Exact Methods for Solving k-Delete Recoverable Robust 0-1 Problems Under Budgeted Uncertainty

Yasmine Beck, Christina Büsing, Ivana Ljubić

Abstract. We study the k-delete recoverable robust 0–1 problem in which a decision-maker solves a combinatorial optimization problem subject to objective uncertainty. The model follows a two-stage robust setup. The decision-maker first commits to an initial plan and may then revoke up to k components of this decision after the uncertainty is revealed. The underlying uncertainty is modeled using a budgeted uncertainty set so that the decision-maker only hedges against a limited number of deviations in the uncertain parameters. We present four reformulations of the k-delete recoverable robust problem, which can be tackled using (i) general-purpose mixed-integer linear programming solvers, (ii) branch-and-cut methods, or (iii) column-and-constraint generation algorithms. For each formulation, we identify suitable solution methods and prove their correctness. Overall, we present eight approaches to solve the k-delete recoverable robust problem, which we assess and compare in an extensive computational study on instances of the assignment problem and the single-source capacitated facility location problem.

1. Introduction

Decision-makers are frequently forced to commit to choices without being able to fully anticipate their consequences as real-world data often contains inherent uncertainties such as measurement errors or incomplete information. However, even small perturbations in the data can render a decision sub-optimal or infeasible for the problem at hand; see, e.g., the case study in Ben-Tal et al. (2009) for an illustrative example. It is thus essential to explicitly account for uncertainty when designing decision-support tools.

One approach to deal with uncertainties in mathematical optimization is to exploit techniques from robust optimization (Ben-Tal et al. 2009; Bertsimas et al. 2011; Soyster 1973). Classic robust models aim for solutions that remain feasible for all possible realizations of the uncertain parameters within a prescribed uncertainty set. However, because feasibility must be ensured even for extreme and often unlikely realizations of uncertainty and performance is evaluated with respect to the worst-case outcome, such models may discard decisions that would perform well in most realistic situations. As a result, classic robust solutions are often overly conservative. This drawback has motivated a growing interest in more flexible robust approaches for dealing with uncertainties, including adjustable robustness (Ben-Tal et al. 2004), light robustness (Fischetti and Monaci 2009), distributional robustness (Goh and Sim 2010; Wiesemann et al. 2014), Γ-robustness (Bertsimas and Sim 2003, 2004; Sim 2004), and recoverable robustness (Liebchen et al. 2009). The latter two are also at the core of this paper. In the Γ-robust approach, also known as the budgeted uncertainty model, the assumption that all uncertain parameters realize in a worst-case sense is relaxed. Instead, the decision-maker only hedges against

Date: May 6, 2026.

2020 *Mathematics Subject Classification.* 90C11, 90C27, 90C57, 90C70.

Key words and phrases. Combinatorial optimization, Mixed-integer programming, Robust optimization, Recoverable robustness, Budgeted uncertainty.

up to Γ deviations in the uncertain parameters that adversely affect the outcome. Recoverable robustness, in contrast, involves a two-stage robust framework. In the first stage, the decision-maker takes some decisions in a here-and-now fashion, i.e., before the uncertainty is revealed. In the second stage, a limited number of recovery actions can then be applied to adjust the first-stage decisions once the actual values of the uncertain parameters become known. These recovery actions are taken in a wait-and-see manner and may depend on the first-stage decisions.

In this paper, we introduce the k-delete recoverable robust problem in which we combine recoverable robustness with a budgeted uncertainty modeling. To this end, we focus on combinatorial optimization problems in which the objective function coefficients are subject to uncertainty. We emphasize that we do not make assumptions about the specific structure of the underlying 0–1 problem. Hence, our setting covers a broad range of well-known combinatorial optimization problems under objective uncertainty, including the assignment problem, the facility location problem, the minimum spanning tree problem, or the knapsack problem, all of which are of practical interest. In particular, our modeling framework is relevant for these problems as their underlying structure—for example, a given graph—typically remains fixed, while only some cost coefficients are affected by uncertainty. In our model, the available recovery actions consist of revoking, i.e., deleting, up to k decisions taken in the first stage. Incorporating the possibility of such recovery actions in the planning phase supports more flexible and informed decision-making. In particular, it can substantially reduce costs compared to withdrawing commitments after solving a deterministic model. To illustrate the practical relevance of this framework, let us briefly discuss two representative applications from telecommunications and preventive maintenance.

Telecommunications. Consider a telecommunication company that needs to decide on the optimal deployment of fiber-to-the-home technology for a given set of customers; see, e.g., Grötschel et al. (2014) for a primer on the optimization and integer programming aspects of optical access network planning. The company has a number of available facilities (e.g., multiplexers or switchers) from which fiber-optic cables must be provided to end-customers. In the first stage, the company needs to commit to a service provision plan by deciding on an optimal assignment between facilities and customers. In the future, the company intends to upgrade the service to a next-generation technology. The corresponding upgrade costs are determined by external suppliers of the underlying equipment and are, thus, unknown at the current stage. Once the actual upgrade prices become available, the company can decide which customers to upgrade. However, only a limited number of customers may receive the upgrade, whereas the remaining ones are left with the old technology. Business solutions in which some of the customers are left unserved or not upgraded to the newest technology are quite common in practice as fully satisfying all customer demand often exceeds the company's available budget.

Preventive Maintenance. Preventive maintenance of machinery and equipment is essential to ensure safe, reliable, and uninterrupted operations across various domains; see, e.g., the surveys in de Jonge and Scarf (2020) and Vasili et al. (2011). Maintenance schedules typically need to be created well in advance, which is why the associated costs—such as labor, spare parts, or downtime—are often uncertain at the planning stage. As these uncertainties realize over time, some maintenance tasks may become too expensive to justify or less critical than initially anticipated. For instance, it may become economically unviable to maintain a machine with a low failure-risk or a component may be in better condition than expected, e.g., because it has just been repaired or replaced. In practice, fully rescheduling all maintenance tasks may be infeasible due to operational, technical, or regulatory constraints. Nevertheless, a company may adjust the plan once the actual maintenance costs are known, but only a limited number of tasks may be postponed or canceled.

Related Literature. Since its introduction in the context of train scheduling by Liebchen et al. (2009), recoverable robustness has gained increasing attention over the last two decades. Early contributions such as Büsing (2009), Cacchiani et al. (2008), and Cicerone et al. (2009) already highlight its relevance for public transportation applications and propose first algorithmic approaches for solving recoverable robust optimization problems. In the following years, recoverable robustness has been studied for various combinatorial optimization problems and uncertainty models. In particular, column generation methods for problems with discrete scenario sets have been proposed in Bouman et al. (2011) and van den Akker et al. (2016), with applications to the size robust knapsack problem and the demand robust shortest path problem. Further applications of recoverable robustness include facility location and allocation problems (Álvarez-Miranda et al. 2015a) as well as telecommunication network design (Álvarez-Miranda et al. 2015b). Moreover, knapsack problems have emerged as one of the most frequently studied settings for recoverable robustness; see, e.g., Büsing et al. (2019, 2011a,b). In Büsing et al. (2011b), the authors consider uncertainties in both the profits and the weights and allow recovery actions that include deleting up to k items from and adding up to ℓ items to the first-stage decision. The uncertainties are modeled using a discrete scenario set. A similar setting is studied in Büsing et al. (2019, 2011a), where uncertain weights are handled using a budgeted uncertainty model and the deletion of up to k items is allowed as recovery action. A recovery structure similar to that in Büsing et al. (2011b) is also considered in Hommelsheim et al. (2023), where the authors study recoverable robust combinatorial optimization problems for problem classes satisfying the hereditary property. More recently, complexity aspects of recoverable robust problems under discrete budgeted uncertainty have been addressed in Grüne and Wulf (2024) for various underlying combinatorial optimization problems.

In this paper, we pursue similar ideas as in Büsing et al. (2019, 2011a) so that the considered recovery actions include deleting up to k components of the first-stage decision. Our contributions complement the aforementioned literature in that we do not make assumptions about the structure of the underlying combinatorial optimization problem in our recoverable robust setup. Hence, our approaches can be applied to any 0–1 problem subject to objective uncertainty. Moreover, whereas much of the existing literature focuses on structural properties and complexity results of specific problem classes, our aim is to present different algorithmic approaches to solve k-delete recoverable robust combinatorial problems to global optimality. To this end, we propose four equivalent reformulations of the problem, which can be tackled using (i) general-purpose mixed-integer linear programming solvers, (ii) branch-and-cut methods, or (iii) column-and-constraint generation algorithms. Overall, we present eight approaches to solve the k-delete recoverable robust problem and assess and compare their performance in an extensive computational study.

Outline. The remainder of this paper is organized as follows. In Section 2, we present the overall problem statement and introduce the k-delete recoverable robust problem. In Section 3, we derive four equivalent reformulations of the problem: a scenario-based, a projection-based, an extended, and a compact reformulation. For each formulation, we discuss exact solution approaches in Section 4. In Section 5, we conduct an extensive computational study on instances of the assignment problem and the single-source capacitated facility location problem to assess the performance of the proposed solution methods. Finally, we derive conclusions in Section 6.

2. Problem Statement and Connections to Other Problem Classes

We study the k-delete recoverable robust 0–1 problem under objective uncertainty. In this setting, a decision-maker determines an initial plan $x \in X \subseteq \{0,1\}^n$ before the uncertainty is revealed, where the vector x represents the first-stage decision. Throughout this paper, we assume that $X \neq \emptyset$ holds. Moreover, let $f \in \mathbb{R}^n_{\geq 0}$ denote the fixed costs associated with the first-stage decision. Then, after the first-stage decision has been made, a scenario $s \in S$ from a compact set S of possible uncertainty realizations is revealed, which determines the corresponding objective function coefficients (or scenario costs) $c^s \in \mathbb{R}^n_{\geq 0}$ for recovery actions. The decision-maker may then apply recovery actions and revoke up to $k \in \{0, \dots, n\}$ components of the first-stage decision. The aim is to select a first-stage decision that minimizes the total cost, consisting of the initial cost of the plan and the worst-case cost incurred for recovery actions. For given $x \in X$ and $s \in S$, the total cost of recovery is defined as

$$R(x,s) := \min_y \left\{ \sum_{i=1}^n c_i^s y_i \colon y \leq x,\ \sum_{i=1}^n y_i \geq \sum_{i=1}^n x_i - k,\ y \in \{0,1\}^n \right\}. \tag{1}$$

Overall, the k-delete recoverable robust problem can thus be stated as

$$\min_x \quad f^\top x + \max_{s \in S} \{R(x,s)\} \quad \text{s.t.} \quad x \in X \subseteq \{0,1\}^n. \tag{2}$$

Note that if $|S| = 1$, the k-delete recoverable robust problem (2) reduces to a deterministic two-stage problem. The recovery actions y correspond to the second-stage decisions and allow the decision-maker to adjust the initial plan by deleting at most k components of x, where the parameter k is used to control the degree of flexibility. Setting $k = 0$ yields a classic robust problem without recovery actions, whereas $k = n$ allows all first-stage decisions to be revised. In what follows, we use the abbreviation $N := \{1, \dots, n\}$ so that Problem (1) can equivalently be stated as

$$R(x,s) = \sum_{i=1}^n c_i^s x_i - \max_{\{K \subseteq N \colon |K| \leq k\}} \sum_{i \in K} c_i^s x_i$$

for given $x \in X$ and $s \in S$.

We point out that, in Problem (2), all decisions can be recovered in principle. This is done for the ease of presentation but we acknowledge that more general formulations in which different parts of the decision are treated independently in terms of uncertainties and allowed recovery actions are possible as well. Such generalizations can, e.g., be obtained by partitioning the index set N into recoverable and non-recoverable decisions. For instance, this is suitable in the context of facility location problems. An opened facility may not be closed again, i.e., no recovery action is allowed, but the connection to an assigned customer may be revoked. Furthermore, cost uncertainties for the facility locations, if present at all, can behave quite differently than uncertainties regarding the assignment costs.

We further mention that our modeling of the k-delete recoverable robust problem allows recovered solutions to violate some constraints of the original model. For instance, some customers may remain unserved in a facility location problem, which can be a common situation in practice. However, if the underlying problem satisfies the downward monotonicity (or hereditary) property, feasibility after recovery is guaranteed. Examples of such problems include the knapsack problem, the maximum clique problem, and other subset-selection problems in which any subset of a feasible solution remains feasible.

Let us now elaborate on the scenario set and the scenario costs considered in this paper. In robust optimization, commonly studied scenario sets include interval-, discrete-, ellipsoidal-, or Γ-scenario sets. In particular, the Γ-scenario approach (Bertsimas and Sim 2003, 2004; Sim 2004) is among the most prominent and frequently used methods to address uncertainties in robust optimization. Hence, and as it seems unlikely that all uncertain costs realize in a worst-case sense, we also adopt such a budgeted uncertainty modeling in this paper. To this end, we assume that the recovery costs are uncertain but known to take values in the uncertainty set

$$\mathcal{U} = \mathcal{U}_1 \times \cdots \times \mathcal{U}_n \quad \text{with} \quad \mathcal{U}_i := [\bar{c}_i, \bar{c}_i + \Delta c_i] \quad \text{for all } i \in N.$$

Here, $\bar{c}_i \geq 0$ is the nominal value of the cost of recovery action $i \in N$ and $\Delta c_i \geq 0$ is the maximum deviation from the nominal value, i.e., the maximum cost increase. Following the Γ-robust approach, the decision-maker now only hedges against a subset of at most $\Gamma \in \{0, \dots, n\}$ deviations in the cost coefficients that adversely affect the solution to the problem at hand. Here, the parameter Γ is used to control the decision-maker's level of conservatism. In particular, our modeling captures the nominal ($\Gamma = 0$) and the strictly robust formulation ($\Gamma = n$) as special cases. For a given Γ, we define

$$\mathcal{U}_\Gamma := \left\{ u \in \{0,1\}^n : \sum_{i=1}^{n} u_i \leq \Gamma \right\} = \left\{ u^1, u^2, \dots, u^M \right\}$$

so that the discrete scenario set under budgeted uncertainty can be stated as

$$S = \{ s \in \{1, 2, \dots, M\} : c_i^s := \bar{c}_i + \Delta c_i u_i^s,\ i \in N,\ u^s \in \mathcal{U}_\Gamma \}.$$

The number of scenarios $|S| = M$ corresponds to the number of binary vectors of length n for which at most Γ many entries are ones. Hence, we have $M \in O(n^\Gamma)$.

2.1. Connections to Decision-Dependent Robust and Bilevel Optimization.

In Problem (2), a decision-maker may delete up to k first-stage decisions after the uncertainty is revealed. Naturally, such deletions are only relevant for decisions that are indeed taken in the first stage and for which the cost increases. Consequently, the recovery actions are inherently dependent on the first-stage decision. This establishes a close relationship between k-delete recoverable robustness and robust optimization with decision-dependent uncertainty sets; see, e.g., Arslan and Poss (2024), Nohadani and Sharma (2018), and Poss (2013, 2014) for influential works in this area. In particular, under certain assumptions, the k-delete recoverable robust problem (2) can be explicitly formulated as a decision-dependent robust optimization problem, as we illustrate in the following.

Proposition 1. *For given $\Gamma, k \in \{0, \dots, n\}$, suppose that*

$$\max_{x \in X} \left\{ \sum_{i=1}^{n} x_i \right\} =: \bar{n} \leq \Gamma + k$$

holds. Then, the k-delete recoverable robust problem (2) can be solved as the robust combinatorial optimization problem

$$\min_{x \in X} \left\{ \sum_{i=1}^{n} f_i x_i + \max_{u \in \mathcal{U}(x)} \left\{ \sum_{i=1}^{n} (\bar{c}_i + \Delta c_i) u_i \right\} \right\} \tag{3}$$

with the discrete decision-dependent uncertainty set

$$\mathcal{U}(x) = \left\{ u \in \{0,1\}^n : x \geq u,\ \sum_{i=1}^{n} u_i \leq \sum_{i=1}^{n} x_i - k \right\}.$$

Proof. The k-delete recoverable robust problem under budgeted uncertainty can equivalently be stated as

$$\min_{x\in X}\left\{f^\top x+\max_{u\in\mathcal{U}_\Gamma}\left\{\min_{y\in\{0,1\}^n}\left\{\sum_{i=1}^n(\bar{c}_i+\Delta c_i u_i)\,y_i\colon y\le x,\ \sum_{i=1}^n y_i\ge\sum_{i=1}^n x_i-k\right\}\right\}\right\}.$$

Let x^* be an optimal first-stage decision to this problem and let y^* be its associated optimal recovery action under a worst-case uncertainty realization. Because all cost coefficients are non-negative, it is optimal to delete as many components as possible, i.e., $\sum_{i=1}^n y_i^*=\sum_{i=1}^n x_i^*-k$ holds. We now show that there exists a worst-case uncertainty realization $u^*\in\mathcal{U}_\Gamma$ with $u^*=y^*$. On the one hand, if $y_i^*=0$ holds for some $i\in N$, setting $u_i^*=1$ does not increase the objective value. Hence, there must exist a worst-case realization u^* with $u_i^*=0$ for all $i\in N$ satisfying $y_i^*=0$. On the other hand, if $y^*=1$ holds for some $i\in N$, setting $u_i^*=1$ leads to an objective value that is at least as large as in the case with $u_i^*=0$ because $\Delta c_i\ge 0$ holds. Hence, we set $u_i^*=1$ for all $i\in N$ with $y_i^*=1$. By construction, this yields $u^*\in\{0,1\}^n$ and

$$\sum_{i=1}^n u_i^*=\sum_{i=1}^n y_i^*=\sum_{i=1}^n x_i^*-k\le\bar{n}-k\le\Gamma,$$

i.e., $u^*\in\mathcal{U}_\Gamma$. In particular, each $i\in N$ is associated with either no recovery cost (if $u_i^*=0$) or the full recovery cost $\bar{c}_i+\Delta c_i$ (if $u_i^*=1$). Defining the decision-dependent uncertainty set accordingly, we conclude that (i) x^* is feasible for Problem (3), (ii) $u^*\in\mathcal{U}(x^*)$ is a worst-case uncertainty realization for x^*, and (iii) the k-delete recoverable robust problem and Problem (3) attain the same objective value.

Let now $x^*\in X$ be an optimal solution to Problem (3) and let $u^*\in\mathcal{U}(x^*)$ be its associated worst-case uncertainty realization. Because u^* is optimal for the inner maximization problem in (3), we have $\sum_{i=1}^n u_i^*=\sum_{i=1}^n x_i^*-k$ and $x^*\ge u^*$. Setting $y^*=u^*$ thus yields a feasible recovery action for the given first-stage decision x^*, implying that Problems (2) and (3) admit the same objective value. Overall, this shows that both problems are equivalent whenever $\bar{n}-k\le\Gamma$. □

The connection between Problem (2) and decision-dependent robust optimization established in Proposition 1 also indicates a close relationship between k-delete recoverable robustness and bilevel optimization, as highlighted by recent results in Goerigk et al. (2025) and Lefebvre et al. (2025).

3. Reformulations

Recoverable robust combinatorial optimization problems are notoriously hard to solve; see, e.g., Grüne and Wulf (2024) for recent complexity results. In addition, the nonlinear objective of the k-delete recoverable robust problem (2) introduces further computational challenges, motivating the need for effective solution approaches. In this section, we present four reformulations of (2)—a scenario-based, a projection-based, an extended, and a compact reformulation—for which we develop exact solution methods in Section 4.

3.1. Scenario-Based Formulation.

The first reformulation of Problem (2) is obtained by introducing binary variables $y^s\in\{0,1\}^n$ and additional constraints to model the recovery actions for each scenario $s\in S$. To hedge against the worst-case realization of the uncertainty, we use an epigraph reformulation so that Problem (2) can equivalently be re-stated as

$$\min_{x,y,\eta} \quad f^\top x + \eta \tag{4a}$$
$$\text{s.t.} \quad \eta \geq \sum_{i=1}^n c_i^s y_i^s, \quad s \in S, \tag{4b}$$
$$\sum_{i=1}^n y_i^s \geq \sum_{i=1}^n x_i - k, \quad s \in S, \tag{4c}$$
$$x \geq y^s, \quad s \in S, \tag{4d}$$
$$x \in X,\ \eta \in \mathbb{R}_{\geq 0},\ y^s \in \{0,1\}^n, \quad s \in S. \tag{4e}$$

Here and in what follows, we abbreviate $v = (v^a)_{a \in A}$ with A being a discrete finite set and v^a, $a \in A$, being a vector of appropriate dimension. The constraints in (4b) capture that the decision-maker hedges against the worst-possible realization of the uncertainty, whereas (4c) guarantees that at most k components of the first-stage decision $x \in X$ are deleted in each scenario $s \in S$. The constraints in (4d) model that only those elements contained in a fist-stage decision can be deleted.

A feature of the scenario-based formulation (4) is that a solution also specifies the recovery action to implement for each scenario. However, this leads to a considerably large model in general, as the number of additional variables and constraints grows as $O(n^{\Gamma+1})$. In general, it is thus not possible to apply general-purpose MILP solvers directly to solve Problem (4).

3.2. **Projection-Based Formulation.** The second reformulation of Problem (2) is obtained by projecting the recovery actions onto the first-stage decisions and considering a new scenario set. For this purpose, we build on the fact that Problem (2) is equivalent to

$$\min_x \quad f^\top x + \beta(x) \quad \text{s.t.} \quad x \in X$$

with

$$\beta(x) = \max_u \left\{ \sum_{i=1}^n \bar{c}_i x_i + \sum_{i=1}^n \Delta c_i x_i u_i - \gamma(x,u) \colon u \in \mathcal{U}_\Gamma \right\}$$

and

$$\gamma(x,u) = \max_y \left\{ \sum_{i=1}^n \left(\bar{c}_i x_i + \Delta c_i x_i u_i\right) y_i \colon \sum_{i=1}^n y_i \leq k,\ y \in \{0,1\}^n \right\}.$$

In addition, we introduce the following notation, which we use throughout the remainder of this paper. We define

$$\mathcal{V} := \{0\} \cup \{\bar{c}_i \colon i \in N\} \cup \{\bar{c}_i + \Delta c_i \colon i \in N\}.$$

Moreover, for each $\nu \in \mathcal{V}$, we define the cost functions

$$c_i(\nu) := \min\left\{\Delta c_i, \nu - \bar{c}_i\right\}, \quad i \in N,$$

and set

$$\bar{c}_i(\nu) := \begin{cases} \nu, & \text{if } \nu < \bar{c}_i, \\ \bar{c}_i, & \text{if } \nu \geq \bar{c}_i, \end{cases} \qquad \text{and} \qquad \Delta c_i(\nu) := \begin{cases} 0, & \text{if } \nu < \bar{c}_i, \\ c_i(\nu), & \text{if } \nu \geq \bar{c}_i. \end{cases}$$

For notational convenience, we further define

$$N(\nu) := \{i \in N \colon c_i(\nu) < 0\} \quad \text{and} \quad P(\nu) := \{i \in N \colon c_i(\nu) \geq 0\},$$

as well as

$$L(\nu) := \{i \in N \colon c_i(\nu) < \Delta c_i\} \quad \text{and} \quad G(\nu) := \{i \in N \colon c_i(\nu) \geq \Delta c_i\}$$

for all $\nu \in \mathcal{V}$. By definition, $N(\nu) \subseteq L(\nu)$, $G(\nu) \subseteq P(\nu)$, and $N = N(\nu) \cup P(\nu)$ holds for any $\nu \geq 0$.

Before we can formally state the projection-based reformulation of Problem (2), we need the following intermediate results. We start with two technical lemmas, whose proofs are deferred to Appendix A to keep this section self-contained.

Lemma 1. *For arbitrarily given $x \in X$, it holds*

$$\beta(x) = \max_{u \in \mathcal{U}_\Gamma, \nu \geq 0} \left\{ \sum_{i=1}^{n} \bar{c}_i x_i - k\nu + \sum_{i \in N(\nu)} c_i(\nu) x_i + \sum_{i \in P(\nu)} c_i(\nu) x_i u_i \right\}. \tag{5}$$

Lemma 2. *Let $\nu^* \geq 0$ be given arbitrarily and let*

$$\underline{\nu} := \max\{\nu \colon \nu \in \mathcal{V},\, \nu \leq \nu^*\} \quad \text{and} \quad \bar{\nu} := \min\{\nu \colon \nu \in \mathcal{V},\, \nu \geq \nu^*\}.$$

Then, the following statements hold:

(i) $N(\bar{\nu}) \subseteq N(\nu^*) \subseteq N(\underline{\nu})$ *and* $P(\underline{\nu}) \subseteq P(\nu^*) \subseteq P(\bar{\nu})$,

(ii) $c_i(\underline{\nu}) + (\nu^* - \underline{\nu}) = c_i(\nu^*) = c_i(\bar{\nu}) - (\bar{\nu} - \nu^*)$ *for all* $i \in L(\nu^*)$,

(iii) $c_i(\underline{\nu}) = c_i(\nu^*) = c_i(\bar{\nu})$ *for all* $i \in G(\nu^*)$,

(iv) $i \in N(\nu^*) \cap P(\bar{\nu}) \implies c_i(\bar{\nu}) = 0$,

(v) $\nu^* \notin \mathcal{V} \implies P(\nu^*) \cap N(\underline{\nu}) = \emptyset$.

We use Lemmas 1 and 2 to establish the following intermediate result.

Proposition 2. *Problem (2) can be solved as*

$$\min_{x \in X} f^\top x + \max_{\nu \in \mathcal{V}} \{\alpha(x, \nu) - k\nu\},$$

where, for given $x \in X$ and $\nu \in \mathcal{V}$, we have

$$\alpha(x, \nu) = \max_{u} \left\{ \sum_{i=1}^{n} \bar{c}_i(\nu) x_i + \sum_{i=1}^{n} \Delta c_i(\nu) x_i u_i \colon u \in \mathcal{U}_\Gamma \right\}.$$

Proof. Let $x \in X$ be given arbitrarily. Further, let (u^*, ν^*) be an optimal solution to Problem (5). We prove the claim by contradiction. To this end, suppose that $\nu^* \notin \mathcal{V}$ holds and let $\underline{\nu}$ and $\bar{\nu}$ be defined as in Lemma 2. We distinguish two cases. First, suppose that

$$\sum_{i \in N(\nu^*)} x_i + \sum_{i \in P(\nu^*) \cap L(\nu^*)} x_i u_i^* \leq k$$

holds. Then, by Lemmas 1 and 2, we obtain

$$\begin{aligned}
\beta(x) &\overset{\text{Lemma 1}}{=} \sum_{i=1}^{n} \bar{c}_i x_i - k\nu^* + \sum_{i \in N(\nu^*)} c_i(\nu^*) x_i + \sum_{i \in P(\nu^*)} c_i(\nu^*) x_i u_i^* \\
&= \sum_{i=1}^{n} \bar{c}_i x_i - k\nu^* + \sum_{i \in N(\nu^*)} c_i(\nu^*) x_i \\
&\quad + \sum_{i \in P(\nu^*) \cap L(\nu^*)} c_i(\nu^*) x_i u_i^* + \sum_{i \in P(\nu^*) \cap G(\nu^*)} c_i(\nu^*) x_i u_i^* \\
&\overset{\substack{\text{Lemma 2} \\ \text{(ii),(iii)}}}{=} \sum_{i=1}^{n} \bar{c}_i x_i - k\underline{\nu} - k(\nu^* - \underline{\nu}) + \sum_{i \in N(\nu^*)} c_i(\underline{\nu}) x_i + \sum_{i \in N(\nu^*)} (\nu^* - \underline{\nu}) x_i \\
&\quad + \sum_{i \in P(\nu^*) \cap L(\nu^*)} c_i(\underline{\nu}) x_i u_i^* + \sum_{i \in P(\nu^*) \cap L(\nu^*)} (\nu^* - \underline{\nu}) x_i u_i^* \\
&\quad + \sum_{i \in P(\nu^*) \cap G(\nu^*)} c_i(\underline{\nu}) x_i u_i^*
\end{aligned}$$

$$\begin{aligned}
&= \sum_{i=1}^{n} \bar{c}_i x_i - k\underline{\nu} + \sum_{i \in N(\nu^*)} c_i(\underline{\nu}) x_i + \sum_{i \in P(\nu^*)} c_i(\underline{\nu}) x_i u_i^* \\
&\quad + (\nu^* - \underline{\nu}) \left(\sum_{i \in N(\nu^*)} x_i + \sum_{i \in P(\nu^*) \cap L(\nu^*)} x_i u_i^* - k \right) \\
&\leq \sum_{i=1}^{n} \bar{c}_i x_i - k\underline{\nu} + \sum_{i \in N(\nu^*)} c_i(\underline{\nu}) x_i + \sum_{i \in P(\nu^*)} c_i(\underline{\nu}) x_i u_i^* \\
&\overset{\text{Lemma 2 (i)}}{=} \sum_{i=1}^{n} \bar{c}_i x_i - k\underline{\nu} + \sum_{i \in N(\underline{\nu})} c_i(\underline{\nu}) x_i + \sum_{i \in P(\underline{\nu})} c_i(\underline{\nu}) x_i u_i^* \\
&\quad + \sum_{i \in P(\nu^*) \cap N(\underline{\nu})} c_i(\underline{\nu}) x_i \left(u_i^* - 1\right) \\
&\overset{\text{Lemma 2 (v)}}{=} \sum_{i=1}^{n} \bar{c}_i x_i - k\underline{\nu} + \sum_{i \in N(\underline{\nu})} c_i(\underline{\nu}) x_i + \sum_{i \in P(\underline{\nu})} c_i(\underline{\nu}) x_i u_i^*,
\end{aligned}$$

i.e., $(u^*, \underline{\nu})$ is an optimal solution to Problem (5) as well.

Second, suppose that

$$\sum_{i \in N(\nu^*)} x_i + \sum_{i \in P(\nu^*) \cap L(\nu^*)} x_i u_i^* > k$$

holds. Then, again by Lemmas 1 and 2, we obtain

$$\begin{aligned}
\beta(x) &\overset{\text{Lemma 1}}{=} \sum_{i=1}^{n} \bar{c}_i x_i - k\nu^* + \sum_{i \in N(\nu^*)} c_i(\nu^*) x_i + \sum_{i \in P(\nu^*)} c_i(\nu^*) x_i u_i^* \\
&= \sum_{i=1}^{n} \bar{c}_i x_i - k\nu^* + \sum_{i \in N(\nu^*)} c_i(\nu^*) x_i \\
&\quad + \sum_{i \in P(\nu^*) \cap L(v^*)} c_i(\nu^*) x_i u_i^* + \sum_{i \in P(\nu^*) \cap G(v^*)} c_i(\nu^*) x_i u_i^* \\
&\overset{\text{Lemma 2 (ii),(iii)}}{=} \sum_{i=1}^{n} \bar{c}_i x_i - k\bar{\nu} - k(\nu^* - \bar{\nu}) + \sum_{i \in N(\nu^*)} c_i(\bar{\nu}) x_i - \sum_{i \in N(\nu^*)} (\bar{\nu} - \nu^*) x_i \\
&\quad + \sum_{i \in P(\nu^*) \cap L(v^*)} c_i(\bar{\nu}) x_i u_i^* - \sum_{i \in P(\nu^*) \cap L(v^*)} (\bar{\nu} - \nu^*) x_i u_i^* \\
&\quad + \sum_{i \in P(\nu^*) \cap G(v^*)} c_i(\bar{\nu}) x_i u_i^* \\
&= \sum_{i=1}^{n} \bar{c}_i x_i - k\bar{\nu} + \sum_{i \in N(\nu^*)} c_i(\bar{\nu}) x_i + \sum_{i \in P(\nu^*)} c_i(\bar{\nu}) x_i u_i^* \\
&\quad + (\bar{\nu} - \nu^*) \left(k - \sum_{i \in N(\nu^*)} x_i - \sum_{i \in P(\nu^*) \cap L(v^*)} x_i u_i^* \right) \\
&< \sum_{i=1}^{n} \bar{c}_i x_i - k\bar{\nu} + \sum_{i \in N(\nu^*)} c_i(\bar{\nu}) x_i + \sum_{i \in P(\nu^*)} c_i(\bar{\nu}) x_i u_i^* \\
&\overset{\text{Lemma 2 (i)}}{=} \sum_{i=1}^{n} \bar{c}_i x_i - k\bar{\nu} + \sum_{i \in N(\bar{\nu})} c_i(\bar{\nu}) x_i + \sum_{i \in P(\bar{\nu})} c_i(\bar{\nu}) x_i u_i^*
\end{aligned}$$

$$+ \sum_{i \in N(\nu^*) \cap P(\bar{\nu})} c_i(\bar{\nu}) x_i \left(1 - u_i^*\right)$$

$$\overset{\text{Lemma 2 (iv)}}{=} \sum_{i=1}^{n} \bar{c}_i x_i - k\bar{\nu} + \sum_{i \in N(\bar{\nu})} c_i(\bar{\nu}) x_i + \sum_{i \in P(\bar{\nu})} c_i(\bar{\nu}) x_i u_i^*.$$

The latter contradicts the optimality of (u^*, ν^*) for Problem (5). To sum up, we can thus assume, w.l.o.g., that $\nu^* \in \mathcal{V}$ holds. By Lemma 1, we thus obtain

$$\begin{aligned}
\beta(x) &= \max_{\nu \in \mathcal{V}} \left\{ \max_{u \in \mathcal{U}_\Gamma} \left\{ \sum_{i=1}^{n} \bar{c}_i x_i + \sum_{i \in N(\nu)} c_i(\nu) x_i + \sum_{i \in P(\nu)} c_i(\nu) x_i u_i \right\} - k\nu \right\} \\
&= \max_{\nu \in \mathcal{V}} \left\{ \max_{u \in \mathcal{U}_\Gamma} \left\{ \sum_{i=1}^{n} \bar{c}_i(\nu) x_i + \sum_{i=1}^{n} \Delta c_i(\nu) x_i u_i \right\} - k\nu \right\},
\end{aligned}$$

where the last equality is due to the definition of $\bar{c}_i(\nu)$ and $\Delta c_i(\nu)$, $i \in N$, $\nu \in \mathcal{V}$. □

Because of $|\mathcal{V}| \in O(n)$, Proposition 2 implies that Problem (1) can be solved in polynomial time if $\bar{c}, \Delta c \in \mathbb{Q}^n$. Next, we state the projection-based formulation of Problem (2).

Theorem 1. *Problem (2) can be solved as the mixed-integer linear problem*

$$\begin{aligned}
\min_{x,\eta} \quad & f^\top x + \eta \\
\text{s.t.} \quad & \eta \geq \sum_{i=1}^{n} c_i^s(\nu) x_i - k\nu, \quad \nu \in \mathcal{V},\ s \in S(\nu), \\
& x \in X,\ \eta \in \mathbb{R}_{\geq 0},
\end{aligned} \tag{6}$$

where, for all $\nu \in \mathcal{V}$, we have

$$S(\nu) := \left\{ s \in \{1, 2, \ldots, M\} : c_i^s(\nu) := \bar{c}_i(\nu) + \Delta c_i(\nu) u_i^s,\ i \in N,\ u^s \in \mathcal{U}_\Gamma \right\}.$$

Proof. By Proposition 2, Problem (2) can be solved as

$$\begin{aligned}
\min_{x,\eta} \quad & f^\top x + \eta \\
\text{s.t.} \quad & \eta \geq \alpha(x, \nu) - k\nu, \quad \nu \in \mathcal{V}, \\
& x \in X,\ \eta \in \mathbb{R}_{\geq 0},
\end{aligned}$$

where, for given $x \in X$ and $\nu \in \mathcal{V}$, we have

$$\alpha(x, \nu) = \max_{u} \left\{ \sum_{i=1}^{n} \bar{c}_i(\nu) x_i + \sum_{i=1}^{n} \Delta c_i(\nu) x_i u_i : u \in \mathcal{U}_\Gamma = \left\{ u^1, u^2, \ldots, u^M \right\} \right\}.$$

The claim then follows from the definition of the projected scenario set $S(\nu)$. □

Note that, for given $\nu \in \mathcal{V}$, the set $S(\nu)$ is again a budgeted uncertainty set. Compared to the set S, the only difference lies in the definition of the nominal values and deviations, which are adjusted according to ν. We further note that no additional variables are introduced in the projection-based formulation (6) to model uncertainties or recovery actions. However, we still need to add a considerable number of constraints to capture these aspects, which grows as $O(n^{\Gamma+1})$.

3.3. **Extended Formulation.** Our third reformulation of Problem (2) builds on the projection-based formulation in Section 3.2. In contrast to the previous formulation, however, we now allow for an extended variable space that includes additional continuous variables for modeling recovery actions.

Theorem 2. *Problem (2) can be solved as the mixed-integer linear problem*

$$\begin{aligned}\min_{x,\eta,w,z} \quad & f^\top x + \eta \\ \text{s.t.} \quad & \eta \geq \sum_{i=1}^{n} \bar{c}_i(\nu) x_i + \Gamma w^\nu + \sum_{i=1}^{n} z_i^\nu - k\nu, \quad \nu \in \mathcal{V}, \\ & w^\nu + z_i^\nu \geq \Delta c_i(\nu) x_i, \quad i \in N,\, \nu \in \mathcal{V}, \\ & x \in X,\, \eta \in \mathbb{R}_{\geq 0},\, w^\nu \in \mathbb{R}_{\geq 0},\, z^\nu \in \mathbb{R}^n_{\geq 0}, \quad \nu \in \mathcal{V}.\end{aligned} \tag{7}$$

The proof of Theorem 2 can be found in Appendix B. In Problem (7), we consider an extended variable space that includes additional continuous variables to model the uncertainties and recovery actions, whose number is in $O(n^2)$. Overall, this results in a considerably smaller model compared to the scenario- and projection-based formulations presented earlier.

3.4. **Compact Formulation.** We now state our fourth and last reformulation of Problem (2), which is our most compact one. For the derivation, we build on the extended formulation (7) and eliminate the continuous variables w^ν and z^ν, $\nu \in \mathcal{V}$. For notational convenience, let us set

$$\Delta c_{n+1} := 0 \quad \text{and} \quad \bar{c}_{n+1} := 0$$

so that we obtain

$$c_{n+1}(\nu) = \min \{\Delta c_{n+1},\, \nu - \bar{c}_{n+1}\} = 0, \quad \nu \in \mathcal{V}.$$

For all $\nu \in \mathcal{V}$, let now $\sigma_\nu : N \to N$ be a perturbation function that orders the components of $\Delta c(\nu)$ in non-increasing order, i.e.,

$$\Delta c_{\sigma_\nu(1)}(\nu) \geq \Delta c_{\sigma_\nu(2)}(\nu) \geq \cdots \geq \Delta c_{\sigma_\nu(n)}(\nu).$$

Moreover, let γ be the largest odd integer such that $\Gamma + \gamma < n + 1$ holds and

$$\mathcal{L} := \{\Gamma + 1, \Gamma + 3, \Gamma + 5, \ldots, \Gamma + \gamma, n + 1\}.$$

For all $\nu \in \mathcal{V}$ and $\ell \in \mathcal{L}$, we further define the modified cost functions

$$\tilde{c}^{\ell}_{\sigma_\nu(i)}(\nu) := \begin{cases} \bar{c}_{\sigma_\nu(i)}(\nu) + \Delta c_{\sigma_\nu(i)}(\nu) - \Delta c_{\sigma_\nu(\ell)}(\nu), & 1 \leq i \leq \ell, \\ \bar{c}_{\sigma_\nu(i)}(\nu), & \ell + 1 \leq i \leq n, \end{cases}$$

as well as protection terms

$$\kappa_\ell(\nu) := \Gamma \Delta c_{\sigma_\nu(\ell)}(\nu) - k\nu.$$

Using the above notation, we obtain the following intermediate result.

Proposition 3. *Problem (2) is equivalent to*

$$\min_{x,\eta} \quad f^\top x + \eta \tag{8a}$$

$$\text{s.t.} \quad \eta \geq \min_{\ell \in \mathcal{L}} \left\{ \kappa_\ell(\nu) + \sum_{i=1}^{n} \tilde{c}^{\ell}_{\sigma_\nu(i)}(\nu) x_{\sigma_\nu(i)} \right\}, \quad \nu \in \mathcal{V}, \tag{8b}$$

$$x \in X,\, \eta \in \mathbb{R}_{\geq 0}. \tag{8c}$$

The proof of Proposition 3 can be found in Appendix C. Because of (8b), Problem (8) cannot be tackled by a general-purpose solver directly. However, an equivalent MILP reformulation of Problem (8) can be obtained by modeling the selection of $\ell \in \mathcal{L}$ using auxiliary binary variables and SOS1 constraints. To this end, we further need to introduce the big-M constants

$$M(\nu) := \max_{\ell \in \mathcal{L}} \left\{ \Gamma \Delta c_{\sigma_\nu(k)}(\nu) + \sum_{i=1}^{n} \max \left\{ 0,\, \tilde{c}^{\ell}_{\sigma_\nu(i)}(\nu) \right\} \right\}$$

for all $\nu \in \mathcal{V}$. Next, we state the compact MILP reformulation of Problem (2).

Theorem 3. *Problem (2) can be solved as the mixed-integer linear problem*

$$\min_{x,\eta,y} \quad f^\top x + \eta \tag{9a}$$

$$\text{s.t.} \quad \eta \geq \kappa_\ell(\nu) + \sum_{i=1}^{n} \tilde{c}^{\ell}_{\sigma_\nu(i)}(\nu) x_{\sigma_\nu(i)} - M(\nu)(1 - y^\nu_\ell), \quad \ell \in \mathcal{L},\, \nu \in \mathcal{V}, \tag{9b}$$

$$\sum_{\ell \in \mathcal{L}} y^\nu_\ell = 1, \quad \nu \in \mathcal{V}, \tag{9c}$$

$$x \in X,\, \eta \in \mathbb{R}_{\geq 0},\, y^\nu \in \{0,1\}^{|\mathcal{L}|}, \quad \nu \in \mathcal{V}. \tag{9d}$$

Proof. We show that, for an optimal solution (x^*, η^*, y^*) to Problem (9), the pair (x^*, η^*) is an optimal solution to Problem (8). We prove this by contradiction. To this end, let (x^*, η^*, y^*) be an optimal solution to Problem (9) and suppose that (x^*, η^*) does not solve Problem (8), i.e., there exists $(\hat{x}, \hat{\eta})$ feasible for Problem (8) with $f^\top \hat{x} + \hat{\eta} < f^\top x^* + \eta^*$. For all $\nu \in \mathcal{V}$, we define

$$\ell^*_\nu := \operatorname*{arg\,min}_{\ell \in \mathcal{L}} \left\{ \kappa_\ell(\nu) + \sum_{i=1}^{n} \tilde{c}^{\ell}_{\sigma_\nu(i)}(\nu) \hat{x}_{\sigma_\nu(i)} \right\} \quad \text{and} \quad \hat{y}^\nu_\ell := \begin{cases} 1, & \ell = \ell^*_\nu, \\ 0, & \ell \in \mathcal{L} \setminus \{\ell^*_\nu\}. \end{cases}$$

By construction, $(\hat{x}, \hat{\eta}, \hat{y})$ with $\hat{y} = (\hat{y}^\nu)_{\nu \in \mathcal{V}}$ satisfies (9c) and (9d). For all $\nu \in \mathcal{V}$, we further have

$$\begin{aligned}
\hat{\eta} &\geq \min_{\ell \in \mathcal{L}} \left\{ \kappa_\ell(\nu) + \sum_{i=1}^{n} \tilde{c}^{\ell}_{\sigma_\nu(i)}(\nu) \hat{x}_{\sigma_\nu(i)} \right\} = \kappa_{\ell^*_\nu}(\nu) + \sum_{i=1}^{n} \tilde{c}^{\ell^*_\nu}_{\sigma_\nu(i)}(\nu) \hat{x}_{\sigma_\nu(i)} \\
&= \kappa_{\ell^*_\nu}(\nu) + \sum_{i=1}^{n} \tilde{c}^{\ell^*_\nu}_{\sigma_\nu(i)}(\nu) \hat{x}_{\sigma_\nu(i)} - M(\nu)(1 - y^\nu_{\ell^*_\nu})
\end{aligned}$$

and

$$\begin{aligned}
\hat{\eta} \geq\ & -k\nu + \Gamma \Delta c_{\sigma_\nu(\ell)}(\nu) + \sum_{i=1}^{n} \max \left\{ 0,\, \tilde{c}^{\ell}_{\sigma_\nu(i)}(\nu) \right\} \\
& - \max_{\ell' \in \mathcal{L}} \left\{ \Gamma \Delta c_{\sigma_\nu(k)}(\nu) + \sum_{i=1}^{n} \max \left\{ 0,\, \tilde{c}^{\ell'}_{\sigma_\nu(i)}(\nu) \right\} \right\} \\
=\ & \kappa_\ell(\nu) + \sum_{i=1}^{n} \max \left\{ 0,\, \tilde{c}^{\ell}_{\sigma_\nu(i)}(\nu) \right\} - M(\nu) \\
\geq\ & \kappa_\ell(\nu) + \sum_{i=1}^{n} \max \left\{ 0,\, \tilde{c}^{\ell}_{\sigma_\nu(i)}(\nu) \right\} \hat{x}_{\sigma_\nu(i)} - M(\nu) \\
\geq\ & \kappa_\ell(\nu) + \sum_{i=1}^{n} \max \left\{ 0,\, \tilde{c}^{\ell}_{\sigma_\nu(i)}(\nu) \right\} \hat{x}_{\sigma_\nu(i)} + \sum_{i=1}^{n} \min \left\{ 0,\, \tilde{c}^{\ell}_{\sigma_\nu(i)}(\nu) \right\} \hat{x}_{\sigma_\nu(i)} - M(\nu) \\
=\ & \kappa_\ell(\nu) + \sum_{i=1}^{n} \tilde{c}^{\ell}_{\sigma_\nu(i)}(\nu) \hat{x}_{\sigma_\nu(i)} - M(\nu)(1 - \hat{y}^\nu_\ell)
\end{aligned}$$

for all $\ell \in \mathcal{L} \setminus \{\ell_\nu^*\}$. Hence, the point $(\hat{x}, \hat{\eta}, \hat{y})$ is feasible for Problem (9) and has a better objective function value than (x^*, η^*, y^*). This is a contradiction to the optimality of (x^*, η^*, y^*). By Proposition 3, this concludes the proof. □

The number of additional variables and constraints introduced for modeling the uncertainties and recovery actions in Problem (9) also grows as $O(n^2)$. Compared to the extended formulation, however, the compact formulation (9) does not make use of additional continuous variables so that the combinatorial structure of the original problem is preserved.

To conclude this section, we summarize the number of additional variables and constraints introduced for modeling uncertainties and recovery actions in the four presented reformulations of the k-delete recoverable robust problem (2) in Table 1.

4. Exact Solution Approaches

We now present exact solution approaches for the k-delete recoverable robust problem (2), which are based on the four reformulations derived in Section 3. We consider three approaches:

(i) solving the full model using a general-purpose MILP solver (MILP approach),
(ii) column-and-constraint generation (CCG), and
(iii) branch-and-cut (BnC).

Table 2 provides an overview of which methods are applicable for each formulation. Overall, we present eight approaches to solve the k-delete recoverable robust problem (2). In Section 4.1, we elaborate on the MILP approach. Afterward, in Section 4.2, we present the column-and-constraint generation algorithms. Finally, we discuss the branch-and-cut frameworks in Section 4.3.

Table 1. The number of additional variables ("# variables") and constraints ("# constraints") introduced for modeling uncertainties and recovery actions in the scenario-based, projection-based, extended, and compact reformulations of Problem (2). In particular, the number of additional continuous ("cont.") and binary variables ("bin.") is shown. Note that $M \in O(n^\Gamma)$ and $|\mathcal{V}|, |\mathcal{L}| \in O(n)$.

formulation	# variables		# constraints
	cont.	bin.	
scenario-based; see (4)	1	Mn	$M(2+n)+1$
projection-based; see (6)	1	0	$M\|\mathcal{V}\|+1$
extended; see (7)	$\|\mathcal{V}\|(n+1)+1$	0	$2\|\mathcal{V}\|(n+1)+1$
compact; see (9)	1	$\|\mathcal{V}\|\|\mathcal{L}\|$	$\|\mathcal{V}\|(\|\mathcal{L}\|+1)+1$

Table 2. Solution methods applicable to each formulation: solving the full model using a general-purpose MILP solver (MILP), column-and-constraint generation (CCG), and branch-and-cut (BnC).

formulation	MILP	CCG	BnC
scenario-based; see (4)	✗	✓	✗
projection-based; see (6)	✗	✗	✓
extended; see (7)	✓	✓	✓
compact; see (9)	✓	✓	✓

4.1. **MILP Approach.** For the extended formulation (7) and the compact formulation (9), the number of additional variables and constraints introduced for modeling uncertainties and recovery actions is in $O(n^2)$; cf. Table 1. Hence, these two reformulations can, in principle, be solved directly using a general-purpose MILP solver. In practice, however, the size of the resulting MILP can still become restrictive for larger instances and the LP bound may be rather weak. We therefore also present CCG and BnC frameworks for these formulations. In contrast, general-purpose MILP solvers cannot tackle the scenario-based formulation (4) and the projection-based formulation (6) directly because of their significant number of additional variables and constraints.

4.2. **Column-and-Constraint Generation.** Column-and-constraint generation (Zeng and Zhao 2013) is an iterative framework that can be used to solve robust optimization problems with a large number of scenarios. The idea is to start from a relaxation of the problem in which only a subset of the variables and constraints associated with specific scenarios is included. New scenarios, along with their corresponding variables and constraints, are then generated and added iteratively. To achieve this, the method alternates between solving a *(relaxed) master problem* and an *adversarial problem*, where the latter is used to identify critical scenarios that are missing from the current master problem. We now discuss how CCG is applied to the scenario-based, compact, and extended reformulations of Problem (2).

4.2.1. *Scenario-Based Formulation.* Problem (4) involves $O(n^\Gamma)$ scenarios, which may render enumerating all of them impractical. Applying CCG, we thus start from a relaxation of the problem in which we only consider a subset of scenarios. In iteration j of the method, we consider the (relaxed) master problem

$$\begin{aligned}
\min_{x,y,\eta} \quad & f^\top x + \eta \\
\text{s.t.} \quad & \eta \geq \sum_{i=1}^{n} c_i^s y_i^s, \quad s \in S^j, \\
& \sum_{i=1}^{n} y_i^s \geq \sum_{i=1}^{n} x_i - k, \quad s \in S^j, \\
& x \geq y^s, \quad s \in S^j, \\
& x \in X,\, \eta \in \mathbb{R}_{\geq 0},\, y^s \in \{0,1\}^n, \quad s \in S^j,
\end{aligned} \tag{10}$$

with $S^j \subseteq S$. To identify critical scenarios that are missing from this problem, we then determine the worst-case recovery costs for a given $x \in X$ by solving the adversarial problem

$$\max_{s\in S} \{R(x,s)\} = \max_{s\in S} \left\{ \min_{y\in\{0,1\}^n} \left\{ \sum_{i=1}^{n} c_i^s y_i \colon y \leq x,\, \sum_{i=1}^{n} y_i \geq \sum_{i=1}^{n} x_i - k \right\} \right\}. \tag{11}$$

In Problem (11), we consider the entire scenario set S again. To tackle this problem effectively, we thus resort to solving an equivalent compact reformulation, which we state in the following proposition.

Proposition 4. *Let $x \in X$ be given arbitrarily. Then, Problem (11) can be solved as the mixed-integer linear problem*

$$\begin{aligned} \max_{z,\lambda,\mu} \quad & \left(\sum_{i=1}^{n} x_i - k\right)\lambda - \sum_{i=1}^{n} x_i \mu_i \\ \text{s.t.} \quad & \sum_{i=1}^{n} z_i \leq \Gamma, \\ & \lambda - \mu_i \leq \bar{c}_i + \Delta c_i z_i, \quad i \in N, \\ & z \in \{0,1\}^n, \ \mu \in \mathbb{R}^n_{\geq 0}, \ \lambda \in \mathbb{R}_{\geq 0}. \end{aligned} \tag{12}$$

The proof of Proposition 4 can be found in Appendix D. The CCG algorithm for solving the k-delete recoverable robust problem (2) using the scenario-based formulation is formally stated in Algorithm 1. In Line 1, we initialize lower and upper bounds for Problem (2) with L and U, respectively. As long as the optimality gap is not closed, i.e., $U > L$ holds, we alternate between solving the master problem (Line 3) and the MILP reformulation of the adversarial problem (Line 7). In Line 12, the z^j component of an optimal solution to the latter is then used to generate a new scenario that is missing from the current formulation. Note that we need to specify an initial subset of scenarios S^0 in Algorithm 1, which can be done in various ways. A straightforward initialization is to consider only the nominal scenario, i.e, to set $S^0 := \{s = 1 \colon c^s = \bar{c}\}$.

Algorithm 1 CCG for the Scenario-Based Formulation

Input: An instance of Problem (2), a subset of scenarios $S^0 \subseteq S$, exact solution methods for Problems (10) and (12)
Output: An optimal solution to Problem (2)
1: Set $L \leftarrow -\infty$ and $U \leftarrow +\infty$.
2: **for** $j = 0, 1, \ldots$ **do**
3: Compute an optimal solution (x^j, y^j, η^j) to Problem (10) with the set S^j.
4: Set $L \leftarrow f^\top x^j + \eta^j$.
5: **if** $L \geq U$ **then**
6: **return** x^j
7: Compute an optimal solution (z^j, λ^j, μ^j) to the x^j-parameterized problem (12) and let $R(x^j)$ denote its optimal objective function value.
8: Set $U \leftarrow \min\{U, f^\top x^j + R(x^j)\}$.
9: **if** $L \geq U$ **then**
10: **return** x^j
11: Define a new scenario s with $c_i^s \leftarrow \bar{c}_i + \Delta c_i z_i^j$ for all $i \in N$.
12: Set $S^{j+1} \leftarrow S^j \cup \{s\}$ and $j \leftarrow j + 1$.

Theorem 4. *Algorithm 1 terminates after finitely many iterations with a globally optimal solution to the k-delete recoverable robust problem (2).*

Proof. Let (x^j, y^j, η^j) be an optimal solution to the problem solved in Line 3 of Algorithm 1 for some $j \in \mathbb{N}$. Because (10) is a relaxation of Problem (4), $f^\top x^j + \eta^j$ is a valid lower bound for its optimal objective function value. Moreover, due to Proposition 4, solving Problem (12) for given $x^j \in X$ yields a valid upper bound for Problem (4). Hence, the bound updates in Lines 4 and 8 are correct and $L \leq U$ holds. If the termination criterion $L \geq U$ is satisfied, we thus have $L = U$, i.e., the optimality gap is closed. Finite termination now follows from the finiteness of the scenario set S, the finiteness of the branch-and-cut methods used to solve the MILPs in Lines 3 and 7, and from the fact that, in the worst case, the problem considered in Line 3 is the one in which all scenarios have been added, i.e., we solve Problem (4). The claim then follows from the equivalence of Problems (2) and (4). □

Next, we present the CCG method applied to the compact formulation of the k-delete recoverable robust problem (2).

4.2.2. *Compact Formulation.* In contrast to the scenario-based formulation (4), the compact formulation (9) does not rely on the scenario set S but on the sets $\mathcal{V}$ and $\mathcal{L}$ instead. Although $|\mathcal{V}|\,|\mathcal{L}| \in O(n^2)$, Problem (9) can still become large for practical instances. Hence, applying CCG may also be beneficial in this setting. In iteration j of the CCG method, we consider the (relaxed) master problem

$$\begin{aligned} \min_{x,\eta,y} \quad & f^\top x + \eta \\ \text{s.t.} \quad & \eta \geq \kappa_\ell(\nu) + \sum_{i=1}^n \tilde{c}^\ell_{\sigma_\nu(i)}(\nu) x_{\sigma_\nu(i)} - M(\nu)(1 - y^\nu_\ell), \quad \ell \in \mathcal{L},\, \nu \in \mathcal{V}^j, \\ & \sum_{\ell \in \mathcal{L}} y^\nu_\ell = 1, \quad \nu \in \mathcal{V}^j, \\ & x \in X,\, \eta \in \mathbb{R}_{\geq 0},\, y^\nu \in \{0,1\}^{|\mathcal{L}|}, \quad \nu \in \mathcal{V}^j, \end{aligned} \tag{13}$$

where $\mathcal{V}^j \subseteq \mathcal{V}$. Due to Proposition 2, the worst case for a given $x \in X$ and $\nu \in \mathcal{V}$ can be obtained by solving

$$\max_u \quad \sum_{i=1}^n \left(\bar{c}_i(\nu) + \Delta c_i(\nu) u_i\right) x_i - k\nu \quad \text{s.t.} \quad u \in \mathcal{U}_\Gamma. \tag{14}$$

Solving Problem (14) avoids computing the cost coefficients $\tilde{c}^\ell_i(\nu)$ for all $i \in N$, $\ell \in \mathcal{L}$, and $\nu \in \mathcal{V}$, which may be quite costly in practice. Moreover, we note that Problem (14) can be solved efficiently using a classic greedy algorithm.

We formally state the CCG method for the compact formulation in Algorithm 2. The first steps of the method are similar to those in Algorithm 1. However, in Line 8, we now solve an auxiliary problem for fixed x^j across all $\nu \in \mathcal{V}$. Note that the problems considered in Line 8 are independent of each other, i.e., they can be solved in parallel. Although one can, in principle, augment the set $\mathcal{V}^j$ with every $\nu \in \mathcal{V}$ for which $\eta^j < R(x^j, \nu)$ holds, we only add the worst-case uncertainty realization $\bar{\nu}$ to control the number of variables and constraints added in each iteration; see Lines 9–13. Moreover, the for-loop in Line 7 can, in principle, be terminated as soon as a $\nu \in \mathcal{V}$ with $\eta^j < R(x^j, \nu)$ is found. This can help reduce the computational burden of the CCG method. In this case, however, the upper bound cannot be updated in Line 9 as $R(x^j, \nu) < \max_{\bar{\nu} \in \mathcal{V}}\{R(x^j, \bar{\nu})\}$ may hold. To achieve a trade-off between the frequency at which the upper bound is updated and the computational burden of the method, we solve Problem (14) for all $\nu \in \mathcal{V}$ only in selected iterations, e.g., every 42nd iteration. In what follows, we refer to this as the *full evaluation frequency.* Finally, we note that finite termination of Algorithm 2 with a globally optimal solution to Problem (2) can be shown in analogy to the proof of Theorem 4.

4.2.3. *Extended Formulation.* Similar to the compact formulation, the extended formulation (7) is based on the projected uncertainty set $\mathcal{V}$ rather than the scenario set S. The CCG framework from Algorithm 2 can also be applied to the extended formulation, with the only difference being the (relaxed) master problem considered

Algorithm 2 CCG for the Compact Formulation

Input: An instance of Problem (2), exact solution methods for Problems (13) and (14)
Output: An optimal solution to Problem (2)

1: Set $L \leftarrow -\infty$, $U \leftarrow +\infty$, and $\mathcal{V}^0 \leftarrow \{0\}$.
2: **for** $j = 0, 1, \dots$ **do**
3: Compute an optimal solution (x^j, η^j, y^j) to Problem (13) with the set $\mathcal{V}^j$.
4: Set $L \leftarrow f^\top x^j + \eta^j$.
5: **if** $L \geq U$ **then**
6: **return** x^j
7: **for** $\nu \in \mathcal{V}$ **do**
8: Solve the (x^j, ν)-parameterized problem (14) and let $R(x^j, \nu)$ denote its optimal objective function value.
9: Set $\bar{\nu} \leftarrow \arg\max_{\nu \in \mathcal{V}} \{R(x^j, \nu)\}$ and $U \leftarrow \min\{U,\, f^\top x^j + R(x^j, \bar{\nu})\}$.
10: **if** $L \geq U$ **then**
11: **return** x^j
12: **if** $\eta^j < R(x^j, \bar{\nu})$ **then**
13: Set $\mathcal{V}^{j+1} \leftarrow \mathcal{V}^j \cup \{\bar{\nu}\}$ and $j \leftarrow j + 1$.

in Line 3 of the method. For iteration j with $\mathcal{V}^j \subseteq \mathcal{V}$, this problem is given by

$$\begin{aligned} \min_{x,\eta,w,z} \quad & f^\top x + \eta \\ \text{s.t.} \quad & \eta \geq \sum_{i=1}^n \bar{c}_i(\nu) x_i + \Gamma w^\nu + \sum_{i=1}^n z_i^\nu - k\nu, \quad \nu \in \mathcal{V}^j, \\ & w^\nu + z_i^\nu \geq \Delta c_i(\nu) x_i, \quad i \in N,\, \nu \in \mathcal{V}^j, \\ & x \in X,\, \eta \in \mathbb{R}_{\geq 0},\, w^\nu \in \mathbb{R}_{\geq 0},\, z^\nu \in \mathbb{R}^n_{\geq 0}, \quad \nu \in \mathcal{V}^j. \end{aligned}$$

4.3. **Branch-and-Cut.** To solve the projection-based, compact, and extended formulations of Problem (2), we now present branch-and-cut frameworks that are similar to (generalized) Benders decomposition (Benders 1962; Geoffrion 1972). In these methods, we only add constraints associated with violated scenarios and keep the sets of variables fixed. This is in contrast to the CCG methods discussed earlier, in which both variables and constraints are added iteratively. We first present the branch-and-cut framework for the projection-based formulation and then discuss what needs to be adapted to tackle the extended and the compact formulations.

4.3.1. *Projection-Based Formulation.* In the projection-based formulation (6), we only add one additional continuous variable (the variable η used for the epigraph reformulation) for modeling the uncertainties and the recovery actions. Hence, we can tackle Problem (6) using branch-and-cut. We start by solving the linear problem

$$\min_{x,\eta} \quad f^\top x + \eta \quad \text{s.t.} \quad (x, \eta) \in \Omega_0 := \left\{ (\bar{x}, \bar{\eta}) \in \bar{X} \times \mathbb{R}_{\geq 0} \right\}, \tag{15}$$

where $\bar{X} \subseteq \mathbb{R}^n$ is a continuous relaxation of X, i.e., the integer points contained in $\bar{X}$ coincide with X. After solving Problem (15), we iteratively augment the set Ω_0 to ensure integer feasibility and to approximate the worst-case recovery costs until a solution (x^*, η^*) to Problem (15) is also feasible for Problem (6). At node j of the branch-and-cut search tree, we consider the problem

$$\min_{x,\eta} \quad f^\top x + \eta \quad \text{s.t.} \quad (x, \eta) \in \Omega_j \subseteq \mathbb{R}^n \times \mathbb{R}_{\geq 0}, \tag{16}$$

where the set Ω_j is obtained from Ω_0 by adding all valid inequalities that have been generated along the path from the root to node j—both to include new scenarios for the uncertainty and to separate fractional solutions—together with all branching decisions made along that path. We state the method for processing node j of the branch-and-cut search tree in Algorithm 3.

Algorithm 3 Processing Node j Using the Projection-Based Formulation

Input: Exact solution methods for Problems (14) and (16), an upper bound U

Output: An indication of whether node j is fathomed or whether two new subproblems are generated due to branching

```
1: Set resolve ← False.
2: Solve Problem (16).
3: if Problem (16) is infeasible then
4:     Fathom the current node, i.e., go back to the main method.
5: Let (x^j, η^j) denote the solution to Problem (16).
6: if f^⊤ x^j + η^j ≥ U then
7:     Fathom the current node, i.e., go back to the main method.
8: if x^j ∉ X then
9:     Either generate cuts valid for Ω_j ∩ (X × ℝ_{≥0}), augment Ω_j, and go to Step 2,
       or branch and go back to the main method.
10: for ν ∈ 𝒱 do
11:     Compute an optimal solution u^j to the (x^j, ν)-parameterized problem (14)
        and let R(x^j, ν) denote its optimal objective function value.
12:     if η^j < R(x^j, ν) then
13:         Set
                Ω_j ← Ω_j ∩ {(x, η): η ≥ Σ_{i=1}^n (c̄_i(ν) + Δc_i(ν) u_i^j) x_i − kν}
            and resolve ← True.
14: if resolve then
15:     Go to Step 1.
16: Set U ← f^⊤ x^j + η^j, update the incumbent solution, and fathom the current
    node, i.e., go back to the main method.
```

If Problem (16) is infeasible or if its optimal objective function value exceeds the current upper bound U, we fathom node j; see Lines 4 and 7 in Algorithm 3. Otherwise, we do the following. First, we check if the variables x^j satisfy the integrality constraints, i.e., we check if $x^j \in X \subseteq \{0,1\}^n$ holds. In the case of fractional solutions, we can use standard cutting planes from mixed-integer linear optimization, see, e.g., Clautiaux and Ljubić (2025) and Cornuéjols (2008), or branch to separate (x^j, η^j); see Line 9. If $x^j \in X$ holds, we proceed by checking whether (x^j, η^j) satisfies

$$\eta^j \geq \sum_{i=1}^n c_i^s(\nu) x_i^j - k\nu, \quad \nu \in \mathcal{V},\ s \in S(\nu).$$

To this end, we solve Problem (14) with $x = x^j$ for all $\nu \in \mathcal{V}$. Because these problems are independent of each other, we can solve them in parallel. If

$$\eta^j \geq \max_u \left\{ \sum_{i=1}^n \left(\bar{c}_i(\nu) + \Delta c_i(\nu) u_i\right) x_i^j - k\nu \colon u \in \mathcal{U}_\Gamma \right\}$$

holds for all $\nu \in \mathcal{V}$, we update the incumbent and fathom the current node; see Line 16. Otherwise, we augment the set Ω_j with a valid inequality that separates the point (x^j, η^j); see Line 13. Note that a cut is added for each $\nu \in \mathcal{V}$ for which $\eta^j < R(x^j, \nu)$ holds, i.e., up to $|\mathcal{V}|$ cuts may be added at each node. However, it is also valid to consider, e.g., adding only the most violated cut for the given $x^j \in X$. We present different cut separation strategies in Section 5.2.3.

Theorem 5. *If we embed Algorithm 3 into a classic branch-and-bound framework, we obtain a method that terminates with (x^*, η^*), where x^* is an optimal solution to the k-delete recoverable robust problem (2), after finitely many iterations and after adding an overall finite number of cuts.*

Proof. Finite termination follows from the finiteness of the number of feasible first-stage decisions $x \in X \subseteq \{0,1\}^n$, the finiteness of the branch-and-cut method used to solve the separation problem (14), and from the fact that a pair (x, η) cannot occur twice during the execution of the method. We show the latter by contradiction. To this end, suppose that there exist solutions $(x^j, \eta^j) = (x^l, \eta^l)$ at nodes j and l with $j < l$. Then, by Line 13 of Algorithm 3, we have

$$\eta^j = \eta^l \geq \sum_{i=1}^{n} c_i^s(\nu) x_i^j - k\nu, \quad \nu \in \mathcal{V},\, s \in S(\nu),$$

i.e., the termination criterion was already satisfied at node j. Hence, an optimal solution cannot be overlooked. Finally, the number of cuts possibly added to the problem formulation is finite because $|X|, |\mathcal{V}| < \infty$. □

Next, we discuss how to adapt the node processing procedure in Algorithm 3 so that it can be applied to the compact formulation of Problem (2).

4.3.2. *Compact Formulation.* In the compact formulation (9), the number of variables needed to represent uncertainties and recovery actions is in $O(n^2)$; cf. Table 1. Hence, we can keep the full set of variables in the master problem and generate the associated constraints on the fly. To process nodes using the compact reformulation (9), we thus make the following modifications to Algorithm 3:

(i) Instead of Problem (16), the problem considered at node j is

$$\min_{x,\eta,y} \quad f^\top x + \eta \quad \text{s.t.} \quad (x, \eta, y) \in \Omega_j \subseteq \mathbb{R}^n \times \mathbb{R}_{\geq 0} \times [0,1]^{|\mathcal{L}| \times |\mathcal{V}|},$$

i.e., we now operate in the (x, η, y)-space. The set Ω_j is obtained from

$$\Omega_0 := \left\{ (x, \eta, y) \in \bar{X} \times \mathbb{R}_{\geq 0} \times [0,1]^{|\mathcal{L}| \times |\mathcal{V}|} \colon \sum_{\ell \in \mathcal{L}} y_\ell^\nu = 1,\, \nu \in \mathcal{V} \right\}$$

by adding all valid inequalities that have been generated along the path from the root to node j as well as all branching decisions made along that path.

(ii) In Line 8 of Algorithm 3, we additionally check whether the y^j component of a solution (x^j, η^j, y^j) satisfies all integrality constraints. As before, fractional solutions are handled by either cutting or branching.

(iii) If a solution (x^j, η^j, y^j) to the problem at node j satisfies $\eta^j < R(x^j, \nu)$ for some $\nu \in \mathcal{V}$, we augment the set Ω_j with $|\mathcal{L}|$ inequality constraints, namely

$$\eta \geq \kappa_\ell(\nu) + \sum_{i=1}^{n} \tilde{c}_{\sigma_\nu(i)}^\ell(\nu) x_{\sigma_\nu(i)} - M(\nu)(1 - y_\ell^\nu), \quad \ell \in \mathcal{L},$$

in Line 13 of Algorithm 3.

Finite termination of the overall branch-and-cut method in which this modified node processing scheme is embedded can be shown in analogy to the proof of Theorem 5.

4.3.3. *Extended Formulation.* The setting for the extended formulation (7) is very similar to that of the compact formulation, with the main difference being the considered variable space. To process node j of the branch-and-cut search tree using the extended formulation, we replace Problem (16) in Algorithm 3 with

$$\min_{x,\eta,w,z} \quad f^\top x + \eta \quad \text{s.t.} \quad (x, \eta, w, z) \in \Omega_j \subseteq \mathbb{R}^n \times \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0}^{n \times |\mathcal{V}|},$$

where the set Ω_j is obtained from

$$\Omega_0 := \left\{ (x, \eta, w, z) \in \bar{X} \times \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0}^{n \times |\mathcal{V}|} \right\}$$

by adding all valid inequalities that have been generated along the path from the root to node j as well as all branching decisions made along that path. If a solution (x^j, η^j, w^j, z^j) to the problem at node j satisfies $\eta^j < R(x^j, \nu^*)$ for some $\nu^* \in \mathcal{V}$, we augment the set Ω_j with $n+1$ inequality constraints, namely

$$\eta \geq \sum_{i=1}^{n} \bar{c}_i(\nu^*) x_i + \Gamma w^{\nu^*} + \sum_{i=1}^{n} z_i^{\nu^*} - k\nu^*, \quad w^{\nu^*} + z_i^{\nu^*} \geq \Delta c_i(\nu^*) x_i, \quad i \in N,$$

in Line 13 of Algorithm 3.

5. Computational Results

We now computationally assess and compare the performance of the exact solution methods presented in Section 4. To this end, we briefly describe the generation of the test instances and the computational setup in Sections 5.1 and 5.2, respectively. In Sections 5.3 and 5.4, we compare the solution methods presented in this paper with respect to (i) runtimes, (ii) the number of instances solved to global optimality, and (iii) optimality gaps. Note that, to the best of our knowledge, there are currently no other methods in the literature that tackle k-delete recoverable robust 0–1 problems under budgeted uncertainty. Hence, there are no alternative methods that we could compare with. In addition to computational metrics, we discuss qualitative aspects of solutions in Section 5.5. To this end, we consider the *gain of recovery*, i.e., the decrease in the optimal objective value obtained by allowing for recovery actions. Moreover, we elaborate on the benefits of using a robust model by analyzing the reduction in the worst-case objective value achieved when considering robust rather than nominal solutions. Our main findings are summarized in Figures 1–8 and Table 4. We provide supplementary results in Appendix E.

5.1. Generation of Test Instances.

In our computational study, we consider instances of two well-known combinatorial optimization problems, both of which capture many practical applications: the assignment problem and the single-source capacitated facility location problem. Comprehensive overviews of these problem classes can, e.g., be found in Burkard et al. (2012) and Saldanha-da-Gama and Wang (2024). The latter particularly addresses facility location problems under uncertainty. In what follows, we describe the problem setting for both classes and explain how we generate the corresponding data.

5.1.1. *Assignment Problem.*

The assignment problem (AP) seeks to assign n agents, i.e., staff, machines, or resources, to n tasks so that each agent is assigned to exactly one task and each task is assigned to exactly one agent. Assigning agent i to task j incurs a cost of $c_{ij} \in \mathbb{R}_{\geq 0}$, and the objective is to determine an assignment that minimizes the total cost. In its deterministic form, the assignment problem reads

$$\begin{aligned} \min_{x} \quad & \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_{ij} = 1, \quad j \in [n], \\ & \sum_{j=1}^{n} x_{ij} = 1, \quad i \in [n], \\ & x_{ij} \in \{0, 1\}, \quad i, j \in [n]. \end{aligned}$$

Table 3. The number of customers ("$|I|$"), potential facility locations ("$|J|$"), and instances ("# instances") for cap41–cap134.

name	$\|I\|$	$\|J\|$	# instances
cap41–cap74	50	16	13
cap81–cap104	50	25	12
cap121–cap134	50	50	12

We generate 40 deterministic instances of the assignment problem as follows. Starting from the assign800 instance of the OR-Library (Beasley 1990a,b), we randomly select a subset of tasks until the desired number n is reached. For each $n \in \{25, 50, 75, 100\}$, we generate 10 instances. To adapt these instances to a recoverable robust setup, we use a procedure similar to that in Büsing et al. (2011b), which works as follows. For all $i, j \in [n]$, we set the first-stage cost to $f_{ij} = \lceil 0.6c_{ij} \rceil$ and the nominal cost to $\bar{c}_{ij} = \lceil 0.2c_{ij} \rceil$. The worst-case cost increase Δc_{ij} is obtained by generating a uniformly distributed random value $\delta_{ij} \in [0.2, 0.4)$ and setting $\Delta c_{ij} = \lceil \delta_{ij} c_{ij} \rceil$. The robustness parameter is set to $\Gamma = \lceil \gamma n \rceil$ with $\gamma \in \{0.1, 0.25, 0.5\}$ and the recovery parameter is set to $k = \lceil \kappa n \rceil$ with $\kappa \in \{0.1, 0.25\}$. To sum up, we consider 240 instances of the k-delete recoverable robust assignment problem.

5.1.2. *Single-Source Capacitated Facility Location Problem.* Let I and J be given sets of customers and potential facility locations, respectively. Each customer $i \in I$ has a demand $d_i \in \mathbb{R}_{\geq 0}$ that needs to be served and each facility $j \in J$ is associated with an opening cost of $b_j \in \mathbb{R}_{\geq}$ and a capacity of $s_j \in \mathbb{R}_{\geq 0}$. Assigning customer i to facility j incurs a cost of $c_{ij} \in \mathbb{R}_{\geq 0}$. Each customer must be assigned to one facility, which supplies its entire demand, and the total demand served by a facility cannot exceed its capacity. The single-source capacitated facility location problem (SSCFLP) seeks to determine which facilities to open and how to assign customers so that the total cost of opening facilities and serving customers is minimized. The deterministic SSCFLP is given by

$$
\begin{aligned}
\min_{x,y} \quad & \sum_{j \in J} b_j y_j + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} \\
\text{s.t.} \quad & \sum_{j \in J} x_{ij} = 1, \quad i \in I, \\
& x_{ij} \leq y_j, \quad i \in I,\, j \in J, \\
& \sum_{i \in I} d_i x_{ij} \leq s_j y_j, \quad j \in J, \\
& x_{ij}, y_j \in \{0, 1\}, \quad i \in I,\, j \in J.
\end{aligned}
$$

We consider the 37 deterministic facility location problem instances from the OR-Library (Beasley 1990a,b) labeled cap41–cap134, for which a summary is given in Table 3. Because these instances correspond to the (multi-source) capacitated facility location problem, we adapted them to the single-source setting by requiring that each customer is served by exactly one facility. To ensure feasibility, we remove all customers whose demand exceeds the maximum capacity, i.e., we remove all i from I for which $d_i > \max\{s_j : j \in J\}$ holds. We assume that only the assignment costs c are uncertain, whereas the facility opening costs b remain deterministic. The robustness parameter is set to $\Gamma = \lceil \gamma \, |I| \rceil$ with $\gamma \in \{0.1, 0.25, 0.5\}$ and the recovery parameter is set to $k = \lceil \kappa \, |I| \rceil$ with $\kappa \in \{0.1, 0.25\}$. All remaining data, i.e., f, $\bar{c}$, and Δc, is generated in the same way as for the instances of the assignment problem. Overall, we consider 222 instances of the k-delete recoverable robust SSCFLP.

5.2. **Computational Setup.** All tests were carried out on an Intel XEON 8468 Sapphire at 2.1 GHz (8 cores) and 32 GB RAM, which is part of the High-Performance Computing cluster "CLAIX" at RWTH Aachen University. We implemented our solution approaches in Python 3.12.3 and use Gurobi 12.0.0 to solve all arising optimization problems. The code, along with the instance data used in our computational study, is publicly available at https://github.com/YasmineBeck/k-delete-recoverable-robust-methods. A time limit of 1 h was set for solving each instance. For the branch-and-cut methods, we add valid inequalities using Gurobi's lazy constraint callback, which requires setting the parameter LazyConstraints to 1. All other parameters were kept at their default settings. We now elaborate on the scenario initialization and the full evaluation frequency for the CCG methods as well as on the cut separation strategies used in our branch-and-cut approaches.

5.2.1. *Initializing the Scenario Set.* For initializing the column-and-constraint-generation methods, we need to specify the scenario sets S^0 and $\mathcal{V}^0$ in Algorithms 1 and 2, respectively. In preliminary computational experiments, we considered several ways to initialize these sets as singletons, including greedy and randomized approaches. These tests revealed that the impact of such scenario initialization strategies is marginal. In what follows, we thus only report the results for the setting in which the CCG methods are initialized using the nominal scenario, i.e., $S^0 := \{s = 1 \colon c^s = \bar{c}\}$ and $\mathcal{V}^0 := \{0\}$.

5.2.2. *Full Evaluation Frequency.* As discussed in Section 4.2.2, we aim for a trade-off between the frequency at which we update the upper bound and the computational effort of the CCG method by adjusting the full evaluation frequency, i.e., the frequency at which we solve Problem (14) for all $\nu \in \mathcal{V}$ in Algorithm 2. In preliminary computational experiments, we compared two strategies: performing a full evaluation (i) in every iteration or (ii) only every 10th iteration. In the latter case, we terminate the for-loop in Line 7 of Algorithm 2 as soon as a $\nu \in \mathcal{V}$ with $\eta^j < R(x^j, \nu)$ is found. Our tests revealed that the most suitable strategy strongly depends on the problem at hand. For AP, we observe that performing a full evaluation only every 10th iteration leads to considerably shorter runtimes, whereas for SSCFLP it seems beneficial to evaluate all $\nu \in \mathcal{V}$ in every iteration. In particular, this suggests that more involved problem classes (such as SSCFLP compared to AP) tend to benefit from evaluating all $\nu \in \mathcal{V}$ and potentially updating the upper bound more frequently. However, if solving the master problem is computationally rather cheap, this seems less advantageous. In what follows, we thus perform a full evaluation in every iteration for SSCFLP and only every 10th iteration for AP.

5.2.3. *Cut Separation Strategies.* As discussed in Section 4.3, cuts associated with multiple uncertainty realizations $\nu \in \mathcal{V}$ may be added at each node of the branch-and-cut search tree. Nevertheless, different cut separation strategies can be used to reduce the number of added cuts and, thus, potentially speed up the solution process. We consider four such strategies.

All-In: Add the cut(s) for all $\nu \in \mathcal{V}$ whose constraints are violated by the solution to the current node problem.
First-In: Iterate over $\nu \in \mathcal{V}$, add the cut(s) for the first uncertainty realization whose constraints are violated by the solution to the current node problem, and break the loop.
Shuffle-First-In: Randomly shuffle the elements in $\mathcal{V}$ and apply First-In.
Max-Violation: Add the cut(s) only for the uncertainty realization $\nu \in \mathcal{V}$ whose constraints are maximally violated by the solution to the current node problem.

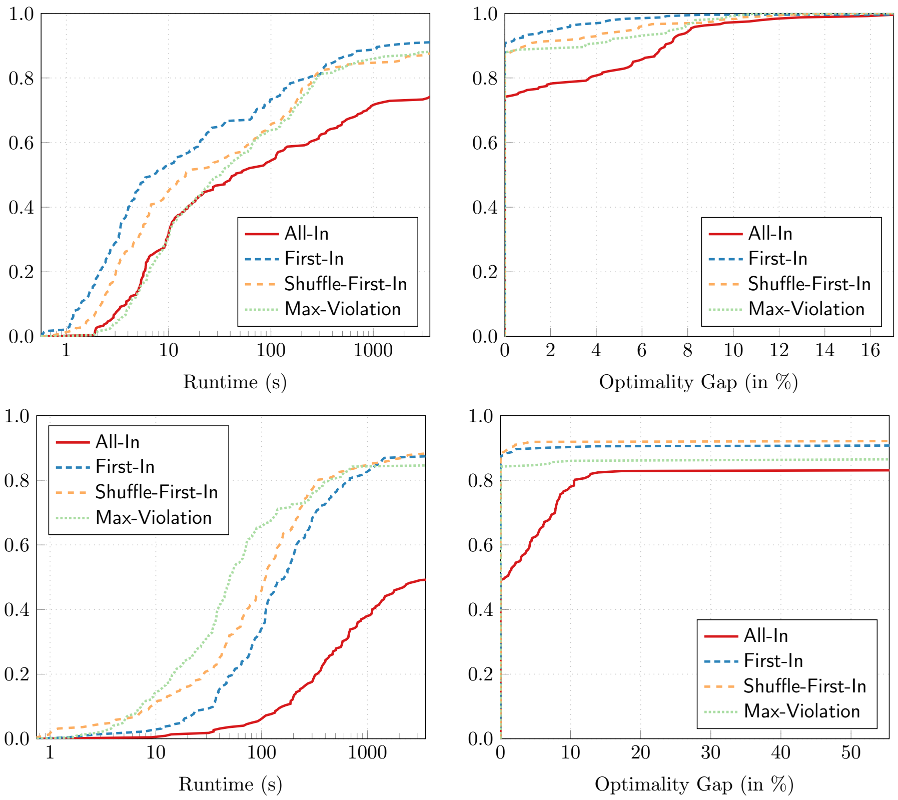

FIGURE 1. Log-scaled ECDFs of the runtimes (in s) and linear-scaled ECDFs of the optimality gaps (in %) for the branch-and-cut approaches with different cut separation strategies applied to the projection-based formulation. Results for AP are shown in the top figures and results for SSCFLP are shown in the bottom figures.

In Figure 1, we show empirical cumulative distribution functions (ECDFs) of the runtimes and the optimality gaps to compare the four considered cut separation strategies. The ECDFs can be interpreted as the percentage of instances (y-axis) that can be solved within a given time or with a given optimality gap (x-axis). Here, we exemplarily focus on the branch-and-cut methods applied to the projection-based formulation, but preliminary computational results revealed that the same qualitative observations can also be made for the approaches applied to the extended and the compact formulation.

For both AP and SSCFLP, we observe that adding only one cut per node, i.e., using strategies First-In, Shuffle-First-In, or Max-Violation, significantly improves the performance of the overall branch-and-cut method in terms of running times and, consequently, the number of instances solved to global optimality; see also Table 5 in Appendix E. Although the most effective choice among the considered strategies seems to depend on the specific problem under consideration, we observe that Shuffle-First-In consistently performs well for both the assignment problem and the single-source capacitated facility location problem. This can be explained by the fact that this cut separation strategy is computationally rather cheap and that randomization helps to avoid repeatedly adding cuts associated with the same scenario in consecutive iterations, resulting in a more balanced exploration of the scenario set. The previous observations align particularly well with the findings of

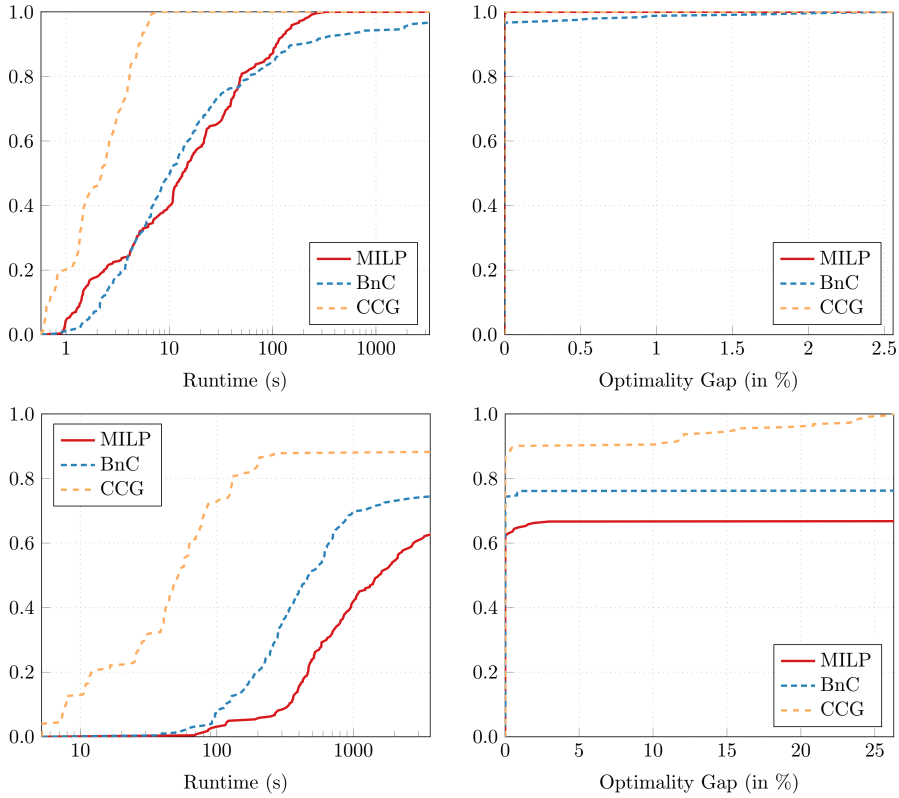

FIGURE 2. Log-scaled ECDFs of the runtimes (in s) and linear-scaled ECDFs of the optimality gaps (in %) for the approaches MILP, BnC, and CCG applied to the extended formulation (7). Results for AP are shown in the top figures and results for SSCFLP are shown in the bottom figures.

the computational studies in Beck (2024) and Beck et al. (2023), in which bilevel problems with Γ-robust followers are studied. As supported by the results in Goerigk et al. (2025), such robust bilevel problems exhibit a multi-level structure similar to that of the k-delete recoverable robust problem studied in this work. To sum up, we thus fix the cut separation strategy to Shuffle-First-In for all branch-and-cut approaches in the following.

5.3. **MILP vs. BnC vs. CCG.** Whereas only a single method (CCG or BnC) is available for solving the scenario-based formulation (4) and the projection-based formulation (6) of Problem (2), we can choose among three methods—the MILP, the branch-and-cut, and the CCG approach—for the extended formulation (7) and the compact formulation (9); cf. Table 2. In what follows, we compare these three approaches for both formulations.

5.3.1. *Extended Formulation* (7). In Figure 2, we show ECDFs of the runtimes and optimality gaps for the three approaches applied to the extended formulation. For AP, we observe that the CCG method and the MILP approach can solve all 240 considered instances to global optimality within the time limit of 1 h, whereas the BnC approach solves 232 instances (96.67 %). Nevertheless, BnC finds a feasible point with finite optimality gap for all instances, with the largest optimality gap observed being 2.56 %. Regarding runtimes, we emphasize that the CCG method

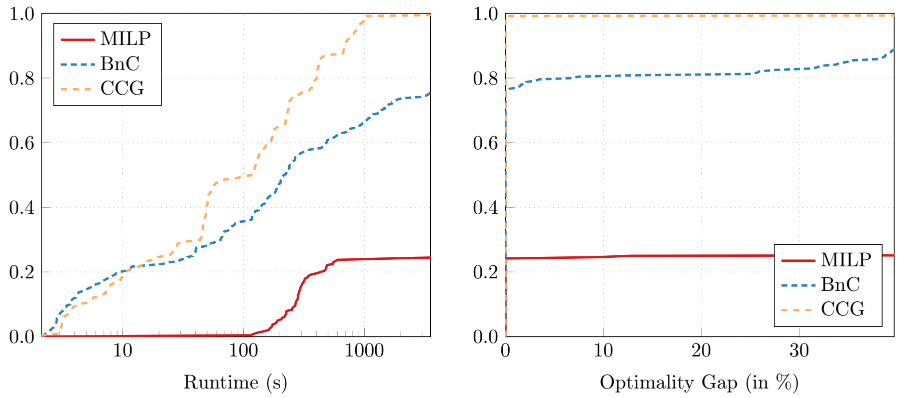

FIGURE 3. Log-scaled ECDFs of the runtimes (in s) and linear-scaled ECDFs of the optimality gaps (in %) for the approaches MILP, BnC, and CCG applied to the compact formulation (9) of AP.

clearly outperforms the other two approaches. The same qualitative behavior can also be observed for the SSCFLP instances. Again, CCG performs significantly better compared to the branch-and-cut and the MILP approach. As a result, more instances are solved to global optimality using CCG and, in particular, a feasible point with finite optimality gap is found for all 222 SSCFLP instances. In contrast, the MILP approach and BnC find feasible points with finite optimality gaps for only 66.67 % and 76.13 % of these instances, respectively. These observations are further supported by the results shown in Table 6 in Appendix E. Overall, applying CCG to the extended formulation thus seems to be the most effective approach.

5.3.2. *Compact Formulation* (9). For the assignment problem, the previous observations regarding the performance of the MILP, BnC, and CCG approaches are even more pronounced when considering the compact formulation. In Figure 3, we show ECDFs of the runtimes and the optimality gaps for the three approaches. It can again be seen that the CCG method performs considerably better than the other two approaches both in terms of running times and solution quality. In particular, because CCG can solve the AP instances more quickly, a larger number of instances can be solved to global optimality. Overall, 238 out of 240 instances (99.17 %) can be solved using CCG within the time limit of 1 h, whereas no feasible point can be computed for the two remaining instances. In contrast, a feasible point with finite optimality gap is found for only 25 % and 88.75 % of the instances using the MILP and the BnC approaches, respectively. The poor performance of the MILP approach for the compact formulation can be explained by the fact that computing the modified cost coefficients $\tilde{c}_i^{\ell}(\nu)$ for all $\ell \in \mathcal{L}$, $\nu \in \mathcal{V}$, and $i \in N$ is extremely expensive. In fact, this computation consumes the majority of both runtime and memory. Consequently, for many instances considered in this computational study, the model cannot be built due to either time or memory constraints.

For the same reasons, no incumbent solution is found for any instance of the single-source capacitated facility location problem using the MILP or CCG approaches. Therefore, we do not show ECDFs for the compact formulation applied to SSCFLP. Nevertheless, the branch-and-cut approach solves 100 out of 222 instances (45.05 %) to global optimality within the time limit of 1 h. For additional 72 instances, a feasible point with an open but finite gap of at most 10.86 % is found, whereas no

Table 4. The number of instances solved to global optimality ("solved"; out of 240 and 222 instances for AP and SSCFLP, respectively), the number of instances for which a feasible point with finite but non-zero gap is found ("open gap"), and the number of instances with infinite gap ("infinite gap") for "winner approaches" of the four considered formulations. For those instances with finite but non-zero gap, also the average gap ("average gap"; in %) is shown.

problem	formulation	solved	open gap	average gap	infinite gap
AP	scenario-based	144	95	3.88	1
	projection-based	210	30	5.19	0
	extended	240	0	–	0
	compact	238	0	–	2
SSCFLP	scenario-based	83	139	2.40	0
	projection-based	196	8	1.87	18
	extended	195	27	14.35	0
	compact	100	72	10.86	50

incumbent solution is found for the remaining 50 instances. All previous observations are also supported by the results shown in Table 7 in Appendix E.

To sum up, the CCG method outperforms the MILP and BnC approach for AP, whereas branch-and-cut is the only approach capable of tackling instances of SSCFLP when considering the compact formulation.

5.4. **Comparison of Formulations.** We now compare the four considered formulations—the scenario-based, projection-based, extended, and compact formulation—by considering their respective best-performing solution methods. This means that we apply CCG to the scenario-based and the extended formulation, and we apply BnC to the projection-based formulation. For the compact formulation, we make the following distinction: we use CCG for AP and BnC for SSCFLP.

In Figure 4, we show ECDFs of the runtimes and the optimality gaps for the approaches considered with these four formulations. Moreover, we report the number of instances solved to global optimality in Table 4. For AP, it can be seen that the only approach capable of solving all of the 240 considered instances to global optimality is the CCG approach applied to the extended formulation. In particular, we observe that this approach also outperforms all remaining ones in terms of running times. For SSCFLP, we observe that the extended formulation again yields an approach that performs considerably better than the remaining ones in terms of runtimes. However, when considering the ECDFs of the optimality gaps in Figure 4, it can be seen that the solution quality obtained from the scenario-based formulation is slightly better. Nevertheless, the CCG method applied to the extended formulation significantly outperforms this approach in terms of running times. We also point out that BnC for the compact formulation performs similarly well compared to the CCG approach applied to the extended formulation. However, as more instances can be solved to global optimality using the extended formulation, we consider the latter the overall best formulation for solving SSCFLP instances.

Nevertheless, let us also compare our methods with respect to so-called idealized parallel runtimes, which reflect the runtime of a method assuming that sufficient capacities are available to solve all arising sub-problems in parallel. As discussed in Section 4, some of our methods can be partially parallelized. Hence, for each instance, we compute the idealized parallel runtime by first solving all sub-problems

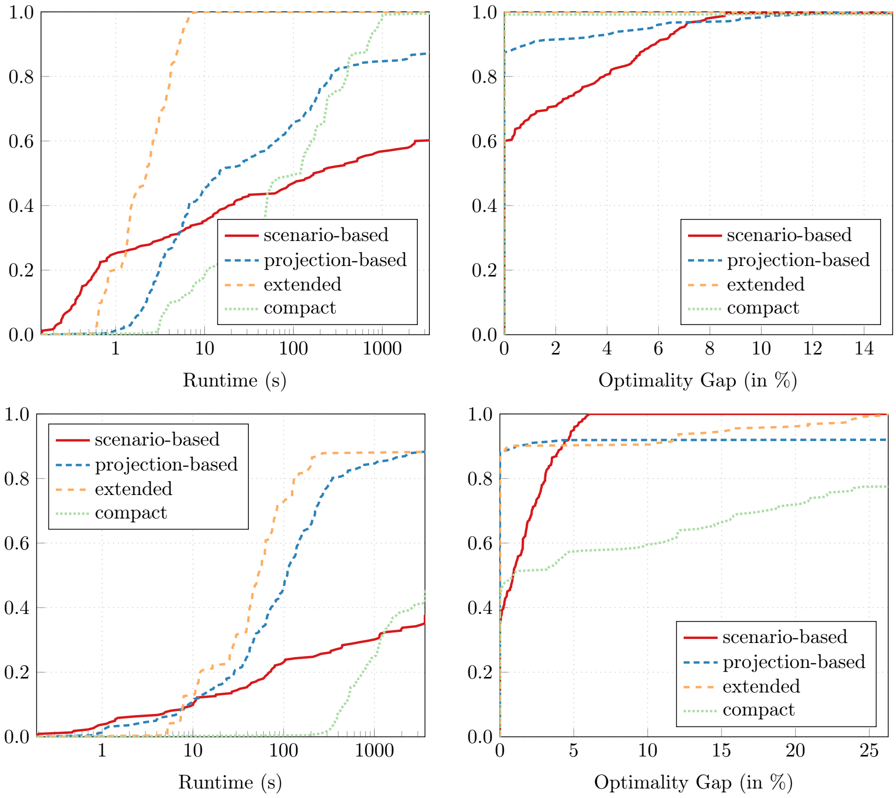

FIGURE 4. Log-scaled ECDFs of the runtimes (in s) and linear-scaled ECDFs of the optimality gaps (in %) for the best-performing approaches applied to the scenario-based, projection-based, extended, and compact formulations of Problem (2). Results for AP are shown in the top figures and results for SSCFLP are shown in the bottom figures.

sequentially and then taking the maximum runtime across all sub-problems. If an instance cannot be solved within the time limit in the sequential setting, it is thus also considered as unsolved in the idealized parallel setting. In Figure 5, we show ECDFs of the idealized parallel runtimes. It can be seen that the previous observations are even more pronounced in this setting. Overall, applying the CCG method to the extended formulation yields an approach that clearly outperforms the remaining ones in terms of idealized parallel runtimes. To sum up, the extended formulation (7) thus seems to be the most effective formulation for solving the k-delete recoverable robust 0–1 problem.

5.5. **Gain of Recovery and Impact of Robust Modeling.** To conclude this computational study, let us now discuss some qualitative aspects of solutions. We start with evaluating the gain of recovery, i.e., the decrease in the optimal objective value obtained by incorporating the possibility of recovery actions in the modeling. To this end, we compare the optimal objective value of the k-delete recoverable robust problem with the objective value obtained by first solving the robust problem without recovery (i.e., Problem (2) with $k = 0$) and then deleting up to k components of the optimal solution ex post for which the worst-case cost increase is realized. In Figure 6, we show boxplots for the gain of recovery for the different choices of the recovery parameter k. Note that we only include instances that have been

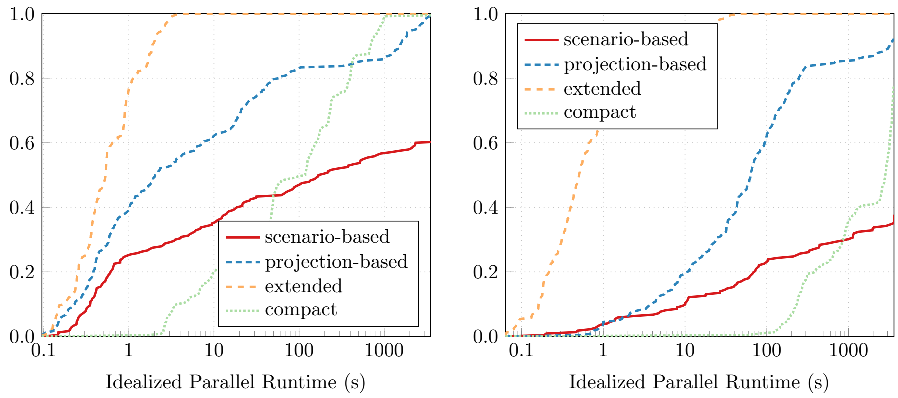

FIGURE 5. Log-scaled ECDFs of the idealized parallel runtimes (in s) for the best-performing approaches applied to the scenario-based, projection-based, extended, and compact formulations of Problem (2). Results for AP are shown on the right, whereas results for SSCFLP are shown on the left.

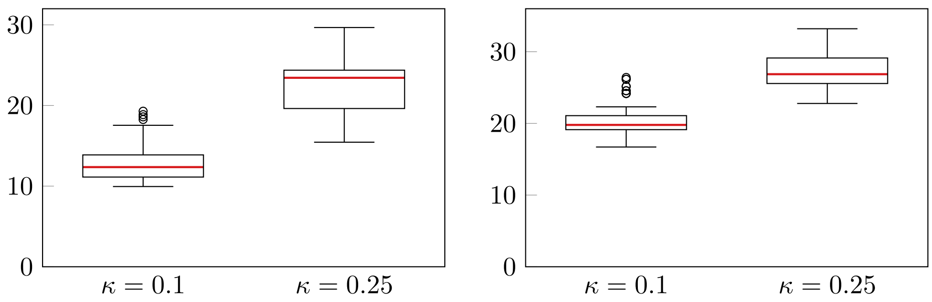

FIGURE 6. Boxplots for the gain of recovery (in %). Results for AP are shown on the left, whereas results for SSCFLP are shown on the right. Instances with the recovery parameter $k = \lceil \kappa n \rceil$ are considered, where n is the number of variables in each instance.

solved to global optimality. In particular, our comparisons are based on the results obtained from the CCG method applied to the extended formulation, which is the most effective solution approach for both AP and SSCFLP instances, in order to include as many instances solved to global optimality as possible. From Figure 6, it can be clearly seen that incorporating the possibility of recovery actions in the planning phase through a two-stage robust framework enables more flexible and informed decision-making, leading to substantially reduced costs. In particular, up to 29.67 % and 33.21 % of the costs can be saved for AP and SSCFLP, respectively, if recovery actions are incorporated in the modeling. Moreover, our results suggest that these cost benefits increase with the number of allowed recovery actions. Nevertheless, these benefits also come at a certain price. We thus also analyze the "price of recovery", i.e., the additional cost incurred by a recoverable robust solution if recovery actions are not allowed after all. To this end, we first compute an optimal first-stage decision x^* for the k-delete recoverable robust problem (2) and then evaluate the worst-case cost of this solution under the assumption that no components of x^* can be deleted. We then compare this objective value to the one obtained by solving the robust problem without recovery (i.e., Problem (2) with $k = 0$). For the price of recovery, we show boxplots in Figure 7. We observe

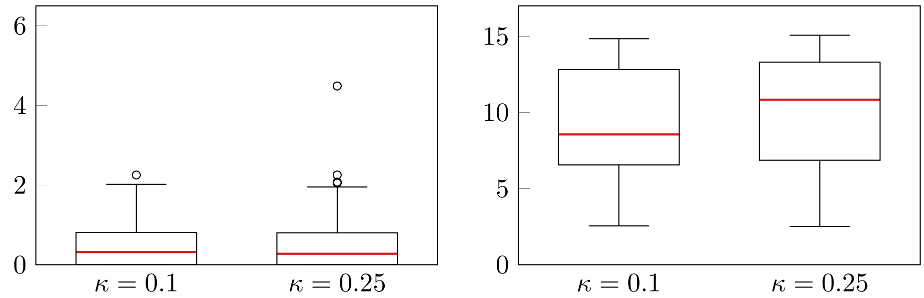

FIGURE 7. Boxplots of the price of recovery (in %). Results for AP are shown on the left, whereas results for SSCFLP are shown on the right. Instances with the recovery parameter $k = \lceil \kappa n \rceil$ are considered, where n is the number of variables in each instance.

that incorporating a recoverable robust solution without being able to delete items afterward can increase costs by up to 4.49 % and 15.07 % for AP and SSCFLP, respectively. Comparing these increases with the potential savings mentioned above shows that it can clearly be beneficial to incorporate recovery actions in the decision-making process. Overall, our results indicate that allowing for recovery actions can yield substantial cost reductions, whereas the associated cost increase without the possibility of recovery remains moderate and manageable.

Let us now comment on the impact of robustness on the costs. To this end, we compare the optimal objective value of a recoverable robust solution to that of the respective deterministic two-stage variant of the problem (i.e., Problem (2) with $\Gamma = 0$), in which the worst-case cost increase is computed ex post. In Figure 8, we show boxplots for the cost savings obtained through considering a robust framework. We observe that these savings increase with larger values of Γ. Overall, up to 16.52 % and 21.74 % of the costs can be saved for AP and SSCFLP, respectively. This demonstrates that our k-delete recoverable robust framework allows for decisions that are more resilient against uncertainties and cost variability. Finally, we note that the previous analysis is closely related to the so-called price of robustness (Bertsimas and Sim 2004), which measures the increase in the optimal objective function due to robustification. However, our metric differs from the price of robustness in the sense that we evaluate the decrease in the optimal objective function when comparing robust and nominal solutions under adverse settings.

6. CONCLUSION

In this paper, we study the k-delete recoverable robust 0–1 problem in which a decision-maker solves a combinatorial optimization problem subject to objective uncertainty. In this setting, the decision-maker first commits to an initial plan and may revoke up to k components of this initial decision after the uncertainty is revealed. The underlying uncertainty is modeled using a budgeted uncertainty set so that the decision-maker only hedges against at most Γ deviations in the uncertain parameters. We derive four equivalent reformulations of the k-delete recoverable robust problem—a scenario-based, a projection-based, an extended, and a compact reformulation. For each formulation, we present exact solution methods, including (i) compact formulations that can be tackled using general-purpose MILP solvers, (ii) branch-and-cut frameworks, and (iii) column-and-constraint generation algorithms. The performance of all presented solution approaches is assessed in an extensive computational study on instances of the assignment problem and the single-source capacitated facility location problem. Our computational results

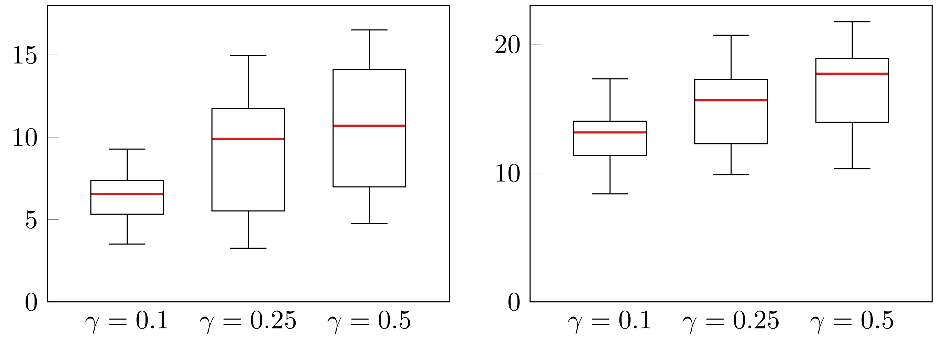

FIGURE 8. Boxplots of the impact of robustness (in %). Results for AP are shown on the left, whereas results for SSCFLP are shown on the right. Instances with the robustness parameter $\Gamma = \lceil \gamma n \rceil$ are considered, where n is the number of variables in each instance.

show that incorporating the possibility of recovery actions in the planning phase can substantially improve the optimal objective value compared to withdrawing commitments after solving a deterministic model. Among all solution methods and reformulations that we consider in this paper, we further observe that using a column-and-constraint-generation method to solve the extended formulation seems to be the most effective one.

Finally, let us mention that our modeling of the k-delete recoverable robust problem allows recovered solutions to violate some constraints of the original model. For instance, some customers may remain unserved in a facility location problem. Although such situations are common in practice, one may still be interested in models that preserve feasibility after recovery. Developing solution methods for such models typically requires different solution techniques, which is beyond the scope of the present paper and left for future research. Another aspect that remains open is the computational complexity of the k-delete recoverable robust problem studied in this work. Similar settings, however, have been studied in Grüne and Wulf (2024), which may provide useful insights for addressing this question in future work. Moreover, it may be interesting to study alternative recovery actions, such as allowing insertions or both insertions and deletions. Extending the solution methods developed here, together with existing approaches in recoverable robust optimization, to handle such recovery actions in general combinatorial problems would provide a natural next step. In addition, future work could consider other types of uncertainty, such as uncertain constraints, and explore robust models beyond the budgeted uncertainty framework studied in this work.

Acknowledgment

This research was funded in part by the French National Research Agency (ANR) under project ANR-24-CE48-3565-03.

References

Álvarez-Miranda, E., E. Fernández, and I. Ljubić (2015a). "The recoverable robust facility location problem." In: *Transportation Research Part B: Methodological* 79, pp. 93–120. DOI: 10.1016/j.trb.2015.06.001.

Álvarez-Miranda, E., I. Ljubić, S. Raghavan, and P. Toth (2015b). "The Recoverable Robust Two-Level Network Design Problem." In: *INFORMS Journal on Computing* 27 (1), pp. 1–19. DOI: 10.1287/ijoc.2014.0606.

Arslan, A. N. and M. Poss (2024). "Uncertainty reduction in robust optimization." In: *Operations Research Letters* 55, p. 107131. DOI: 10.1016/j.orl.2024.107131.

Beasley, J. E. (1990a). *OR-Library*. Last accessed October 30, 2025. URL: https://people.brunel.ac.uk/~mastjjb/jeb/info.html.

Beasley, J. E. (1990b). "OR-Library: Distributing Test Problems by Electronic Mail." In: *Journal of the Operational Research Society* 41.11, pp. 1069–1072. DOI: 10.1057/jors.1990.166.

Beck, Y. (2024). "Mixed-Integer Optimization Techniques for Robust Bilevel Problems with Here-and-Now Followers." PhD thesis. Trier University, Department of Mathematics. DOI: 10.25353/ubtr-923d-98c6-ddfc. URL: https://ubt.opus.hbz-nrw.de/frontdoor/index/index/docId/2432.

Beck, Y., I. Ljubić, and M. Schmidt (2023). "Exact methods for discrete Γ-robust interdiction problems with an application to the bilevel knapsack problem." In: *Mathematical Programming Computation* 15 (4), pp. 733–783. DOI: 10.1007/s12532-023-00244-6.

Ben-Tal, A., A. Goryashko, E. Guslitzer, and A. Nemirovski (2004). "Adjustable robust solutions of uncertain linear programs." In: *Mathematical Programming* 99.2, pp. 351–376. DOI: 10.1007/s10107-003-0454-y.

Ben-Tal, A., L. Ghaoui, and A. Nemirovski (2009). *Robust Optimization*. DOI: 10.1515/9781400831050.

Benders, J. F. (1962). "Partitioning procedures for solving mixed-variables programming problems." In: *Numerische Mathematik* 4, pp. 238–252. DOI: 10.1007/BF01386316.

Bertsimas, D., D. Brown, and C. Caramanis (2011). "Theory and Applications of Robust Optimization." In: *SIAM Review* 53.3, pp. 464–501. DOI: 10.1137/080734510.

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

Bertsimas, D. and M. Sim (2004). "The Price of Robustness." In: *Operations Research* 52.1, pp. 35–53. DOI: 10.1287/opre.1030.0065.

Bouman, P. C., J. M. van den Akker, and J. A. Hoogeveen (2011). "Recoverable Robustness by Column Generation." In: *Algorithms — ESA 2011*. Ed. by C. Demetrescu and M. M. Halldórsson. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 215–226.

Burkard, R., M. Dell'Amico, and S. Martello (2012). *Assignment Problems*. Society for Industrial and Applied Mathematics. DOI: 10.1137/1.9781611972238.

Büsing, C. (2009). "The Exact Subgraph Recoverable Robust Shortest Path Problem." In: *Robust and Online Large-Scale Optimization: Models and Techniques for Transportation Systems*. Ed. by R. K. Ahuja, R. H. Möhring, and C. D. Zaroliagis. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 231–248. DOI: 10.1007/978-3-642-05465-5_9.

Büsing, C., S. Goderbauer, A. M. C. A. Koster, and M. Kutschka (2019). "Formulations and algorithms for the recoverable Γ-robust knapsack problem." In: *EURO Journal on Computational Optimization* 7 (1), pp. 15–45. DOI: 10.1007/s13675-018-0107-9.

Büsing, C., A. M. C. A. Koster, and M. Kutschka (2011a). "Recoverable Robust Knapsacks: Γ-Scenarios." In: *Network Optimization*. Ed. by J. Pahl, T. Reiners, and S. Voß. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 583–588. DOI: 10.1007/978-3-642-21527-8_65.

Büsing, C., A. M. C. A. Koster, and M. Kutschka (2011b). "Recoverable robust knapsacks: the discrete scenario case." In: *Optimization Letters* 5 (3), pp. 379–392. DOI: 10.1007/s11590-011-0307-1.

Cacchiani, V., A. Caprara, L. Galli, L. Kroon, G. Maróti, P. Toth, et al. (2008). "Recoverable robustness for railway rolling stock planning." In: *Open Access Series in Informatics* 9, pp. 1–13.

Cicerone, S., G. D'Angelo, G. Di Stefano, D. Frigioni, A. Navarra, M. Schachtebeck, and A. Schöbel (2009). "Recoverable Robustness in Shunting and Timetabling." In: *Robust and Online Large-Scale Optimization: Models and Techniques for Transportation Systems*. Ed. by R. K. Ahuja, R. H. Möhring, and C. D. Zaroliagis. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 28–60. DOI: 10.1007/978-3-642-05465-5_2.

Clautiaux, F. and I. Ljubić (2025). "Last fifty years of integer linear programming: A focus on recent practical advances." In: *European Journal of Operational Research* 324.3, pp. 707–731. DOI: 10.1016/j.ejor.2024.11.018.

Cornuéjols, G. (2008). "Valid inequalities for mixed integer linear programs." In: *Mathematical Programming* 112, pp. 3–44. DOI: 10.1007/s10107-006-0086-0.

de Jonge, B. and P. A. Scarf (2020). "A review on maintenance optimization." In: *European Journal of Operational Research* 285.3, pp. 805–824. DOI: 10.1016/j.ejor.2019.09.047.

Fischetti, M. and M. Monaci (2009). "Light Robustness." In: *Robust and Online Large-Scale Optimization: Models and Techniques for Transportation Systems*. Ed. by R. K. Ahuja, R. H. Möhring, and C. D. Zaroliagis. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 61–84. DOI: 10.1007/978-3-642-05465-5_3.

Geoffrion, A. M. (1972). "Generalized Benders decomposition." In: *Journal of Optimization Theory and Applications* 10, pp. 237–260. DOI: 10.1007/BF00934810.

Goerigk, M., J. Kurtz, M. Schmidt, and J. Thürauf (2025). "Connections between Robust and Bilevel Optimization." In: *Open Journal of Mathematical Optimization* 6.2, pp. 1–17. DOI: 10.5802/ojmo.38.

Goh, J. and M. Sim (2010). "Distributionally Robust Optimization and Its Tractable Approximations." In: *Operations Research* 58.4-part-1, pp. 902–917. DOI: 10.1287/opre.1090.0795.

Grötschel, M., C. Raack, and A. Werner (2014). "Towards optimizing the deployment of optical access networks." In: *EURO Journal on Computational Optimization* 2.1, pp. 17–53. DOI: 10.1007/s13675-013-0016-x.

Grüne, C. and L. Wulf (2024). *On the Complexity of Recoverable Robust Optimization in the Polynomial Hierarchy*. URL: https://arxiv.org/abs/2411.18590.

Hommelsheim, F., N. Megow, K. Muluk, and B. Peis (2023). *Recoverable Robust Optimization with Commitment*. Tech. rep. URL: https://arxiv.org/abs/2306.08546.

Lee, T. and C. Kwon (2014). "A short note on the robust combinatorial optimization problems with cardinality constrained uncertainty." In: *4OR* 12, pp. 373–378. DOI: 10.1007/s10288-014-0270-7.

Lefebvre, H., M. Schmidt, S. Stevens, and J. Thürauf (2025). *Solving Decision-Dependent Robust Problems as Bilevel Optimization Problems*. URL: https://optimization-online.org/?p=29420.

Liebchen, C., M. Lübbecke, R. Möhring, and S. Stiller (2009). "The Concept of Recoverable Robustness, Linear Programming Recovery, and Railway Applications." In: *Robust and Online Large-Scale Optimization: Models and Techniques for Transportation Systems*. Ed. by R. K. Ahuja, R. H. Möhring, and C. D. Zaroliagis. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 1–27. DOI: 10.1007/978-3-642-05465-5_1.

Nohadani, O. and K. Sharma (2018). "Optimization under Decision-Dependent Uncertainty." In: *SIAM Journal on Optimization* 28.2, pp. 1773–1795. DOI: 10.1137/17M1110560.

Poss, M. (2013). "Robust combinatorial optimization with variable budgeted uncertainty." In: *4OR* 11, pp. 75–92. DOI: 10.1007/s10288-012-0217-9.

Poss, M. (2014). "Robust combinatorial optimization with variable cost uncertainty." In: *European Journal of Operational Research* 237.3, pp. 836–845. DOI: 10.1016/j.ejor.2014.02.060.

Saldanha-da-Gama, F. and S. Wang (2024). *Facility Location Under Uncertainty: Models, Algorithms and Applications*. International Series in Operations Research & Management Science. Springer Cham. DOI: 10.1007/978-3-031-55927-3.

Sim, M. (2004). "Robust Optimization." PhD thesis. Massachusetts Institute of Technology, Sloan School of Management. URL: https://dspace.mit.edu/handle/1721.1/17725.

Soyster, A. L. (1973). "Technical Note—Convex Programming with Set-Inclusive Constraints and Applications to Inexact Linear Programming." In: *Operations Research* 21.5, pp. 1154–1157. DOI: 10.1287/opre.21.5.1154.

van den Akker, J., P. Bouman, J. Hoogeveen, and D. Tönissen (2016). "Decomposition approaches for recoverable robust optimization problems." In: *European Journal of Operational Research* 251.3, pp. 739–750. DOI: 10.1016/j.ejor.2015.12.008.

Vasili, M., T. S. Hong, N. Ismail, and M. Vasili (2011). "Maintenance optimization models: a review and analysis." In: *Proceedings of the 2011 International Conference on Industrial Engineering and Operations Management*. URL: https://ieomsociety.org/ieom2011/pdfs/IEOM173.pdf.

Wiesemann, W., D. Kuhn, and M. Sim (2014). "Distributionally Robust Convex Optimization." In: *Operations Research* 6, pp. 1358–1376. DOI: 10.1287/opre.2014.1314.

Wolsey, L. A. (2020). *Integer Programming*. John Wiley & Sons, Ltd. DOI: 10.1002/9781119606475.

Zeng, B. and L. Zhao (2013). "Solving two-stage robust optimization problems using a column-and-constraint generation method." In: *Operations Research Letters* 41.5, pp. 457–461. DOI: 10.1016/j.orl.2013.05.003.

Appendix A. Proofs of Lemmas 1 and 2

In this section, we provide the omitted proofs of Lemmas 1 and 2, which are used in the proof of Proposition 2.

Proof of Lemma 1. Let $x \in X$ and $u \in \mathcal{U}_\Gamma$ be given arbitrarily. Then, we obtain

$$\begin{aligned}
\gamma(x,u) &= \max_y \left\{ \sum_{i=1}^n (\bar{c}_i x_i + \Delta c_i x_i u_i)\, y_i \colon \sum_{i=1}^n y_i \le k,\; y \in [0,1]^n \right\} \\
&= \min_{\nu, w \ge 0} \left\{ k\nu + \sum_{i=1}^n w_i \colon \nu + w_i \ge \bar{c}_i x_i + \Delta c_i x_i u_i,\; i \in N \right\} \\
&= \min_{\nu \ge 0} \left\{ k\nu + \sum_{i=1}^n \max\{0,\; \bar{c}_i x_i + \Delta c_i x_i u_i - \nu\} \right\} \\
&= -\max_{\nu \ge 0} \left\{ -k\nu - \sum_{i=1}^n \max\{0,\; \bar{c}_i x_i + \Delta c_i x_i u_i - \nu\} \right\}.
\end{aligned}$$

Here, the first equality follows from the total unimodularity of the constraint matrix, whereas the second one is due to strong duality. The third equality follows from optimality arguments. Hence, we obtain

$$\beta(x) = \max_{u \in \mathcal{U}_\Gamma, \nu \geq 0} \left\{ \sum_{i=1}^n \bar{c}_i x_i + \sum_{i=1}^n \Delta c_i x_i u_i - k\nu - \sum_{i=1}^n \max\{0,\, \bar{c}_i x_i + \Delta c_i x_i u_i - \nu\} \right\}.$$

For all $i \in N$, we have

$$\begin{aligned} &\max\{0,\, \bar{c}_i x_i + \Delta c_i x_i u_i - \nu\} \\ =\ &\max\{0,\, \bar{c}_i - \nu\} x_i(1-u_i) + \max\{0,\, \bar{c}_i + \Delta c_i - \nu\} x_i u_i \end{aligned}$$

and we further distinguish the following two cases.

(i) If $c_i(\nu) < 0$, i.e., $i \in N(\nu)$, we obtain

$$c_i(\nu) = \min\{\Delta c_i,\, \nu - \bar{c}_i\} = \nu - \bar{c}_i = \min\{0,\, \nu - \bar{c}_i\}$$

because $\Delta c_i \geq 0$ holds.

(ii) If $c_i(\nu) \geq 0$, i.e., $i \in P(\nu)$, we obtain $\nu \geq \bar{c}_i$ and, thus, $\min\{0,\, \nu - \bar{c}_i\} = 0$.

Taking all previous considerations into account yields

$$\begin{aligned} &\sum_{i=1}^n \bar{c}_i x_i + \sum_{i=1}^n \Delta c_i x_i u_i - k\nu - \sum_{i=1}^n \max\{0,\, \bar{c}_i x_i + \Delta c_i x_i u_i - \nu\} \\ =\ &\sum_{i=1}^n \bar{c}_i x_i + \sum_{i=1}^n \Delta c_i x_i u_i - k\nu + \sum_{i=1}^n \min\{0,\, \nu - \bar{c}_i\} x_i(1-u_i) \\ &+ \sum_{i=1}^n \min\{\Delta c_i,\, \nu - \bar{c}_i\} x_i u_i - \sum_{i=1}^n \Delta c_i x_i u_i \\ =\ &\sum_{i=1}^n \bar{c}_i x_i - k\nu + \sum_{i=1}^n \min\{0,\, \nu - \bar{c}_i\} x_i(1-u_i) + \sum_{i=1}^n c_i(\nu) x_i u_i \\ =\ &\sum_{i=1}^n \bar{c}_i x_i - k\nu + \sum_{i \in N(\nu)} \min\{0,\, \nu - \bar{c}_i\} x_i(1-u_i) \\ &+ \sum_{i \in P(\nu)} \min\{0,\, \nu - \bar{c}_i\} x_i(1-u_i) + \sum_{i=1}^n c_i(\nu) x_i u_i \\ =\ &\sum_{i=1}^n \bar{c}_i x_i - k\nu + \sum_{i \in N(\nu)} c_i(\nu) x_i(1-u_i) + \sum_{i=1}^n c_i(\nu) x_i u_i \\ =\ &\sum_{i=1}^n \bar{c}_i x_i - k\nu + \sum_{i \in N(\nu)} c_i(\nu) x_i + \sum_{i \in P(\nu)} c_i(\nu) x_i u_i, \end{aligned}$$

which concludes the proof. □

Proof of Lemma 2. (i) The claim immediately follows from the definition of the respective sets.

(ii) Let $i \in L(\nu^*)$ be given arbitrarily. By construction, we then have $\Delta c_i > \nu^* - \bar{c}_i \geq \underline{\nu} - \bar{c}_i$, which yields

$$\begin{aligned} c_i(\underline{\nu}) + (\nu^* - \underline{\nu}) &= \min\{\Delta c_i,\, \underline{\nu} - \bar{c}_i\} + (\nu^* - \underline{\nu}) = (\underline{\nu} - \bar{c}_i) + (\nu^* - \underline{\nu}) \\ &= \nu^* - \bar{c}_i = \min\{\Delta c_i,\, \nu^* - \bar{c}_i\} \\ &= c_i(\nu^*). \end{aligned}$$

This proves the first equality in (ii). We now show the second one. To this end, let us first assume that $\bar{\nu} > \bar{c}_i + \Delta c_i$ holds. Because $i \in L(\nu^*)$, we then obtain $\bar{\nu} > \bar{c}_i + \Delta c_i > \nu^*$, which contradicts the definition of $\bar{\nu}$. Hence, $\bar{\nu} \leq \bar{c}_i + \Delta c_i$ has to hold and we obtain

$$\begin{aligned} c_i(\nu^*) &= \min\{\Delta c_i,\, \nu^* - \bar{c}_i\} = \nu^* - \bar{c}_i \\ &= (\bar{\nu} - \bar{c}_i) - (\bar{\nu} - \nu^*) = \min\{\Delta c_i,\, \bar{\nu} - \bar{c}_i\} - (\bar{\nu} - \nu^*) \\ &= c_i(\bar{\nu}) - (\bar{\nu} - \nu^*). \end{aligned}$$

(iii) Let $i \in G(\nu^*)$ be given arbitrarily, i.e., $\nu^* \geq \bar{c}_i + \Delta c_i$. By construction, we have $\underline{\nu} \geq \bar{c}_i + \Delta c_i$ as well as

$$c_i(\underline{\nu}) = \min\{\Delta c_i,\, \underline{\nu} - \bar{c}_i\} = \Delta c_i = \min\{\Delta c_i,\, \nu^* - \bar{c}_i\} = c_i(\nu^*).$$

Moreover, we have $\bar{\nu} \geq \nu^* \geq \bar{c}_i + \Delta c_i$, which yields

$$c_i(\bar{\nu}) = \min\{\Delta c_i,\, \bar{\nu} - \bar{c}_i\} = \Delta c_i = \min\{\Delta c_i,\, \nu^* - \bar{c}_i\} = c_i(\nu^*).$$

(iv) From $i \in N(\nu^*) \cap P(\bar{\nu})$, we obtain $\bar{\nu} \geq \bar{c}_i > \nu^*$ and, thus, $\bar{\nu} = \bar{c}_i$ has to hold by the definition of $\bar{\nu}$. As a result, we have $c_i(\bar{\nu}) = \min\{\Delta c_i,\, \bar{\nu} - \bar{c}_i\} = 0$ because of $\Delta c_i \geq 0$.

(v) We prove the claim by contradiction. Hence, suppose that $\nu^* \notin \mathcal{V}$ holds and that there exists $i \in P(\nu^*) \cap N(\underline{\nu})$. From $i \in P(\nu^*)$, $\nu^* \notin \mathcal{V}$, and $\bar{c}_i \in \mathcal{V}$, we obtain $\nu^* > \bar{c}_i$. Because $i \in N(\underline{\nu})$, we further have $\bar{c}_i > \underline{\nu}$. Overall, this yields $\nu^* > \bar{c}_i > \underline{\nu}$, which is a contradiction to $\underline{\nu} = \max\{\nu \colon \nu \in \mathcal{V},\, \nu \leq \nu^*\} \geq \bar{c}_i$.

□

Appendix B. Proof of Theorem 2

By Proposition 2, Problem (2) can be solved as

$$\begin{aligned} \min_{x,\eta} \quad & f^\top x + \eta \\ \text{s.t.} \quad & \eta \geq \alpha(x,\nu) - k\nu, \quad \nu \in \mathcal{V}, \\ & x \in X,\ \eta \in \mathbb{R}_{\geq 0}. \end{aligned}$$

Let now $x \in X$ and $\nu \in \mathcal{V}$ be given arbitrarily. Because $\alpha(x,\nu)$ corresponds to a classic discrete budgeted uncertainty model, we obtain

$$\begin{aligned} \alpha(x,\nu) &= \max_u \left\{ \sum_{i=1}^n \bar{c}_i(\nu) x_i + \sum_{i=1}^n \Delta c_i(\nu) x_i u_i \colon \sum_{i=1}^n u_i \leq \Gamma,\ u \in [0,1]^n \right\} \\ &= \sum_{i=1}^n \bar{c}_i(\nu) x_i + \min_{w^\nu, z^\nu \geq 0} \left\{ \Gamma w^\nu + \sum_{i=1}^n z_i^\nu \colon w^\nu + z_i^\nu \geq \Delta c_i(\nu) x_i,\ i \in N \right\}. \end{aligned}$$

Here, the first equality follows from the total unimodularity of the constraint matrix (cf. Proposition 3.3 in Wolsey (2020)), whereas the second one is due to strong duality. Overall, we thus obtain

$$\eta \geq \sum_{i=1}^n \bar{c}_i(\nu) x_i + \min_{w^\nu, z^\nu \geq 0} \left\{ \Gamma w^\nu + \sum_{i=1}^n z_i^\nu \colon w^\nu + z_i^\nu \geq \Delta c_i(\nu) x_i,\ i \in N \right\} - k\nu.$$

If, for fixed $x \in X$ and $\nu \in \mathcal{V}$, there is a pair (w^ν, z^ν) that is feasible for the problem

$$\min_{w^\nu, z^\nu \geq 0} \quad \Gamma w^\nu + \sum_{i=1}^n z_i^\nu \quad \text{s.t.} \quad w^\nu + z_i^\nu \geq \Delta c_i(\nu) x_i, \quad i \in N, \tag{17}$$

and that satisfies the inequality

$$\eta \geq \sum_{i=1}^n \bar{c}_i(\nu) x_i + \Gamma w^\nu + \sum_{i=1}^n z_i^\nu - k\nu,$$

the latter is particularly satisfied for an optimal solution to Problem (17). This concludes the proof. □

Appendix C. Proof of Proposition 3

By Theorem 2, Problem (2) can be solved as

$$\begin{aligned}\min_{x,\eta} \quad & f^\top x + \eta \\ \text{s.t.} \quad & \eta \geq \sum_{i=1}^n \bar{c}_i(\nu)x_i + \varphi(x,\nu) - k\nu, \quad \nu \in \mathcal{V}, \\ & x \in X,\, \eta \in \mathbb{R}_{\geq 0},\end{aligned}$$

where, for fixed $x \in X$ and $\nu \in \mathcal{V}$, we have

$$\varphi(x,\nu) := \min_{w^\nu, z^\nu \geq 0} \left\{ \Gamma w^\nu + \sum_{i=1}^n z_i^\nu \colon w^\nu + z_i^\nu \geq \Delta c_i(\nu) x_i,\, i \in N \right\}.$$

Let now $x \in X$ and $\nu \in \mathcal{V}$ be given arbitrarily. By the definition of the perturbation function σ_ν, applying the same steps as in the proof of Theorem 3 in Bertsimas and Sim (2003) and the proofs of the results in Lee and Kwon (2014) then yields

$$\varphi(x,\nu) = \min_{\ell \in \mathcal{L}} \left\{ \Gamma \Delta c_{\sigma_\nu(\ell)}(\nu) + \sum_{i=1}^{\ell} \left(\Delta c_{\sigma_\nu(i)}(\nu) - \Delta c_{\sigma_\nu(\ell)}(\nu) \right) x_{\sigma_\nu(i)} \right\}.$$

Hence, for all $\nu \in \mathcal{V}$, we obtain

$$\begin{aligned}\eta &\geq \sum_{i=1}^n \bar{c}_i(\nu)x_i + \varphi(x,\nu) - k\nu \\ &= \min_{\ell \in \mathcal{L}} \left\{ \Gamma \Delta c_{\sigma_\nu(\ell)}(\nu) - k\nu + \sum_{i=1}^n \bar{c}_i(\nu) x_i + \sum_{i=1}^{\ell} \left(\Delta c_{\sigma_\nu(i)}(\nu) - \Delta c_{\sigma_\nu(\ell)}(\nu) \right) x_{\sigma_\nu(i)} \right\} \\ &= \min_{\ell \in \mathcal{L}} \left\{ \kappa_\ell(\nu) + \sum_{i=1}^n \tilde{c}^{\,\ell}_{\sigma_\nu(i)}(\nu) x_{\sigma_\nu(i)} \right\}.\end{aligned}$$

□

Appendix D. Proof of Proposition 4

Per definition, we have $c_i^s = \bar{c}_i + \Delta c_i u_i^s$ for all $i \in N$ and $s \in S$ with $u^s \in \mathcal{U}_\Gamma$. Hence, Problem (11) can equivalently be stated as the max-min problem

$$\max_{z \in Z} \min_{y \in Y(x)} \quad \sum_{i=1}^n \left(\bar{c}_i + \Delta c_i z_i \right) y_i$$

with

$$Z := \left\{ z \in \{0,1\}^n : \sum_{i=1}^n z_i \leq \Gamma \right\}$$

and

$$Y(x) := \left\{ y \in \{0,1\}^n : \sum_{i=1}^n y_i \geq \sum_{i=1}^n x_i - k,\, x \geq y \right\}.$$

Applying Proposition 3.3 in Wolsey (2020), we can relax the integrality restrictions $y \in \{0,1\}^n$, i.e., we consider

$$\min_{y \in [0,1]^n} \quad \sum_{i=1}^n \left(\bar{c}_i + \Delta c_i z_i \right) y_i \quad \text{s.t.} \quad \sum_{i=1}^n y_i \geq \sum_{i=1}^n x_i - k,\, x \geq y \tag{18}$$

as the inner minimization problem. Problem (18) is a continuous linear optimization problem for fixed $x \in X \subseteq \{0,1\}^n$ and the constraints $y_i \leq 1$ are redundant because of $x \geq y$. Hence, the dual of Problem (18) is given by

$$\begin{aligned} \max_{\lambda,\mu} \quad & \left(\sum_{i=1}^{n} x_i - k\right)\lambda - \sum_{i=1}^{n} x_i\mu_i \\ \text{s.t.} \quad & \lambda - \mu_i \leq \bar{c}_i + \Delta c_i z_i, \quad i \in N, \\ & \mu \in \mathbb{R}^n_{\geq 0},\ \lambda \in \mathbb{R}_{\geq 0}. \end{aligned}$$

Finally, the claim follows by applying strong duality. □

Appendix E. Supplementary Tables

Table 5. The number of instances solved to global optimality ("solved"; out of 240 and 222 instances for AP and SSCFLP, respectively), the number of instances for which a feasible point with finite but non-zero gap is found ("open gap"), and the number of instances with infinite gap ("infinite gap") for the branch-and-cut approach applied to the projection-based formulation. For those instances with finite but non-zero gap, also the average gap ("average gap"; in %) is shown.

problem	cut strategy	solved	open gap	average gap	infinite gap
AP	All-In	178	61	6.19	1
	First-In	218	21	3.04	1
	Shuffle-First-In	210	30	5.19	0
	Most-Violated	211	29	6.36	0
SSCFLP	All-In	109	75	6.31	38
	First-In	194	7	3.90	21
	Shuffle-First-In	196	8	1.87	18
	Most-Violated	187	5	16.93	30

Table 6. The number of instances solved to global optimality ("solved"; out of 240 and 222 instances for AP and SSCFLP, respectively), the number of instances for which a feasible point with finite but non-zero gap is found ("open gap"), and the number of instances with infinite gap ("infinite gap") for the approaches applied to the extended formulation. For those instances with finite but non-zero gap, also the average gap ("average gap"; in %) is shown.

problem	approach	solved	open gap	average gap	infinite gap
AP	MILP	240	0	–	0
	BnC	232	8	1.17	0
	CCG	240	0	–	0
SSCFLP	MILP	139	9	1.17	74
	BnC	165	4	0.84	53
	CCG	195	27	14.35	0

Table 7. The number of instances solved to global optimality ("solved"; out of 240 and 222 instances for AP and SSCFLP, respectively), the number of instances for which a feasible point with finite but non-zero gap is found ("open gap"), and the number of instances with infinite gap ("infinite gap") for the approaches applied to the compact formulation. For those instances with finite but non-zero gap, also the average gap ("average gap"; in %) is shown.

problem	approach	solved	open gap	average gap	infinite gap
AP	MILP	58	2	11.10	180
	BnC	181	32	21.62	27
	CCG	238	0	–	2
SSCFLP	MILP	0	0	–	222
	BnC	100	72	10.86	50
	CCG	0	0	–	222

(Y. Beck) Eindhoven University of Technology, Department of Industrial Engineering and Innovation Sciences, PO Box 513, 5600 MB Eindhoven, the Netherlands
Email address: y.beck@tue.nl

(I. Ljubić) ESSEC Business School, Department of Information Systems, Data Analytics and Operations, 3 Avenue Bernard Hirsch, 95021 Cergy-Pontoise Cedex, France
Email address: ljubic@essec.edu

(C. Büsing) Teaching and Research Area Combinatorial Optimization, RWTH Aachen University, Ahornstrasse 55, 52074 Aachen, Germany
Email address: buesing@combi.rwth-aachen.de