(SEC - \text{BILEVEL})(\hat{y})$.

Finally, let us consider a new instance, obtained by replacing in G the capacities of the arcs (r, b) and (r, c) by 4 and 2, respectively, and where, on the one hand, the cost of the arc (r, b) is equal to the sum of the costs of the arcs (r, a) and (r, c), and, on the other hand, these two arc costs are equal: for instance, we can choose 2 as the cost of (r, b), and 1 as the cost of both (r, a) and (r, c). An initial solution $\hat{y}$, optimal with respect to the constraints associated with the cutsets incident to r and s, can be obtained by picking the arcs (r, a) and (r, c) and the four arcs incident to s: then, $\tilde{y}$ and the three cutsets C_1, C_2 and C_3 are unchanged. An optimal solution y^* obtained by solving $(LIN - BILEVEL)$ again, after adding the one Constraint (3.7) obtained by solving $(SEC-\text{BILEVEL})(\tilde{y})$ (i.e., the constraint associated with the cutset C_2), is given in Figure A2. Notice that we have $\sum_{(i,j)\in A} u_{ij} y^*_{ij} \hat{\lambda}^{\text{aux}}_{ij} = \sum_{(i,j)\in A} u_{ij} y^*_{ij} \hat{\lambda}^{\text{sec}}_{ij} = 2 \leq 3 = |T|-1$, while obviously $\sum_{(i,j)\in A} u_{ij} y^*_{ij} \tilde{\lambda}^{\text{sec}}_{ij} = 4 = |T|$, which implies that the solution y^* would not have been feasible if, starting from $\hat{y}$, we had solved (LIN-BILEVEL) again, after adding the one Constraint (3.7) obtained by solving either $(\text{AUX} - \text{BILEVEL})(\hat{y})$ (i.e., the constraint associated with the cutset C_1) or $(SEC - \text{BILEVEL})(\hat{y})$ (i.e., the constraint associated with the cutset C_3), instead of $(SEC - \text{BILEVEL})(\tilde{y})$. ■

By modifying the costs in a suitable way, the proof of the previous proposition also yields the two following corollaries:

Corollary A.1. *The optimal value of the continuous relaxation of (LIN-BILEVEL) obtained after adding a new constraint generated by solving either* $(AUX - BILEVEL)(\hat{y})$ *or* $(SEC - BILEVEL)(\tilde{y})$ *is not always better or worse than the one obtained after adding a new constraint generated by solving* $(SEC - BILEVEL)(\hat{y})$*, even without any breakdown.*

Proof. We begin by defining the (partial) solution y^3 for the digraph $G = (V, A, U)$ given in Figure A1. This solution is obtained from the (partial) solution y^* given in Figure A2 by adding the arcs (r, a) and (r, c). It can be noticed that, whatever the (positive) arc costs are, y^3 is an optimal solution obtained by starting from the solution $\hat{y}$, defined in the proof of Proposition 3.5, and then solving (LIN-BILEVEL) once again, after adding the one Constraint (3.7) obtained by solving $(SEC - \text{BILEVEL})(\tilde{y})$ (i.e., the constraint associated with the cutset C_2, defined in the proof of Proposition 3.5).

On the one hand, the costs of the solutions y^1 (defined in the proof of Proposition 3.5) and y^3 can be made arbitrarily large by increasing the cost of the arc (b, d). Since, from the proof of Proposition 3.5, the optimal solution

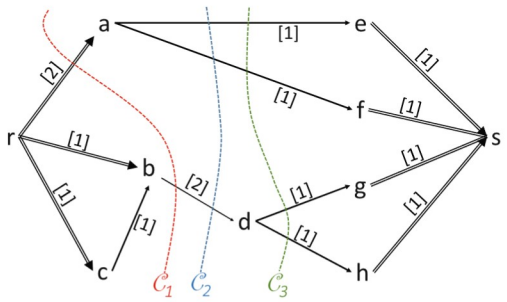

FIGURE A1 A digraph with arc capacities and three cutsets to illustrate Proposition (3.5).

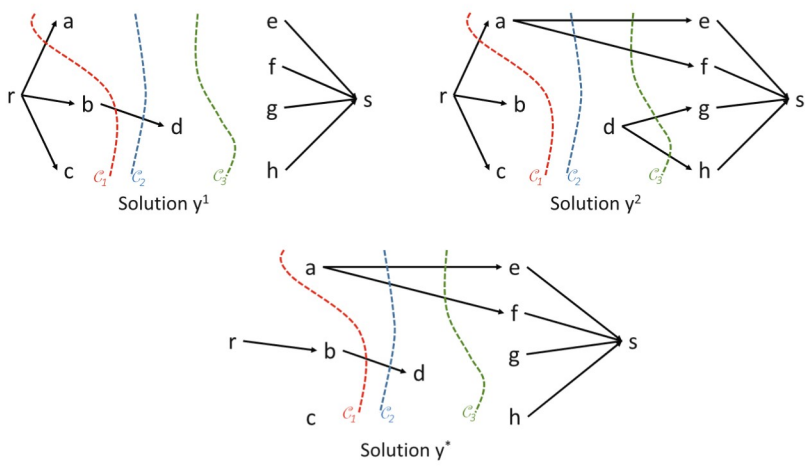

FIGURE A2 Three solutions for the instance given in Figure A1.

y^2 of the continuous relaxations of (LIN-BILEVEL) obtained after adding a new constraint generated by solving (*SEC* − BILEVEL)($\hat{y}$) does not contain this arc, we have that the optimal value of the continuous relaxation of (LIN-BILEVEL) obtained after adding a new constraint generated by solving either (AUX−BILEVEL)($\hat{y}$) or (*SEC*− BILEVEL)($\tilde{y}$) can be arbitrarily better than the optimal value of the continuous relaxation of (LIN-BILEVEL) obtained after adding a new constraint generated by solving (*SEC* − BILEVEL)($\hat{y}$).

On the other hand, from the proof of Proposition 3.5, the cost of the solution y^2 can be made arbitrarily large by increasing the cost of the arc (d, g) or (d, h), for instance. Since the optimal solution of the continuous relaxations of (LIN-BILEVEL) obtained after adding a new constraint generated by solving either (AUX − BILEVEL)($\hat{y}$) or (*SEC* − BILEVEL)($\tilde{y}$) does not contain any of these two arcs, we have that the optimal value of the continuous relaxation of (LIN-BILEVEL) obtained after adding a new constraint generated by solving (*SEC* − BILEVEL)($\hat{y}$) can be arbitrarily better than the optimal value of the continuous relaxation of (LIN-BILEVEL) obtained after adding a new constraint generated by solving either (AUX − BILEVEL)($\hat{y}$) or (*SEC* − BILEVEL)($\tilde{y}$). ■

Corollary A.2. *The optimal value of the continuous relaxations of (LIN-BILEVEL) obtained after adding a new constraint generated by solving* (*SEC* − *BILEVEL*)($\tilde{y}$) *can be arbitrarily better than the one obtained after adding a new constraint generated by solving* (*AUX* − *BILEVEL*)($\hat{y}$)*, even without any breakdown.*

Proof. From the proof of the previous corollary, the cost of the optimal solution y^3 of the continuous relaxations of (LIN-BILEVEL) obtained after adding a new constraint generated by solving (*SEC* − BILEVEL)($\tilde{y}$) can be made arbitrarily large by increasing the cost of the arc (a, e) or (a, f), for instance. Since, from the proof of Proposition 3.5, the optimal solution y^1 of the continuous relaxations of (LIN-BILEVEL) obtained after adding a new constraint generated by solving (*AUX* − *BILEVEL*)($\hat{y}$) does not contain any of these two arcs, the corollary follows. ■

Note that neither of these two corollaries implies that the optimal value of the continuous relaxations of (LIN-BILEVEL) obtained after adding a new constraint generated by solving (AUX − BILEVEL)($\hat{y}$) can be better than the one obtained after adding a new constraint generated by solving (*SEC* − BILEVEL)($\tilde{y}$). Actually, we do not know whether this is indeed the case, or whether the optimal value of the continuous relaxations of (LIN-BILEVEL) obtained after adding a new constraint generated by solving (*SEC* − BILEVEL)($\tilde{y}$) is *always* better than the one obtained after adding a new constraint generated by solving (AUX − BILEVEL)($\hat{y}$).