Mathematical Programming (2025) 210:97–146
https://doi.org/10.1007/s10107-024-02135-7

FULL LENGTH PAPER

Series B

Recycling valid inequalities for robust combinatorial optimization with budgeted uncertainty

Christina Büsing[1] · **Timo Gersing**[1] · **Arie M. C. A. Koster**[2]

Received: 29 June 2023 / Accepted: 30 July 2024 / Published online: 29 August 2024

Abstract

Robust combinatorial optimization with budgeted uncertainty is one of the most popular approaches for integrating uncertainty into optimization problems. The existence of a compact reformulation for (mixed-integer) linear programs and positive complexity results give the impression that these problems are relatively easy to solve. However, the practical performance of the reformulation is quite poor when solving robust integer problems, in particular due to its weak linear relaxation. To overcome this issue, we propose procedures to derive new classes of valid inequalities for robust combinatorial optimization problems. For this, we recycle valid inequalities of the underlying deterministic problem such that the additional variables from the robust formulation are incorporated. The valid inequalities to be recycled may either be readily available model constraints or actual cutting planes, where we can benefit from decades of research on valid inequalities for classical optimization problems. We first demonstrate the strength of the inequalities theoretically, by proving that recycling yields a facet-defining inequality in many cases, even if the original valid inequality was not facet-defining. Afterwards, we show in an extensive computational study that using recycled inequalities can lead to a significant improvement of the computation time when solving robust optimization problems.

Keywords Robust optimization · Combinatorial optimization · Integer programming · Polyhedral combinatorics · Computation

Mathematics Subject Classification 90C11 · 90C17 · 90C27 · 90C57

✉ Timo Gersing
gersing@combi.rwth-aachen.de

Christina Büsing
buesing@combi.rwth-aachen.de

Arie M. C. A. Koster
koster@math2.rwth-aachen.de

[1] Combinatorial Optimization, RWTH Aachen University, Ahornstraße 55, 52074 Aachen, Germany

[2] Discrete Optimization, RWTH Aachen University, Pontdriesch 10-12, 52062 Aachen, Germany

1 Introduction

Uncertain parameters are a common issue in decision-making problems and should be treated with caution, as a seemingly optimal decision may reveal itself to be impractical once it is implemented in the real world [1]. A popular approach for dealing with uncertainties is robust optimization, in which one aims to optimize against the worst-case of a set of scenarios. Robust optimization was proposed first by Soyster [2] in the early 1970s. Kouvelis and Yu [3] considered it for combinatorial optimization problems and discrete uncertainty sets in the 1990s. The concept was then further analyzed by Ben-Tal and Nemirovski [1, 4, 5] as well as Bertsimas and Sim [6, 7] at the beginning of this century. An overview on the topic can be found in [8–10].

The concept of budgeted uncertainty by Bertsimas and Sim has received particular attention. This is illustrated by the fact that their seminal paper [7] is the most cited document in the literature databases Scopus and Web of Science containing "robust optimization" in its title, keywords or abstract. This popularity is not least due to the scenarios being constructed in an intuitive way, where an *uncertainty budget* is used to control the extent to which one wants to hedge against uncertainties. Bertsimas and Sim show theoretically and experimentally that the "price of robustness" is rather small, as the uncertainty budget can be chosen such that a high level of probabilistic protection is achieved, while the loss in the objective value compared to non-robust solutions is rather small [7]. In addition, the possibility to formulate the robust counterpart of (mixed-integer) linear optimization problems again as (mixed-integer) linear problems, and the existence of positive theoretical complexity results suggest that robust optimization problems with budgeted uncertainty are easily solvable. However, despite their popularity and the amount of research devoted to solving these kinds of robust problems, instances of practical size often still pose a considerable challenge for MILP solvers [11]. To help overcoming this challenge, we propose new classes of valid inequalities for robust combinatorial optimization problems that are easy to compute and often lead to a significant reduction of the computation time.

We first define a standard, so called *nominal,* combinatorial problem without uncertainties

$$\text{(NOM)} \qquad \begin{aligned} \min \ & \sum_{i\in[n]} c_i x_i \\ \text{s.t. } & Ax \le b, x \in \{0,1\}^n , \end{aligned}$$

where $c \in \mathbb{R}^n$ is an objective vector, $A \in \mathbb{R}^{m\times n}$ a constraint matrix with a right-hand side $b \in \mathbb{R}^m$ and $[n] = \{1, \ldots, n\}$. We now replace the objective coefficients c_i with uncertain coefficients c_i' from an interval $\left[c_i, c_i + \hat{c}_i\right]$ and say that c_i' can *deviate* from its *nominal value* c_i by up to the *deviation* $\hat{c}_i \in \mathbb{R}_{\ge 0}$. In the following, we assume that at least one deviation $\hat{c}_i$ is strictly positive, as otherwise all c_i' are fixed to their nominal values c_i. We call a vector of possible objective coefficients $\left\{c' \in \mathbb{R}^n \middle| c_i' \in \left[c_i, c_i + \hat{c}_i\right] \ \forall i \in [n]\right\}$ a *scenario*. Note that for any feasible solution x, the objective value is worst if all coefficients c_i' are equal to their maximum value $c_i + \hat{c}_i$. In practice, however, this extreme scenario is usually very unlikely such that

it would be overly conservative to assume that it actually occurs. To adjust the level of conservatism, Bertsimas and Sim [7] propose a robust counterpart to NOM, in which they restrict the set of considered uncertain scenarios by defining an *uncertainty budget* $\Gamma \in [0, n]$. Given such a budget, we only consider these scenarios in which at most $\lfloor \Gamma \rfloor$ coefficients c_i' deviate to $c_i + \hat{c}_i$ and one coefficient deviates to $c_i + (\Gamma - \lfloor \Gamma \rfloor)\, \hat{c}_i$. This robust counterpart can be stated as

$$\begin{aligned} \min \sum_{i\in[n]} c_i x_i + \max_{\substack{S\cup\{t\}\subseteq[n]:\\ |S|\leq\lfloor\Gamma\rfloor, t\notin S}} \left((\Gamma - \lfloor\Gamma\rfloor)\, \hat{c}_t x_t + \sum_{i\in S} \hat{c}_i x_i \right) \\ \text{s.t. } Ax \leq b, x \in \{0,1\}^n . \end{aligned}$$

Bertsimas and Sim [7] also show that we can resolve the non-linearity of the above problem. For this, we write the inner maximization problem as

$$\begin{aligned} \max \sum_{i\in[n]} \hat{c}_i x_i \delta_i \\ \text{s.t. } \sum_{i\in[n]} \delta_i \leq \Gamma \\ \delta \in [0,1]^n . \end{aligned}$$

Note that this is a linear program when considering a fixed $x \in \{0,1\}^n$. By dualizing this problem, we obtain a linear minimization problem, which we can use to substitute the inner maximization problem above. This results in the compact robust problem

$$\begin{aligned} \min \Gamma z + \sum_{i\in[n]} (c_i x_i + p_i) \\ \text{s.t. } (x, p, z) \in \mathcal{P}^{\text{ROB}}, x \in \{0,1\}^n \end{aligned} \tag{ROB}$$

with

$$\mathcal{P}^{\text{ROB}} = \left\{ (x, p, z) \;\middle|\; \begin{array}{ll} Ax \leq b & \\ p_i + z \geq \hat{c}_i x_i & \forall i \in [n] \\ x \in [0,1]^n,\ p \in \mathbb{R}^n_{\geq 0},\ z \in \mathbb{R}_{\geq 0} & \end{array} \right\}.$$

Unfortunately, the formulation $\mathcal{P}^{\text{ROB}}$ is quite weak, potentially leading to much higher computation times for solving ROB compared to NOM. In fact, the integrality gap of the formulation $\mathcal{P}^{\text{ROB}}$, that is the relative difference between the values of an optimal fractional solution and an optimal integer-feasible solution, may be arbitrarily large, even if the integrality gap of the corresponding nominal problem is zero. This is shown in the following example from [11].

Example 1 Consider the trivial problem of selecting the cheapest of n elements

$$\min \sum_{i\in[n]} c_i x_i$$

$$\text{s.t.} \sum_{i\in[n]} x_i = 1, x \in \{0,1\}^n .$$

The integrality gap of the above problem is zero for all $c \in \mathbb{R}^n$. However, if we consider an instance of the uncertain counterpart ROB with $c \equiv 0$, $\hat{c} \equiv 1$, and $\Gamma = 1$

$$\begin{aligned}
\min\ & z + \sum_{i\in[n]} p_i \\
\text{s.t.}\ & \sum_{i\in[n]} x_i = 1 \\
& p_i + z \geq x_i && \forall i \in [n] \\
& x \in \{0,1\}^n,\ p \in \mathbb{R}^n_{\geq 0}, z \in \mathbb{R}_{\geq 0},
\end{aligned}$$

then $(x, p, z) = \left(\frac{1}{n}, \ldots, \frac{1}{n}, 0, \ldots, 0, \frac{1}{n}\right)$ is the unique optimal fractional solution of value $\frac{1}{n}$, while the objective value of an optimal integer solution is 1. Hence, the integrality gap is $\frac{1-1/n}{1/n} = n - 1$, and thus grows arbitrarily large as $n \to \infty$.

The above example shows that optimal continuous solutions for ROB tend to be highly fractional, as small values of x_i allow for covering all right-hand sides $\hat{c}_i x_i$ in the constraints $p_i + z \geq \hat{c}_i x_i$ with a small value of z, while choosing $p_i = 0$. On the one hand, such solutions are exactly what we aim for when striving for robustness, as we distribute the risk as much as possible. On the other hand, highly fractional optimal solutions for the linear relaxation imply the need for much branching, and thus a high computational effort when solving ROB.

In the literature, several alternative approaches to solve ROB have been developed and evaluated. Bertsimas et al. [12] as well as Fischetti and Monaci [13] test the practical performance of the compact reformulation $\mathcal{P}^{\text{ROB}}$ compared to a separation approach using an alternative formulation with exponentially many inequalities, each one modeling an extreme scenario from the uncertainty set. Unfortunately, the alternative formulation is, despite its size, as weak as $\mathcal{P}^{\text{ROB}}$ and performs worse for robust integer problems (but better for continuous problems). Joung and Park [14] propose cuts that dominate the classic scenario inequalities and can be separated by considering the robustness term as a submodular function and greedily solving a maximization problem over the corresponding polymatroid. Atamtürk [15] addresses the issue by proposing four different problem-independent strong formulations. The strongest of these is theoretically as strong as possible, as it preserves the integrality gap of the nominal problem. However, the four formulations are very large and are thus computationally outperformed by the standard formulation $\mathcal{P}^{\text{ROB}}$, as shown in [11].

A famous approach to completely avoid the issues arising from the weak formulation $\mathcal{P}^{\text{ROB}}$ is to solve ROB via resorting to its nominal counterpart. Bertsimas and Sim [6] show that there always exists an optimal solution (x, p, z) to ROB such that $z \in \{0, \hat{c}_1, \ldots, \hat{c}_n\}$. Furthermore, solving ROB for a fixed z is equivalent to solving its nominal counterpart with different objective values. Hence, ROB can be solved by solving $\left|\{0, \hat{c}_1, \ldots, \hat{c}_n\}\right| \leq n + 1$ nominal problems. Álvarez-Miranda et al. [16] as well as Park and Lee [17] showed independently that it is sufficient to solve only

$n + 2 - \Gamma$, or $n + 1 - \Gamma$ nominal problems respectively. Lee and Kwon [18] even improved these results later, showing that solving $\lceil \frac{n-\Gamma}{2} \rceil + 1$ nominal problems already suffices. Still, the computational effort of this approach is too high if n is large. To address this, Hansknecht et al. [19] propose a divide and conquer approach, in which one also solves nominal problems, but reduces the computational effort by pruning many non-optimal values for z using bounds on the respective optimal objective value. In [11], we proposed a branch and bound approach, in which non-optimal values for z are pruned even more efficiently. The branch and bound makes use of structural insights and strong linearizations derived from the following bilinear formulation

$$\mathcal{P}^{\text{BIL}} = \left\{ (x, p, z) \middle| \begin{array}{ll} Ax \leq b & \\ p_i + x_i z \geq \hat{c}_i x_i & \forall i \in [n] \\ x \in [0,1]^n,\ p \in \mathbb{R}^n_{\geq 0},\ z \in \mathbb{R}_{\geq 0} & \end{array} \right\}.$$

This bilinear formulation strengthens the original robustness constraints $p_i + z \geq \hat{c}_i x_i$ by multiplying z with x_i. This is valid, as the bilinear inequality is equivalent to $p_i + z \geq \hat{c}_i x_i$ for $x_i = 1$ and $p_i \geq 0$ for $x_i = 0$. While the bilinearity is rather hindering for practical purposes, as non-linear problems are in practice harder to solve, $\mathcal{P}^{\text{BIL}}$ is theoretically very strong. In fact, there exists no polyhedral formulation $\mathcal{P}$ for ROB with $\mathcal{P} \subsetneq \mathcal{P}^{\text{BIL}}$ [11].

Contribution

In this paper, we use the bilinear formulation $\mathcal{P}^{\text{BIL}}$ as a foundation for the new class of *recycled inequalities*. To obtain these, we combine the strength of the bilinear inequalities with the structural properties provided by valid inequalities for the nominal problem NOM. By doing so, we can use valid inequalities for NOM a second time to improve the formulation $\mathcal{P}^{\text{ROB}}$.

In its simplest and most effective version, our recycling relies on the underlying valid inequality to be a knapsack inequality. We will show that in this case the corresponding recycled inequality often defines a facet of the convex hull of integer-feasible solutions

$$\mathcal{C}^{\text{ROB}} = \text{conv}\left(\left\{ (x, p, z) \in \mathcal{P}^{\text{ROB}} \middle| x \in \{0,1\}^n \right\} \right).$$

Additionally, we show how to efficiently separate such recycled inequalities in a branch and cut algorithm. We also discuss how to recycle inequalities that are not of the knapsack type, to make use of a wider range of valid inequalities.

In an extensive computational study on robust versions of both classical combinatorial problems and real-world instances from MIPLIB 2017 [20], we verify that recycled inequalities can substantially strengthen the formulation $\mathcal{P}^{\text{ROB}}$, which is expressed by drastic reductions of the integrality gap. Together with the efficient separation of recycled inequalities, this leads to a significant improvement of solving times.

All implemented algorithms and generated test instances are published, together with a package of algorithms for solving robust combinatorial optimization problems [21] and benchmark instances [22].

Note that this is an extended version of a paper that appeared in the proceedings of IPCO 2023 [23].

Outline
In Sect. 2, we show how to derive the new class of recycled inequalities from valid knapsack inequalities. In Sect. 3, we characterize inequalities for which the corresponding recycled inequality is facet-defining. In Sect. 4, we discuss different efficient approaches of separating recycled inequalities. In Sect. 5, we show how to recycle non-knapsack inequalities. In Sect. 6, we conduct our computational study.

2 Recycling valid inequalities

As already mentioned, the bilinear inequalities $p_i + x_i z \geq \hat{c}_i x_i$ play a crucial role for our recycled inequalities. To understand their strength intuitively, we recall our observations from Example 1. There, we noticed that choosing fractional values for x_i is tempting, as we are then able to meet the inequalities $p_i + z \geq \hat{c}_i x_i$ with a small value of z and $p_i = 0$. However, this advantage vanishes for the bilinear inequalities $p_i + x_i z \geq \hat{c}_i x_i$, as we always have $z \geq \hat{c}_i$ for $x_i \neq 0$ and $p_i = 0$. To make use of this in practice, it would be beneficial to carry over the strength of the bilinear inequalities to a linear formulation.

Multiplying linear inequalities with variables as an intermediate step in order to achieve a stronger linear formulation is not a new approach. For the Reformulation–Linearization-Technique by Sherali and Adams [24], one multiplies constraints with variables and linearizes the resulting products afterwards via substitution with auxiliary variables. When taken to the extreme, where all constraints are multiplied with all possible combinations of variables, one obtains a formulation with exponentially many variables and constraints, whose projection onto the space of original variables equals $\mathcal{C}^{\mathrm{ROB}}$. Our approach is different in the sense that we don't directly linearize the bilinear inequalities, and thus don't create auxiliary variables. Instead, we combine several of the bilinear inequalities in order to estimate the non-linear terms against a linear term, using a valid inequality for the corresponding nominal problem. From now on, let

$$\mathcal{C}^{\mathrm{NOM}} = \mathrm{conv}\left(\left\{x \in \{0,1\}^n \,\middle|\, Ax \leq b\right\}\right)$$

be the convex hull of all integer nominal solutions. Then we combine the bilinear inequalities and valid inequalities for $\mathcal{C}^{\mathrm{NOM}}$ as follows.

Theorem 1 *Let $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ be a valid inequality for $\mathcal{C}^{NOM}$ with $\pi \in \mathbb{R}^{n+1}_{\geq 0}$. Then the inequality*

$$\pi_0 z + \sum_{i\in[n]} \pi_i p_i \geq \sum_{i\in[n]} \pi_i \hat{c}_i x_i \tag{1}$$

is valid for $\mathcal{C}^{ROB}$.

Proof Summing the bilinear constraints $p_i + x_i z \geq \hat{c}_i x_i$, each with a weight of π_i, we obtain

$$\sum_{i\in[n]} \pi_i p_i + \sum_{i\in[n]} \pi_i x_i z \geq \sum_{i\in[n]} \pi_i \hat{c}_i x_i,$$

which is a valid inequality for $\mathcal{C}^{\mathrm{ROB}}$ due to $\pi \geq 0$. Now, since $z \geq 0$ holds, we have $\sum_{i\in[n]} \pi_i x_i z \leq \pi_0 z$, and thus the validity of (1). □

Hence, we can reuse the valid inequality $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ to strengthen the formulation $\mathcal{P}^{\text{ROB}}$ by adding the corresponding inequality (1). This motivates the following definition.

Definition 1 We call $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ *recyclable* if it is valid for $\mathcal{C}^{\text{NOM}}$ and $\pi \geq 0$. We call $\pi_0 z + \sum_{i\in[n]} \pi_i p_i \geq \sum_{i\in[n]} \pi_i \hat{c}_i x_i$ the corresponding *recycled inequality*.

In the following, we will only consider recyclable inequalities $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ consisting exclusively of variables with uncertain objective coefficients, i.e., $\pi_i = 0$ for all $i \in [n]$ with $\hat{c}_i = 0$.

Definition 2 We call an inequality $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ and its corresponding coefficients π *uncertainty-exclusive* if $\pi_i = 0$ holds for all $i \in [n]$ with $\hat{c}_i = 0$.

Note that uncertainty-exclusive inequalities are the only interesting ones for recycling, because we can always recycle $\sum_{i\in[n]\setminus\{j\}} \pi_i x_i \leq \pi_0$ when $\hat{c}_j = 0$ holds. While this inequality is weaker than $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ for the nominal problem, the corresponding recycled inequality is stronger. This is because we remove $\pi_i p_i$ from the left-hand side while the right-hand side does not change due to $\pi_i \hat{c}_i x_i = 0$.

By focusing on uncertainty-exclusive inequalities, we obtain the following statement.

Proposition 1 *Let $\pi \in \mathbb{R}^{n+1}$ be uncertainty-exclusive. If $\pi_0 z + \sum_{i\in[n]} \pi_i p_i \geq \sum_{i\in[n]} \pi_i \hat{c}_i x_i$ is valid for $\mathcal{C}^{ROB}$, then $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ is a recyclable inequality.*

Proof First, note that the validity of $\pi_0 z + \sum_{i\in[n]} \pi_i p_i \geq \sum_{i\in[n]} \pi_i \hat{c}_i x_i$ already implies $\pi \geq 0$, since p and z are unbounded, while the right-hand side $\sum_{i\in[n]} \pi_i \hat{c}_i x_i$ is not. Second, note that $\hat{c}_i x_i = 0$ implies $\pi_i x_i = 0$ for all $i \in [n]$, since we have $x_i = 0$ or we have $\hat{c}_i = 0$ and thus $\pi_i = 0$ due to the uncertainty-exclusiveness. Now, assume that $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ is not valid for $\mathcal{C}^{\text{NOM}}$, i.e., there exists a vector $\tilde{x} \in \mathcal{C}^{\text{NOM}}$ with $\sum_{i\in[n]} \pi_i \tilde{x}_i > \pi_0$. Then there exists an index $i \in [n]$ with $\hat{c}_i \tilde{x}_i > 0$, as otherwise $\pi_i \tilde{x}_i = 0$ for all $i \in [n]$ and therefore $\pi_0 < \sum_{i\in[n]} \pi_i \tilde{x}_i = 0$. We define $(\tilde{x}, \tilde{p}, \tilde{z}) \in \mathcal{C}^{\text{ROB}}$ with $\tilde{z} = \min\left\{\hat{c}_i \middle| i \in [n], \hat{c}_i \tilde{x}_i > 0\right\}$ as well as $\tilde{p}_i = \left(\hat{c}_i - \tilde{z}\right)^+ \tilde{x}_i$ for all $i \in [n]$. Note that we write $(y)^+ = \max\{y, 0\}$ for arbitrary $y \in \mathbb{R}$. For $i \in [n]$, we either have $\hat{c}_i \tilde{x}_i = 0$, implying $\pi_i \tilde{x}_i = 0$, or we have $\hat{c}_i \tilde{x}_i > 0$, implying $\tilde{z} \leq \hat{c}_i$ by the choice of $\tilde{z}$ and thus $\left(\hat{c}_i - \tilde{z}\right)^+ = \hat{c}_i - \tilde{z}$. This yields $\pi_i \tilde{p}_i = \pi_i \left(\hat{c}_i - \tilde{z}\right)^+ \tilde{x}_i = \pi_i \left(\hat{c}_i - \tilde{z}\right) \tilde{x}_i$ for all $i \in [n]$. Together with $\tilde{z} > 0$, it follows

$$\pi_0 \tilde{z} + \sum_{i\in[n]} \pi_i \tilde{p}_i = \tilde{z} \left(\pi_0 - \sum_{i\in[n]} \pi_i \tilde{x}_i \right) + \sum_{i\in[n]} \pi_i \hat{c}_i \tilde{x}_i < \sum_{i\in[n]} \pi_i \hat{c}_i \tilde{x}_i,$$

which proves that $\pi_0 z + \sum_{i\in[n]} \pi_i p_i \geq \sum_{i\in[n]} \pi_i \hat{c}_i x_i$ cannot be valid for $\mathcal{C}^{\text{ROB}}$. □

The proposition above gives a first indication of the strength of the recycling approach: We can obtain all non-dominated valid inequalities for $\mathcal{C}^{\text{ROB}}$ of the form $\pi_0 z + \sum_{i\in[n]} \pi_i p_i \geq \sum_{i\in[n]} \pi_i \hat{c}_i x_i$ by recycling a valid nominal inequality. Hence, if there exists an inequality of the form (1) that dominates a recycled inequality, then

there necessarily exists a nominal recyclable inequality that leads to this stronger inequality through recycling.

In order to see in which cases recycled inequalities are particularly effective, let us consider how they compare to the bilinear inequalities over the course of their construction. First, note that the sum of the bilinear inequalities is weaker than the bilinear inequalities themselves. Hence, when separating a recycled inequality to cut off a fractional solution $(\tilde{x}, \tilde{p}, \tilde{z}) \in \mathcal{P}^{\mathrm{NOM}}$, then the inequality to be recycled ideally only has $\pi_i > 0$ for indices $i \in [n]$ for which the bilinear inequality $\tilde{p}_i + \hat{c}_i \tilde{x}_i \geq \hat{c}_i \tilde{x}_i$ is violated or tight. A second potential weakening occurs when applying the estimation $\sum_{i\in[n]} \pi_i x_i z \leq \pi_0 z$. This implies that recycling $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ is especially interesting if it is binding for $(\tilde{x}, \tilde{p}, \tilde{z})$.

Reconsider Example 1, for which we can recycle the valid inequality $\sum_{i\in[n]} x_i \leq 1$ implied by $\sum_{i\in[n]} x_i = 1$. The recycled inequality $z + \sum_{i\in[n]} p_i \geq \sum_{i\in[n]} x_i$ yields $z + \sum_{i\in[n]} p_i \geq 1$, and thus the optimal objective value of the linear relaxation is now equal to the optimal integer objective value. This intuitively highlights the strength of the recycled inequalities in the case where both properties, a binding recyclable valid inequality and the violation of bilinear inequalities corresponding to indices $i \in [n]$ with $\pi_i > 0$, coincide.

2.1 Recycling inequalities for uncertain constraints

Recycling can also be applied when considering uncertain constraints with budgeted uncertainty sets. In this setting, the j-th row $\sum_{i\in[n]} a_{ji} x_i \leq b_j$ of the constraint matrix $Ax \leq b$ becomes $\Gamma_j z_j + \sum_{i\in[n]} \left(a_{ji} x_i + p_{ji}\right) \leq b_j$ with additional robustness variables $z_j \in \mathbb{R}_{\geq 0}$ and $p_j \in \mathbb{R}^n_{\geq 0}$ as well as robustness constraints $z_j + p_{ji} \geq \hat{a}_{ji} x_i$ for $i \in [n]$ when considering deviations $\hat{a}_j \in \mathbb{R}^n$ and a constraint-specific uncertainty budget $\Gamma_j \in \mathbb{R}_{\geq 0}$ [7]. Since the structure obtained for uncertain constraints is analogous to that obtained for uncertain objective functions, we can apply the same ideas as above. Let $\mathcal{C}^{\mathrm{ROB}}$ be the convex hull of the set of feasible solutions for the robust problem. Then we obtain the following result.

Theorem 2 *Consider an uncertain constraint with deviations $\hat{a}_j \in \mathbb{R}^n$. Let $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ be a valid inequality for $\mathcal{C}^{ROB}$ with $\pi \in \mathbb{R}^{n+1}_{\geq 0}$. Then the inequality*

$$\pi_0 z_j + \sum_{i\in[n]} \pi_i p_{ji} \geq \sum_{i\in[n]} \pi_i \hat{a}_{ji} x_i$$

is also valid for $\mathcal{C}^{ROB}$.

The proof is analogous to that of Theorem 1. Note that the recyclable inequality only has to be valid for $\mathcal{C}^{\mathrm{ROB}}$ and not for $\mathcal{C}^{\mathrm{NOM}}$. This difference is important, as the uncertain constraints have an impact on the set of feasible solutions. Thus, an uncertain constraint may imply a valid inequality for $\mathcal{C}^{\mathrm{ROB}}$ that can be used for recycling with respect to another uncertain constraint. We only consider recycling for uncertain objective functions in the following, as we suppose that studying the reciprocal effects between different uncertain constraints is complex enough to motivate a separate paper. In the

next section, we will investigate the strength of recycled inequalities for uncertain objectives from a polyhedral point of view.

3 Facet-defining recycled inequalities

In this section, we show that recycled inequalities often define facets of the convex hull of the robust problem $\mathcal{C}^{\mathrm{ROB}}$. Formally, for a polyhedron $P \subseteq \mathbb{R}^n$, each valid inequality $\sum_{i\in[n]} \pi_i x \leq \pi_0$ defines a *face* $F = \left\{x \in P \middle| \sum_{i\in[n]} \pi_i x = \pi_0\right\}$ of P. A face F is called a *facet* if $\dim(F) = \dim(P) - 1$ holds, where the dimension of a polyhedron P is defined as the maximum number of affinely independent points within P minus one [25]. We say that an inequality is *facet-defining* if its induced face is a facet. Facet-defining inequalities are highly interesting, as they are the best inequalities to describe a polyhedron P in the sense that a minimal representation

$$P = \left\{ x \in \mathbb{R}^n \middle| \begin{array}{l} A^{=}x = b^{=} \\ A^{<}x \leq b^{<} \end{array} \right\},$$

with as few constraints as possible, consists of $n - \dim(P)$ equations $A^{=}x = b^{=}$ and otherwise only proper inequalities $A^{<}x \leq b^{<}$, all of which are facet-defining [25].

In order to prove that recycled inequalities can be facet-defining for $\mathcal{C}^{\mathrm{ROB}}$, we first determine the dimension of $\mathcal{C}^{\mathrm{ROB}}$. For the sake of simplicity, we assume for the rest of this paper that the solution sets $\mathcal{C}^{\mathrm{NOM}}$ and $\mathcal{C}^{\mathrm{ROB}}$ are non-empty.

Lemma 1 *We have* $\dim\left(\mathcal{C}^{ROB}\right) = \dim\left(\mathcal{C}^{NOM}\right) + n + 1$.

Proof For a polyhedron $P \subseteq \mathbb{R}^n$, the number $n - \dim(P)$ equals the maximum number of linearly independent equations that are met by all vectors in P. Let $\sum_{i\in[n]} (\omega_i x_i + \omega_{n+i} p_i) + \omega_{2n+1} z = \omega_0$ be satisfied by all $(x, p, z) \in \mathcal{C}^{\mathrm{ROB}}$. Since p and z can be raised arbitrarily and $\mathcal{C}^{\mathrm{ROB}} \neq \emptyset$, we have $\omega_{n+1} = \cdots = \omega_{2n+1} = 0$, and thus $\sum_{i\in[n]} \omega_i x_i = \omega_0$. Hence, the equations that are met by all $(x, p, z) \in \mathcal{C}^{\mathrm{ROB}}$ are exactly the equations that are met by all $x \in \mathcal{C}^{\mathrm{NOM}}$, which implies

$$\dim\left(\mathcal{C}^{\mathrm{ROB}}\right) = 2n + 1 - \left(n - \dim\left(\mathcal{C}^{\mathrm{NOM}}\right)\right) = \dim\left(\mathcal{C}^{\mathrm{NOM}}\right) + n + 1.$$

□

Knowing the dimension of $\mathcal{C}^{\mathrm{ROB}}$, we are now able to study facet-defining recycled inequalities. For this, we let

$$F^{\mathrm{NOM}}(\pi) = \left\{ x \in \mathcal{C}^{\mathrm{NOM}} \middle| \sum_{i\in[n]} \pi_i x_i = \pi_0 \right\}$$

be the face of $\mathcal{C}^{\text{NOM}}$ induced by $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ and

$$F^{\text{ROB}}(\pi) = \left\{ (x, p, z) \in \mathcal{C}^{\text{ROB}} \middle| \pi_0 z + \sum_{i\in[n]} \pi_i p_i = \sum_{i\in[n]} \pi_i \hat{c}_i x_i \right\}$$

be the face of $\mathcal{C}^{\text{ROB}}$ induced by the corresponding recycled inequality. Furthermore, for some $F \subseteq \mathcal{C}^{\text{NOM}}$ and $S \subseteq [n]$, let $\text{proj}_S(F)$ be the projection on the subspace of variables $\{x_i | i \in S\}$. The following theorem states the dimension of $F^{\text{ROB}}(\pi)$ based on the dimension of the projection of $F^{\text{NOM}}(\pi)$ onto the subspace of variables x_i with $\pi_i > 0$. Together with Lemma 1, this yields a characterization of facet-defining recycled inequalities. Remember that we only consider uncertainty-exclusive inequalities, as these are the only non-dominated ones.

Theorem 3 *Let $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ be a recyclable, uncertainty-exclusive inequality and $S = \{i \in [n] | \pi_i > 0\}$. Then the face $F^{ROB}(\pi)$ has dimension*

$$\dim\left(\text{proj}_S\left(F^{NOM}(\pi)\right)\right) + \dim\left(\mathcal{C}^{NOM}\right) + 1 + n - |S|.$$

Hence, the recycled inequality from Definition 1 is facet-defining for $\mathcal{C}^{ROB}$ if and only if

$$\dim\left(\text{proj}_S\left(F^{NOM}(\pi)\right)\right) = |S| - 1$$

holds.

Proof There always exist $\dim\left(\mathcal{C}^{\text{NOM}}\right) + 1 + n - |S|$ affinely independent vectors $(x, p, z) \in F^{\text{ROB}}(\pi)$. To see this, let $\left\{x^0, \ldots, x^{\dim(\mathcal{C}^{\text{NOM}})}\right\} \subseteq \mathcal{C}^{\text{NOM}}$ be affinely independent. We construct two families of vectors from these. For the first familiy of $\dim\left(\mathcal{C}^{\text{NOM}}\right) + 1$ vectors, let $(\bar{x}, \bar{p}, \bar{z})^j = \left(x^j, \hat{c} \odot x^j, 0\right)$ for $j \in \left\{0, \ldots, \dim\left(\mathcal{C}^{\text{NOM}}\right)\right\}$, where $\hat{c} \odot x^j$ refers to the component-wise multiplication, i.e., $\left(\hat{c} \odot x^j\right)_i = \hat{c}_i x_i^j$. By definition, $(\bar{x}, \bar{p}, \bar{z})^j$ is within $\mathcal{C}^{\text{ROB}}$ and we have

$$\pi_0 \bar{z}^j + \sum_{i\in[n]} \pi_i \bar{p}_i^j = \pi_0 0 + \sum_{i\in[n]} \pi_i \left(\hat{c} \odot x^j\right)_i = \sum_{i\in[n]} \pi_i \hat{c}_i \bar{x}_i^j.$$

For the second family of $n - |S|$ vectors, we choose $\left(\mathring{x}, \mathring{p}, \mathring{z}\right)^j = \left(x^0, \hat{c} \odot x^0 + e^j, 0\right)$ for each $j \in [n] \setminus S$, with $e^j \in \mathbb{R}^n$ being the j-th canonical vector. Again, each vector is within $\mathcal{C}^{\text{ROB}}$ and due to $\pi_j = 0$ it follows

$$\pi_0 \mathring{z}^j + \sum_{i\in[n]} \pi_i \mathring{p}_i^j = \pi_0 0 + \sum_{i\in[n]} \pi_i \left(\hat{c} \odot x^0 + e^j\right)_i = \pi_j + \sum_{i\in[n]} \pi_i \hat{c}_i x_i^0 = \sum_{i\in[n]} \pi_i \hat{c}_i \mathring{x}_i^0.$$

We extend the $\dim\left(\mathcal{C}^{\text{NOM}}\right) + 1 + n - |S|$ vectors above with a set of additional vectors $\left\{(\tilde{x}, \tilde{p}, \tilde{z})^j \middle| j \in [k]\right\} \subseteq \mathcal{C}^{\text{ROB}}$ for some $k \in \mathbb{Z}_{\geq 0}$. We say that such an *extension*

is *valid* if the additional vectors are in $F^{\mathrm{ROB}}(\pi)$ and are affinely independent together with the vectors above. Let $k' \in \mathbb{Z}_{\geq 0}$ be the maximum number such that there exists a valid extension with k' vectors. In order to prove the theorem, we show that $k' = \dim\left(\mathrm{proj}_S\left(F^{\mathrm{NOM}}(\pi)\right)\right) + 1$ holds. For this, we can restrict ourselves without loss of generality to extensions with binary $\tilde{x}^j$. If $\tilde{x}^j$ is not binary, then $(\tilde{x}, \tilde{p}, \tilde{z})^j$ is a convex combination of integer-feasible solutions within $\mathcal{C}^{\mathrm{ROB}}$. Convex combinations are a special case of affine combinations, and thus one of these integer-feasible solutions must be affinely independent and can replace $(\tilde{x}, \tilde{p}, \tilde{z})^j$. Otherwise, $(\tilde{x}, \tilde{p}, \tilde{z})^j$ itself would not be affinely independent.

To show $k' = \dim\left(\mathrm{proj}_S\left(F^{\mathrm{NOM}}(\pi)\right)\right) + 1$, we claim that an extension $\left\{(\tilde{x}, \tilde{p}, \tilde{z})^j \middle| j \in [k]\right\}$ is valid if and only if the following four properties hold:

1. $\tilde{p}_i^j = \left(\hat{c}_i - \tilde{z}^j\right)\tilde{x}_i^j$ for all $j \in [k], i \in S$,
2. $\tilde{z}^j > 0$ for all $j \in [k]$,
3. $\left\{\mathrm{proj}_S\left(\tilde{x}^j\right) \middle| j \in [k]\right\}$ are affinely independent,
4. $\left\{\tilde{x}^j \middle| j \in [k]\right\} \subseteq F^{\mathrm{NOM}}(\pi)$.

Assume that the above claim is true. Then properties 3 and 4 imply $k' \leq \dim\left(\mathrm{proj}_S\left(F^{\mathrm{NOM}}(\pi)\right)\right) + 1$. In order to prove $k' \geq \dim\left(\mathrm{proj}_S\left(F^{\mathrm{NOM}}(\pi)\right)\right) + 1$, let $\left\{\tilde{x}^0, \ldots, \tilde{x}^{\dim\left(\mathrm{proj}_S\left(F^{\mathrm{NOM}}(\pi)\right)\right)}\right\} \subseteq F^{\mathrm{NOM}}(\pi)$ be affinely independent in the components corresponding to $\{x_i | i \in S\}$. These vectors fulfill properties 3 and 4, and thus it suffices to construct $\tilde{p}^j, \tilde{z}^j$ for each $j \in \left\{0, \ldots, \dim\left(\mathrm{proj}_S\left(F^{\mathrm{NOM}}(\pi)\right)\right)\right\}$ such that $(\tilde{x}, \tilde{p}, \tilde{z})^j$ satisfies properties 1 and 2. We choose $\tilde{z}^j = \min\left\{\hat{c}_i \middle| i \in [n], \hat{c}_i > 0\right\}$ and $\tilde{p}_i^j = \left(\hat{c}_i - \tilde{z}^j\right)^+ \tilde{x}_i^j$ for all $i \in [n]$. Then $(\tilde{x}, \tilde{p}, \tilde{z})^j$ is by definition within $\mathcal{C}^{\mathrm{ROB}}$ and satisfies $\tilde{z}^j > 0$, since we assumed that there exists at least one deviation $\hat{c}_i$ that is strictly positive. Since π is uncertainty-exclusive, we have $\hat{c}_i > 0$ for all $i \in S$, which yields $\hat{c}_i \geq \tilde{z}^j$ and thus $\tilde{p}_i^j = \left(\hat{c}_i - \tilde{z}^j\right)\tilde{x}_i^j$ for all $i \in S$. Therefore, the vectors $(\tilde{x}, \tilde{p}, \tilde{z})^j$ satisfy properties 1 to 4, and thus constitute a valid extension.

It remains to show the equivalency between the validity of extensions and the properties 1 to 4. For this, let $\left\{(\tilde{x}, \tilde{p}, \tilde{z})^j \middle| j \in [k]\right\} \subseteq F^{\mathrm{ROB}}(\pi)$ be a valid extension of arbitrary size $k \in \mathbb{Z}_{\geq 0}$. Since we can assume $\tilde{x}^j \in \{0, 1\}^n$ for all $j \in [k]$, we have $\tilde{p}_i^j \geq \left(\hat{c}_i - \tilde{z}^j\right)\tilde{x}_i^j$ for all $i \in [n]$. If property 1 did not hold, then there would exist indices $j \in [k]$ and $i \in S$ with $\tilde{p}_i^j > \left(\hat{c}_i - \tilde{z}^j\right)\tilde{x}_i^j$, and thus $\pi_i \tilde{p}_i^j > \pi_i\left(\hat{c}_i - \tilde{z}^j\right)\tilde{x}_i^j$, since $\pi_i > 0$ holds for $i \in S$. This is a contradiction due to

$$\begin{aligned}
\pi_0 \tilde{z}^j + \sum_{i \in [n]} \pi_i \tilde{p}_i^j &> \pi_0 \tilde{z}^j + \sum_{i \in [n]} \pi_i \left(\hat{c}_i - \tilde{z}^j\right)\tilde{x}_i^j \\
&\overset{(*)}{\geq} \sum_{i \in [n]} \pi_i \tilde{x}_i^j \tilde{z}^j + \sum_{i \in [n]} \pi_i \left(\hat{c}_i - \tilde{z}^j\right)\tilde{x}_i^j \\
&= \sum_{i \in [n]} \pi_i \hat{c}_i \tilde{x}_i^j,
\end{aligned}$$

which implies $(\tilde{x}, \tilde{p}, \tilde{z})^j \notin F^{\mathrm{ROB}}(\pi)$, where $(*)$ follows from $\sum_{i\in[n]} \pi_i x_i \le \pi_0$.

The affine independency of the extension is equivalent to the linear independency of the $\dim\left(\mathcal{C}^{\mathrm{NOM}}\right) + n - |S| + k$ vectors obtained from subtracting $(\bar{x}, \bar{p}, \bar{z})^0$ of all others, that is, $\left\{(\bar{x}, \bar{p}, \bar{z})^j - (\bar{x}, \bar{p}, \bar{z})^0 \middle| j \in \left[\dim\left(\mathcal{C}^{\mathrm{NOM}}\right)\right]\right\}$ and $\left\{\left(\mathring{x}, \mathring{p}, \mathring{z}\right)^j - (\bar{x}, \bar{p}, \bar{z})^0 \middle| j \in [n] \setminus S\right\}$ as well as $\left\{(\tilde{x}, \tilde{p}, \tilde{z})^j - (\bar{x}, \bar{p}, \bar{z})^0 \middle| j \in [k]\right\}$. We write these into the following matrix, using property 1 and with indices ordered such that $S = \{1, \ldots, |S|\}$. The vertical lines separate the different families of vectors and the horizontal lines separate the components corresponding to the variables x, p, and z.

$$\left(\begin{array}{ccc|ccc|ccc}
\ldots & x_1^j - x_1^0 & \ldots & 0 & \ldots & 0 & \ldots & \tilde{x}_1^j - x_1^0 & \ldots \\
 & \vdots & & \vdots & & \vdots & & \vdots & \\
\ldots & x_n^j - x_n^0 & \ldots & 0 & \ldots & 0 & \ldots & \tilde{x}_n^j - x_n^0 & \ldots \\
\hline
\ldots & \hat{c}_1\left(x_1^j - x_1^0\right) & \ldots & 0 & \ldots & 0 & \ldots & \left(\hat{c}_1 - \tilde{z}^j\right)\tilde{x}_1^j - \hat{c}_1 x_1^0 & \ldots \\
 & \vdots & & \vdots & & \vdots & & \vdots & \\
 & \vdots & & 0 & \ldots & 0 & & \left(\hat{c}_{|S|} - \tilde{z}^j\right)\tilde{x}_{|S|}^j - \hat{c}_{|S|} x_{|S|}^0 & \\
 & \vdots & & 1 & & 0 & & \tilde{p}_{|S|+1}^j - \hat{c}_{|S|+1} x_{|S|+1}^0 & \\
 & \vdots & & & \ddots & & & \vdots & \\
\ldots & \hat{c}_n\left(x_n^j - x_n^0\right) & \ldots & 0 & & 1 & \ldots & \tilde{p}_n^j - \hat{c}_n x_n^0 & \ldots \\
\hline
\ldots & 0 & \ldots & 0 & \ldots & 0 & \ldots & \tilde{z}^j & \ldots
\end{array}\right).$$

For each $i \in [n]$, we subtract row i from row $n + i$ with factor $\hat{c}_i$ and obtain

$$\left(\begin{array}{ccc|ccc|ccc}
\ldots & x_1^j - x_1^0 & \ldots & 0 & \ldots & 0 & \ldots & \tilde{x}_1^j - x_1^0 & \ldots \\
 & \vdots & & \vdots & & \vdots & & \vdots & \\
\ldots & x_n^j - x_n^0 & \ldots & 0 & \ldots & 0 & \ldots & \tilde{x}_n^j - x_n^0 & \ldots \\
\hline
\ldots & 0 & \ldots & 0 & \ldots & 0 & \ldots & -\tilde{z}^j \tilde{x}_1^j & \ldots \\
 & \vdots & & \vdots & & \vdots & & \vdots & \\
 & \vdots & & 0 & \ldots & 0 & & -\tilde{z}^j \tilde{x}_{|S|}^j & \\
 & \vdots & & 1 & & 0 & & \tilde{p}_{|S|+1}^j - \hat{c}_{|S|+1} \tilde{x}_{|S|+1}^j & \\
 & \vdots & & & \ddots & & & \vdots & \\
\ldots & 0 & \ldots & 0 & & 1 & \ldots & \tilde{p}_n^j - \hat{c}_n \tilde{x}_n^j & \ldots \\
\hline
\ldots & 0 & \ldots & 0 & \ldots & 0 & \ldots & \tilde{z}^j & \ldots
\end{array}\right).$$

We use the middle columns containing the canonical vectors to eliminate the corresponding rows. We have $\tilde{x}^j \in \mathcal{C}^{\mathrm{NOM}}$ and thus $\tilde{x}^j - x^0$ is linearly dependent of the vectors $\left\{x^1 - x^0, \ldots, x^{\dim(\mathcal{C}^{\mathrm{NOM}})} - x^0\right\}$ due to the dimension of $\mathcal{C}^{\mathrm{NOM}}$. Hence, we

can also eliminate the first n rows in the last k columns and obtain

$$\left(\begin{array}{ccc|ccc|ccc} \ldots & x_1^j - x_1^0 & \ldots & 0 & \ldots & 0 & \ldots & 0 & \ldots \\ & \vdots & & \vdots & & \vdots & & \vdots & \\ \ldots & x_n^j - x_n^0 & \ldots & 0 & \ldots & 0 & \ldots & 0 & \ldots \\ \hline \ldots & 0 & \ldots & 0 & \ldots & 0 & \ldots & -\tilde{z}^j \tilde{x}_1^j & \ldots \\ & \vdots & & \vdots & & \vdots & & \vdots & \\ & \vdots & & 0 & \ldots & 0 & & -\tilde{z}^j \tilde{x}_{|S|}^j & \\ & \vdots & & 1 & & 0 & & 0 & \\ & \vdots & & & \ddots & & & \vdots & \\ \ldots & 0 & \ldots & 0 & & 1 & \ldots & 0 & \ldots \\ \hline \ldots & 0 & \ldots & 0 & \ldots & 0 & \ldots & \tilde{z}^j & \ldots \end{array}\right).$$

These columns are linearly independent if and only if $\tilde{z}^j \neq 0$ holds for all $j \in [k]$ and $\left\{\text{proj}_S\left(\tilde{x}^j\right) \middle| j \in [k]\right\}$ are affinely independent. Thus, if property 1 holds, then the affine independency of the extension is equivalent to properties 2 and 3. Since we have already shown above that property 1 is implied, we know that the validity of the extension implies properties 1 to 3. Conversely, properties 1 to 3 imply the affine independency of the extension. Furthermore, given property 1, we have $\tilde{p}_i^j = \left(\hat{c}_i - \tilde{z}^j\right)\tilde{x}_i^j$ for all $i \in S$ and $\pi_i = 0$ for all $i \notin S$, and thus $\pi_i \tilde{p}_i^j = \pi_i \left(\hat{c}_i - \tilde{z}^j\right)\tilde{x}_i^j$ for all $i \in [n]$. Together with property 2, this yields

$$\pi_0 \tilde{z}^j + \sum_{i\in[n]} \pi_i \tilde{p}_i^j = \sum_{i\in[n]} \pi_i \hat{c}_i \tilde{x}_i^j \;\Leftrightarrow\; \pi_0 \tilde{z}^j = \sum_{i\in[n]} \pi_i \tilde{z}^j \tilde{x}_i^j \;\Leftrightarrow\; \pi_0 = \sum_{i\in[n]} \pi_i \tilde{x}_i^j,$$

and thus $\left\{\tilde{x}^j \middle| j \in [k]\right\} \subseteq F^{\text{NOM}}(\pi)$. Hence, property 4 holds if and only if $\left\{(\tilde{x}, \tilde{p}, \tilde{z})^j \middle| j \in [k]\right\}$ fulfill the recycled inequality with equality. In summary, this shows that the validity of the extension is equivalent to properties 1 to 4. □

A powerful implication of Theorem 3 is that recycling an uncertainty-exclusive inequality yields always a facet-defining inequality if $\dim\left(F^{\text{NOM}}(\pi)\right) = n - 1$ holds. This is because there exist n affinely independent vectors satisfying $\sum_{i\in[n]} \pi_i x = \pi_0$, of which $|S|$ must be affinely independent when projected on the subspace of the variables $\{x_i | i \in S\}$. Note that $\dim\left(F^{\text{NOM}}(\pi)\right) = n - 1$ holds if $F^{\text{NOM}}(\pi)$ is either a facet of a full-dimensional polyhedron $\mathcal{C}^{\text{NOM}}$ or if $\sum_{i\in[n]} \pi_i x \leq \pi_0$ is actually an equation with $F^{\text{NOM}}(\pi) = \mathcal{C}^{\text{NOM}}$ and $\dim\left(\mathcal{C}^{\text{NOM}}\right) = n - 1$. This is summarized in the following corollary.

Corollary 1 *Let $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ be a recyclable, uncertainty-exclusive inequality. The recycled inequality from Definition 1 is facet-defining for $\mathcal{C}^{ROB}$ if one of the following holds:*

- *$\mathcal{C}^{NOM}$ is full-dimensional and $F^{NOM}(\pi)$ is a facet of $\mathcal{C}^{NOM}$,*

- $\dim\left(\mathcal{C}^{NOM}\right) = n - 1$ *and* $F^{NOM}(\pi) = \mathcal{C}^{NOM}$.

Contrary to first intuition, it is also possible to obtain facet-defining inequalities by recycling weaker inequalities that are neither facet-defining nor equations. This is because the dimension of the face $F^{\mathrm{NOM}}(\pi)$ can shrink by less than $n - |S|$ when projected on the subspace of the variables $\{x_i | i \in S\}$. Thus, inequalities defining low-dimensional faces can also yield facet-defining recycled inequalities if $\pi_i = 0$ holds for many $i \in [n]$. For example, consider an independent set problem on a graph $G = (V, E)$ with nodes $V = [n]$. The set of feasible solutions can be stated as $\mathcal{I} = \left\{x \in \{0,1\}^n \middle| x_i + x_j \leq 1 \; \forall \{i, j\} \in E\right\}$, with $x_i = 1$ if $i \in V$ belongs to the independent set and $x_i = 0$ otherwise. If $Q \subseteq V$ is a clique in G, then the *clique inequality* $\sum_{i \in Q} x_i \leq 1$ is valid for $\mathcal{I}$. Furthermore, the inequality strictly dominates all inequalities $\sum_{i \in Q'} x_i \leq 1$ with $Q' \subsetneq Q$ and it is facet-defining for $\mathrm{conv}(\mathcal{I})$ if and only if Q is a maximal clique with respect to inclusion [25]. However, if $\hat{c}_i > 0$ holds for all $i \in Q$, then the recycled inequality $z + \sum_{i \in Q'} p_i \geq \sum_{i \in Q'} \hat{c}_i x_i$ defines a facet of the corresponding $\mathcal{C}^{\mathrm{ROB}}$ for all cliques $Q' \subseteq Q$. This is because the projection $\mathrm{proj}_{Q'}\left(F^{\mathrm{NOM}}(\pi)\right)$, that is $S = Q'$, contains $\left\{e^j \middle| j \in Q'\right\}$ and thus has dimension $\left|Q'\right| - 1$. Other examples include odd hole inequalities for the independent set problem [26] and minimal cover inequalities for the knapsack problem [25]. These are in general not facet-defining for their respective polytope $\mathcal{C}^{\mathrm{NOM}}$, but yield facet-defining recycled inequalities for the polytope $\mathcal{C}^{\mathrm{ROB}}$ of the robust counterpart.

One now might raise the question whether recycling dominated inequalities is actually of practical interest or whether these do not matter due to the special structure of the objective function, with all p_i having an objective coefficient of 1. The following example demonstrates that it can be beneficial to weaken an inequality before it is recycled.

Example 2 Consider the robust problem

$$\begin{aligned}
\min\ & 2z + \sum_{i \in [5]} -x_i + p_i \\
\text{s.t. } & 3x_5 + \sum_{i \in [4]} x_i \leq 3 \\
& z + p_i \geq x_i && \forall i \in [5] \\
& x \in \{0,1\}^5,\ p \in \mathbb{R}^5_{\geq 0},\ z \in \mathbb{R}_{\geq 0}.
\end{aligned}$$

Choosing $x = \left(\frac{3}{4}, \ldots, \frac{3}{4}, 0\right)$, $p = 0$, and $z = \frac{3}{4}$ yields an optimal solution for the linear relaxation with objective value $-\frac{3}{2}$. Recycling constraint $3x_5 + \sum_{i \in [4]} x_i \leq 3$ yields $3z + 3p_5 + \sum_{i \in [4]} p_i \geq 3x_5 + \sum_{i \in [4]} x_i$. After adding the recycled inequality, an optimal solution is given by $x = \left(\frac{3}{4}, \ldots, \frac{3}{4}, 0\right)$, $p = \left(0, \ldots, 0, \frac{1}{4}\right)$, and $z = \frac{3}{4}$, with an objective value of $-\frac{5}{4}$. Note that we now choose $p_5 > 0$ even though $x_5 = 0$ holds. Since the bilinear inequality $p_5 + x_5 z \geq \hat{c}_5 x_5$ now has a slack of $\frac{1}{4}$, our observation from the last section suggests that it may be beneficial to drop x_5 from the valid inequality for recycling. In fact, when recycling the dominated inequality

$\sum_{i\in[4]} x_i \leq 3$ instead of the model constraint, we obtain $3z + \sum_{i\in[4]} p_i \geq \sum_{i\in[4]} x_i$ and an optimal solution is now given by $x = (1, 1, 1, 0, 0)$, $p = 0$, and $z = 1$, which yields an objective value of -1.

After discussing the strong implications of Theorem 3, let us now consider its limitations. The next example shows that uncertainty-exclusiveness is indeed necessary for the correctness of Theorem 3 and Corollary 1.

Example 3 Consider the full-dimensional polyhedron

$$\mathcal{C}^{\text{ROB}} = \text{conv}\left(\left\{(x, p, z) \in \{0, 1\}^3 \times \mathbb{R}^4_{\geq 0} \;\middle|\; \begin{array}{rl} x_1 + x_2 + x_3 \leq 2 & \\ z + p_i \geq \hat{c}_i x_i & \forall i \in [3] \end{array}\right\}\right),$$

with $\hat{c}_1 = \hat{c}_2 = 1$ and $\hat{c}_3 = 0$. The constraint $x_1 + x_2 + x_3 \leq 2$ is facet-defining for the corresponding $\mathcal{C}^{\text{NOM}}$, and thus meets all requirements of Corollary 1 with the exception that it is not uncertainty-exclusive. Indeed, the recycled inequality $2z + p_1 + p_2 + p_3 \geq x_1 + x_2$ is not facet-defining for $\mathcal{C}^{\text{ROB}}$, as it is dominated by the sum of the constraints $z + p_1 \geq x_1$ and $z + p_2 \geq x_2$. Note that this does not change when recycling the corresponding uncertainty-exclusive inequality $x_1 + x_2 \leq 2$ instead.

The observation in the example above is quite intuitive. While we can always transform an arbitrary recyclable inequality into an uncertainty-exclusive one by dropping all x_i with $\hat{c}_i = 0$, we loose information during this process and cannot expect to obtain a facet of $\mathcal{C}^{\text{ROB}}$. Less obvious is the importance of the dimension of the nominal polytope $\mathcal{C}^{\text{NOM}}$, which is a prerequisite for Corollary 1. In the following, we consider another example which shows that inequalities recycled from facet-defining inequalities are not necessarily facet-defining if we do not have $\dim\left(\mathcal{C}^{\text{NOM}}\right) = n$. Hence, the dimension of the nominal problem is important for the correctness of Corollary 1. After the example, we show how the corollary can be generalized for lower-dimensional polyhedra.

Example 4 Consider the four-dimensional polyhedron with five variables

$$\mathcal{C}^{\text{ROB}} = \text{conv}\left(\left\{(x, p, z) \in \{0, 1\}^2 \times \mathbb{R}^3_{\geq 0} \;\middle|\; \begin{array}{rl} x_1 + x_2 = 1 & \\ z + p_i \geq x_i & \forall i \in [2] \end{array}\right\}\right).$$

The inequality $2x_1 + x_2 \leq 2$ defines a facet for the corresponding 1-dimensional $\mathcal{C}^{\text{NOM}}$. However, the recycled inequality $2z + 2p_1 + p_2 \geq 2x_1 + x_2$ is the sum of $z + p_1 \geq x_1$ and $z + p_1 + p_2 \geq x_1 + x_2$, where the latter is recycled from $x_1 + x_2 = 1$. Hence, the face of $\mathcal{C}^{\text{ROB}}$ induced by $2z + 2p_1 + p_2 \geq 2x_1 + x_2$ is the intersection of the faces induced by $z + p_1 \geq x_1$ and $z + p_1 + p_2 \geq x_1 + x_2$. Since these are not equal, the face induced by $2z + 2p_1 + p_2 \geq 2x_1 + x_2$ has a lower dimension, and thus cannot be a facet.

Note that the mapping from the set of recyclable inequalities to the corresponding recycled inequalities is a homomorphism in the sense that the recycled inequality

of $\sum_{i\in[n]}\left(\pi_i^1+\pi_i^2\right)x_i \leq \pi_0^1+\pi_0^2$ equals the sum of the recycled inequalities of $\sum_{i\in[n]}\pi_i^1 x_i \leq \pi_0^1$ and $\sum_{i\in[n]}\pi_i^2 x_i \leq \pi_0^2$. As recycled inequalities are proper inequalities, their sum is weaker than the separate inequalities (if they do not define the same face) and it is better to recycle each inequality individually. This applies to the example, where $2x_1+x_2 \leq 2$ is the sum of the recyclable inequality $x_1 \leq 1$ and the recyclable equation $x_1+x_2=1$. Although $x_1 \leq 1$ and $2x_1+x_2 \leq 2$ define the same face, recycling $x_1 \leq 1$ yields a facet-defining inequality, while recycling $2x_1+x_2 \leq 2$ does not. Hence, for lower dimensional $\mathcal{C}^{\mathrm{NOM}}$, one cannot decide whether recycling yields a facet-defining inequality by solely relying on the face induced by the recyclable inequality.

However, as in the example, we can eliminate equations from an inequality and recycle the resulting one, which is equivalent for $\mathcal{C}^{\mathrm{NOM}}$. Formally, we call two inequalities $\sum_{i\in[n]}\pi_i' x_i \leq \pi_0'$ and $\sum_{i\in[n]}\pi_i x_i \leq \pi_0$ *equivalent for a polyhedron* P if there exist equations $\sum_{i\in[n]}\omega_i^k x_i = \omega_0^k$ with $k\in[\ell]$, satisfied by all $x\in P$, such that $\left\{\pi', \omega^1, \ldots, \omega^\ell\right\}$ are linearly independent and $\lambda_0\pi' + \sum_{k\in[\ell]}\lambda_k\omega^k = \pi$ holds for some $\lambda_0 > 0$ and $\lambda_1, \ldots, \lambda_\ell \in \mathbb{R}$ [27]. Note that two equivalent inequalities define the same face.

Algorithm 1: Procedure for eliminating equations from π.

Input: A recyclable, uncertainty-exclusive inequality $\sum_{i\in[n]}\pi_i' x_i \leq \pi_0'$ and equations $\sum_{i\in[n]}\omega_i^k x_i = \omega_0^k$ for $k\in[\ell]$, satisfied by all $x\in\mathcal{C}^{\mathrm{NOM}}$, such that $\left\{\pi', \omega^1, \ldots, \omega^\ell\right\}$ are linearly independent

Output: An equivalent recyclable, uncertainty-exclusive inequality $\sum_{i\in[n]}\pi_i x_i \leq \pi_0$ with $|\{i\in[n] | \pi_i = 0\}| \geq \ell$

1 Set $\pi = \pi'$
2 **for** $k\in[\ell]$ **do**
3 Choose $i^* \in \operatorname{argmin}\left\{\left|\frac{\pi_i}{\omega_i^k}\right| \middle| i\in[n], \omega_i^k \neq 0,\right\}$
4 Update $\pi \leftarrow \pi - \frac{\pi_{i^*}}{\omega_{i^*}^k}\omega^k$
5 **for** $k'\in\{k+1,\ldots,\ell\}$ **do**
6 Update $\omega^{k'} \leftarrow \omega^{k'} - \frac{\omega_{i^*}^{k'}}{\omega_{i^*}^k}\omega^k$
7 **return** $\sum_{i\in[n]}\pi_i x \leq \pi_0$

We already noted below Corollary 1 that having many zero coefficients in a recyclable inequality is beneficial for obtaining facet-defining recycled inequalities. Accordingly, when eliminating equations from a recyclable inequality, we want to increase the number of zero coefficients, if possible. For given equations $\sum_{i\in[n]}\omega_i^k x_i = \omega_0^k$, with $k\in[\ell]$, Algorithm 1 performs a specialized Gaussian elimination on a recyclable inequality $\sum_{i\in[n]}\pi_i' x_i \leq \pi_0'$ in order to obtain an equivalent inequality $\sum_{i\in[n]}\pi_i x_i \leq \pi_0$ with at least ℓ zero coefficients. For this, Algorithm 1 subtracts for each equation given by ω^k in line 4 a multiple $\frac{\pi_{i^*}}{\omega_{i^*}^k}$ of ω^k from π such that the coefficient π_{i^*} becomes or stays zero. The index i^* is chosen with respect to a

bottleneck condition in line 3, which ensures that $\frac{\pi_{i^*}}{\omega_{i^*}^k}$ is the multiple with the smallest absolute value for all $i \in [n]$ with $\omega_i^k \neq 0$. This has two desirable implications. First, if π_i was positive before, then it will be non-negative after subtracting $\frac{\pi_{i^*}}{\omega_{i^*}^k}\omega_i^k$. Second, if π_i was already zero before, then it will still be zero afterwards. This implies that if $\sum_{i\in[n]} \pi_i' x_i \leq \pi_0'$ is recyclable and uncertainty-exclusive, then this also holds for the resulting $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$. After eliminating π_{i^*}, we make sure in line 6 that $\omega_{i^*}^k = 0$ holds for all remaining equations such that i^* will be different in every iteration. By doing so, we guarantee that we have at least ℓ indices $i \in [n]$ with $\pi_i = 0$ at the end of Algorithm 1.

The following proposition uses Algorithm 1 to generalize Corollary 1 for lower-dimensional $\mathcal{C}^{\mathrm{ROB}}$.

Proposition 2 *Let $\sum_{i\in[n]} \pi_i' x_i \leq \pi_0'$ be a recyclable, uncertainty-exclusive inequality such that $F^{NOM}\left(\pi'\right)$ is either a facet of $\mathcal{C}^{NOM}$ or $F^{NOM}\left(\pi'\right) = \mathcal{C}^{NOM}$. Let furthermore ℓ be the maximum number of equations $\sum_{i\in[n]} \omega_i^k x_i = \omega_0^k$ for $k \in [\ell]$, satisfied by all $x \in \mathcal{C}^{NOM}$, such that $\left\{\pi', \omega^1, \ldots, \omega^\ell\right\}$ are linearly independent. Then Algorithm 1 computes an equivalent recyclable inequality $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ whose recycled inequality is facet-defining for $\mathcal{C}^{ROB}$.*

Proof Since the returned inequality $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ from Algorithm 1 is recyclable and uncertainty-exclusive, it only remains to show by Theorem 3 that $\dim\left(\mathrm{proj}_S\left(F^{\mathrm{NOM}}(\pi)\right)\right) = |S| - 1$ holds, with $S = \{i \in [n] | \pi_i > 0\}$. Remember that $F^{\mathrm{NOM}}(\pi) = F^{\mathrm{NOM}}\left(\pi'\right)$ holds. Due to the choice of ℓ, we have

$$\ell = \begin{cases} n - \dim\left(\mathcal{C}^{\mathrm{NOM}}\right) - 1, & \text{if } F^{\mathrm{NOM}}(\pi) = \mathcal{C}^{\mathrm{NOM}} \\ n - \dim\left(\mathcal{C}^{\mathrm{NOM}}\right), & \text{otherwise.} \end{cases}$$

Hence, there exists a set $\left\{x^1, \ldots, x^{n-\ell}\right\} \subseteq F^{\mathrm{NOM}}(\pi)$ of affinely independent vectors. Let furthermore $T \subseteq [n] \setminus S$ consist of the ℓ indices i^* that were chosen in Algorithm 1. We show that the vectors $\left\{\mathrm{proj}_{[n]\setminus T}\left(x^1\right), \ldots, \mathrm{proj}_{[n]\setminus T}\left(x^{n-\ell}\right)\right\}$ are affinely independent, i.e., $\dim\left(\mathrm{proj}_{[n]\setminus T}\left(F^{\mathrm{NOM}}(\pi)\right)\right) \geq n - |T| - 1$. Since $S \subseteq [n] \setminus T$ holds, this implies $\dim\left(\mathrm{proj}_S\left(F^{\mathrm{NOM}}(\pi)\right)\right) \geq |S| - 1$. Furthermore, since the equation induced by π is only on the variables $\{x_i | i \in S\}$, we have $\dim\left(\mathrm{proj}_S\left(F^{\mathrm{NOM}}(\pi)\right)\right) < |S|$, which then proves the proposition.

Assume that the projections $\mathrm{proj}_{[n]\setminus T}\left(x^j\right)$ are not affinely independent. Then there exist coefficients $\lambda \in \mathbb{R}^{n-\ell}$ with $\lambda \neq 0$, $\sum_{j\in[n-\ell]} \lambda_j = 0$, and $\sum_{j\in[n-\ell]} \lambda_j x_i^j = 0$ for all $i \in [n] \setminus T$. Consider a fixed but arbitrary index $i^* \in T$. Since i^* was chosen in Algorithm 1, there exists an equation $\sum_{i\in[n]} \omega_i^k x_i = \omega_0^k$ with $\omega_{i^*}^k \neq 0$. Without loss of generality, we can assume $\omega_{i^*}^k = 1$ and obtain

$$x_{i^*}^j = \omega_0^k - \sum_{i\in[n]\setminus\{i^*\}} \omega_i^k x_i^j$$

for all $j \in [n-\ell]$, and thus

$$\sum_{j\in[n-\ell]} \lambda_j x_{i^*}^j = \sum_{j\in[n-\ell]} \lambda_j \left(\omega_0^k - \sum_{i\in[n]\setminus\{i^*\}} \omega_i^k x_i^j\right)$$
$$= \omega_0^k \underbrace{\sum_{j\in[n-\ell]} \lambda_j}_{=0} - \sum_{i\in[n]\setminus\{i^*\}} \omega_i^k \underbrace{\sum_{j\in[n-\ell]} \lambda_j x_i^j}_{=0} = 0.$$

However, as this applies for all $i^* \in T$, we have $\sum_{j\in[n-\ell]} \lambda_j x_i^j = 0$ for all $i \in [n]$, since this was already true for all $i \in [n] \setminus T$. This implies that the vectors $\{x^1, \ldots, x^{n-\ell}\}$ are affinely dependent, which contradicts their choice and completes the proof. □

Note that we do not always know the dimension of $\mathcal{C}^{\text{NOM}}$ in practice, let alone all equations $\sum_{i\in[n]} \omega_i^k x_i = \omega_0^k$. We tested Algorithm 1 using the already present equations in the constraint matrix $Ax \leq b$ of the robust instances generated from the MIPLIB 2017, which we use in our computational study in Sect. 6.6. Interestingly, we observed no improvement in the dual bound provided by the linear relaxation compared to the setting in which we didn't use Algorithm 1. Thus, Algorithm 1 is more of a theoretical tool for Proposition 2.

Now that we have established a good theoretical understanding of the strength of recycled inequalities, we discuss in the next section how to use them in practice.

4 Separating recycled inequalities

In the previous section, we have seen that recycling can yield a vast number of facet-defining inequalities. For example, in the case of the independent set problem, every clique inequality can be recycled to a facet-defining inequality. Therefore, potentially exponentially many facet-defining recycled inequalities exist, which raises the need for an efficient separation.

4.1 Separation of recycled constraints

A straightforward separation approach is to recycle the constraints $Ax \leq b$ of the nominal problem. Given a row $\sum_{i\in[n]} a_{ji} x_i \leq b_j$ of the constraint matrix, we transform it into a recyclable inequality by tentatively fixing $x_i = 1$ for all $i \in [n]$ with $a_{ji} < 0$. Since the variables x_i are binary,

$$\sum_{i\in[n]:a_{ji}\geq 0} a_{ji} x_i \leq b_j - \sum_{i\in[n]:a_{ji}<0} a_{ji}$$

is a recyclable inequality for $\mathcal{C}^{\text{ROB}}$. We may either add the corresponding recycled inequalities directly to the formulation $\mathcal{P}^{\text{ROB}}$ or precalculate and store them for later

separation during branch and cut. In both cases, we restrict ourselves to inequalities with

$$\sum_{i\in[n]:a_{ji}\geq 0} a_{ji} > b_j - \sum_{i\in[n]:a_{ji}<0} a_{ji},$$

as the corresponding recycled inequality is otherwise dominated by the robustness constraints $p_i + z \geq \hat{c}_i x_i$. When using the precalculated recyclable inequalities for separation to cut off a fractional solution $(\tilde{x}, \tilde{p}, \tilde{z}) \in \mathcal{P}^{\text{ROB}}$, we also make sure to only include variables x_i with $\hat{c}_i\tilde{x}_i - \tilde{p}_i \geq 0$, as this maximizes the violation $\sum_{i\in[n]} \pi_i \left(\hat{c}_i\tilde{x}_i - \tilde{p}_i\right) - \pi_0\tilde{z}$ of the recycled inequality. That is, we iterate over every row in the constraint matrix $Ax \leq b$ and recycle

$$\sum_{\substack{i\in[n]:a_{ji}\geq 0,\\ \hat{c}_i\tilde{x}_i-\tilde{p}_i\geq 0}} a_{ji}x_i \leq b_j - \sum_{i\in[n]:a_{ji}<0} a_{ji}$$

if the resulting recycled inequality is violated. We will see in our computational study that this simple approach already improves the solvers performance drastically in many cases.

4.2 Separation of recycled cuts

Another approach is to benefit from the research on the nominal problem and recycle well studied cutting planes, i.e., inequalities that are valid for the convex hull $\mathcal{C}^{\text{NOM}}$ but not for the linear relaxation $\mathcal{P}^{\text{NOM}}$. Let $\Pi \subseteq \mathbb{R}^{n+1}_{\geq 0}$ be such that $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ is a recyclable inequality for all $\pi \in \Pi$. When separating inequalities to cut off a fractional solution to the nominal problem $\tilde{x} \in \mathcal{P}^{\text{NOM}}$, we would usually search for a $\pi \in \Pi$ with positive violation $\sum_{i\in[n]} \pi_i\tilde{x}_i - \pi_0 > 0$. When separating recycled inequalities for a given solution $(\tilde{x}, \tilde{p}, \tilde{z}) \in \mathcal{P}^{\text{ROB}}$, we require $\sum_{i\in[n]} \pi_i \left(\hat{c}_i\tilde{x}_i - \tilde{p}_i\right) - \pi_0\tilde{z} > 0$ instead. Note that the coefficients $\hat{c}_i\tilde{x}_i - \tilde{p}_i$ and $\tilde{z}$ are fixed in this case. Therefore, the same algorithms for finding a violated nominal inequality can be applied for separating violated recycled inequalities, provided these do not rely on some special structure of the objective function of the separation problem. In our computational study, we will test a heuristic separation of recycled clique inequalities for the robust independent set problem. We will show that these facet-defining recycled inequalities improve the formulation significantly.

4.3 Exact separation via recycling combined constraints

An exact separation of violated recycled inequalities is $\mathcal{NP}$-hard in general. For example, in the case of recycled clique inequalities, an exact separation would require solving a maximum weighted clique problem. However, we show below how we can separate recycled inequalities from valid inequalities for $\mathcal{P}^{\text{NOM}}$ in polynomial time in the size of the constraint matrix $Ax \leq b$ via solving an LP. In particular, if we already know the convex hull of the nominal problem, i.e., $\mathcal{C}^{\text{NOM}} = \mathcal{P}^{\text{NOM}}$, then an exact separation of recycled inequalities can be done in polynomial time. To see

this, note that an inequality $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ is valid for $\mathcal{P}^{\text{NOM}}$ if and only if it can be expressed as a conic combination of the rows in $Ax \leq b$ as well as the box constraints $x_i \leq 1$ and $-x_i \leq 0$. That is, we have $\nu_i - \mu_i + \sum_{j\in[m]} a_{ji}\lambda_j = \pi_i$ and $\sum_{i\in[n]} \nu_i + \sum_{j\in[m]} b_j\lambda_j = \pi_0$ for some $\lambda \in \mathbb{R}^m_{\geq 0}$, $\mu, \nu \in \mathbb{R}^n_{\geq 0}$. Hence, there exists a recyclable inequality for $\mathcal{P}^{\text{NOM}}$ whose recycled inequality is violated by $(\tilde{x}, \tilde{p}, \tilde{z}) \in \mathcal{P}^{\text{ROB}}$ if and only if the following separation LP has a solution with positive objective value

$$
\begin{aligned}
\max \ & \sum_{i\in[n]} \pi_i \left(\hat{c}_i \tilde{x}_i - \tilde{p}_i\right) - \pi_0 \tilde{z} \\
\text{s.t. } & \nu_i - \mu_i + \sum_{j\in[m]} a_{ji}\lambda_j = \pi_i && \forall i \in [n] \\
& \sum_{i\in[n]} \nu_i + \sum_{j\in[m]} b_j\lambda_j = \pi_0 \\
& \pi \in \mathbb{R}^{n+1}_{\geq 0},\ \lambda \in \mathbb{R}^m_{\geq 0},\ \mu, \nu \in \mathbb{R}^n_{\geq 0}.
\end{aligned}
\tag{SLP}
$$

Note that we demand $\pi \geq 0$ so that the inequality $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ is recyclable. As the optimal objective value of the SLP is either zero or unbounded due to arbitrary scaling, we normalize the recyclable inequality $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ by fixing $\pi_0 = 1$. This imposes no restriction on finding violated recycled inequalities, as we always have $\pi_0 > 0$ for all relevant recyclable inequalities and can thus achieve $\pi_0 = 1$ via scaling. This is because the left-hand side $\sum_{i\in[n]} \pi_i x_i$ is non-negative, and thus $\pi_0 = 0$ would imply $x_i = 0$ for all $x \in \mathcal{P}^{\text{NOM}}$ and $i \in [n]$ with $\pi_i > 0$. In this case, the right-hand side of the recycled inequality $\sum_{i\in[n]} \pi_i \hat{c}_i \tilde{x}_i$ would always be zero, which renders the inequality redundant.

Remember that recycling is a homomorphism on the set of recyclable inequalities. Hence, recycling a conic combination of inequalities, as given by a solution to SLP, is only reasonable if some of the combined inequalities are not recyclable themselves. That is, if we have $\pi = \pi^1 + \pi^2$, with π^1, π^2 linearly independent and both define recyclable inequalities, then it would be better to recycle each of these inequalities separately. In practice, this is achieved by fixing $\pi_0 = 1$, as a combination of π^1 and π^2 is never a vertex of SLP and can only be an optimal solution if the violation of their respective recycled inequalities is equal.

We observe two issues when using SLP for separation in practice. First, solving SLP is relatively time consuming if the number of inequalities is large. Second, we obtain only one optimal solution when solving SLP, and can thus only separate one recycled inequality at a time. However, MILP solvers usually perform better when several cuts are added at once. The following proposition helps in this regard, showing that we can partition the constraints into sets that can be considered independently for combination. Doing so, we can solve one smaller LP for each set of the partition, yielding multiple (possibly violated) recycled inequalities within the same separation round.

Proposition 3 *Let $A = \left(a_{ji}\right)_{j\in[m], i\in[n]}$ be the left-hand side of the constraints $Ax \leq b$ (not including $0 \leq x \leq 1$) and let $G = (V, E)$ be the graph with nodes $V = [m]$*

and edges $E = \left\{\{j, j'\} \in \binom{V}{2} \middle| \exists i \in [n] : a_{ji} < 0 < a_{j'i}\right\}$. *Let* $\{C_1, \ldots, C_k\} \subseteq 2^V$ *be the connected components in* G. *Then every inequality that is recycled from a valid inequality for* $\mathcal{P}^{NOM}$ *is dominated by inequalities recycled from recyclable inequalities of the form*

$$\sum_{i\in[n]} \left(v_i^\ell - \mu_i^\ell + \sum_{j\in C_\ell} a_{ji}\lambda_j \right) x_i \leq \sum_{i\in[n]} v_i^\ell + \sum_{j\in C_\ell} b_j\lambda_j \tag{2}$$

with $\ell \in [k]$, $\lambda \in \mathbb{R}^m_{\geq 0}$, *and* $\mu^\ell, \nu^\ell \in \mathbb{R}^n_{\geq 0}$.

Proof Let $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ be a valid inequality for $\mathcal{P}^{\text{NOM}}$. We write it as

$$\sum_{i\in[n]} \left(v_i - \mu_i + \sum_{j\in[m]} a_{ji}\lambda_j \right) x_i \leq \sum_{i\in[n]} v_i + \sum_{j\in[m]} b_j\lambda_j,$$

which is a conic combination of $Ax \leq b$ and $-x \leq 0$ as well as $x \leq 1$ with coefficients $\lambda \in \mathbb{R}^m_{\geq 0}$, $\mu, \nu \in \mathbb{R}^n_{\geq 0}$. We show that if $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ is recyclable, then it is also a conic combination of the recyclable inequalities from the statement. For this, we show that $v_i^\ell - \mu_i^\ell + \sum_{j\in C_\ell} a_{ji}\lambda_j \geq 0$ holds for all $i \in [n]$ and $\ell \in [k]$. Then we also have $\sum_{i\in[n]} v_i^\ell + \sum_{j\in C_\ell} b_j\lambda_j \geq 0$ for all $\ell \in [k]$, since the inequalities (2) are valid with non-negative left-hand sides.

Note that we can assume $v_i = 0$ or $\mu_i = 0$ for all $i \in [n]$, since we would otherwise decrease both and then recycle $\sum_{i\in[n]} \pi_i x_i \leq \pi_0 - \sum_{i\in[n]} \min\{v_i, \mu_i\}$ instead. We can also assume $v_{i'} = \left(-\sum_{j\in[m]} a_{ji'}\lambda_j\right)^+$ for all $i' \in [n]$, as otherwise $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ would be a combination of $x_{i'} \leq 1$ and the recyclable inequality obtained by decreasing $v_{i'}$ to $\left(-\sum_{j\in[m]} a_{ji'}\lambda_j\right)^+$. It follows that $\mu_{i'} = 0$ holds for all $i' \in [n]$ with $\sum_{j\in[m]} a_{ji'}\lambda_j < 0$, because we have $v_{i'} = -\sum_{j\in[m]} a_{ji'}\lambda_j > 0$ in this case. If $\sum_{j\in[m]} a_{ji'}\lambda_j \geq 0$ holds, then we can assume $\mu_{i'} \in \left\{0, \sum_{j\in[m]} a_{ji'}\lambda_j\right\}$. This is because both values result in recyclable inequalities and all values in between would imply that $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ is a convex combination of the two inequalities obtained by choosing $\mu_{i'} = 0$ or $\mu_{i'} = \sum_{j\in[m]} a_{ji'}\lambda_j$. We conclude that we only choose $\mu_{i'} > 0$ or $v_{i'} > 0$ if we want to obtain $\pi_{i'} = 0$. Otherwise, we have $\mu_{i'} = v_{i'} = 0$ and thus $\pi_{i'} = \sum_{j\in[m]} a_{ji'}\lambda_j$. Hence, we have $\mu_{i'} = 0$ for $\pi_{i'} > 0$ and $\mu_{i'} = \left(\sum_{j\in[m]} a_{ji'}\lambda_j\right)^+$ for $\pi_{i'} = 0$. Together with $v_{i'} = \left(-\sum_{j\in[m]} a_{ji'}\lambda_j\right)^+$ from above, we can rewrite $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ as

$$\sum_{i\in[n]} \left(\left(-\sum_{j\in[m]} a_{ji}\lambda_j \right)^+ + \sum_{j\in[m]} a_{ji}\lambda_j \right) x_i - \sum_{i\in[n]:\pi_i=0} \left(\sum_{j\in[m]} a_{ji}\lambda_j \right)^+ x_i$$

$$\leq \sum_{i\in[n]} \left(-\sum_{j\in[m]} a_{ji}\lambda_j \right)^+ + \sum_{j\in[m]} \lambda_j b_j.$$

Note that $\sum_{j\in[m]} a_{ji}\lambda_j$ and $\sum_{j\in C_\ell} a_{ji}\lambda_j$ have the same sign for all $i \in [n]$ and $\ell \in [k]$. Otherwise, there existed $i \in [n]$ and $\ell, \ell' \in [k]$ with $\sum_{j\in C_\ell} a_{ji}\lambda_j < 0$ and $\sum_{j\in C_{\ell'}} a_{ji}\lambda_j > 0$. Then there exists $j \in C_\ell$ and $j' \in C_{\ell'}$ such that $a_{ji} < 0 < a_{j'i}$ holds. However, this implies that constraints j and j' are adjacent in the graph G, and thus $\ell = \ell'$. It follows $\left(\pm\sum_{j\in[m]} a_{ji}\lambda_j\right)^+ = \sum_{\ell\in[k]} \left(\pm\sum_{j\in C_\ell} a_{ji}\lambda_j\right)^+$, and thus we can rewrite $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ again as

$$\sum_{\ell\in[k]} \left(\sum_{i\in[n]} \left(\nu_i^\ell - \mu_i^\ell + \sum_{j\in C_\ell} a_{ji}\lambda_j \right) x_i \right) \leq \sum_{\ell\in[k]} \left(\sum_{i\in[n]} \nu_i^\ell + \sum_{j\in C_\ell} b_j\lambda_j \right),$$

with $\mu_i^\ell = \left(\sum_{j\in C_\ell} a_{ji}\lambda_j\right)^+$ for $\pi_i = 0$ and $\mu_i^\ell = 0$ for $\pi_i > 0$ as well as $\nu_i^\ell = \left(-\sum_{j\in C_\ell} a_{ji}\lambda_j\right)^+$.

Thus, the above inequality decomposes into the k inequalities (2), all of which are recyclable since $\nu_i^\ell - \mu_i^\ell + \sum_{j\in C_\ell} a_{ji}\lambda_j \geq 0$ holds by the definition of ν^ℓ, μ^ℓ. □

In the special case where all constraints are recyclable, the graph G from the proposition above contains no edges. Thus, we don't have to consider any combinations of constraints but can solely rely on recycling constraints as in Sect. 4.1.

Corollary 2 *Let all constraints in $Ax \leq b$ be recyclable. Then every inequality that is recycled from a valid inequality for $\mathcal{P}^{NOM}$ is dominated by the recycled inequalities from $\sum_{i\in I} a_{ji}x_i \leq b_j$ for $j \in [m]$ and $I \subseteq [n]$.*

In the case where we have $\mathcal{P}^{\text{NOM}} = \mathcal{C}^{\text{NOM}}$ and all constraints are recyclable, the above corollary shows together with Proposition 1 that we can separate inequalities of the form $\pi_0 z + \sum_{i\in[n]} \pi_i p_i \geq \sum_{i\in[n]} \pi_i \hat{c}_i x_i$ exactly in linear time. Given such favorable conditions, one might ask whether we even obtain the convex hull $\mathcal{C}^{\text{ROB}}$ by separating recycled inequalities. This seems especially plausible since $\mathcal{P}^{\text{NOM}} = \mathcal{C}^{\text{NOM}}$ implies that ROB can be solved in polynomial time according to the famous result of Bertsimas and Sim [6]. However, the following example shows that we do not obtain $\mathcal{C}^{\text{ROB}}$ for a robust bipartite matching problem, although all constraints are recyclable in this case and we have $\mathcal{P}^{\text{NOM}} = \mathcal{C}^{\text{NOM}}$ for the bipartite matching polytope [28].

Example 5 Consider a bipartite graph $G = (V, E)$ with $V = \{1, 2\} \cup \{3, 4\}$ and $E = \{e_1, \ldots, e_4\} = \{\{1, 3\}, \{1, 4\}, \{2, 3\}, \{2, 4\}\}$. The convex hull of the robust weighted matching problem with variables $x_i \in \{0, 1\}$ and deviations $\hat{c}_i = i$ for every

$e_i \in E$ is

$$\mathcal{C}^{\text{ROB}} = \text{conv}\left(\left\{(x, p, z) \in \{0,1\}^4 \times \mathbb{R}^5_{\geq 0} \;\middle|\; \begin{array}{l} x_1 + x_2 \leq 1 \\ x_3 + x_4 \leq 1 \\ x_1 + x_3 \leq 1 \\ x_2 + x_4 \leq 1 \\ p_i + z \geq \hat{c}_i x_i \quad \forall i \in [4] \end{array}\right\}\right).$$

We use PORTA [29] to compute a representation of $\mathcal{C}^{\text{ROB}}$ for this problem and see that the inequality $p_2 + p_3 + z \geq 2x_2 + x_3$ is facet-defining for $\mathcal{C}^{\text{ROB}}$. The validity is easily verified, because the inequality is implied by $p_2 + z \geq 2x_2$ for $x_3 = 0$ and by $p_3 + z \geq 3x_3$ for $x_3 = 1$ due to $2x_2 + x_3 \leq 3 = 3x_3$. To see that it is also facet-defining, one verifies that the following $\dim\left(\mathcal{C}^{\text{ROB}}\right) = 9$ solutions are affinely independent and satisfy the inequality with equality

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ z \end{pmatrix} \in \left\{ \begin{pmatrix} 0\\0\\0\\0\\0\\0\\0\\0\\0 \end{pmatrix}, \begin{pmatrix} 1\\0\\0\\0\\1\\0\\0\\0\\0 \end{pmatrix}, \begin{pmatrix} 1\\0\\0\\0\\2\\0\\0\\0\\0 \end{pmatrix}, \begin{pmatrix} 0\\1\\0\\0\\0\\2\\0\\0\\0 \end{pmatrix}, \begin{pmatrix} 0\\0\\1\\0\\0\\0\\3\\0\\0 \end{pmatrix}, \begin{pmatrix} 0\\0\\0\\1\\0\\0\\0\\4\\0 \end{pmatrix}, \begin{pmatrix} 0\\1\\0\\0\\0\\0\\0\\0\\2 \end{pmatrix}, \begin{pmatrix} 0\\0\\1\\0\\0\\0\\0\\0\\3 \end{pmatrix}, \begin{pmatrix} 0\\1\\1\\0\\0\\0\\1\\0\\2 \end{pmatrix} \right\}.$$

Note that $p_2 + p_3 + z \geq 2x_2 + x_3$ is not a recycled inequality, since the quotient of the coefficients of x_3 and p_3 is not $\hat{c}_3 = 3$.

In our computational study, we show that recycled inequalities nevertheless almost completely close the gap between optimal integer and continuous solutions for some robust bipartite matching problems. This highlights the large potential of recycled inequalities when all constraints are recyclable and $\mathcal{P}^{\text{NOM}} = \mathcal{C}^{\text{NOM}}$ holds.

Even if not all constraints are recyclable, an optimal solution to SLP often corresponds to an already recyclable constraint in $Ax \leq b$ in practice. Hence, we observe that it is beneficial to first check whether we can separate violated recycled inequalities from constraints, as described in Sect. 4.1. Only if none of these are violated, we solve SLP to check whether there exists a violated recycled inequality from a combined inequality. We will see in our tests on robust instances generated from the MIPLIB 2017 that solving SLP sometimes yields very strong recycled inequalities, even if recycling the constraints in $Ax \leq b$ has no effect at all (cf. Sec 6.6).

5 Partially recycling of non-recyclable inequalities

Let $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ be a non-recyclable valid inequality for $\mathcal{C}^{\text{NOM}}$. In the previous section, we transformed such inequalities into $\sum_{i\in[n]:\pi_i>0} \pi_i x_i \leq \pi_0 - \sum_{i\in[n]:\pi_i<0} \pi_i$

for recycling, by tentatively fixing $x_i = 1$ for all $i \in [n]$ with $\pi_i < 0$. Intuitively, the resulting recycled inequality seems to be unnecessarily weak if $\tilde{x}_i$ is actually (near to) zero for an $i \in [n]$ with $\pi_i < 0$ and a fractional solution $(\tilde{x}, \tilde{p}, \tilde{z}) \in \mathcal{P}^{\text{ROB}}$ to be cut off. To resolve this, we propose another procedure, using the recyclable part of generally non-recyclable inequalities. For the remainder of this section, let $S^+ = \{i \in [n] | \pi_i > 0\}$ and $S^- = \{i \in [n] | \pi_i < 0\}$ be the sets of indices corresponding to positive and negative coefficients, respectively.

Note that $\sum_{i \in S^+} \pi_i x_i \leq \pi_0$ is a recyclable inequality for the restricted nominal solution space $\left\{x \in \mathcal{C}^{\text{NOM}} \middle| x_i = 0 \, \forall i \in S^-\right\}$, and can thus be recycled to a valid inequality for $\left\{(x, p, z) \in \mathcal{C}^{\text{ROB}} \middle| x_i = 0 \, \forall i \in S^-\right\}$. In order to obtain a valid inequality for $\mathcal{C}^{\text{ROB}}$, we can *lift* the fixed variables into the recycled inequality. For this, we compute *lifting coefficients* $\alpha_i \in \mathbb{R}$ for $i \in S^-$ such that

$$\pi_0 z + \sum_{i \in S^+} \pi_i p_i \geq \sum_{i \in S^+} \pi_i \hat{c}_i x_i + \sum_{i \in S^-} \alpha_i x_i$$

is a valid inequality for $\mathcal{C}^{\text{ROB}}$.

In general, one wants to choose maximal lifting coefficients α, so that the *lifted inequality* is as strong as possible. Whether one obtains a facet-defining inequality is not trivial to say, as this not only depends on the inequality to be lifted and the maximality of the lifting coefficients but also on the considered polyhedron. However, roughly speaking, lifting is more likely to yield a facet-defining inequality if the original inequality is facet-defining for the restricted solution space, where the variables to be lifted are fixed to zero [30–32]. Using Theorem 3, we can state in which case this applies for our recycled inequalities.

Corollary 3 *Let $\sum_{i \in [n]} \pi_i x_i \leq \pi_0$ be a valid inequality for $\mathcal{C}^{NOM}$ with $\pi_0 \geq 0$. The recycled inequality $\pi_0 z + \sum_{i \in S^+} \pi_i p_i \geq \sum_{i \in S^+} \pi_i \hat{c}_i x_i$ is facet-defining for the restricted solution space $\left\{(x, p, z) \in \mathcal{C}^{ROB} \middle| x_i = 0 \, \forall i \in S^-\right\}$ if $\hat{c}_i > 0$ holds for all $i \in S^+$, i.e., it is uncertainty-exclusive on $\left\{x_i \middle| i \in S^+\right\}$, and*

$$\dim\left(\text{proj}_{S^+}\left(\left\{x \in F^{NOM}(\pi) \middle| x_i = 0 \, \forall i \in S^-\right\}\right)\right) = \left|S^+\right| - 1.$$

The approach of fixing, recycling, and lifting is especially promising if the original inequality meets the criteria of the corollary above and if we are able to compute high lifting coefficients α. Computing maximal lifting coefficients often involves solving multiple $\mathcal{NP}$-hard optimization problems. For example, when only lifting the variable x_ℓ into the recycled inequality $\pi_0 z + \sum_{i \in S^+} \pi_i p_i \geq \sum_{i \in S^+} \pi_i \hat{c}_i x_i$, then we need to solve

$$\begin{aligned} \min \; & \pi_0 z + \sum_{i \in S^+} \pi_i p_i - \sum_{i \in S^+} \pi_i \hat{c}_i x_i \\ \text{s.t.} \; & (x, p, z) \in \mathcal{C}^{\text{ROB}}, x_\ell = 1. \end{aligned}$$

That is, we minimize the slack of the inequality to be lifted while fixing $x_\ell = 1$. This (in our case non-positive) slack is then the maximal lifting coefficient of x_ℓ. The

theoretical complexity of lifting implies the need for an efficient heuristic approach. The following proposition shows how to compute lifting coefficients by solving a sequence of easy fractional knapsack problems.

Proposition 4 *Let $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ be a valid inequality for $\mathcal{C}^{NOM}$ with $\pi_0 \geq 0$. Consider the fractional knapsack problem*

$$f(y) = \max\left\{ \sum_{i\in S^+} \pi_i \hat{c}_i x_i \,\middle|\, \sum_{i\in S^+} \pi_i x_i \leq y,\, x \in [0,1]^n \right\}$$

for a capacity $y \in \mathbb{R}_{\geq 0}$ and let $\alpha_i = f(\pi_0) - f(\pi_0 - \pi_i)$ for $i \in S^-$. Then

$$\pi_0 z + \sum_{i\in S^+} \pi_i p_i \geq \sum_{i\in S^+} \pi_i \hat{c}_i x_i + \sum_{i\in S^-} \alpha_i x_i$$

is a valid inequality for $\mathcal{C}^{ROB}$.

Proof We sequentially lift the variables $\{x_{i_1}, \ldots, x_{i_k}\} = \{x_i | i \in S^-\}$ and show via induction over $\ell \in \{0, \ldots, k\}$ that

$$\pi_0 z + \sum_{i\in S^+} \pi_i p_i \geq \sum_{i\in S^+} \pi_i \hat{c}_i x_i + \sum_{j\in[\ell]} \alpha_{i_j} x_{i_j},$$

with $[0] = \emptyset$ and $\alpha_{i_j} = f(\pi_0) - f(\pi_0 - \pi_{i_j})$, is a valid inequality for the restricted solution space $\{(x, p, z) \in \mathcal{C}^{\text{ROB}} | x_{i_{\ell+1}} = \cdots = x_{i_k} = 0\}$. The induction statement holds true for the base case $\ell = 0$, where no variables are lifted. Now, consider some $\ell \in [k]$ and assume that the induction statement holds true for $\ell - 1$. Then the following lifting problem yields the maximal lifting coefficient for variable x_{i_ℓ}

$$\begin{aligned} \min\ & \pi_0 z + \sum_{i\in S^+} \pi_i p_i - \sum_{i\in S^+} \pi_i \hat{c}_i x_i - \sum_{j\in[\ell-1]} \alpha_{i_j} x_{i_j} \\ \text{s.t. } & x_{i_\ell} = 1,\ x_{i_{\ell+1}} = \cdots = x_{i_k} = 0, \\ & (x, p, z) \in \mathcal{C}^{\text{ROB}}. \end{aligned}$$

We relax the lifting problem in order to obtain a valid but not necessarily maximal lifting coefficient. We can assume that the bilinear constraints $p_i + x_i z \geq \hat{c}_i x_i$ are met, since it is sufficient to only consider integer-feasible solutions for the lifting problem. In the relaxation, we only consider the bilinear constraints as well as the reduced constraint $\sum_{i\in S^+} \pi_i x_i \leq \pi_0 - \pi_{i_\ell} - \sum_{j\in[\ell-1]} \pi_{i_j} x_{i_j}$ obtained from the original constraint $\sum_{i\in[n]} \pi_i x_i \leq \pi_0$ and $x_{i_\ell} = 1$ as well as $x_{i_{\ell+1}} = \cdots = x_{i_k} = 0$. Furthermore, we allow all variables but $x_{i_1}, \ldots, x_{i_\ell}$ to be fractional. By assuming that $S \subseteq [\ell - 1]$ defines an optimal choice for the already lifted variables $x_{i_1}, \ldots, x_{i_{\ell-1}}$,

with $x_{i_j} = 1$ iff $j \in S$, we obtain the following relaxed lifting problem

$$\text{(RLP)} \qquad \begin{aligned} \min\ & \pi_0 z + \sum_{i \in S^+} \pi_i p_i - \sum_{i \in S^+} \pi_i \hat{c}_i x_i - \sum_{j \in S} \alpha_{i_j} \\ \text{s.t.}\ & \sum_{i \in S^+} \pi_i x_i \le y \\ & p_i + x_i z \ge \hat{c}_i x_i && \forall i \in S^+ \\ & x \in [0,1]^{S^+},\ p \in \mathbb{R}^{S^+}_{\ge 0}, z \in \mathbb{R}_{\ge 0} \end{aligned}$$

with $y = \pi_0 - \pi_{i_\ell} - \sum_{j \in S} \pi_{i_j}$. We will first show that the optimal solution value of the RLP equals $f(\pi_0) - f(y) - \sum_{j \in S} \alpha_{i_j}$ for all $y \ge \pi_0$ and $S \subseteq [\ell - 1]$. Afterwards, we show that $S = \emptyset$ is an optimal choice, which proves that $f(\pi_0) - f\left(\pi_0 - \pi_{i_\ell}\right) = \alpha_{i_\ell}$ is a feasible lifting coefficient.

Since $f(y)$ is a fractional knapsack problem with capacity y, values $\pi_i \hat{c}_i$, and weights π_i, we can sort the variables with respect to $\frac{\pi_i \hat{c}_i}{\pi_i} = \hat{c}_i$ and greedily fill the knapsack until the capacity is reached. Let x^* be such an optimal greedy solution to $f(y)$. We show that x^*, together with appropriate p^*, z^*, is also an optimal solution to RLP. For this, let (x, p, z) be an optimal solution to RLP. We can assume $p_i = \left(\hat{c}_i - z\right)^+ x_i$, and thus obtain for the objective of RLP

$$\pi_0 z + \sum_{i \in S^+} \pi_i p_i - \sum_{i \in S^+} \pi_i \hat{c}_i x_i - \sum_{j \in S} \alpha_{i_j} = \pi_0 z - \sum_{i \in S^+} \pi_i \min\left\{\hat{c}_i, z\right\} x_i - \sum_{j \in S} \alpha_{i_j}.$$

Hence, for fixed z, we have a fractional knapsack problem with values $\pi_i \min\left\{\hat{c}_i, z\right\}$ and weights π_i. Since sorting with respect to $\hat{c}_i$ also yields a sorting with respect to $\frac{\pi_i \min\{\hat{c}_i, z\}}{\pi_i} = \min\left\{\hat{c}_i, z\right\}$, the above greedy solution x^* is again optimal.

Now, we choose

$$z^* = \min\left\{ z \in \left\{0, \hat{c}_1, \ldots, \hat{c}_n\right\} \,\middle|\, \sum_{i \in S^+, \hat{c}_i > z} \pi_i x_i^* \le \pi_0 \right\}$$

together with $p_i^* = \left(\hat{c}_i - z^*\right)^+ x_i^*$. We first show that the value of this solution equals $f(\pi_0) - f(y) - \sum_{j \in S} \alpha_{i_j}$ and show afterwards that it is optimal.

If $\sum_{i \in S^+} \pi_i \le \pi_0$ holds, then we have $z^* = 0$, and thus

$$\begin{aligned} & \pi_0 z^* + \sum_{i \in S^+} \pi_i p_i^* - \sum_{i \in S^+} \pi_i \hat{c}_i x_i^* - \sum_{j \in S} \alpha_{i_j} \\ & = \sum_{i \in S^+} \pi_i \left(\hat{c}_i - 0\right)^+ x_i^* - \sum_{i \in S^+} \pi_i \hat{c}_i x_i^* - \sum_{j \in S} \alpha_{i_j} = -\sum_{j \in S} \alpha_{i_j}. \end{aligned}$$

Since all items $i \in S^+$ already fit into the knapsack of capacity π_0, increasing the capacity up to $y \geq \pi_0$ has no effect on the solution value. Hence, we have $f(\pi_0) = f(y)$ and thus $f(\pi_0) - f(y) - \sum_{j \in S} \alpha_{i_j} = -\sum_{j \in S} \alpha_{i_j}$.

Otherwise, if $\sum_{i \in S^+} \pi_i > \pi_0$ holds, we assume $0 =: \hat{c}_0 \leq \hat{c}_1 \leq \cdots \leq \hat{c}_n$ and let $j^* \in \{0, \ldots, n\}$ be the smallest index such that $\sum_{i \in S^+ : i > j^*} \pi_i \leq \pi_0$. It follows $z^* = \hat{c}_{j^*}$ by the definition of z^* and we can assume $x_i^* = 1$ for all $i \in S^+$ with $i > j^*$ by the definition of our greedy solution. This implies that (x^*, p^*, z^*) is a solution to RLP of value

$$\begin{aligned}
&\pi_0 z^* + \sum_{i \in S^+} \pi_i p_i^* - \sum_{i \in S^+} \pi_i \hat{c}_i x_i^* - \sum_{j \in S} \alpha_{i_j} \\
&= \pi_0 \hat{c}_{j^*} + \sum_{i \in S^+ : i > j^*} \pi_i \left(\hat{c}_i - \hat{c}_{j^*} \right) - f(y) - \sum_{j \in S} \alpha_{i_j} \\
&= \left(\pi_0 - \sum_{i \in S^+ : i > j^*} \pi_i \right) \hat{c}_{j^*} + \sum_{i \in S^+ : i > j^*} \pi_i \hat{c}_i - f(y) - \sum_{j \in S} \alpha_{i_j} \\
&= f(\pi_0) - f(y) - \sum_{j \in S} \alpha_{i_j}.
\end{aligned}$$

Here, the last equation holds since $x_i^* = 1$ for all $i \in S^+$ with $i > j^*$ and

$$x_{j^*}^* = \frac{\pi_0 - \sum_{i \in S^+ : i > j^*} \pi_i}{\pi_{j^*}}$$

is an optimal solution to $f(\pi_0)$.

To see that the choice of p^*, z^* is actually optimal, first consider $z' > z^*$ and p' with $p_i' \geq \left(\hat{c}_i - z' \right) x_i^*$. By definition of z^*, we have $\sum_{i \in S^+ : \hat{c}_i > z^*} \pi_i x_i^* \leq \pi_0$, and thus

$$\begin{aligned}
\pi_0 z^* + \sum_{i \in S^+} \pi_i p_i^* &= \pi_0 z^* + \sum_{i \in S^+ : \hat{c}_i > z^*} \pi_i \left(z' - z^* + \hat{c}_i - z' \right) x_i^* \\
&\leq \pi_0 z^* + \pi_0 \left(z' - z^* \right) + \sum_{i \in S^+ : \hat{c}_i > z^*} \pi_i \left(\hat{c}_i - z' \right) x_i^* \\
&\leq \pi_0 z' + \sum_{i \in S^+ : \hat{c}_i > z'} \pi_i \left(\hat{c}_i - z' \right) x_i^* \\
&\leq \pi_0 z' + \sum_{i \in S^+} \pi_i p_i'.
\end{aligned}$$

Second, consider $z' < z^*$ with an appropriate p'. Due to the minimality of z^* and $0 \leq z' < z^*$, we have $\sum_{i \in S^+ : \hat{c}_i \geq z^*} \pi_i x_i^* > \pi_0$, and thus

$$\pi_0 z^* + \sum_{i \in S^+} \pi_i p_i^* = \pi_0 \left(z' + z^* - z' \right) + \sum_{i \in S^+ : \hat{c}_i \geq z^*} \pi_i \left(\hat{c}_i - z^* \right) x_i^*$$

$$
\begin{aligned}
&< \pi_0 z' + \sum_{i \in S^+ : \hat{c}_i \geq z^*} \pi_i \left(z^* - z' + \hat{c}_i - z^* \right) x_i^* \\
&\leq \pi_0 z' + \sum_{i \in S^+ : \hat{c}_i > z'} \pi_i \left(\hat{c}_i - z' \right) x_i^* \\
&\leq \pi_0 z' + \sum_{i \in S^+} \pi_i p_i',
\end{aligned}
$$

which shows the optimality of z^* and p^*.

We have shown that $f(\pi_0) - f\left(\pi_0 - \pi_{i_\ell} - \sum_{j \in S} \pi_{i_j}\right) - \sum_{j \in S} \alpha_{i_j}$ is the optimal value of RLP for some $S \subseteq [\ell - 1]$. Thus, it only remains to show that $S = \emptyset$ is optimal. To see this, note that f is submodular due to the diminishing utility of additional capacity. That is, we have $f\left(y' + \varepsilon\right) - f\left(y'\right) \geq f(y + \varepsilon) - f(y)$ for $y' \leq y$ and $\varepsilon \geq 0$. Since all π_{i_j} are negative, this implies

$$
\begin{aligned}
\sum_{j \in S} \left(f\left(\pi_0 - \pi_{i_j}\right) - f(\pi_0) \right) &\geq \sum_{j \in S} f\left(\pi_0 - \pi_{i_j} - \sum_{\substack{j^* \in S: \\ j^* < j}} \pi_{i_{j^*}} \right) - f\left(\pi_0 - \sum_{\substack{j^* \in S: \\ j^* < j}} \pi_{i_{j^*}} \right) \\
&= f\left(\pi_0 - \sum_{j \in S} \pi_{i_j} \right) - f(\pi_0),
\end{aligned}
$$

where the equality follows from the second term being a telescope sum. Using $\alpha_{i_j} = f\left(\pi_0 - \pi_{i_j}\right) - f(\pi_0)$ for all $j \in [\ell - 1]$, we obtain

$$
\begin{aligned}
&f(\pi_0) - f\left(\pi_0 - \pi_{i_\ell} - \sum_{j \in S} \pi_{i_j} \right) - \sum_{j \in S} \alpha_{i_j} \\
&= f(\pi_0) - f\left(\pi_0 - \pi_{i_\ell} - \sum_{j \in S} \pi_{i_j} \right) + \sum_{j \in S} \left(f\left(\pi_0 - \pi_{i_j}\right) - f(\pi_0) \right) \\
&\geq f\left(\pi_0 - \sum_{j \in S} \pi_{i_j} \right) - f\left(\pi_0 - \pi_{i_\ell} - \sum_{j \in S} \pi_{i_j} \right) \\
&\geq f(\pi_0) - f\left(\pi_0 - \pi_{i_\ell}\right),
\end{aligned}
$$

which proves the statement. □

In practice, when cutting off a fractional solution $(\tilde{x}, \tilde{p}, \tilde{z}) \in \mathcal{P}^{\mathrm{ROB}}$ with a lifted recycled inequality, we again drop all variables x_i from the inequality with $\pi_i > 0$ and $\hat{c}_i \tilde{x}_i < \tilde{p}_i$, as these negatively impact the violation of the recycled inequality. We do this before lifting the variables x_i with $\pi_i < 0$, as doing so restricts the lifting problem RLP, and thus yields potentially better lifting coefficients.

Note that we require $\pi_0 \geq 0$ in the above proposition, as a negative coefficient of the unbounded variable z would imply infinite lifting coefficients α for obtaining a valid inequality. Hence, if the original inequality has $\pi_0 < 0$, we first have to tentatively fix $x_i = 1$ for some $i \in S^-$ to obtain a non-negative right-hand side. This raises the question of which variables should be fixed and which should be lifted. Moreover, even if $\pi_0 \geq 0$ holds, it is reasonable to check whether lifting or fixing a variable yields a higher violation. For example, when cutting off a fractional solution $(\tilde{x}, \tilde{p}, \tilde{z})$ with $\tilde{z} = 0$, we obtain a higher violation by fixing $x_i = 1$ for all $i \in S^-$ and recycling $\sum_{i \in S^+} \pi_i x_i \leq \pi_0 - \sum_{i \in S^-} \pi_i$, as the higher coefficient of z in the recycled inequality is irrelevant in this case. Contrary to that, if $\tilde{z} > 0$ and $\tilde{x}_i = 0$ hold, then it is preferable to lift x_i.

Since we add $\alpha_i \tilde{x}_i$ to the violation when lifting and $\pi_i \tilde{z}$ when fixing, we want to lift those variables with $\alpha_i \tilde{x}_i > \pi_i \tilde{z}$. However, if we decide to fix $x_i = 1$ for $i \in S \subseteq S^-$, then we obtain a new inequality with a greater right-hand side $\pi_0 - \sum_{i \in S} \pi_i$, which influences the lifting coefficients $\alpha_j(S) = f\left(\pi_0 - \sum_{i \in S} \pi_i\right) - f\left(\pi_0 - \sum_{i \in S} \pi_i - \pi_j\right)$ of the other variables x_j, and thus our lifting decision. Let $\{i_1, \ldots, i_k\} = S^-$ such that $\tilde{x}_{i_1} \geq \cdots \geq \tilde{x}_{i_k}$. Since variables with higher solution values $\tilde{x}_i$ are less preferable for lifting, it is reasonable to assume that a good decision for S consists of variables $x_{i_1}, \ldots, x_{i_j}$ for some $j \in [k]$. Therefore, we first set $S = \emptyset$ and assume that all variables will be lifted. Afterwards, we greedily decide for $i \in \{i_1, \ldots, i_k\}$ whether x_i should better not be lifted and instead added to S. For this, we check whether

$$\pi_i \tilde{z} + \sum_{j \in \{i_1, \ldots, i_k\} \setminus (S \cup \{i\})} \alpha_j(S \cup \{i\}) \tilde{x}_j \geq \sum_{j \in \{i_1, \ldots, i_k\} \setminus S} \alpha_j(S) \tilde{x}_j$$

holds, i.e., whether the change of the violation is positive when not lifting x_i. Note that the values $\alpha_j(S \cup \{i\})$ can be updated efficiently from $\alpha_j(S)$ by greedily extending the solutions of the corresponding fractional knapsack problems.

We use this approach in the following computational study, which shows the practical relevance of recycling in general and also indicates the potential of partially recycling.

6 Computational study

In this section, we assess the performance of recycled inequalities computationally. We first discuss numerical pitfalls that can occur in practice when using recycled inequalities and present a remedy for these. Afterwards, we lay out our methodology for measuring an algorithm's performance. Using this methodology, we test different aspects of recycling inequalities for different classes of robust combinatorial optimization problems.

With the study of the robust independent set problem in Sect. 6.3, we examine the contribution of recycling problem specific cuts. For this, we heuristically separate recycled clique inequalities, which are always facet-defining for the robust problem (cf. Sect. 3). In Sect. 6.4, we test the recycling of model constraints for the robust bipar-

tite matching problem. Since the standard formulation of the nominal version consists exclusively of recyclable inequalities and also describes the convex hull $\mathcal{C}^{\text{NOM}}$ [28], every non-dominated recycled inequality corresponds to a model constraint (cf. Corollary 2). This allows us to test the influence of recycled inequalities to their limits. In Sect. 6.5, we consider the robust bipartite matching problem with penalties, in which we allow the violation of matching constraints at the cost of a penalty. Using the adapted model constraints, which are no longer recyclable, we test the partially recycling of non-recyclable inequalities.

After considering the combinatorial problems above, we evaluate the practical relevance of recycling on a broad set of robustified real world instances from the MIPLIB 2017 [20], which have been used for benchmarking in [11]. For these instances, we also test the generic approach of separating recycled inequalities via solving SLP. Finally, we close our study by comparing the performance of our recycling approach with the performance of different approaches from the literature.

All algorithms have been implemented in Java 11 and tests are performed on a single core of a Linux machine with an Intel® Core™ i7-5930K CPU @ 3.50GHz, with 2 GB RAM reserved for each calculation. All LPs and MILPs are solved using Gurobi version 9.5.0 [33] in single thread mode and all other settings at default. Furthermore, we use a time limit of 3600 seconds for each algorithm and instance.

All implemented algorithms [21] and generated test instances [22] are freely available online.

6.1 Dealing with numerical issues

MILP solvers relying on numeric arithmetic have to face the threat of numerical instability, leading to inconsistent results. One source of numerical instability is a constraint matrix $Ax \leq b$ with a high range in the order of magnitude of coefficients a_{ij}, e.g., with $a_{11} = 10^{-4}$ and $a_{12} = 10^{10}$. In fact, the Gurobi documentation recommends that the range of coefficients in the constraint matrix should be within six orders of magnitude [34]. In the case of recycled inequalities $\pi_0 z + \sum_{i\in[n]} \pi_i p_i \geq \sum_{i\in[n]} \pi_i \hat{c}_i x_i$, the coefficients $\pi_i \hat{c}_i$ on the right-hand side might violate this desirable property if $\hat{c}_i$ and π_i are both very large or both very small. As a consequence, we observed for three instances in our computational study on the MIPLIB that sub-optimal solutions were reported as optimal.

To tackle this problem, we scale the deviations $\hat{c}_i$ as well as the variables p, z in an attempt to reduce the range of coefficients in the recycled inequalities. Let $\hat{c}_{\max} = \max\{\hat{c}_1, \ldots, \hat{c}_n\}$ and $\hat{c}_{\min} = \min\{\hat{c}_i \,|\, i \in [n], \hat{c}_i > 0\}$ be the maximum and minimum (proper) deviations. If $\hat{c}_{\max}$ is very large and $\hat{c}_{\min}$ simultaneously very small, then our problem is predisposed to be numerically unstable anyway. However, if both are either very large or very small, then we can scale the deviations such that $\hat{c}_{\max}$ and $\hat{c}_{\min}$ are closer to one. For this, we divide all deviations $\hat{c}_i$ by $\lambda = \sqrt{\hat{c}_{\max}\hat{c}_{\min}}$. This implies $\frac{\hat{c}_{\max}}{\lambda}\frac{\hat{c}_{\min}}{\lambda} = 1$, i.e., the scaled maximum and minimum deviation have the same distance to one in orders of magnitudes. To compensate this change, we multiply z and p in the objective function with λ. Thus, our new problem, which is equivalent to

ROB, reads

$$\begin{aligned} \min\ & \lambda\Gamma z + \sum_{i\in[n]} (c_i x_i + \lambda p_i) \\ \text{s.t. } & Ax \le b \\ & p_i + z \ge \frac{\hat{c}_i}{\lambda} x_i && \forall i \in [n] \\ & x \in \{0,1\}^n,\ p \in \mathbb{R}^n_{\ge 0},\ z \in \mathbb{R}_{\ge 0} \end{aligned}$$

and the recycled inequalities are

$$\pi_0 z + \sum_{i\in[n]} \pi_i p_i \ge \sum_{i\in[n]} \pi_i \frac{\hat{c}_i}{\lambda} x_i .$$

This small change resolves the observed issues for the MIPLIB instances. For comparability, we will always use the scaled problem when solving ROB. However, for the sake of simplicity, we will only write down the non-scaled problem in the following sections.

6.2 Performance indicators

Rating the performance of generic algorithms is not trivial, as different use cases imply different requirements for an algorithm. While we aim to find an optimal solution as fast as possible for some practical problems, it is important to find any good solution within seconds for other problems. Therefore, we need performance indicators that appropriately reflect the spectrum of use cases.

To reflect the aim of solving problems as fast as possible to optimality, we consider the elapsed time required to solve an instance. As it is standard in the literature, we set the elapsed time to the time limit in case an algorithm is not able to solve an instance within this limit. Note that this favors algorithms that often hit the time limit.

In addition, we use the *primal-dual integral*, which was proposed by Berthold [35] with the aim that this metric "reflects the development of the solution quality over the complete optimization process". The primal-dual integral is defined to be the integral of the gap between the current primal and dual bound for each point in time. Since the optimality gap, as reported by, e.g., Gurobi, is in general not bounded and at the start even infinite, the primal-dual integral is defined over an adapted gap. Let $\overline{v}(t)$ be the primal bound and $\underline{v}(t)$ be the dual bound at time step t, with $\overline{v}(t) = \infty$ or $\underline{v}(t) = -\infty$ respectively if no bound is known. We define the step function

$$g(t) = \begin{cases} 1, & \text{if } \overline{v}(t) = \infty \text{ or } \underline{v}(t) = -\infty \text{ or } \underline{v}(t) \cdot \overline{v}(t) < 0, \\ 0, & \text{if } \underline{v}(t) = \overline{v}(t), \\ \frac{\overline{v}(t) - \underline{v}(t)}{\max\{|\overline{v}(t)|, |\underline{v}(t)|\}}, & \text{else,} \end{cases}$$

with respect to the piecewise constant bounds $\overline{v}(t), \underline{v}(t)$, for which we can easily compute the primal-dual integral

$$G(T) = \int_{t=0}^{T} g(t) \, \mathrm{d}\, t,$$

with T being the time at which the algorithm is terminated or finishes. The primal-dual integral reflects improvements of the gap over the whole computation process, and is thus a reasonable additional performance indicator alongside the computation time.

Since displaying our performance indicators for all algorithms and instances is impractical, we give aggregate values using the *shifted geometric mean* [36]. This is defined as $\left(\prod_{i=1}^{k} (v_i + s)^{1/k}\right) - s$ for values $v_1, \ldots, v_k \in \mathbb{R}_{\geq 0}$ and a *shifting parameter* $s \in \mathbb{R}_{\geq 0}$. In the following, we always use $s = 1$ second for computation times and $s = 100\%$ for primal-dual integrals, which corresponds to the integral of one second at maximum gap.

Besides computation times and primal-dual integrals, we will also report *integrality gaps*, that is the relative difference between the optimal integer solution value and the optimal continuous solution value, in order to compare the strength of the linear relaxation with and without recycled inequalities. Let v be the objective value of an optimal integer solution and v^{R} the objective value of an optimal continuous solution. Then we compute the integrality gap as $\frac{|v^{\mathrm{R}} - v|}{|v|}$ for $v \neq 0$. For $v = 0$, we define it to be zero if $v^{\mathrm{R}} = 0$ holds, and ∞ otherwise. In practice, if we are not able to compute the optimal objective value v, then we instead use the best primal bound computed by any approach. For v^{R}, we use the dual bound obtained by separating recycled inequalities for subsequent continuous relaxations until no violated inequalities are found or until we reach a total time limit of 3600 seconds. For aggregating integrality gaps, we use the shifted geometric mean with $s = 1\%$.

6.3 Robust independent set

To show the effect of recycling a class of well-known valid inequalities in a separation procedure, we consider the robust maximum weighted independent set problem on a graph G with nodes V and edges E. The robust counterpart of the standard formulation with decision variables $x_v \in \{0, 1\}$ for each $v \in V$ and *edge constraints* $x_v + x_w \leq 1$ for $\{v, w\} \in E$ reads

$$\begin{aligned}
\max \ & \sum_{v \in V} c_v x_v - \left(\Gamma z + \sum_{v \in V} p_v \right) \\
\text{s.t. } & x_v + x_w \leq 1 && \forall \{v, w\} \in E \\
& p_v + z \geq \hat{c}_v x_v && \forall v \in V \\
& x \in \{0, 1\}^V, \ p \in \mathbb{R}^V_{\geq 0}, z \in \mathbb{R}_{\geq 0}.
\end{aligned}$$

As seen in Sect. 3, recycling a clique inequality $\sum_{v \in Q} x_v \leq 1$ yields a facet-defining inequality for all cliques $Q \subseteq V$. We compare the separation of recycled clique inequalities in the root node of the branching tree against the robust default formulation $\mathcal{P}^{\text{ROB}}$, which solely uses the constraints $p_i + z \geq \hat{c}_i x_i$. For this, we use Gurobi's callback to add the recycled inequalities as user cuts [33]. Every time Gurobi invokes the callback in the root node and reports a current optimal fractional solution $(\tilde{x}, \tilde{p}, \tilde{z}) \in \mathcal{P}^{\text{ROB}}$, we heuristically separate cliques $Q \subseteq V$ for which the recycled inequality $z + \sum_{v \in Q} p_v \geq \sum_{v \in Q} \hat{c}_v x_v$ is violated. We do so as in Sect. 4.2, that is, we heuristically solve maximum weighted clique problems on the graph $G = (V, E)$ with weights $\hat{c}_v \tilde{x}_v - \tilde{p}_v$. To separate many recycled inequalities at once, we extend each node $v \in V$ with $\hat{c}_v \tilde{x}_v - \tilde{p}_v > 0$ greedily to a clique $Q_v \subseteq V$ with $v \in Q_v$. For this, we start with $Q_v = \{v\}$ and then iteratively add $v' \in N(Q_v)$ such that $\hat{c}_{v'} \tilde{x}_{v'} - \tilde{p}_{v'}$ is maximal and non-negative. Finally, we return the corresponding recycled inequality to Gurobi as a user cut if its violation is positive.

As a basis for our instances, we use the graphs of the second DIMACS implementation challenge on the clique problem [37]. Of the 66 DIMACS graphs, we choose the 46 graphs that have at most 500 nodes, as otherwise the nominal problem is already very hard. For each node $v \in V$, we generate independent and uniformly distributed values $c_v \in \{900, \ldots, 1000\}$ and correlated deviations $\hat{c}_v = \lceil \xi_v c_v \rceil$, with $\xi_v \in [0.45, 0.55]$ being an independent and uniformly distributed random variable. Since robust problems tend to be hard for Γ being somewhere around half the number of variables with $x_i = 1$ [11], we greedily compute a maximal independent set $S \subseteq V$ and define $\Gamma = \left\lfloor \frac{|S|}{2} \right\rfloor$. Using this procedure, we randomly generate five robust independent set problems for each of the 46 DIMACS graphs, leaving us with 230 robust instances.

We show computational results for the default formulation (DEF) and the separation of recycled clique inequalities (RECsepClq) in Table 1. The table shows the number of instances that could not be solved within the time limit (timeout), the shifted geometric mean of the computation times (time), the shifted geometric mean of the primal-dual integrals (P-D integral) and the shifted geometric mean of the integrality gaps (int gap). We show results once with Gurobi's own cutting planes enabled and once with them disabled (GCuts).

We see that the shifted geometric mean of the integrality gaps is reduced absolutely by roughly 220% from 1427.09% to 1206.91% when using recycled clique inequalities. While the absolute reduction of the integrality gap is quite impressive, the relative reduction does not adequately reflect the strength of the recycled inequalities. This is due to the large integrality gap of the nominal problem, which constitutes a major part of the total gap. To reduce the integrality gap of the nominal problem, we test an additional formulation for the independent set problem. Here, we replace every constraint $x_v + x_w \leq 1$ for an edge $\{v, w\} \in E$ with a constraint $\sum_{v \in Q} x_v \leq 1$ for a clique $Q \subseteq V$ with $\{v, w\} \subseteq Q$. This *clique formulation* has a much tighter linear relaxation compared to the previous *edge formulation*, and thus reduces the contribution of the nominal problem to the integrality gap. Indeed, Table 1 shows that separating recycled clique inequalities reduces the integrality gap by more than half when using the clique formulation. Figure 1 gives a more detailed view of the improvement, by showing for how many instances the integrality gap is reduced by at least a specific percentage.

Table 1 Computational results for 230 instances of the robust maximum weighted independent set problem

Formulation	GCuts	DEF				RECsepClq			
		Timeout	Time	P-D integral	Int Gap (%)	Timeout	Time	P-D integral	Int Gap (%)
Edge	Enable	22	26.13	14.12	1427.09	21	33.22	16.99	1206.91
	Disable	42	51.23	22.91		20	20.93	11.72	
Clique	Enable	61	116.13	67.66	135.50	64	135.15	77.06	56.49
	Disable	74	168.47	83.30		53	82.07	49.97	

We use different nominal formulations and test with Gurobi's own cuts enabled or disabled

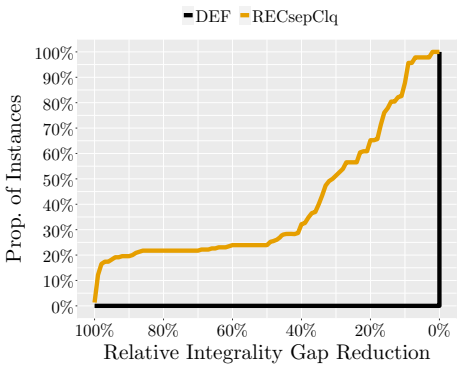

Fig. 1 Cumulative distribution of integrality gap reductions using the clique formulation

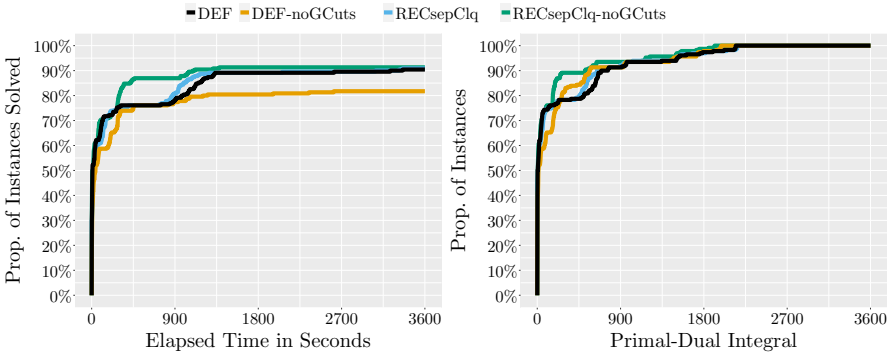

Fig. 2 Cumulative distribution of computation times and primal-dual integrals when using the edge formulation

Here, we see that recycling cliques reduces the integrality gap by at least 30% for more than 50% of all instances. Moreover, we have a reduction of 90% for almost 20% of the instances.

Apart from the analysis of the integrality gap, the clique formulation is not of practical interest, as Gurobi seems to be better trained on the edge formulation. Table 1 shows for the edge formulation that we solve one more instance when recycling clique inequalities but have an increase in the computation time and the primal-dual integral. This seems to be due to some interference with Gurobi's own cutting planes. When disabling Gurobi's cutting planes, then recycling is much better than using the default formulation, as it approximately halves the computation time and the primal-dual integral. In fact, disabling Gurobi's cuts and using recycled clique inequalities (RECsepClq-noGCuts) is the overall best performing approach, solving the most instances in the least amount of computation time. This is supported by Fig. 2, which shows for each approach the proportion of instances whose computation time or primal-dual integral are below a specific value. While DEF, RECsepClq, and RECsepClq-noGCuts solve roughly the same number of instances within the first 280 s and up to a primal-dual integral of 160, RECsepClq-noGCuts clearly performs better afterwards on the harder instances.

6.4 Robust bipartite matching

To study the recycling of model constraints, we now consider the robust maximum weighted matching problem on a bipartite graph with nodes V and edges E. Using decision variables $x_e \in \{0, 1\}$ for each edge $e \in E$, the robust problem reads

$$\begin{aligned} \max \ & \sum_{e \in E} c_e x_e - \left(\Gamma z + \sum_{e \in E} p_e \right) \\ \text{s.t.} \ & \sum_{e \in \delta(v)} x_e \leq 1 && \forall v \in V \\ & p_e + z \geq \hat{c}_e x_e && \forall e \in E \\ & x \in \{0, 1\}^E, \, p \in \mathbb{R}^E_{\geq 0}, z \in \mathbb{R}_{\geq 0} \end{aligned}$$

As mentioned above, the bipartite matching problem has the interesting property that the constraints $\sum_{e \in \delta(v)} x_e \leq 1$ for $v \in V$ and $0 \leq x_e$ for $e \in E$ already define the convex hull of the nominal problem [25]. Moreover, since all constraints are recyclable, the properties from Corollary 2 are fulfilled, which allows for an exact separation of recycled inequalities in linear time, and thus enables us to test their strength to the limit.

We randomly generate instances by first dividing a given set of nodes $V = [n]$ into two partitions $U = \left[\left\lceil \frac{n}{2} \right\rceil\right]$ and $W = \left\{\left\lceil \frac{n}{2} \right\rceil + 1, \ldots, n\right\}$. Afterwards, we sample for each node $u \in U$ a random number $\phi_u \in [0, 1]$, modeling the probability with which an edge incident to u exists. Then for every $w \in W$, we add the edge $\{u, w\}$ with probability ϕ_u. Given the constructed graph, we generate weights c_e and deviations $\hat{c}_e$ analogously to the independent set problem. Every weight is a random number $c_e \in \{900, \ldots, 1000\}$ and the correlated deviations are $\hat{c}_e = \lceil \xi_e c_e \rceil$ with $\xi_e \in [0.45, 0.55]$. Finally, as the number of edges in a solution will most likely be near to $\frac{n}{2}$, we set $\Gamma = \left\lfloor \frac{n}{4} \right\rfloor$. We use this procedure to generate ten instances for different numbers of nodes $n \in \{50, 100, 150\}$.

Table 2 shows computational results for the robust default formulation (DEF) and two different approaches for using recycled inequalities. The first approach recycles all constraints $\sum_{e \in \delta(v)} x_e \leq 1$ for $v \in V$ (RECcons). The second approach additionally separates violated recycled inequalities corresponding to the nominal inequalities $\sum_{e \in E'} x_e \leq 1$ with $E' \subseteq \delta(v)$ for $v \in V$ in the root node of the branch and bound tree (RECconsSepCons).

It is evident that recycling inequalities is significantly better than solely using the default formulation. We observe a considerable strengthening of the formulation, leading to a reduction of the integrality gap to nearly one-hundredth for $n = 150$ nodes. This strength also translates to a higher number of instances solved and much lower computation times. For $n = 150$ with Gurobi's cuts enabled, RECcons has 93% lower computation times compared to DEF. Still, the primal-dual integral is quite low for DEF, suggesting that the solver is very close to optimality from the beginning. This changes once we disable Gurobi's cuts. In this case, DEF is not even able to solve any

Table 2 Computational results for the robust maximum weighted bipartite matching problem

Nodes	GCuts	DEF				RECcons				RECconsSepCons			
		Timeout	Time	P-D integral	Int Gap (%)	Timeout	Time	P-D integral	Int Gap (%)	Timeout	Time	P-D integral	Int Gap (%)
50	Enable	0	1.73	0.04	19.532	0	0.48	0.04	0.326	0	0.77	0.04	0.319
	Disable	10	3600.00	192.48		0	0.25	0.05		0	0.35	0.05	
100	Enable	9	2269.14	3.49	22.820	0	4.50	0.16	0.319	0	6.28	0.17	0.316
	Disable	10	3600.00	550.96		0	15.16	0.21		0	16.26	0.20	
150	Enable	7	2223.68	2.56	23.660	0	150.40	0.59	0.269	0	168.12	0.61	0.265
	Disable	10	3600.00	635.91		6	1887.56	2.51		8	1960.65	2.65	

We generate ten instances per number of nodes and test with Gurobi's own cuts enabled or disabled

instance. Furthermore, the primal-dual integrals of DEF are 253-times as large as the ones of RECcons for $n = 150$.

The recycling of dominated inequalities $\sum_{e \in E'} x_e \leq 1$ compared to the sole recycling of constraints $\sum_{e \in \delta(v)} x_e \leq 1$ yields an improvement of the integrality gap. However, as the recycled constraints already perform well for these instances, the improvement in the linear relaxation is very small. In fact, the minor strengthening of the linear relaxation cannot compensate for the computational load imposed by the additional inequalities, which leads to higher computation times. Our study on the MIPLIB instances will reveal that recycling dominated inequalities can have a much greater effect on the integrality gap, and thus lead to lower computation times.

6.5 Robust bipartite matching with penalties

Until now, we only considered problems for which all valid inequalities are recyclable. In order to test our approach of partially recycling from Sect. 5, we alter the bipartite matching problems from above such that none of the constraints is recyclable. To this end, we allow a solution to match each node $v \in V$ up to two times. However, when matching v more than once, we have to pay a penalty $c_v > 0$. We introduce decision variables $y_v \in \{0, 1\}$ indicating whether node $v \in V$ is matched twice and obtain the following robust problem

$$\begin{aligned}
\max \ & \sum_{e \in E} c_e x_e - \sum_{v \in V} c_v y_v - \left(\Gamma z + \sum_{e \in E} p_e \right) \\
\text{s.t.} \ & \sum_{e \in \delta(v)} x_e - y_v \leq 1 && \forall v \in V \\
& p_e + z \geq \hat{c}_e x_e && \forall e \in E \\
& x \in \{0, 1\}^E, \, y \in \{0, 1\}^V, \, p \in \mathbb{R}^E_{\geq 0}, \, z \in \mathbb{R}_{\geq 0},
\end{aligned}$$

We use the same graphs and parameters as in the previous section, together with random penalties $c_v \in \{450, \ldots, 500\}$, which is on average half the value $c_e \in \{900, \ldots, 1000\}$ of the edge $e \in E$. Note that we do not consider uncertainties on the penalty coefficients.

Table 3 shows computational results for the robust default formulation (DEF) as well as the separation of recycled inequalities via tentatively fixing $y_v = 1$, which yields $\sum_{e \in \delta(v)} x_e \leq 2$, as in Sect. 4.1 (RECsepFix), and the separation of partially recycled inequalities (RECsepPart), as in Sect. 5. Again, we only separate within the root node of the branch and bound tree.

The separation of recycled inequalities via fixing still significantly improves the formulation, cutting the integrality gap in half. However, the effect is clearly weaker compared to the improvement for the original matching problem. The partially recycling approach is considerably stronger, reducing the integrality gap to one-tenth. We have a closer look at the computed solutions in order to assess whether this observed reduction is meaningful. Note that partially recycled inequalities are especially strong when the variables to be lifted are zero. Hence, if we had $y_v = 0$ for all $v \in V$,

Table 3 Computational results for the robust maximum weighted bipartite matching with penalties problem

Nodes	GCuts	DEF				RECsepFix				RECsepPart			
		Timeout	Time	P-D integral	Int Gap (%)	Timeout	Time	P-D integral	Int Gap (%)	Timeout	Time	P-D integral	Int Gap (%)
50	Enable	0	65.67	0.35	18.14	0	67.20	0.37	10.58	0	27.07	0.17	1.67
	Disable	10	3600.00	289.70		9	2907.90	72.20		0	127.79	0.48	
100	Enable	10	3600.00	17.79	20.89	10	3600.00	17.73	11.02	10	3600.00	13.84	2.02
	Disable	10	3600.00	524.65		10	3600.00	252.57		10	3600.00	35.51	
150	Enable	10	3600.00	28.15	20.80	10	3600.00	27.00	9.81	10	3600.00	21.27	2.03
	Disable	10	3600.00	575.09		10	3600.00	267.58		10	3600.00	45.15	

We generate ten instances per number of nodes and test with Gurobi's own cuts enabled or disabled

then the reduction might only be due to a favorable problem generation. However, we observe that we have $y_v = 1$ for approximately three-fourths of all nodes $v \in V$ in the computed solutions, showing that the instance generation is not in our favor.

Even with the partially recycling procedure, the matching with penalties is apparently much harder to solve. Since all approaches always hit the time limit for $n \in \{100, 150\}$, we cannot compare computation times. However, we still see that the partially recycling results in significantly smaller primal-dual integrals compared to the other approaches, especially when Gurobi's cuts are disabled.

6.6 Robustified MIPLIB instances

So far, we have seen that, given the right setting, recycling inequalities can have a significant impact on the strength of the formulation, and thus on the computational performance. To evaluate the use of recycled inequalities for practical instances, we also perform tests on a broad set of robust instances that have been generated in [11]. The test set contains 804 robust instances based on 67 nominal benchmark instances from MIPLIB 2017 [20]. For each nominal instance, 12 robust instances were generated by combining three different ranges of deviations and four different values for Γ.

We consider four different approaches for integrating recycled inequalities in the optimization process. Note that we don't have any insight into the structure of the nominal problems of our test instances, and thus we only recycle inequalities in a generic fashion based on the constraints in $Ax \leq b$. Our first two approaches are as described in Sect. 4.1: We test the direct addition of recycled constraints to the default formulation (RECcons) and the separation of violated recycled (weakened) constraints in the root node of the branch and bound tree (RECsepCons). For the third approach, we first separate recycled constraints as for RECsepCons. Once we do not find any violated recycled constraints, we solve SLP from Sect. 4.3 for a more refined separation (RECsepLP). The fourth approach is as RECsepCons, but we also consider partially recyclable constraints as in Sect. 5 (RECsepPart).

Table 4 shows computational results for the four recycling approaches and the default formulation. To show the effect of scaling deviations together with variables p, z from Sect. 6.1, we also test the default formulation without scaling (DEFnoScale). We see that scaling clearly improves the performance of the approach, with 7% fewer timeouts, 10% lower computation times, and 15% lower primal-dual integrals over all instances when Gurobi's cuts are enabled. In the following, we compare our recycling approaches only with the default formulation using scaling (DEF).

As for the combinatorial problems in the last sections, recycling inequalities is very effective when Gurobi's cuts are disabled. In this setting, the recycling approaches require between 39% and 45% less time in the shifted geometric mean of the computation time over all instances. When only considering the affected instances, that are the instances for which at least one of the recycling approaches provides a better integrality gap compared to the default formulation, the speed-up is even higher. Out of the 804 instances in our test set, 608 were affected by recycling. For these, the recy-

Table 4 Computational results for robustified MIPLIB instances

Instances	GCuts	DEF				DEFnoScale				RECcons			
		Timeout	Time	P-D integral	Int Gap (%)	Timeout	Time	P-D integral	Int Gap (%)	Timeout	Time	P-D integral	Int Gap (%)
All	Enable	348	226.51	35.35	15.90	373	251.99	41.39	15.90	333	198.96	35.95	8.61
	Disable	497	670.44	99.48		501	678.82	101.37		394	368.21	51.14	
Affected	Enable	273	303.20	47.59	24.64	299	340.76	57.89	24.64	254	252.92	45.17	11.15
	Disable	398	842.90	147.33		403	845.00	148.49		295	377.53	60.50	
Instances	GCuts	RECsepCons				RECsepLP				RECsepPart			
		Timeout	Time	P-D integral	Int Gap (%)	Timeout	Time	P-D integral	Int Gap (%)	Timeout	Time	P-D integral	Int Gap (%)
All	Enable	309	193.71	31.97	7.87	308	199.96	33.16	6.60	316	195.51	32.33	7.86
	Disable	409	395.63	51.69		403	405.30	52.97		408	400.15	52.53	
Affected	Enable	235	246.40	41.63	9.93	234	251.87	42.65	7.92	242	247.77	41.99	9.92
	Disable	311	416.76	61.85		305	425.28	62.88		310	423.32	63.24	

We test with Gurobi's own cuts enabled or disabled and show results aggregated for all 804 instances as well as only the 608 on which at least one recycling approach had an effect

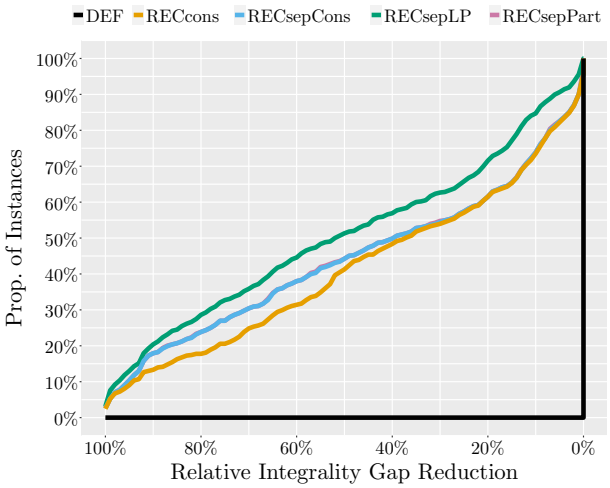

Fig. 3 Cumulative distribution of integrality gap reductions for affected instances

cling approaches require between 49% and 55% less time, which clearly highlights the practical potential of recycling inequalities.

This performance boost is due to the substantially strengthened linear relaxations. RECcons already cuts the integrality gap nearly in half, from 15.90% to 8.61%. RECsepCons yields an even better integrality gap, since we also recycle dominated nominal inequalities in this approach. Note that the relative improvement of the integrality gap using RECsepCons is much larger compared to our observations for the robust bipartite matching problem. We deduce from this that considering dominated inequalities is more important when the coefficients in the constraints are not all the same, as in Example 2 in contrast to the matching problem.

RECsepPart yields nearly no improvement of the integrality gap compared to RECsepCons. While we were able to prove the great potential of this approach for the matching with penalties, the considered instances of the MIPLIB apparently don't contain many constraints of the necessary structure with both positive and negative coefficients on the left-hand side.

In contrast, RECsepLP yields another substantial improvement down to 6.60%. In comparison with the integrality gap of the default formulation, this is a relative reduction of 59% in the shifted geometric mean over all instances. When only considering the affected instances, we even see an improvement from 24.64% to 7.92%, which is a relative reduction of 68%.

To get a better picture of the improvement on the affected instances, we show in Fig. 3 for how many of these the integrality gap is reduced by at least a specific percentage. Note that RECsepCons and RECsepPart have nearly identical lines, as they mostly compute the same cuts for these instances. Of the 608 affected instances, RECcons, RECsepCons, and RECsepPart close the integrality gap completely for 19 and RECsepLP even for 22 instances. Interestingly, this includes not only instances with a low default integrality gap, but 12 instances with a default gap of more than

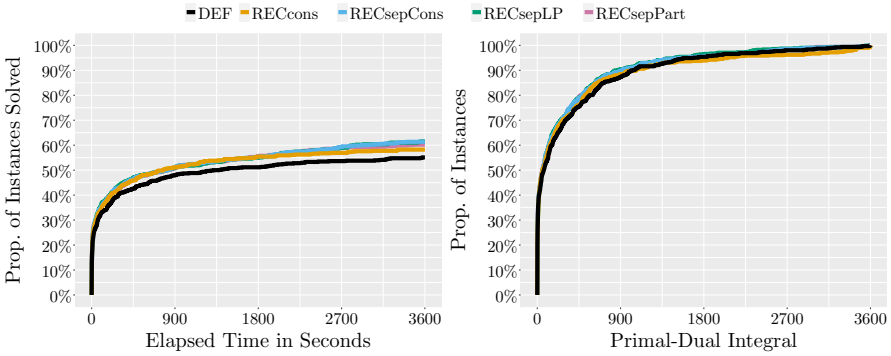

Fig. 4 Cumulative distribution of computation times and primal-dual integrals for the set of affected instances

10%, of which one is even 74, 417%. Moreover, these 12 instances are based on 7 different nominal instances from the MIPLIB 2017. That is, for more than every tenth nominal instance, there is at least one corresponding robust instance for which we close the integrality gap completely.

In addition to these extreme cases, we see that RECsepLP is able to halve the integrality gap for 51% of the instances. Furthermore, RECsepLP achieves a reduction for some problems on which RECcons, RECsepCons, and RECsepPart have no effect. This gives hope that practitioners with a good understanding of their problem might be able to benefit from problem specific fast separations of recycled inequalities that don't correspond directly to the constraints $Ax \leq b$.

The strong linear relaxations also translate to an improved performance when Gurobi's cuts are enabled, with all recycling approaches solving more instances in shorter time. Table 4 shows that RECsepCons has the lowest shifted geometric mean for the computation time and primal-dual integral. Compared to the default formulation, the computation times are 14.5% lower for all instances and 18.7% for the affected ones. Moreover, the lower primal-dual integral implies that separating recycled constraints usually improves the performance across the whole optimization process. RECsepLP solves one instance more, but is on average slightly slower than RECsepCons. This is because the overhead of handling and solving SLP only pays off for specific instances. RECsepPart performs worse than RECsepCons, as both compute almost the same cuts, with RECsepPart requiring more time doing so. RECcons is on average clearly slower compared to the other recycling approaches because many of the added recycled constraints are actually uninteresting for strengthening the linear relaxation, and thus impose unnecessary computational load due to the bigger constraint matrix. Nevertheless, we will see later that RECcons can actually be very useful in practice.

Just like for the integrality gap, Fig. 4 shows the cumulative distribution of performance indicators for each approach on the set of affected instances. We see that each recycling approach solves at any point in time more instances than DEF. The same holds for the primal-dual integral with the exception of RECcons, which has a high primal-dual integral for more instances than DEF. While the differences in

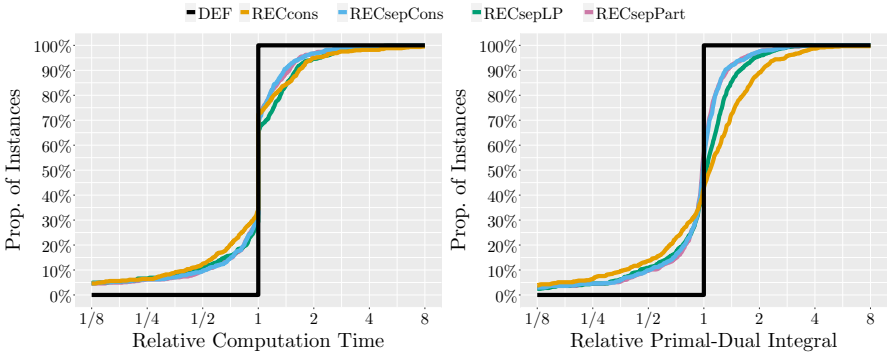

Fig. 5 Cumulative distribution of computation times and primal-dual integrals relative to the default formulation for the set of affected instances

the primal-dual integrals appear to be small, a paired Wilcoxon signed-rank test [38] reveals that for all algorithms but RECcons the improvement in the computation time and the primal-dual integral is significant with a confidence level of at least 98%.

Figure 5 displays the performance indicators of our recycling approaches relative to the default formulation. In the left graphic, we see that all recycling approaches are 8-times as fast for roughly 5%, while requiring 8-times as much time for only 0.5% of the instances. The most balanced of all approaches is RECsepCons, which is 2-times as fast for 9.5% and halve as fast for 3.3% of the instances. Similar observations can be made for the primal-dual integral in the right graphic. However, the most interesting observation about Fig. 5 is that RECcons' and RECsepLP's performance is quite extreme. Regarding computation time and primal-dual integral, both approaches perform badly for more instances than RECsepCons, but the number of instances on which they perform very well is also higher.

The extreme performance is no surprise for RECsepLP, as we already observed above that the higher effort in separating cuts pays off for some specific instances. For RECcons, we see that, given the proper problem structure, recycling constraints directly is not only the most easy approach, but also very efficient. This is good news for the practical use of recycled inequalities, as practitioners will often know whether their optimization problem contains promising recyclable constraints, like clique constraints or almost binding capacity restrictions, that are worth recycling. Recycling precisely these constraints, and not all as we do here for RECcons, might result in a good speed-up for the respective problem. In comparison to RECsepCons, this yields the advantage that the added recycled inequalities are present from the beginning of the optimization process, which is beneficial because the solver can use the additional information for preprocessing. When integrated into a general robust optimization solver, some further engineering might enable us to combine the stable performance of RECsepCons with the performance peaks of RECcons.

6.7 Comparing algorithms from the literature

We close our study by comparing the performance of our recycling approach with that of the following approaches from the literature

DEF	solves the problem over the default formulation $\mathcal{P}^{\text{ROB}}$.
SCE	starts with solving NOM and then separates constraints corresponding to the objective coefficients c' of scenarios from the budgeted uncertainty set, as described by Bertsimas et al. [12].
SUB	solves the problem over the default formulation $\mathcal{P}^{\text{ROB}}$ and separates submodular-cuts dominating the above scenario cuts of SCE, as proposed by Joung and Park [14].
RP1,...,**RP4**	solve the corresponding reformulations of Atamtürk [15].
BS	iterates over all $z \in \{\hat{c}_0, \ldots, \hat{c}_n\}$ for fixing z and solves the corresponding nominal problems, as proposed by Bertsimas and Sim [6].
DnC	also solves nominal problems by fixing $z \in \{\hat{c}_0, \ldots, \hat{c}_n\}$ but reduces the number of iterations by using a relation between the nominal problems to prune non-optimal choices of z, as proposed by Hansknecht et al. [19].
BnB	branches on the variable z in order to prune non-optimal choices for z and resorts to simpler robust subproblems by using the obtained bounds on z, as proposed in [11] and developed further in [39].

A description of the implementation of all considered approaches is given in [39]. For the sake of comparability, we always use scaled deviations and variables p, z, as in Sect. 6.1, for DEF and SUB, which both rely on the formulation $\mathcal{P}^{\text{ROB}}$, just like RECsepCons.

Just like in the previous section, we try to solve the 804 robust problems based on the MIPLIB instances within a time limit of 3,600 s. Table 5 shows for the approaches above and RECsepCons the number of timeouts and shifted geometric means of computation times and primal-dual integrals.

We see that RECsepCons is better than any other algorithm from the literature except for BnB in terms of computation times and primal-dual integrals. The separation approach SCE is clearly worse than DEF, and thus also worse than RECsepCons. The separation of submodular cuts in SUB yields an improvement over DEF, but still performs worse than RECsepCons. The reformulations RP1, RP3, and RP4 of Atamtürk are theoretically strong but too big to be practical. For each of these three approaches, we run out of memory for more than 50% of the instances. Together with timeouts for instances that did not run out of memory, we obtain more than 650 unsolved instances for each of these approaches. The reformulation RP2, which is implemented as a

Table 5 Computational results for different approaches from the literature for robustified MIPLIB instances

Instances	RECsepCons			DEF			SCE			SUB		
	Timeout	Time	P-D integral	Timeout	Time	P-D integral	Timeout	Time	P-D integral	Timeout	Time	P-D integral
All	309	193.71	31.97	348	226.51	35.35	547	710.85	131.67	319	204.67	33.18
Affected	235	246.40	41.63	273	303.20	47.59	452	1216.70	206.85	243	264.00	43.73

Instances	RP1			RP2			RP3			RP4		
	Timeout/ Memory	Time	P-D integral	Timeout	Time	P-D integral	Timeout/Memory	Time	P-D integral	Timeout/Memory	Time	P-D integral
All	652	1758.92	1089.27	382	290.66	49.35	652	1716.20	1348.13	680	2607.05	2253.52
Affected	552	2528.57	1966.90	307	410.28	71.96	552	2436.01	2125.63	550	2768.17	2624.06

Instances	BS			DnC			BnB		
	Timeout	Time	P-D integral	Timeout	Time	P-D integral	Timeout	Time	P-D integral
All	480	901.20	901.20	302	414.42	94.14	83	67.79	12.02
Affected	403	1276.22	1276.22	244	571.96	120.91	67	88.40	14.71

We show results aggregated for all 804 instances as well as only the 608 on which the recycling had an effect

separation approach, does not run out of memory but still yields no computational improvement over DEF. Solving all nominal problems for $z \in \{\hat{c}_0, \ldots, \hat{c}_n\}$ in BS appears to be slower than DEF, as the number of iterations grows too large. Note that the computation time and the primal-dual integral for this approach are equal, as BS yields no dual bound before all nominal problems are solved. The divide and conquer approach DnC dominates BS, as it solves less nominal problems. In fact, DnC solves more problems than RECsepCons, but it is slower in the shifted geometric mean. The branch and bound approach BnB clearly outperforms every other approach, with 73% fewer timeouts than DnC as well as 65% lower computation times and 62% lower primal-dual integrals than RECsepCons. However, BnB is much more complex than the recycling approach, and thus RECsepCons might be a practical alternative to BnB. Moreover, the recycling might be interesting to improve other approaches that benefit from a strengthened formulation. In particular, the recycling of inequalities might be useful within the branch and bound algorithm to improve the performance of BnB even more.

7 Conclusions

In this paper, we proposed and analyzed recycled inequalities for robust combinatorial optimization problems with budgeted uncertainty. Given a valid knapsack inequality for the nominal problem, the corresponding recycled inequality can be derived in linear time, which gives the possibility to reuse model-constraints and well known classes of valid inequalities in order to strengthen the linear relaxation of the robust problem. We highlighted the theoretical strength of such recycled inequalities by proving that they often define facets of the convex hull of the robust problem, even when the underlying valid inequality is dominated.

To make recycled inequalities usable in practice, we discussed different separation procedures that either depend on separation algorithms for classical cutting planes or simply work on the constraint matrix in a generic fashion. One of these separation procedures even implies that recycled inequalities can be separated exactly in polynomial time if the convex hull of the nominal problem is known. Furthermore, we showed that inequalities that are not of the knapsack type can be partially recycled on a restricted solution space and lifted afterwards to obtain a valid inequality for the robust problem.

To test the strength of recycled inequalities and the practicability of their separation, we conducted an extensive computational study on robust versions of three classes of combinatorial problems and a set of nominal benchmark instances. Our experiments show that recycled inequalities are not only interesting from a theoretical point of view, but can also yield a substantial speed-up in the optimization process over the standard formulation.

For future research, it would be interesting to further analyze the recycling of non-knapsack inequalities and evaluate whether one can obtain facet-defining robust inequalities from specific classes of nominal inequalities. Furthermore, recycling should be tested in combination with other tailored approaches and the effect of recycling should be evaluated for robust problems with uncertain constraints. Another

interesting line of research is to investigate whether the recycling approach can be generalized to other uncertainty sets that have similar structures as the budgeted uncertainty set.

Author contributions Christina Büsing: Conceptualization, Writing - Review & Editing, Supervision, Project administration, Funding acquisition
Timo Gersing: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization
Arie Koster: Conceptualization, Writing - Review & Editing, Supervision, Project administration, Funding acquisition

Funding Open Access funding enabled and organized by Projekt DEAL. This work was partially supported by the German Federal Ministry of Education and Research (grants no. 05M16PAA) within the project "HealthFaCT - Health: Facility Location, Covering and Transport", the Freigeist-Fellowship of the Volkswagen Stiftung, and the German research council (DFG) Research Training Group 2236 UnRAVeL.

Data availibility All test instances used in our computational study are published and available for download, sharing, and reuse, see [22].

Code availability All tested algorithms have been implemented in Java and are available on GitHub, see [21].

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Ben-Tal, A., Nemirovski, A.: Robust solutions of linear programming problems contaminated with uncertain data. Mathematical Programming **88**(3), 411–424 (2000). https://doi.org/10.1007/PL00011380
2. Soyster, A.L.: Convex programming with set-inclusive constraints and applications to inexact linear programming. Operations Research **21**(5), 1154–1157 (1973). https://doi.org/10.1287/opre.21.5.1154
3. Kouvelis, P., Yu, G.: Robust Discrete Optimization and Its Applications. Springer, USA (1997). https://doi.org/10.1007/978-1-4757-2620-6
4. Ben-Tal, A., Nemirovski, A.: Robust convex optimization. Mathematics of Operations Research **23**(4), 769–805 (1998). https://doi.org/10.1287/moor.23.4.769
5. Ben-Tal, A., Nemirovski, A.: Robust solutions of uncertain linear programs. Operations Research Letters **25**(1), 1–13 (1999). https://doi.org/10.1016/S0167-6377(99)00016-4
6. Bertsimas, D., Sim, M.: Robust discrete optimization and network flows. Mathematical Programming **98**(1–3), 49–71 (2003). https://doi.org/10.1007/s10107-003-0396-4
7. Bertsimas, D., Sim, M.: The price of robustness. Operations Research **52**(1), 35–53 (2004). https://doi.org/10.1287/opre.1030.0065
8. Ben-Tal, A., El Ghaoui, L., Nemirovski, A.: Robust Optimization. Princeton Series in Applied Mathematics. Princeton University Press, USA (2009). https://doi.org/10.1515/9781400831050

9. Bertsimas, D., Brown, D.B., Caramanis, C.: Theory and applications of robust optimization. SIAM Review **53**(3), 464–501 (2011). https://doi.org/10.1137/080734510
10. Gabrel, V., Murat, C., Thiele, A.: Recent advances in robust optimization: An overview. European Journal of Operational Research **235**(3), 471–483 (2014). https://doi.org/10.1016/j.ejor.2013.09.036
11. Büsing, C., Gersing, T., Koster, A.M.: A branch and bound algorithm for robust binary optimization with budget uncertainty. Mathematical Programming Computation (2023). https://doi.org/10.1007/s12532-022-00232-2
12. Bertsimas, D., Dunning, I., Lubin, M.: Reformulation versus cutting-planes for robust optimization. Computational Management Science **13**(2), 195–217 (2016). https://doi.org/10.1007/s10287-015-0236-z
13. Fischetti, M., Monaci, M.: Cutting plane versus compact formulations for uncertain (integer) linear programs. Mathematical Programming Computation **4**(3), 239–273 (2012). https://doi.org/10.1007/s12532-012-0039-y
14. Joung, S., Park, S.: Robust mixed 0–1 programming and submodularity. INFORMS Journal on Optimization **3**(2), 183–199 (2021). https://doi.org/10.1287/ijoo.2019.0042
15. Atamtürk, A.: Strong formulations of robust mixed 0–1 programming. Mathematical Programming **108**(2–3), 235–250 (2006). https://doi.org/10.1007/s10107-006-0709-5
16. Álvarez-Miranda, E., Ljubić, I., Toth, P.: A note on the bertsimas & sim algorithm for robust combinatorial optimization problems. 4OR **11**(4), 349–360 (2013). https://doi.org/10.1007/s10288-013-0231-6
17. Park, K., Lee, K.: A note on robust combinatorial optimization problem. Management Science and Financial Engineering **13**(1), 115–119 (2007)
18. Lee, T., Kwon, C.: A short note on the robust combinatorial optimization problems with cardinality constrained uncertainty. 4OR **12**(4), 373–378 (2014). https://doi.org/10.1007/s10288-014-0270-7
19. Hansknecht, C., Richter, A., Stiller, S.: Fast robust shortest path computations. In: 18th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2018). OpenAccess Series in Informatics (OASIcs), vol. 65, pp. 5–1521. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2018). https://doi.org/10.4230/OASIcs.ATMOS.2018.5
20. Gleixner, A., Hendel, G., Gamrath, G., Achterberg, T., Bastubbe, M., Berthold, T., Christophel, P.M., Jarck, K., Koch, T., Linderoth, J., Lübbecke, M., Mittelmann, H.D., Ozyurt, D., Ralphs, T.K., Salvagnin, D., Shinano, Y.: MIPLIB 2017: data-driven compilation of the 6th mixed-integer programming library. Mathematical Programming Computation (2021). https://doi.org/10.1007/s12532-020-00194-3
21. Gersing, T.: Algorithms for Robust Binary Optimization. Zenodo (2022). https://doi.org/10.5281/zenodo.7463371
22. Gersing, T., Büsing, C., Koster, A.: Benchmark Instances for Robust Combinatorial Optimization with Budgeted Uncertainty. Zenodo (2022). https://doi.org/10.5281/zenodo.7419028
23. Büsing, C., Gersing, T., Koster, A.: Recycling inequalities for robust combinatorial optimization with budget uncertainty. In: Del Pia, A., Kaibel, V. (eds.) Integer Programming and Combinatorial Optimization IPCO 2023 Lecture Notes in Computer Science, vol. 13904, pp. 58–71. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-32726-1_5
24. Sherali, H.D., Adams, W.P.: A Reformulation-linearization Technique for Solving Discrete and Continuous Nonconvex Problems, vol. 31. Springer, New York (2013). https://doi.org/10.1007/978-1-4757-4388-3
25. Conforti, M., Cornuéjols, G., Zambelli, G., et al.: Integer Programming, vol. 271. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11008-0
26. Padberg, M.W.: On the facial structure of set packing polyhedra. Mathematical programming **5**(1), 199–215 (1973). https://doi.org/10.1007/BF01580121
27. Wolsey, L.A., Nemhauser, G.L.: Integer and Combinatorial Optimization, vol. 55. John Wiley & Sons, New York (1999). https://doi.org/10.1002/9781118627372
28. Korte, B., Vygen, J.: Combinatorial Optimization. Springer, Heidelberg (2018). https://doi.org/10.1007/978-3-662-56039-6
29. Christof, T., Loebel, A.: POlyhedron Representation Transformation Algorithm (PORTA). https://porta.zib.de/. Accessed: 2023-10-24
30. Gu, Z., Nemhauser, G.L., Savelsbergh, M.W.: Sequence independent lifting in mixed integer programming. Journal of Combinatorial Optimization **4**, 109–129 (2000). https://doi.org/10.1023/A:1009841107478

31. Wolsey, L.A.: Facets and strong valid inequalities for integer programs. Operations research **24**(2), 367–372 (1976). https://doi.org/10.1287/opre.24.2.367
32. Zemel, E.: Lifting the facets of zero–one polytopes. Mathematical Programming **15**, 268–277 (1978). https://doi.org/10.1007/BF01609032
33. Gurobi Optimization, LLC: Gurobi Optimizer Reference Manual, Version 9.5 (2022). http://www.gurobi.com
34. Gurobi Optimization, LLC: Advanced user scaling. https://www.gurobi.com/documentation/9.5/refman/advanced_user_scaling.html. Accessed: 2022-09-27
35. Berthold, T.: Measuring the impact of primal heuristics. Operations Research Letters **41**(6), 611–614 (2013). https://doi.org/10.1016/j.orl.2013.08.007
36. Achterberg, T.: Constraint integer programming. Ph. D. Thesis, Technische Universitat Berlin (2007) https://doi.org/10.14279/depositonce-1634
37. Johnson, D.S., Trick, M.A.: Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, October 11-13, 1993 vol. 26. American Mathematical Soc., USA (1996)
38. Wilcoxon, F.: Individual comparisons by ranking methods. Biometrics **1**(6), 80–83 (1945). https://doi.org/10.2307/3001968
39. Gersing, T.: Algorithms for robust combinatorial optimization with budgeted uncertainty and fair planning of the out-of-hours service for pharmacies. Dissertation, RWTH Aachen University, Aachen (2024). https://doi.org/10.18154/RWTH-2024-05270