Folding Mixed-Integer Linear Programs and Reflection Symmetries

Rolf van der Hulst

University of Twente, Enschede, The Netherlands.

Abstract

For mixed-integer linear programming and linear programming it is well known that symmetries can have a negative impact on the performance of branch-and-bound and linear optimization algorithms. A common strategy to handle symmetries in linear programs is to reduce the dimension of the linear program by aggregating symmetric variables and solving a linear program of reduced dimension. In their work *Dimension Reduction via Color Refinement* (DRCR), Grohe, Kersting, Mladenov and Selman show that it is sufficient to run a fast color refinement algorithm to detect permutation symmetries and reduce the dimension of the linear program. We extend DRCR in two directions. First, we show that DRCR can be extended to reflection symmetries, which generalize permutation symmetries. Second, we show the folklore result that DRCR can be applied to the continuous columns of mixed-integer linear programs. In order to derive additional reductions on the integer variables we use *affine totally unimodular decompositions* to reformulate mixed-integer linear programs into mixed-integer linear programs with fewer integer variables. Computational experiments on MIPLIB 2017 collection set using SCIP 10 show that DRCR is an effective tool for handling symmetries. For the linear programming relaxations, DRCR with reflection symmetries yields a modest reduction in running time compared to the original DRCR procedure. For mixed-integer linear programming models, DRCR is very effective at reducing the solution time compared to the default configuration of SCIP. Moreover, the developed DRCR detection algorithms are fast and scale well to large problem instances.

Keywords: linear programming, reflection symmetry, mixed-integer linear programming, symmetry, total unimodularity

1 Introduction

In many problems in mathematical optimization, symmetries arise naturally. For the representative MIPLIB 2017 collection [1], at least 47% of the instances in the collection contains some permutation or reflection symmetry [2]. Symmetry can pose a serious challenge for (spatial) branch-and-cut algorithms in mixed-integer linear programming (MILP) and mixed-integer nonlinear programming (MINLP). If symmetries are not handled adequately, they may cause a branch-and-cut algorithm to repeatedly explore symmetric parts of the search space [3]. In the worst case, all symmetric solutions to a problem are enumerated, which can cause a major performance degradation in the branch-and-cut algorithm, and make it difficult to solve the given optimization problem efficiently.

Previous works that develop symmetry handling methods for mathematical optimization can loosely be categorized into two categories. The first category consists of symmetry handling methods that remove symmetries from the problem formulation by *projection*, which remove symmetric parts of the feasible region by projecting onto the fixed space of the symmetry group. By doing so, one obtains a symmetry-free formulation with fewer variables and constraints, which is attractive from a computational perspective. This method is used primarily for convex optimization problems such as linear programming [4] and semidefinite programs [5].

In contrast to the symmetry handling methods for convex optimization problems, mixed-integer linear programming solvers typically handle symmetry by *breaking* it, either through specialized branching rules [6] or by adding valid inequalities [7]. Due to the non-convexity of the integrality constraints, the projection-based methods used for convex optimization problems no longer apply as they do not necessarily preserve integrality. Although breaking symmetries is still very popular for mixed-integer linear programming, a few recent works, such as the work on Orbital Shrinking [8–10], challenge the idea that symmetry handling methods for mixed-integer linear programming must necessarily break symmetry. This work falls into the same category and formulates symmetry-handling methods that bridge the gap between the strong symmetry handling methods for linear programs to the more challenging setting of mixed-integer linear programs. As our main result, we show that the strong dimension reduction method for linear programs from [11] can be extended to mixed-integer linear programs. Secondly, we show that the dimension reduction method from [11] can be extended to include the class of reflection symmetries, which generalizes the permutation symmetries that are handled by their dimension reduction method. Next, we consider some of the relevant previous works in further detail.

Symmetry handling for linear programs

For linear programs, Bödi, Herr and Joswig [4] show that one can exploit the convexity of the linear program by projecting the linear program onto the fixed space of its symmetry group. This method is computationally attractive, as it both removes the symmetries from the linear program and reduces its dimension. Furthermore, Bödi, Herr and Joswig present an algorithm that exploits the projection onto the

fixed space to solve highly symmetric integer programs whose symmetry group is (almost) transitive. Their work was inspired by the symmetry handling for Semidefinite Programs [5, 12, 13], where symmetries of Semidefinite Programs are exploited by applying results from invariant theory and representation theory to reduce the dimensions of the Semidefinite Programs.

One downside of projecting onto the fixed space of the symmetry group is that the computation of the symmetry group of the Linear Program may be too expensive compared to simply solving the symmetric original program. Typically, the symmetry group is computed by detecting automorphisms of an auxilliary graph representing the (Mixed-Integer) Linear Program [3]. The exact complexity of the graph automorphism problem is unknown, and no polynomial time algorithms are known [14]. However, there are several software packages such as `bliss` [15], `nauty` [16] and `saucy` [17] that can handle large graphs in practice.

Grohe, Kersting, Mladenov and Selman [11] observed that instead of computing the symmetry group via graph automorphism on an auxilliary graph, it is sufficient to run a color refinement method to reduce the dimension of the underlying LP, which they call *Dimension Reduction via Color Refinement* (DRCR). Some sources also refer to their technique as *LP folding* or *weak symmetry*. Their approach has two key advantages compared to the projection onto the fixed space of the symmetry group. First of all, it generalizes the handling of permutation symmetries and may provide additional reductions. Secondly, the color refinement approach is fast in theory and fast in practice since it runs in linearithmic time. One downside of DRCR is that the color refinement algorithm presented in [11] only detects permutation symmetries, whereas Bödi, Herr and Joswig [4] show that projecting to the fixed space of the symmetry group works for the more general linear symmetry group. However, this is not a major limitation in practice, as the majority of symmetries in (Mixed-Integer) Linear Programs are permutation symmetries [2]. The developers of the FICO Xpress solver report that after implementing DRCR that they 'observed improvements of an average of up to 30% on standard LP benchmark sets' [18]. In this case, the average is a somewhat skewed measure; for the affected linear programming models that can be reformulated, applying DRCR even leads to solving times that are 'often more than 5 times as fast on average'. However, the technique is only applicable to a limited set of models, which make up roughly 10% of their test set. Similar results are reported by Gurobi [19]. To the best of our knowledge, DRCR has currently not yet been implemented in any open source solver. This is somewhat surprising, as a speedup of up to 30% for linear programming solvers from a single technique is very rare. DRCR can be considered one of the most important algorithmic advances for linear programming software in the last 15 years.

Symmetry handling for mixed-integer linear programming

Although the symmetry handling methods for Linear Programming are very powerful, they are unfortunately not directly applicable to Mixed-Integer Linear Programming, as the dimension reduction by projecting to the fixed space relies on the convexity of the problem. For Mixed-Integer Linear Programs, the non-convex integrality

constraints invalidate the use of the reduction. Instead, there is a vast collection of symmetry handling methods, where most methods handle symmetry by breaking it in a static or dynamic fashion. Some methods attempt to eliminate the symmetry by adding symmetry-breaking constraints [7, 20]. Other methods, such as Isomorphism Pruning [21, 22] or Orbital Branching [6, 23] handle symmetry dynamically by pruning symmetric branches from the branch-and-bound tree. For a more complete overview, we refer the reader to the surveys by Margot [3] and Pfetsch and Rehn [24]. One symmetry handling method that does not break symmetry is discussed by Pfetsch and Rehn. They consider the fixing of continuous variables to the fixed space of the symmetry group whose automorphisms only permute the continuous variables. From their experiments, they conclude that "fixing the continuous variables slightly speeds up LP-solving, but does not result in a significant overall speed-up" compared to other symmetry handling methods.

One method for symmetry handling for MILP which is relevant to the current work is *Orbital Shrinking*, which was introduced by Fischetti and Liberti [8], and whose theory was later solidified in [10]. The key idea of Orbital Shrinking is to aggregate variables in each orbit of the symmetry group to a single variable per orbit to obtain a smaller mixed-integer program. This is similar to the projection to the fixed space of the symmetry group for linear programs, which also aggregates variables in each orbit. However, due to the non-convexity of the integrality constraints, the reduced mixed-integer problem defines a relaxation of the original problem. Thus, solutions to the relaxed problem are checked by solving a subproblem to check feasibility in the original model. Frequently, this feasibility problem is solved using Constraint Programming techniques [9, 25]. Because orbital shrinking defines a relaxation, any dual bound generated for the relaxed model is also valid for the original model. Due to the smaller size of the relaxed model and the fact that the reduced model contains fewer/no symmetries, the computed dual bounds from the reduced model are typically quite strong for highly symmetric instances [9, 25].

Another innovative method for MILP that preserves symmetry is the core point method [26], which extend the more theoretical work by Bödi, Herr and Joswig [4]. Although they do show that their algorithm is effective for integer programs with (nearly) transitive symmetry groups, their algorithm does not perform as well as orbital branching on arbitrary MILP instances. For a careful comparison between Orbital Shrinking and the core point method, we recommend [10, section 3].

Finally, most modern MIP solvers have fast presolving procedures for detecting simple symmetries arising from parallel columns and parallel rows [27], which are typically implemented using 2-level hashing schemes [27, 28]. This can also be considered a form of variable aggregation to remove symmetries, although the treated symmetries are very simple. In [27, Section 7.1] a special structure is presented where non-overlapping symmetries can be aggregated. In particular, if there is an automorphism of the ILP such that variables contained in the same orbit appear in disjoint constraints, then one can aggregate all the variables in the orbit into one variable without breaking linear and integer feasibility.

Contribution

We extend the DRCR procedure from [11] in several directions. First of all, we show that DRCR for Linear Programs can be extended to reflection symmetries, which arise from the action of signed permutation matrices on linear programs. Our main idea for realizing this uses two transformations. First, we reformulate the original LP by centering the origin at the center of the variable domains using an affine transformation. Then, we split each variable into two variables, where one variable corresponds to the positive domain and one variable corresponds to the negative domain. We show that by applying DRCR to this reformulated linear program, one can recover the variable complementation and row scaling that achieves the best possible reduction among all linear programs obtained by complementing variables and negating rows. The procedure can be implemented efficiently with only minor modifications to the original DRCR procedure.

For our second contribution, we aim to close the gap between the strong symmetry handling methods for linear programs and the symmetry handling methods for mixed-integer linear programs by specializing DRCR to mixed-integer linear programs. In particular, we show the 'folklore' result that DRCR can also be applied to the continuous columns of mixed-integer linear programs. Additionally, we further investigate the application of DRCR in MILP to also aggregate integer variables. We use *implied integrality* [29] and *affine TU decompositions* [30] to infer redundant integrality constraints on a subset of variables. Then, the implied convexity of these variables is exploited by applying DRCR to aggregate these variables. Our approach can also be thought of as *Exact Orbital Shrinking*, where one shows that the orbital shrinking relaxation has the same strength as the original problem, and the feasibility subproblem is an integer program that is given by a perfect formulation and can be solved using linear programming. We formulate an algorithm that specializes DRCR for mixed-integer linear programs. One of its crucial components are fast detection algorithms for *network matrices* [31, 32], which are a subclass of totally unimodular matrices [33]. The developed algorithm does not handle the complete permutation symmetry group, but only aggregates variables that appear in automorphisms where all the permuted variables have no integrality constraints or have integrality constraints that can be reformulated using affine TU decompositions or implied integrality.

We conduct experiments on the MIPLIB 2017 collection set [1]. For the linear programs, we consider the linear relaxations, and show that taking reflection symmetries into account in DRCR can lead to additional reductions and a reduced model size and soution time. For the DRCR algorithm for MILP, we show that it has two favorable characteristics. First, the aggregation of variables to remove symmetries from the model is effective in reducing the running time of affected mixed-integer linear programs. In a computational comparison using SCIP 10 [34], the affected mixed-integer linear programming instances in MIPLIB 2017 that can be solved by SCIP or our method are solved, on average, more than twice as fast compared to the default configuration of SCIP 10. Second, we observe that our detection algorithm for DRCR in

MILP is fast and scales well to large problems, much like in the linear programming case.

Outline

In Section 2 we introduce DRCR in further detail and define relevant notation. Section 3 extends DRCR to also handle reflection symmetries that arise from signed permutation matrices. In Section 4, we explore how DRCR can be extended and specialized for mixed-integer linear programs. In Section 5, we formulate a DRCR algorithm that detects reflection symmetries in MILP problems. Section 6 presents computational experiments on MIPLIB 2017.

2 Dimension Reduction via Color Refinement

Let V and W be two disjoint finite sets. Throughout this work, we will index matrices with the sets V and W for the rows and columns, respectively, and denote any matrix A indexed by V and W using $A \in \mathbb{R}^{V \times W}$. Its elements are indexed using $A_{v,w}$ for $v \in V$, $w \in W$. For $P \subseteq V$, $Q \subseteq W$ we use $A_{P,Q}$ to denote the submatrix induced by P and Q that contains all elements $A_{v,w}$ for $(v,w) \in P \times Q$. For vectors, we use similar notation such as $b \in \mathbb{R}^V$ and b_v for $v \in V$ to denote the vector and its elements, respectively. The order of the elements of matrices will not be relevant, unless noted otherwise. For two vectors $y, z \in \mathbb{R}^W$, we use $\max(y,z)$ to be the element-wise maximum of y and z, such that for all $w \in W$ we have $\max(y,z)_w \coloneqq \max(y_w, z_w)$. Note that for any vector $x \in \mathbb{R}^W$, we have the identity $x \coloneqq \max(x, \mathbb{0}) - \max(-x, \mathbb{0})$. For a partition $\mathcal{P}$ of a set V and some set $V' \subseteq V$, we say that $\mathcal{P}$ *is compatible with* V' if for all $P \in \mathcal{P}$ either $P \subseteq V'$ or $P \cap V' = \emptyset$ holds.

2.1 Equitable partitions and fractional automorphisms

First, we introduce equitable partitions of linear programs as presented in [11]. Given a matrix $A \in \mathbb{R}^{V \times W}$, DRCR iteratively computes partitions $\mathcal{P}_i$ and $\mathcal{Q}_i$ of V and W, the rows and columns of A. Let $\mathcal{P}_0 = \{V\}$ and $\mathcal{Q}_0 = \{W\}$. A partition $(\mathcal{P}, \mathcal{Q})$ of the rows and columns of A is said to be *equitable* if for all $P \in \mathcal{P}, Q \in \mathcal{Q}$ it holds that

$$\sum_{w \in Q} A_{v,w} = \sum_{w \in Q} A_{v',w} \text{ for all } v, v' \in P \tag{1}$$

$$\sum_{v \in P} A_{v,w} = \sum_{v \in P} A_{v,w'} \text{ for all } w, w' \in Q. \tag{2}$$

In words, for every block of A defined by P, Q, the row sums and columns sums restricted to elements in the block must be equal. For a vector $b \in \mathbb{R}^V$, we similarly say that $\mathcal{P}$ is an *equitable partition* of b if $b_v = b_{v'}$ holds for all $v, v' \in P$ for all $P \in \mathcal{P}$. If for $(P,Q) \in \mathcal{P} \times \mathcal{Q}$ both (1) and (2) hold, we also say that (P,Q) defines an *equitable block* of A. A partition $\mathcal{P}_1$ is said to *refine* a partition $\mathcal{P}_2$ if for every $P_1 \in \mathcal{P}_1$ there exists some $P_2 \in \mathcal{P}_2$ such that $P_1 \subseteq P_2$. Color refinement works by iteratively refining $\mathcal{P}_i$ and $\mathcal{Q}_i$, by splitting any classes $P \in \mathcal{P}_i$ or $Q \in \mathcal{Q}_i$ into multiple classes in

$\mathcal{P}_{i+1}$ and $\mathcal{Q}_{i+1}$ if they do not satisfy the conditions (1) and (2) respectively. For some iteration i, this iterative process satisfies $\mathcal{P}_i = \mathcal{P}_{i+1}$ and $\mathcal{Q}_i = \mathcal{Q}_{i+1}$. For this i, the partition $(\mathcal{P}_\infty, \mathcal{Q}_\infty) \coloneqq (\mathcal{P}_i, \mathcal{Q}_i)$ is known as the *coarsest equitable partition* of A. The authors of [11] adapt the well-known fast color refinement algorithm by Paige and Tarjan [35], that is based on ideas by Hopcroft [36], to compute the coarsest equitable partition. For $n = |V| + |W|$ and the total bitlength m of the entries of A, they show that one can compute the coarsest equitable partition in $\mathcal{O}((n+m)\log n)$ time.

Given a partition $\mathcal{P}$ of a set V, its *partition matrix* is denoted by $\Pi_\mathcal{P} \in \{0,1\}^{V \times \mathcal{P}}$, for which $(\Pi)_{vP} = 1 \iff v \in P$ holds for all $v \in V$ and $P \in \mathcal{P}$. A *scaled partition matrix* $\widetilde{\Pi_\mathcal{P}}$ is a normalized partition matrix defined such that $(\widetilde{\Pi_\mathcal{P}})_{vP} = \begin{cases} \frac{1}{|P|} \text{ if } v \in P \\ 0 \text{ otherwise} \end{cases}$ holds for all $v \in V$ and $P \in \mathcal{P}$. Proposition 1 summarizes a few identities for partition matrices and scaled partition matrices that are established and used in [11].

Proposition 1 ([11]) *Given a partition $\mathcal{P}$ of a set V, the following hold:*

(i) $\widetilde{\Pi_\mathcal{P}}^\intercal \Pi_\mathcal{P} = \Pi_\mathcal{P}^\intercal \widetilde{\Pi_\mathcal{P}} = I_{\mathcal{P}\times\mathcal{P}}$.

(ii) For $v, v' \in V$, $(\widetilde{\Pi_\mathcal{P}}\Pi_\mathcal{P}^\intercal)_{v,v'} = (\Pi_\mathcal{P}\widetilde{\Pi_\mathcal{P}}^\intercal)_{v',v} = \begin{cases} \frac{1}{|P|} \text{ if } v, v' \in P \text{ for some } P \in \mathcal{P} \\ 0 \text{ otherwise.} \end{cases}$

(iii) If $\mathcal{P}$ is an equitable partition of $b \in \mathbb{R}^V$, then $b = \widetilde{\Pi_\mathcal{P}}\Pi_\mathcal{P}^\intercal b = \Pi_\mathcal{P}\widetilde{\Pi_\mathcal{P}}^\intercal b$ holds.

Proof For (i), consider any $P, P' \in \mathcal{P}$ and note that $(\widetilde{\Pi_\mathcal{P}}^\intercal \Pi_\mathcal{P})_{P,P'} = \begin{cases} \sum_{v \in P} \frac{1}{|P|} \text{ if } P = P' \\ 0 \text{ otherwise} \end{cases} = I_{\mathcal{P}\times\mathcal{P}}$ holds. The identity in (i) follows since $I_{\mathcal{P}\times\mathcal{P}} = (\widetilde{\Pi_\mathcal{P}}^\intercal \Pi_\mathcal{P})^\intercal = \Pi_\mathcal{P}^\intercal \widetilde{\Pi_\mathcal{P}}$ holds. The proof of (ii) follows by the definition of matrix transpose and multiplication. For (iii) we have for $v \in V$ and $P \in \mathcal{P}$ such that $v \in P$ that $(\widetilde{\Pi_\mathcal{P}}^\intercal \Pi_\mathcal{P} b)_v = \sum_{v' \in P} \frac{1}{|P|} b_{v'} = b_v$, where the last equality holds since $\mathcal{P}$ is an equitable partition of b, which implies that $b_v = b_{v'}$ holds for all $v' \in P$. □

A matrix $A \in \mathbb{R}^{V \times W}$ is said to be *stochastic* if it is nonnegative and $\sum_{w \in W} A_{v,w} = 1$ for all $v \in V$, and it is *doubly stochastic* if both A and $A^\intercal$ are stochastic. For a matrix $A \in \mathbb{R}^{V \times W}$, a pair $(X, Y) \in \mathbb{R}^{V \times V} \times \mathbb{R}^{W \times W}$ of doubly stochastic matrices is said to be a *fractional automorphism* of A if $XA = AY$ holds. One of the key insights made by the authors of [11] is given in Proposition 2, and relates fractional automorphisms of A to its equitable partitions.

Proposition 2 *([11]) If $(\mathcal{P}, \mathcal{Q})$ is an equitable partition of a matrix $A \in \mathbb{R}^{V \times W}$, then $(\Pi_\mathcal{P}\widetilde{\Pi_\mathcal{P}}^\intercal, \Pi_\mathcal{Q}\widetilde{\Pi_\mathcal{Q}}^\intercal)$ is a fractional automorphism of A.*

In Proposition 2, it can be easily checked that $\Pi_\mathcal{P}\widetilde{\Pi_\mathcal{P}}^\intercal$ and $\Pi_\mathcal{Q}\widetilde{\Pi_\mathcal{Q}}^\intercal$ are both doubly stochastic due to Proposition 1(ii). From Proposition 1(ii) and Proposition 2 it also

follows that if $(\mathcal{P}, \mathcal{Q})$ is an equitable partition, that $(\widetilde{\Pi_{\mathcal{P}}}\Pi_{\mathcal{P}}^\intercal, \widetilde{\Pi_{\mathcal{Q}}}\Pi_{\mathcal{Q}}^\intercal)$ is a fractional automorphism of A.

2.2 Dimension Reduction of a Linear Program

Throughout this work, we consider the following linear program in standard form.

$$\min c^\intercal x \text{ s.t. } Ax = b,\ \ell \leq x \leq u \qquad (F(A, b, \ell, u, c))$$

with $A \in \mathbb{R}^{V \times W}$, $b \in \mathbb{R}^V$, $\ell_w \in \mathbb{R} \cup \{-\infty\}$ and $u \in \mathbb{R} \cup \{\infty\}$ such that $\ell_w \leq u_w$ for all $w \in W$. We use $F(A, b, \ell, u, c)$ to denote the linear program, and we define its feasible region as $P(A, b, \ell, u) \coloneqq \{x \in \mathbb{R}^W \mid Ax = b, \ell \leq x \leq u\}$. For the linear program $F(A, b, \ell, u, c)$ and a partition $\mathcal{P}$ of V and a partition $\mathcal{Q}$ of W, $(\mathcal{P}, \mathcal{Q})$ is an *equitable partition of* $F(A, b, \ell, u, c)$ if $(\mathcal{P}, \mathcal{Q})$ is an equitable partition of A, $\mathcal{P}$ is an equitable partition of b, and $\mathcal{Q}$ is an equitable partition of ℓ, u, and c. The key result necessary for DRCR is given in [11, Lemma 7.1] and shows that a linear program that has an equitable partition $(\mathcal{P}, \mathcal{Q})$ can be reduced to a linear program with a constraint matrix with dimension $|\mathcal{P}| \times |\mathcal{Q}|$. We show a variation of this result which has essentially the same proof, but which produces a slightly different but equivalent linear program whose variables and constraints are scaled differently. The motivation for deviating from [11] in Proposition 3 will become clear in Section 4, where we consider mixed-integer linear programs.

Proposition 3 (Grohe, Kersting, Mladenov and Selman [11]) *For given $A \in \mathbb{R}^{V \times W}$, $b \in \mathbb{R}^V$, $c \in \mathbb{R}^W$ and vectors ℓ, u with $\ell_w \in \mathbb{R} \cup \{-\infty\}$ and $u_w \in \mathbb{R} \cup \{\infty\}$ such that $\ell_w \leq u_w$ holds for all $w \in W$, consider the linear program $F(A, b, \ell, u, c)$. Let $(\mathcal{P}, \mathcal{Q})$ be an equitable partition of $F(A, b, \ell, u, c)$. Define $A' \coloneqq \Pi_{\mathcal{P}}^\intercal A \widetilde{\Pi_{\mathcal{Q}}}$, $b' \coloneqq \Pi_{\mathcal{P}}^\intercal b$, $\ell' \coloneqq \Pi_{\mathcal{Q}}^\intercal \ell$, $u' \coloneqq \Pi_{\mathcal{Q}}^\intercal u$ and $c' \coloneqq \widetilde{\Pi_{\mathcal{Q}}}^\intercal c$, and consider the reduced linear program $F(A', b', \ell', u', c')$. Then, the following hold:*

(i) If $x \in \mathbb{R}^W$ is feasible for $P(A, b, \ell, u)$, then $y \coloneqq \Pi_{\mathcal{Q}}^\intercal x$ is feasible for $P(A', b', \ell', u')$ and $c'^\intercal y = c^\intercal x$ holds.

(ii) If $y \in \mathbb{R}^{\mathcal{Q}}$ is feasible for $P(A', b', \ell', u')$, then $x \coloneqq \widetilde{\Pi_{\mathcal{Q}}} y$ is feasible for $P(A, b, \ell, u)$ and $c^\intercal x = c'^\intercal y$ holds

Proof Let us prove the first point. Let $x \in P(A, b, \ell, u)$ be given and consider $y \coloneqq \Pi_{\mathcal{Q}}^\intercal x$. First of all, we show that for each $Q \in \mathcal{Q}$, $\ell'_Q \leq y_Q \leq u'_Q$ holds. Note that $\ell'_Q = (\Pi_{\mathcal{P}}^\intercal \ell)_Q = \sum_{w \in Q} \ell_w$ and $u'_Q = (\Pi_{\mathcal{P}}^\intercal u)_Q = \sum_{w \in Q} u_w$ and $y_Q = \sum_{w \in Q} x_w$ hold. Then, $\ell'_Q \leq y_Q \leq u'_Q$ is equivalent to $\sum_{w \in Q} \ell_w \leq \sum_{w \in Q} x_w \leq \sum_{w \in Q} u_w$, which is clearly valid by summing the inequalities $\ell_w \leq x_w \leq u_w$ for all $w \in Q$. Then, we show that y is feasible by showing that the equation $A'y = b'$ holds. The following equations hold.

$$A'y = \Pi_{\mathcal{P}}^\intercal A \widetilde{\Pi_{\mathcal{Q}}} \Pi_{\mathcal{Q}}^\intercal x \overset{\text{(a)}}{=} \Pi_{\mathcal{P}}^\intercal \widetilde{\Pi_{\mathcal{P}}} \Pi_{\mathcal{P}}^\intercal A x \overset{\text{(b)}}{=} \Pi_{\mathcal{P}}^\intercal A x = \Pi_{\mathcal{P}}^\intercal b = b'$$

The equality in (a) follows from the fact that $(\mathcal{P}, \mathcal{Q})$ is an equitable partition of A, which implies that $(\widetilde{\Pi_{\mathcal{P}}}\Pi_{\mathcal{P}}^\intercal, \widetilde{\Pi_{\mathcal{Q}}}\Pi_{\mathcal{Q}}^\intercal)$ is a fractional automorphism of A using Proposition 2. The equality in (b) follows from Proposition 1(i). Thus, $y \in P(A', b', \ell', u')$ holds, and y is indeed feasible. Finally, since $\mathcal{Q}$ is an equitable partition of c it follows that $c'^\intercal y = c^\intercal \widetilde{\Pi_{\mathcal{Q}}} \Pi_{\mathcal{Q}}^\intercal x = c^\intercal x$, where we use Proposition 1(iii) in the final step.

Next, we proceed with the proof of second point. Let $y \in P(A', b', \ell', u')$ be a given feasible point, and consider $x \coloneqq \widetilde{\Pi_{\mathcal{Q}}} y$. First, note that since $\mathcal{Q}$ is an equitable partition of ℓ and u, that for any $Q \in \mathcal{Q}$ we have $\ell'_Q = |Q|\ell_w$ for all $w \in Q$ and $u'_Q = |Q|u_w$, since the l_w and u_w entries must be identical for all $w \in Q$ and are summed by definition of $\Pi_{\mathcal{Q}}^{\intercal}$. Additionally, note that for each $w \in W$, there exists exactly one $Q \in \mathcal{Q}$ such that $x_w = \frac{1}{|Q|} y_Q$. Then, we have for every $w \in W$ and $Q \in \mathcal{Q}$ with $w \in Q$ that $\ell'_Q \leq y_Q \leq u'_Q$ holds, which implies that $\frac{1}{|Q|}\ell'_Q \leq \frac{1}{|Q|} y_Q \leq \frac{1}{|Q|} u'_Q$ holds. By substituting using $\ell'_Q = |Q|\ell_w$, $u'_Q = |Q|u_w$ and $x_w = \frac{1}{|Q|} y_Q$, we see that $\ell_w \leq x_w \leq u_w$ holds for all $w \in W$.

Then, similar to the proof of the first point, we consider Ax and show that it equals b.

$$Ax = A\widetilde{\Pi_{\mathcal{Q}}} y \overset{\text{(c)}}{=} A\widetilde{\Pi_{\mathcal{Q}}}\Pi_{\mathcal{Q}}^{\intercal}\widetilde{\Pi_{\mathcal{Q}}} y \overset{\text{(d)}}{=} \widetilde{\Pi_{\mathcal{P}}}\Pi_{\mathcal{P}}^{\intercal} A\widetilde{\Pi_{\mathcal{Q}}} y = \widetilde{\Pi_{\mathcal{P}}} A' y = \widetilde{\Pi_{\mathcal{P}}} b' = \widetilde{\Pi_{\mathcal{P}}}\Pi_{\mathcal{P}}^{\intercal} b \overset{\text{(e)}}{=} b$$

Here, (c) holds by Proposition 1(i) and (d)holds since $(\mathcal{P}, \mathcal{Q})$ is an equitable partition of A, which implies a fractional automorphism using Proposition 2. Additionally, (e) follows by Proposition 1(iii) and the fact $\mathcal{P}$ is an equitable partition of b. Thus, x is feasible for $P(A, b, \ell, u)$. Finally, note that $c^{\intercal} x = c^{\intercal}\widetilde{\Pi_{\mathcal{Q}}}\Pi_{\mathcal{Q}}^{\intercal} x = c'^{\intercal} y$ holds, where the first equality follows from Proposition 1(iii) and the fact that $\mathcal{Q}$ is an equitable partition of c. □

Occasionally, we will use the notation $R^{\mathcal{P},\mathcal{Q}}(F(A, b, \ell, u, c)) \coloneqq F(\Pi_{\mathcal{P}}^{\intercal} A\widetilde{\Pi_{\mathcal{Q}}}, \Pi_{\mathcal{P}}^{\intercal} b, \Pi_{\mathcal{Q}}^{\intercal}\ell, \Pi_{\mathcal{Q}}^{\intercal} u, \widetilde{\Pi_{\mathcal{Q}}}^{\intercal} c)$ to denote the reduced linear program in Proposition 3.

The key difference in the proof of Proposition 3 compared to the proof in [11, Lemma 7.1] is that we use the fractional automorphism $(\widetilde{\Pi_{\mathcal{P}}}\Pi_{\mathcal{P}}^{\intercal}, \widetilde{\Pi_{\mathcal{Q}}}\Pi_{\mathcal{Q}}^{\intercal})$, instead of $(\Pi_{\mathcal{P}}\widetilde{\Pi_{\mathcal{P}}}^{\intercal}, \Pi_{\mathcal{Q}}\widetilde{\Pi_{\mathcal{Q}}}^{\intercal})$, which leads to different variable and constraint scaling in the reduced linear programs and in the mappings from and to the original linear program.

Although we present our results in standard form, equitable partitions can also be applied to linear programs in inequality form by first introducing slack variables and turning them into equality form. In Proposition 4 we show that this procedure does not affect equitability of the constraint matrix.

Proposition 4 *Let $(\mathcal{P}, \mathcal{Q})$ be an equitable partition of $A \in \mathbb{R}^{V \times W}$. Then $(\mathcal{P}, \mathcal{Q} \cup \mathcal{P})$ is an equitable partition of $\begin{bmatrix} A & I \end{bmatrix}$.*

Proof Consider any block $(P, Q) \in \mathcal{P} \times (\mathcal{Q} \cup \mathcal{P})$. If $Q \in \mathcal{Q}$, then $\begin{bmatrix} A & I \end{bmatrix}_{PQ} = A_{PQ}$ holds, and A_{PQ} satisfies the equitability conditions (1) and (2) since $(\mathcal{P}, \mathcal{Q})$ is an equitable partition of A. Otherwise, for $Q \in \mathcal{P}$, we have that $\begin{bmatrix} A & I \end{bmatrix}_{PQ} = I_{P,P'}$ for some $P' \in \mathcal{P}$. If $P = P'$, then $I_{P,P}$ is a $P \times P$ identity matrix, which satisfies (1) and (2). Otherwise, if $P \neq P'$, then $I_{P,P'}$ is an all-zero matrix, which also satisfies (1) and (2). Thus, $\mathcal{P} \times (\mathcal{Q} \cup \mathcal{P})$ is an equitable partition of $\begin{bmatrix} A & I \end{bmatrix}$. □

Note that the introduced slack variables will all have identical bounds and do not participate in the objective, so the equitability conditions for the variable bounds and objective are also satisfied under the addition of slack variables. However, the standard form may have additional reductions compared to inequality form if the slack variables can be aggregated with non-slack variables. In [37] the authors observe that additional reductions on the slack variables in equality form lead to large running time improvements for algorithms that enumerate all non-symmetric solutions of

certain MILP. Intuitively, this can be explained by the fact that certain relationships between the slack of the constraints and the variables become more explicit upon adding the slack variables. Thus, we consider the more general standard form in the remainder of this work.

The authors of [11] show that DRCR is advantageous primarily because it is fast to compute and because it generalizes permutation symmetries. We have not yet discussed one of its more relevant downsides, which is that a basic optimal solution to the reduced LP is not necessarily mapped to a basic solution of the original LP. This is particularly relevant for branch-and-bound approaches that utilize an LP relaxation, as for these the reduction for the LP may not be valid for the MILP, and thus we need to obtain a solution to the original LP. For MILP, basic linear programming solutions are necessary in an efficient branch-and-cut implementation, which relies heavily on the dual simplex algorithm to branch quickly and to efficiently find cutting planes such as those formulated by Gomory [38] from the simplex tableau. For DRCR, the original solution obtained from the the reduced solution may lie in the interior of the optimal face of the original LP. Then, to obtain a basic solution, one has to run a crossover procedure for the original problem to obtain a feasible vertex from the interior point [39]. In some cases, the crossover procedure has a runtime that may be orders of magnitude larger than solving the reduced linear program. For example, for problems with a transitive symmetry group, the linear program computed by DRCR consists of a single variable and is easily solved, but the crossover procedure may be as expensive as solving the original linear program.

3 Extending DRCR to reflection symmetries

As mentioned in the introduction, the DRCR algorithm presented in [11] only detects automorphisms that arise from permutation matrices, whereas the theory they formulate supports more general linear symmetries of the constraint matrix. In [4] it is shown that for linear programs the projection to the fixed space of the symmetry group works for any linear group, rather than just the symmetries arising from permutation matrices. The algorithm presented in [4] uses reflection symmetries generated by the signed permutation group to solve highly symmetric integer programs. Reflection symmetries generalize permutation symmetries to signed permutations and additionally take the action of complementing a variable x to $u - x$ into account, where u is its upper bound.

Most MILP solvers and LP solvers only perform symmetry handling for permutation symmetries. SCIP is a notable exception, as it handles reflection symmetries since version 9.0 [40]. More information on how reflection symmetries are detected and handled in SCIP can be found in [2]. As far as we are aware, the only other work that treats reflection symmetries computationally for MILP solvers is by Christophel, Güzelsoy and Pólik [41] and explores how reflection symmetries may be used in primal heuristics. Reflection symmetries that are not permutation symmetries are somewhat uncommon in Mixed Integer Programs; Hojny [2] reports that

they were detected for only 60 out of 1055 tested MIPLIB 2017 instances and 6 out of 486 tested MINLPLIB instances, and that handling of reflection symmetries for mixed-integer linear programs does not produce large speedups. In fact, the default settings for symmetry handling in SCIP 9.0 disable the detection and handling of reflection symmetries. This can perhaps be explained by the interesting result by Geyer, Bulutoglu and Ryan [37], who show that for feasible linear programs that are full-dimensional and do not contain redundant inequalities, permutation symmetries can be used to detect the complete linear symmetry group of the linear program. To do so, they first normalize the linear program using a singular-value decomposition and then formulate a complicated algorithm to detect the full symmetry group. They also detect reflection symmetries and exploit them in an enumeration procedure for an integer linear program used for classifying orthogonal arrays.

As our first contribution, we will show that DRCR can be extended to detect reductions that take the complements of every variable into account. We will consider the original linear program given by $F(A, b, \ell, u, c)$, and transform it to obtain a representation where DRCR is more effective. Our approach somewhat resembles the detection strategies in [2, 41], which both reduce the problem of finding reflection symmetries to finding permutation symmetries by adding both the original and the complemented variables to the auxilliary graph used for symmetry detection. We will use two transformations to achieve something similar using DRCR. First, the origin is moved to the center of the variable domains using an affine transformation, so that $\ell \leq x \leq u$ is transformed into $-\nu \leq x' \leq \nu$, where $\nu = \frac{u-\ell}{2}$. Then, each variable x'_w for $w \in W$ is split into two variables x'^+_w and x'^-_w such that $x'_w = x'^+_w - x'^-_w$ holds, where $0 \leq x'^+ \leq \nu$ and $0 \leq x'^- \leq \nu$ hold. Crucially, we show that by computing an equitable partition on this linear program with x'^+ and x'^-, one can recover the best equitable partition possible for the original linear program under the operation of complementing the variables.

We illustrate the main ideas of our method with an example linear program given in (3).

$$\min\ x_1 - x_2 \tag{3a}$$

$$\begin{bmatrix} 2 & 1 & 1 \\ -1 & -2 & -1 \\ -1 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leq \begin{bmatrix} 5 \\ -3 \\ 1 \end{bmatrix} \tag{3b}$$

$$0 \leq x_i \leq 2 \qquad \forall i \in \{1, 2, 3\} \tag{3c}$$

The polyhedron that defines the feasible region of (3) has two reflection symmetries, one reflection in the hyperplane $x_1 + x_2 = 2$ and one reflection in the hyperplane $x_3 = 1$. We will show how both symmetries can be projected out by considering a reformulation of (3). First, we apply an affine transformation to (3) that yields variables that have identical negative and positive domain sizes by substituting $x = x' + \mathbb{1}$. Furthermore, we add the slack variables s to turn the inequalities into equalities.

$$\min\ x'_1 - x'_2 \tag{4a}$$

$$\begin{bmatrix} 2 & 1 & 1 & 1 & 0 & 0 \\ -1 & -2 & -1 & 0 & 1 & 0 \\ -1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x'_1 \\ x'_2 \\ x'_3 \\ s_1 \\ s_2 \\ s_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \tag{4b}$$

$$-1 \le x'_i \le 1 \qquad \forall i \in \{1,2,3\} \tag{4c}$$

$$s_i \ge 0 \qquad \forall i \in \{1,2,3\} \tag{4d}$$

Next, we introduce redundant variables and constraints in the following manner. First, we split each variable x'_w using $x'_w = x'^{+}_w - x'^{-}_w$ for $w \in \{1,2,3\}$. After doing so, we additionally add a negated copy of each equation. Then, the mappings defined by $x'_w \leftarrow x'^{+}_w - x'^{-}_w$ and $(x'^{+}_w, x'^{-}_w) \leftarrow (\max(x_w, 0), \max(-x_w, 0))$ can be used to show that feasibility and optimality are preserved. This yields the following equivalent linear program, which we will call the *split reformulation* more generally.

$$\min\ x'^{+}_1 - x'^{-}_1 - x'^{+}_2 + x'^{-}_2 \tag{5a}$$

$$\begin{bmatrix} 2 & 1 & 1 & -2 & -1 & -1 & 1 & 0 & 0 \\ -1 & -2 & -1 & 1 & 2 & 1 & 0 & 1 & 0 \\ -1 & 1 & 0 & 1 & -1 & 0 & 0 & 0 & 1 \\ -2 & -1 & -1 & 2 & 1 & 1 & -1 & 0 & 0 \\ 1 & 2 & 1 & -1 & -2 & -1 & 0 & -1 & 0 \\ 1 & -1 & 0 & -1 & 1 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x'^{+}_1 \\ x'^{+}_2 \\ x'^{+}_3 \\ x'^{-}_1 \\ x'^{-}_2 \\ x'^{-}_3 \\ s_1 \\ s_2 \\ s_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ -1 \end{bmatrix} \tag{5b}$$

$$0 \le x'^{j}_i \le 1 \quad \forall i \in \{1,2,3\}, \forall j \in \{+,-\} \tag{5c}$$

$$s_i \ge 0 \quad \forall i \in \{1,2,3\} \tag{5d}$$

Next, we compute the equitable partition of the split reformulation (5). We reorder the variables to clarify the partition.

$$\begin{array}{c|cc|cc|cc|ccc|c} & x'^{+}_1 & x'^{-}_2 & x'^{-}_1 & x'^{+}_2 & x'^{+}_3 & x'^{-}_3 & s_1 & s_2 & s_3 & \\ \hline & 2 & -1 & -2 & 1 & 1 & -1 & 1 & 0 & 0 & 1 \\ & -1 & 2 & 1 & -2 & -1 & 1 & 0 & 1 & 0 & 1 \\ \hline & -1 & -1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hline & -2 & 1 & 2 & -1 & -1 & 1 & -1 & 0 & 0 & -1 \\ & 1 & -2 & -1 & 2 & 1 & -1 & 0 & -1 & 0 & -1 \\ \hline & 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & -1 & -1 \end{array}$$

Then, we apply Proposition 3 to the given equitable partition. Doing so aggregates some of the positive and negative index variables with each other.

$$\min y_{1^+,2^-} - y_{1^-,2^+} \tag{6a}$$

$$\begin{bmatrix} 1 & -1 & 0 & 1 & 0 \\ -1 & 1 & 0 & 0 & 1 \\ -1 & 1 & 0 & -1 & 0 \\ 1 & -1 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} y_{1^+,2^-} \\ y_{1^-,2^+} \\ y_{3^+,3^-} \\ s_{1,2} \\ s_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ -2 \\ -1 \end{bmatrix} \tag{6b}$$

$$0 \leq y_{1^+,2^-}, y_{1^-,2^+}, y_{3^+,3^-} \leq 2 \tag{6c}$$

$$s_{1,2}, s_3 \geq 0 \tag{6d}$$

$$\tag{6e}$$

We can notice two things about (6). First of all, we note that $y_{3^+,3^-}$ vanishes from the given equations and thus any $0 \leq y_{3^+,3^-} \leq 2$ is always feasible, so we may remove it from the Linear Program. This corresponds to removing the reflection symmetry around the hyperplane $x_3 = 1$, which implies that x'_{3^+} and x'_{3^-} are symmetric to each other. Secondly, we can observe that the redundancy that we introduced by adding negated variables and rows is still present in (6). By removing the duplicated rows and setting $z_{1^\pm,2^\mp} = y_{1^+,2^-} - y_{1^-,2^+}$, we obtain the following one-dimensional linear program.

$$\begin{aligned} \min z_{1^\pm,2^\mp} \\ z_{1^\pm,2^\mp} + s_{1,2} &= 2 \\ -z_{1^\pm,2^\mp} + s_3 &= 1 \\ -2 \leq z_{1^\pm,2^\mp} &\leq 2 \\ s_{1,2}, s_3 &\geq 0 \end{aligned}$$

In this final linear program, we have additionally removed the reflection symmetry around the hyperplane $x_1 + x_2 = 2$ in the original problem. Finally, we can conclude that applying the equitable partition to the split reformulation was sufficient to remove the reflection symmetries from our problem. In the next sections, we show more formally that the illustrated procedure can be used to aggregate the reflection symmetries of any linear program.

3.1 Centering the linear program

Initially, we consider the linear program $F(A, b, \ell, u, c)$. In order to detect reflection symmetries, we would like to center each variable at the center of its domain. This can be achieved using an affine transformation that offsets the variables.

Proposition 5 *For any scalar $\delta \in \mathbb{R}$, $x \in P(A, b, \ell, u)$ holds if and only if $x' := x - \delta \in P(A, b - A\delta, \ell - \delta, u - \delta)$ holds. Moreover, $c^\intercal x' = c^\intercal x - c^\intercal \delta$ holds.*

Proof It is sufficient to observe that $Ax = b \iff Ax - A\delta = b - A\delta \iff Ax' = b - A\delta$ and that $\ell \leq x \leq u \iff \ell - \delta \leq x - \delta \leq u - \delta \iff \ell - \delta \leq x' \leq u - \delta$. Moreover, $c^\intercal x' = c^\intercal x - c^\intercal \delta$ follows by definition of x'. □

The main idea that we use Proposition 5 for in the context of equitable partitions is that that $F(A, b - A\delta, \ell - \delta, u - \delta, c)$ may admit a coarser equitable partition than $F(A, b, \ell, u, c)$. We note that there are infinitely many choices for δ to do so, so it is not clear how δ may be chosen. Frequently, we use Proposition 5 to center the variables at their domains by using $\delta_w \coloneqq \frac{u_w + \ell_w}{2}$. For variables with infinite lower or upper bounds, such a centered δ_w value is not well-defined. Later on, we will clarify how one should choose δ_w in such cases. For now, we analyze the equitable partitions of $F(A, b - A\delta, \ell - \delta, u - \delta, c)$ for any finite $\delta \in \mathbb{R}$.

Furthermore, we note that δ corresponds to an offset of the primal variables. It is also possible to expand the results in this work by using an offset for the dual variables, which corresponds to adding weighted multiples of the rows to the objective. One could hope that doing so may help to construct additional variables that have identical objective value, which would lead to potentially coarser equitable partitions. In order to simplify the presentation, we do not consider this extension here.

3.2 Split reformulations of linear programs

In both papers [2] and [41], the authors reduce the problem of finding reflection symmetries to finding permutation symmetries by adding both the original and the complemented variables to the auxiliary graph used for symmetry detection. For linear programs, a similar effect can be achieved by splitting each variable x as $x = x^+ - x^-$, where x^+ represents the positive domain of x and x^- corresponds to the negative domain of x. We introduce the notion of a *split reformulation* to achieve such a reformulation.

Definition 6 *Let $F(A, b, \ell, u, c)$ be any linear program with $A \in \mathbb{R}^{V \times W}$, $b \in \mathbb{R}^V$, $c \in \mathbb{R}^W$ and vectors ℓ, u with $\ell_w \in \mathbb{R} \cup \{-\infty\}$ and $u \in \mathbb{R} \cup \{\infty\}$ for all $w \in W$. If $\ell \leq 0$ and $u \geq 0$ hold, we define the* split reformulation *of $F(A, b, \ell, u, c)$ to be the following linear program with variables $x^+ \in \mathbb{R}^W$ and $x^- \in \mathbb{R}^W$:*

$$\begin{aligned} \min\ & c^\intercal x^+ - c^\intercal x^- & \\ & Ax^+ - Ax^- = b & \\ & -Ax^+ + Ax^- = -b & \qquad (F_S(A, b, \ell, u, c))) \\ & 0 \leq x^+ \leq u & \\ & 0 \leq x^- \leq -\ell & \end{aligned}$$

We use $F_S(A, b, \ell, u, c)$ to denote the linear program that is the split reformulation of $F(A, b, \ell, u, c)$, and we use $P_S(A, b, \ell, u)$ to denote its feasible region. Furthermore, we use $\widehat{M} = \begin{bmatrix} A & -A \\ -A & A \end{bmatrix} \in \mathbb{R}^{(\widehat{V}^+ \cup \widehat{V}^-) \times (\widehat{W}^+ \cup \widehat{W}^-)}$, to denote the constraint matrix and $\widehat{V} \coloneqq \widehat{V}^+ \cup \widehat{V}^-$ and $\widehat{W} \coloneqq \widehat{W}^+ \cup \widehat{W}^-$ to denote the row and column index sets.

Note that the split reformulation $F_S(A, b, \ell, u)$ includes the constraints $-Ax^+ + Ax^- = -b$, indexed by $\widehat{V}_2$, which are redundant as they are directly implied by negating $Ax^+ - Ax^- = b$. Thus, they may be left out without changing the feasible region. Previous works do not consider the duplication of constraints. However, we will see that these redundant equations may be useful for the derivation of coarse equitable partitions. In particular, they may help to establish equitable partitions that arise if a subset of the rows are multiplied by -1.

In Lemma 7, we show that the split reformulation of any linear program is equivalent to its original linear program.

Lemma 7 *Let $A \in \mathbb{R}^{V \times W}$, $b \in \mathbb{R}^V$, $c \in \mathbb{R}^W$ and vectors ℓ, u, with $l_w \in \mathbb{R}_{\leq 0} \cup \{-\infty\}$ and $u_w \in \mathbb{R}_{\geq 0} \cup \{\infty\}$ for all $w \in W$. Then, the following hold:*

(i) If $x \in P(A, b, \ell, u)$, then for $x^+ \coloneqq \max(x, \mathbb{O})$ and $x^- \coloneqq \max(-x, \mathbb{O})$ it holds that $(x^+, x^-) \in P_S(A, b, \ell, u)$ and $c^\intercal x = c^\intercal x^+ - c^\intercal x^-$.

(ii) If $(x^+, x^-) \in P_S(A, b, \ell, u)$ then for $x \coloneqq x^+ - x^-$ it holds that $x \in P(A, b, \ell, u)$ and $c^\intercal x = c^\intercal x^+ - c^\intercal x^-$.

Proof To prove (i), let $x \in P(A, b, \ell, u)$ be given and consider $x^+ = \max(x, \mathbb{O})$ and $x^- = \max(-x, \mathbb{O})$. Then, we have:

$$Ax^+ - Ax^- = A(x^+ - x^-) = A(\max(x, \mathbb{O}) - \max(-x, \mathbb{O})) = Ax \overset{\text{(a)}}{=} b,$$

where (a) follows since $x \in P(A, b, \ell, u)$ holds. It follows directly that also $-Ax^+ + Ax^- = -(Ax^+ - Ax^-) = -b$ holds. Furthermore, note that $x^+ \geq 0$ and $x^- \geq 0$ hold by definition of the max operator. The upper bounds $x^+ \leq u$ and $x^- \leq -\ell$ follow from $x \leq u$ and $-x \leq -\ell$ which are satisfied since $x \in P(A, b, \ell, u)$ holds. Then, since $\ell \leq 0$ and $u \geq 0$, it follows that (x^+, x^-) is feasible for $P_S(A, b, \ell, u)$. Finally, note that $c^\intercal x = c^\intercal(\max(x, \mathbb{O}) - \max(-x, \mathbb{O})) = c^\intercal x^+ - c^\intercal x^-$ shows that the objectives are equal.

To prove (ii), let $(x^+, x^-) \in P_S(A, b, \ell, u)$ be given and consider $x \coloneqq x^+ - x^-$. Then, $Ax = Ax^+ - Ax^- = b$ holds since $(x^+, x^-) \in P_S(A, b, \ell, u)$. Since $x^+ \geq 0$ and $x^- \geq 0$ hold in $P_S(A, b, \ell, u)$, we also have that $x = x^+ - x^- \leq u + \mathbb{O} = u$ and $x = x^+ - x^- \geq \mathbb{O} - -\ell = \ell$, which shows that $\ell \leq x \leq u$ holds, and thus it follows that $x \in P(A, b, \ell, u)$. Finally, we have by definition of x that $c^\intercal x = c^\intercal(x^+ - x^-) = c^\intercal x^+ - c^\intercal x^-$. □

First of all, we note that $\ell \leq 0$ and $u \geq 0$ are necessary conditions to guarantee that the split reformulation exists in the proof of Lemma 7. For any linear programs with $\ell_w \leq u_w$ for variable x_w, one can achieve this by transforming x_w into x'_w using $x'_w = x_w + \delta_w$ for any δ_w that satisfies $\ell_w \leq \delta_w \leq u_w$. If $\ell_w \leq u_w$ does not hold, then the linear program is clearly infeasible.

For the detection of reflection symmetries, previous approaches detect reflection symmetries by detecting permutation symmetries on a mixed-integer linear program obtained from a split reformulation. Thus, a logical next step is to consider the application of DRCR to a split reformulation $F_S(A, b, \ell, u, c)$ for any linear program $F(A, b, \ell, u, c)$ with $\ell \leq \mathbb{O}$ and $u \geq \mathbb{O}$. To do so, we first need to understand some of the special properties of equitable partitions of split reformulations.

3.3 Equitable partitions of split reformulations

Throughout this section, we investigate the structure of equitable partitions of the split reformulation $F_S(A, b, \ell, u, c)$ for an arbitrary linear program $F(A, b, \ell, u, c)$ with $\ell \leq \mathbb{O}$ and $u \geq \mathbb{O}$.

Recall that for a split reformulation, its variable index sets $\widehat{W} = \widehat{W}^+ \cup \widehat{W}^-$ and $\widehat{V} = \widehat{V}^+ \cup \widehat{V}^-$ correspond to the original and negated variables and constraints, respectively. For $x \in \{-1, 1\}$, we will use the synonyms $\widehat{V}^x \coloneqq \begin{cases} \widehat{V}^+ \text{ if } x = 1 \\ \widehat{V}^- \text{ if } x = -1 \end{cases}$ and $\widehat{W}^x \coloneqq \begin{cases} \widehat{W}^+ \text{ if } x = 1 \\ \widehat{W}^- \text{ if } x = -1 \end{cases}$ to denote the corresponding signed sets and for $v \in V$ ($w \in W$) we similarly use v^x (w^x) to indicate that v lies in $\widehat{V}^x$ ($\widehat{W}^x$). For any $P \subseteq V$ and $\gamma \in \{-1, 1\}^P$, and, we use $P^\gamma \coloneqq \{v^{\gamma_v} \in \widehat{V}^{\gamma_v} \mid v \in P\}$ to denote the signed set of indices corresponding in $\widehat{V}^+$ (if $\gamma_v = 1$) and $\widehat{V}^-$ (if $\gamma_v = -1$). For example, for $P = \{v_1, v_2, v_3\}$ and $\gamma \coloneqq \begin{bmatrix} \gamma_{v_1} & \gamma_{v_2} & \gamma_{v_3} \end{bmatrix} = \begin{bmatrix} 1 & -1 & -1 \end{bmatrix}$, we would have that $P^\gamma = \{v_1^+, v_2^-, v_3^-\}$ and that $P^{-\gamma} = \{v_1^-, v_2^+, v_3^+\}$, where $v_1^+, v_2^+, v_3^+ \in \widehat{V}_2^+$ and $v_1^-, v_2^-, v_3^- \in \widehat{V}_2^-$. Similarly, for any $Q \subseteq W$ and $\lambda \in \{-1, 1\}^Q$, we define $Q^\lambda \coloneqq \{w^{\lambda_w} \in \widehat{W}^{\lambda_w} \mid w \in Q\}$.

Then, we consider the structure of equitable partitions $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ of $F_S(A, b, \ell, u, c)$ in further detail. These equitable partitions have many symmetries that are a result of the redundancies introduced by the split reformulation. One property that will be important in this context, is the *polarity* of parts $\widehat{P} \in \widehat{\mathcal{P}}$. If for some $v \in V$, both $v^+ \in \widehat{P}$ and $v^- \in \widehat{P}$ hold then $\widehat{P}$ is said to be *bipolar*. If $\widehat{P} \subseteq \widehat{V}$ is not bipolar, then $|\widehat{P} \cap \{v^+, v^-\}| \leq 1$ holds for all $v \in V$ and we say it is *unipolar*. Note that for each unipolar part $\widehat{P}$ there exists a part $P \subseteq V$ and $\gamma \in \{-1, 1\}^V$ such that $\widehat{P} = P^\gamma$ holds. Similarly, $\widehat{Q} \subseteq \widehat{W}$ is *bipolar* if there exists some $w \in W$ such that $w^+ \in \widehat{Q}$ and $w^- \in Q$ hold, and $\widehat{Q}$ is said to be *unipolar* if $|\widehat{Q} \cap \{w^+, w^-\}| \leq 1$ holds for all $w \in W$. If $\widehat{Q}$ is unipolar, then there exists $Q \subseteq W$ and $\lambda \in \{-1, 1\}^W$ such that $\widehat{Q} = Q^\lambda$ holds. In Definition 8, we define the symmetric structure of equitable partitions of the split reformulation $F_S(A, b, \ell, u, c)$ of any linear program $F(A, b, \ell, u, c)$ with row and column indices V and W, respectively. The sets $\widehat{V}$ and $\widehat{W}$ are the row and column indices of $F_S(A, b, \ell, u, c)$ as defined in Definition 6.

Definition 8 *Let $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ be a partition of $\widehat{V} \times \widehat{W}$. We say that $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ is* symmetric *if the following hold.*

(i) for every unipolar part $P \in \widehat{\mathcal{P}}$ such that $P = P^\gamma$ holds, $P^{-\gamma} \in \widehat{\mathcal{P}}$ holds too.

(ii) for every bipolar part $P \in \widehat{\mathcal{P}}$ and any $v \in V$, P either contains both v^+ and v^- or none.

(iii) for every unipolar part $Q \in \widehat{\mathcal{Q}}$ such that $Q = Q^\lambda$ holds, $Q^{-\lambda} \in \widehat{\mathcal{Q}}$ holds too.

(iv) for every bipolar part $Q \in \widehat{\mathcal{Q}}$ and any $w \in W$, Q either contains both w^+ and w^- or none.

For a symmetric equitable partition $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ of $F_S(A, b, \ell, u, c)$, we use $\widehat{\mathcal{P}}_B \coloneqq \{P \in \widehat{\mathcal{P}} \mid P \text{ is bipolar}\}$ and $\widehat{\mathcal{P}}_U \coloneqq \{P \in \widehat{\mathcal{P}} \mid P \text{ is unipolar}\}$ and note that $\widehat{\mathcal{P}}_U$ and $\widehat{\mathcal{P}}_B$ are a disjoint partition of $\widehat{\mathcal{P}}$. Similarly, we define that $\widehat{\mathcal{Q}}_B \coloneqq \{Q \in \widehat{\mathcal{Q}} \mid Q \text{ is bipolar}\}$ and $\widehat{\mathcal{Q}}_U \coloneqq \{Q \in \widehat{\mathcal{Q}} \mid Q \text{ is unipolar}\}$, which also partitions $\widehat{Q}$. Furthermore, we use $\widehat{V}_U \coloneqq \bigcup_{P \in \widehat{\mathcal{P}}_U} P$, $\widehat{V}_B \coloneqq \bigcup_{P \in \widehat{\mathcal{P}}_B} P$, $\widehat{W}_U \coloneqq \bigcup_{Q \in \widehat{\mathcal{Q}}_U} Q$ and $\widehat{W}_B \coloneqq \bigcup_{Q \in \widehat{\mathcal{Q}}_B} Q$ for the corresponding partitioned subsets of $\widehat{V}$ and $\widehat{W}$. To denote the corresponding sets in V and W we use $V_U \coloneqq \{v \in V \mid \{v^+, v^-\} \cap \widehat{V}_U \neq \emptyset\}$, $V_B \coloneqq \{v \in V \mid \{v^+, v^-\} \subseteq \widehat{V}_B\}$, $W_U \coloneqq \{w \in W \mid \{w^+, w^-\} \cap \widehat{W}_U \neq \emptyset\}$ and $W_B \coloneqq \{w \in W \mid \{w^+, w^-\} \subseteq \widehat{W}_B\}$.

The notion of bipolar and unipolar parts in Definition 8 is very similar to an observation by Bödi, Herr and Joswig [4], who show that the orbits of reflection symmetry groups are either unipolar or bipolar, where bipolar orbits must consist of only pairs w^+, w^- for some subset of W. Similarly, we will show that the equitable partitions of $F_S(A, b, \ell, u, c)$, that the associated equitable partitions are indeed symmetric and satisfy Definition 8.

Next, we show one important property of symmetric partitions: bipolar parts correspond to zero row and column sums.

Lemma 9 *Let $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ be any partition of $F_S(A, b, \ell, u, c)$ and let $(P', Q') \in \widehat{\mathcal{P}} \times \widehat{\mathcal{Q}}$. Then, the following hold:*

(i) If $P' = P^{\mathbb{1}} \cup P^{-\mathbb{1}}$ holds for some set $P \subseteq V$, then $\sum_{v \in P'} \widehat{M}_{v,w} = 0$ holds for all $w \in W$.

(ii) If $Q' = Q^{\mathbb{1}} \cup Q^{-\mathbb{1}}$ holds for some set $Q \subseteq W$, then $\sum_{w \in Q'} \widehat{M}_{v,w} = 0$ holds for all $v \in V$.

Proof To prove (i), consider any $w \in W$. Then, we have that

$$\sum_{v \in P'} \widehat{M}_{v,w} = \sum_{v^+ \in P^{\mathbb{1}}} \widehat{M}_{v^+,w} + \sum_{v^- \in P^{-\mathbb{1}}} \widehat{M}_{v^-,w} = \sum_{v \in P} A_{v,w} + \sum_{v \in P} -A_{v,w} = 0,$$

where the first equality is given by the condition. The second equality can be observed through case distinction whether $w \in W^+$ or $w \in W^-$ holds. If $w \in W^+$ holds then by definition of $\widehat{M}$ we have $\widehat{M}_{v^+,w} = A_{v,w}$ and $\widehat{M}_{v^-,w} = -A_{v,w}$. For $w \in W^-$ we have $\widehat{M}_{v^+,w} = -A_{v,w}$ and $\widehat{M}_{v^-,w} = A_{v,w}$. In both cases, the equality follows.

The proof for (ii) is similar, where we have for all $v \in V$ that:

$$\sum_{w \in Q'} \widehat{M}_{v,w} = \sum_{w^+ \in Q^{\mathbb{1}}} \widehat{M}_{v,w^+} + \sum_{w^- \in Q^{-\mathbb{1}}} \widehat{M}_{v,w^-} = \sum_{w \in Q} A_{v,w} + \sum_{w \in Q} -A_{v,w} = 0.$$

□

Corollary 10 Let $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ be any symmetric equitable partition of $F_S(A, b, \ell, u, c)$ and let $(P, Q) \in (\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$. If at least one of P and Q is bipolar, then $\sum_{v \in P} \widehat{M}_{v,w} = 0$ holds for all $w \in Q$ and $\sum_{w \in Q} \widehat{M}_{v,w} = 0$ holds for all $v \in P$.

Proof We have $\sum_{v\in P}\sum_{w\in Q}\widehat{M}_{v,w} = \sum_{w\in Q}\sum_{v\in P}\widehat{M}_{v,w} = 0$, where the final equality holds by Lemma 9, since bipolarity of either Q or P implies that $\sum_{w\in Q}\widehat{M}_{v,w} = 0$ holds for all $v \in P$ or $\sum_{v\in P}\widehat{M}_{v,w} = 0$ holds for all $w \in Q$. By equitability of the block (P, Q), the row and column sums are identical, and it follows that $\sum_{w\in Q}\widehat{M}_{v,w} = \frac{1}{|P|}\sum_{v'\in P}\sum_{w\in Q}\widehat{M}_{v',w} = 0$ holds for all $v \in P$ and $\sum_{v\in P}\widehat{M}_{v,w} = \frac{1}{|Q|}\sum_{w'\in Q}\sum_{v\in P}\widehat{M}_{v,w'} = 0$ holds for all $w' \in Q$. □

In order to investigate symmetric equitable partitions of split reformulations, we need a key assumption that limits the structure of the equitable partitions for the split reformulation. One disadvantage of the split reformulation is that the x^+- and x^--variables may interact in non-identical ways, which makes it tough to interpret the computed partition of $\widehat{V}$ and $\widehat{W}$ in the context of the original linear program $F(A, b, \ell, u, c)$. For example, for variables $w_1, w_2 \in W$, it may be the case that there exists a part $\{w_1^+, w_1^-, w_2^+\}$ which acts identically on the whole domain of w_1 but only on the positive domain of w_2. To avoid such issues, we consider only partitions of the split reformulation such that for each part $Q \in \widehat{\mathcal{Q}}$, the associated variables in Q of the linear program $F(A, b, \ell, u, c)$ have identical variable bounds.

Definition 11 *A partition $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ of $\widehat{V} \times \widehat{W}$ is said to be* bound-respecting *if for all $Q \in \widehat{\mathcal{Q}}$ and any $w_1^{s_1}, w_2^{s_2} \in Q$ we have:*

(i) $\ell_{w_1} = \ell_{w_2}$ *and* $u_{w_1} = u_{w_2}$ *if* $s_1 = s_2$ *holds*

(ii) $\ell_{w_1} = -u_{w_2}$ *and* $u_{w_1} = -\ell_{w_2}$ *if* $s_1 \neq s_2$ *holds*

Then, we show that the split reformulation $F_S(A, b, \ell, u, c)$ admits a symmetric equitable partition in two steps. First, we show in Lemma 12 that the initial coarsest bound-respecting equitable partition is symmetric. Second, we show in Lemma 13 that any refinements of the initial partition from the constraint matrix preserve the symmetry of the partition.

Lemma 12 *Let $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ be the coarsest bound-respecting partition of $F_S(A, b, \ell, u, c)$ such that $\widehat{\mathcal{P}}$ is equitable for $\widehat{b} \coloneqq \begin{bmatrix} b & -b \end{bmatrix}$ and $\widehat{\mathcal{Q}}$ is equitable for the $\widehat{c} \coloneqq \begin{bmatrix} c & -c \end{bmatrix}^\intercal$ and $\widehat{u} \coloneqq \begin{bmatrix} u & -\ell \end{bmatrix}^\intercal$. Then, $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ is symmetric. Moreover, $b_v = 0$ holds for all $v \in \widehat{V}_B$ and $c_w = 0$ and $u_w = -\ell_w$ hold for all $w \in W_B$.*

Proof First, consider the coarsest equitable partition $\widehat{\mathcal{P}}$ of the right-hand side $\widehat{b} \in \mathbb{R}^{\widehat{V}^+ \cup \widehat{V}^-}$. Consider any bipolar part $P \in \widehat{\mathcal{P}}$ such that for some $v \in V$, $\{v^+, v^-\} \subseteq P$ holds. Then, since $\widehat{\mathcal{P}}$ is an equitable partition of $\widehat{b}$, it follows that $b_v = \widehat{b}_{v^+} = \widehat{b}_{v^-} = -b_v$ holds, which implies that $b_v = 0$. As this holds for all $v \in V$ with $b_v = 0$, it follows that the block P in the coarsest equitable partition is exactly formed by those rows of b that have right-hand side 0, i.e. $P = \{v^+, v^- \mid \text{ such that } b_v = 0 \text{ for } v \in V\}$ is the only bipolar part in $\widehat{\mathcal{P}}$. Clearly, P satisfies Definition 8(ii).

For any unipolar part $\widehat{P} \in \widehat{\mathcal{P}}$ holds such that $\widehat{P} = P^\gamma$ for suitable $P \subseteq V$ and $\gamma \in \{-1, 1\}^V$, it follows from equitability of $\widehat{\mathcal{P}}$ for $\widehat{b}$ that for all $v_1^\gamma, v_2^\gamma \in \widehat{P}$ that $\widehat{b}_{v_1^\gamma} = \gamma_{v_1} b_{v_1} = \gamma_{v_2} b_{v_2} = \widehat{b}_{v_2^\gamma}$ holds. However, this directly implies for the set $P^{-\gamma} \subseteq \widehat{V}$ that

$\widehat{b}_{v_1^{-\gamma}} = -\gamma_{v_1} b_{v_1} = -\gamma_{v_2} b_{v_2} = \widehat{b}_{v_1^{-\gamma}}$ holds, which shows that $P^{-\gamma}$ is also an equitable block. Thus, $\widehat{\mathcal{P}}$ can only be the coarsest equitable partition of $\widehat{b}$ if there exists some block $\widehat{P}'$ such that $P^{-\gamma} \subseteq \widehat{P}'$. One can show using a similar argument that $P^{-\gamma} = \widehat{P}'$ holds: existence of any $v^{-\gamma} \in \widehat{P}' \setminus P^{-\gamma}$ contradicts that $\widehat{\mathcal{P}}$ is the coarsest equitable partition of $\widehat{b}$ as then the part containing v^{γ} can be merged with $\widehat{P}$ to form a coarser partition. Since the above reasoning holds for all unipolar blocks, it follows that Definition 8(i) is satisfied.

Second, we consider the coarsest bound-respecting equitable partition $\widehat{\mathcal{Q}}$ of $\widehat{c}$ and $\widehat{u}$. Then, for any $Q \in \widehat{\mathcal{Q}}$ we have for $\widehat{w}_1, \widehat{w}_2 \in Q$ that $\widehat{c}_{\widehat{w}_1} = \widehat{c}_{\widehat{w}_2}$ and $\widehat{u}_{\widehat{w}_1} = \widehat{u}_{\widehat{w}_2}$ hold. First, consider the case where $Q \in \widehat{\mathcal{Q}}$ is bipolar, and let $w_1 \in W$ be some index such that $\{w_1^+, w_1^-\} \subseteq Q$ holds. Then, it follows from equitability for $\widehat{c}$ that $c_{w_1} = \widehat{c}_{w_1^+} = \widehat{c}_{w_1^-} = -c_{w_1}$ holds, which implies that $c_{w_1} = 0$. Furthermore, equitability for $\widehat{u}$ implies that $u_{w_1} = \widehat{u}_{w_1^+} = \widehat{u}_{w_1^-} = -\ell_{w_1}$ holds. Then, assume for the sake of contradiction that there exists any $w_2 \in W$ such that $|\{w_2^+, w_2^-\} \cap Q| = 1$ holds. If $w_2^+ \in Q$ holds, then $\widehat{u}_{w_2^-} = -\ell_{w_2} = -\ell_{w_1} = \widehat{u}_{w_1^-} = \widehat{u}_{w_2^+}$ shows that w_2^- has the same bound for $\widehat{u}$, where the second equality follows since $\widehat{\mathcal{Q}}$ is a bound-respecting partition and w_2^+ and w_1^+ are both included in Q. Similarly, if $w_2^- \in Q$ holds, then $\widehat{u}_{w_2^+} = u_{w_2} = u_{w_1} = \widehat{u}_{w_1^+} = \widehat{u}_{w_2^-}$ holds since $\widehat{\mathcal{Q}}$ is a bound-respecting partition and Q contains both w_1^- and w_2^-. In either case, it follows that $\widehat{u}_{w_2^+} = \widehat{u}_{w_2^-}$ holds. As one of w_2^+ and w_2^- is in Q, it follows that $\widehat{c}_{w_2^+} = \widehat{c}_{w_2^-} = c_{w_2} = 0$ also holds, which contradicts that $\widehat{\mathcal{Q}}$ is the coarsest bound-respecting partition as then both w_2^+ and w_2^- may be included in Q. Thus, bipolar parts must always contain pairs of variables and Definition 8(iv) follows.

Finally, consider the case where $\widehat{Q} \in \widehat{\mathcal{Q}}$ is unipolar and given by $\widehat{Q} = Q^{\lambda}$ with $Q \subseteq W$ and $\lambda \in \{-1, 1\}^W$. By equitability of $\widehat{\mathcal{Q}}$, we must have for all $w_1^{\lambda}, w_2^{\lambda} \in Q^{\lambda}$ that $\widehat{c}_{w_1^{\lambda}} = \widehat{c}_{w_2^{\lambda}}$ and $\widehat{u}_{w_1^{\lambda}} = \widehat{u}_{w_2^{\lambda}}$ hold. Then, consider $Q^{-\lambda} \subseteq \widehat{W}$. First of all, note that for any $w_1^{-\lambda}, w_2^{-\lambda} \in Q^{-\lambda}$ that we have: $\widehat{c}_{w_1^{-\lambda}} = -\widehat{c}_{w_1^{\lambda}} = -\widehat{c}_{w_2^{\lambda}} = \widehat{c}_{w_2^{-\lambda}}$, where the first and last equalities follow by definition of $\widehat{c}$ and the second equality follows since w_1^{λ} and w_2^{λ} lie in the equitable part Q^{λ}. Now, let us show that for the bounds $\widehat{u}_{w_1^{-\lambda}} = \widehat{u}_{w_2^{-\lambda}}$ holds too:

$$\widehat{u}_{w_1^{-\lambda}} = \begin{cases} -\ell_{w_1} & \text{if } \lambda_{w_1} = 1 \\ u_{w_1} & \text{if } \lambda_{w_1} = -1 \end{cases} = \begin{cases} -\ell_{w_1} & \text{if } \lambda_{w_1} = 1 \text{ and } \lambda_{w_2} = 1 \\ -\ell_{w_1} & \text{if } \lambda_{w_1} = 1 \text{ and } \lambda_{w_2} = -1 \\ u_{w_1} & \text{if } \lambda_{w_1} = -1 \text{ and } \lambda_{w_2} = 1 \\ u_{w_1} & \text{if } \lambda_{w_1} = -1 \text{ and } \lambda_{w_2} = -1 \end{cases}$$

$$\overset{\text{(a)}}{=} \begin{cases} -\ell_{w_2} & \text{if } \lambda_{w_1} = 1 \text{ and } \lambda_{w_2} = 1 \\ u_{w_2} & \text{if } \lambda_{w_1} = 1 \text{ and } \lambda_{w_2} = -1 \\ -\ell_{w_2} & \text{if } \lambda_{w_1} = -1 \text{ and } \lambda_{w_2} = 1 \\ u_{w_2} & \text{if } \lambda_{w_1} = -1 \text{ and } \lambda_{w_2} = -1 \end{cases} = \begin{cases} -\ell_{w_2} & \text{if } \lambda_{w_2} = 1 \\ u_{w_2} & \text{if } \lambda_{w_2} = -1 \end{cases} = \widehat{u}_{w_2^{-\lambda}}.$$

Here, (a) follows since $\widehat{\mathcal{Q}}$ is a bound-respecting partition and w_1^{λ} and w_2^{λ} are in $Q \in \widehat{\mathcal{Q}}$. Thus $Q^{-\lambda} \subseteq Q'$ must hold for some $Q' \in \widehat{\mathcal{Q}}$. Using a similar argument, one can show that existence of any $w_x^{-\lambda} \in Q' \setminus Q^{-\lambda}$ shows that the part containing w_x^{λ} can be merged with Q^{λ}, which contradicts that $\widehat{\mathcal{Q}}$ is the coarsest partition. Thus $Q^{-\lambda}$ belongs to $\widehat{\mathcal{Q}}$, which shows that Definition 8(iii) is satisfied. □

Note in Lemma 12 that the lower bounds of variables x^+ and x^- are not considered as they are all zero and thus do not affect the equitable partition of $\widehat{W}$. Next, let us show that refinement of symmetric partitions through the constraint matrix again yields a symmetric partition.

Lemma 13 *Consider the split reformulation $F_S(A, b, \ell, u, c)$ of a linear program, where the constraint matrix of $F_S(A, b, \ell, u, c)$ is given by $\widehat{M} = \begin{bmatrix} A & -A \\ -A & A \end{bmatrix}$, and let $(\widehat{\mathcal{P}}_0, \widehat{\mathcal{Q}}_0)$ be a symmetric partition of $F(A, b, \ell, u, c)$. Then, the coarsest equitable partition $(\widehat{\mathcal{P}}_\infty, \widehat{\mathcal{Q}}_\infty)$ that refines $(\widehat{\mathcal{P}}_0, \widehat{\mathcal{Q}}_0)$ with respect to $\widehat{M}$ is symmetric.*

Proof By Theorem 2 in [35], the coarsest equitable partition is unique. Thus, it suffices to show that there exists a sequence of refinements such that the coarsest equitable partition is symmetric. We prove the statement by showing that there exists a refinement algorithm that maintains a symmetric partition at every step. In particular, we assume that $(\widehat{\mathcal{P}}_i, \widehat{\mathcal{Q}}_i)$ is symmetric, and show that existence of any refinement of $(\widehat{\mathcal{P}}_i, \widehat{\mathcal{Q}}_i)$ implies that there exists a refinement into a partition $(\widehat{\mathcal{P}}_{i+1}, \widehat{\mathcal{Q}}_{i+1})$ that is symmetric. The result then follows inductively, where the base case is satisfied by assumption.

Then, assume that $(\widehat{\mathcal{P}}_i, \widehat{\mathcal{Q}}_i)$ is a symmetric partition of $F_S(A, b, \ell, u, c)$ such that there exists some $(P', Q') \in (\widehat{\mathcal{P}}_i, \widehat{\mathcal{Q}}_i)$ such that $\widehat{M}_{P',Q'}$ is not an equitable block. We consider the case where $\widehat{M}_{P',Q'}$ does not satisfy (1) for some $v, v' \in P'$ such that $\sum_{w \in Q'} \widehat{M}_{v,w} \neq \sum_{w \in Q'} \widehat{M}_{v',w}$, and show that then $\widehat{\mathcal{P}}_i$ can be refined symmetrically so that $(\widehat{\mathcal{P}}_{i+1}, \widehat{\mathcal{Q}}_i)$ is a symmetric partition that refines $(\widehat{\mathcal{P}}_i, \widehat{\mathcal{Q}}_i)$. The argumentation for the case where $\widehat{M}_{P',Q'}$ does not satisfy (2) is identical up to transposition.

First, note that if Q' is bipolar, that then it follows by symmetry of $\widehat{P}_i$ and Lemma 9 that $\sum_{v,w} \widehat{M}_{v,w} = 0$ holds for all $v \in P'$. Hence (1) is satisfied in this case, and it is sufficient to consider the case where Q' is unipolar in the following. We let $Q' = Q^\lambda$ for some $Q \subseteq W$ and $\lambda \in \{-1, 1\}^W$ and use $Q^+ \coloneqq \{w \in Q \mid w^+ \in Q^\lambda\}$ and $Q^- \coloneqq \{w \in Q \mid w^- \in Q^\lambda\}$ to denote the positive and negative variables in Q, respectively. Next, we show that refinement of the block $\widehat{M}_{P',Q'}$ always produces a symmetric partition. To do so, we distinguish the cases whether P' is bipolar or unipolar.

Case 1: P' is bipolar.

If P' is bipolar, then there exists $P \subseteq V$ such that $P' = P^{\mathbb{1}} \cup P^{-\mathbb{1}}$. Then, since Q' is unipolar, we have that:

$$\widehat{M}_{P',Q'} = \widehat{M}_{P',Q^\lambda} = \begin{bmatrix} \widehat{M}_{P^{\mathbb{1}},Q^\lambda \cap W_2^+} & \widehat{M}_{P^{\mathbb{1}},Q^\lambda \cap W_2^-} \\ \widehat{M}_{P^{-\mathbb{1}},Q^\lambda \cap W_2^+} & \widehat{M}_{P^{-\mathbb{1}},Q^\lambda \cap W_2^-} \end{bmatrix} = \begin{bmatrix} A_{P,Q^+} & -A_{P,Q^-} \\ -A_{P,Q^+} & A_{P,Q^-} \end{bmatrix} = \begin{bmatrix} G \\ -G \end{bmatrix},$$

where $G \coloneqq \begin{bmatrix} A_{P,Q^+} & -A_{P,Q^-} \end{bmatrix}$. Then, consider the refinement of $\widehat{\mathcal{P}}_i$ into $\widehat{\mathcal{P}}_{i+1}$ obtained by splitting P' into k different classes based on the different row sums of $\widehat{M}_{P',Q'}$. Let P'_j for $j = 1, \cdots, k$ denote these parts, where we have for all $v, v' \in P'_j$ that $\sum_{w \in Q'} \widehat{M}_{v,w} = \sum_{w \in Q'} \widehat{M}_{v',w}$, and define $\widehat{\mathcal{P}}_{i+1} \coloneqq (\widehat{\mathcal{P}}_i \setminus \{P'\}) \cup \bigcup_{j=1}^k P'_j$. We show that (i) and (ii) in Definition 8 are satisfied for all P'_j, which is sufficient since they are satisfied by assumption for all $P'' \in \widehat{\mathcal{P}}_i \setminus \{P'\}$.

First, consider the case where P'_j is bipolar. Since P'_j contains at least one pair $v^+, v^- \in P'_j$ for some $v \in V_2$, we can deduce that $\sum_{w \in Q'} G_{v,w} = \sum_{w \in Q'} \widehat{M}_{v,w} = \sum_{w \in Q'} \widehat{M}_{v',w} = \sum_{w \in Q'} -G_{v,w} = 0$ holds. Then, note that for any $\gamma \in \{-1, 1\}$ such that $v^\gamma \in P'_j$, the rows of v^γ and $v^{-\gamma}$ both sum to zero, which implies that also $v^{-\gamma} \in P'_j$ holds by definition of P'_j, where we use the assumption that Definition 8(ii) holds for $\widehat{\mathcal{P}}_i$ to infer that $v^{-\gamma} \in P'$ holds. Thus, this shows that Definition 8(ii) is satisfied by $\widehat{\mathcal{P}}_{i+1}$ in the case where P' is bipolar.

If P'_j is unipolar, then there exists some $\gamma \in \{-1,1\}^V$ and $P \subseteq V$ such that $P'_j = P^\gamma$, and we have for all $v \in P'_j$ that $\sum_{w\in Q'} \widehat{M}_{v,w} = \gamma_v \sum_{w\in Q'} G_{v,w} = \alpha$ for some $\alpha \in \mathbb{R}\setminus\{0\}$, where we can exclude $\alpha = 0$ because this corresponds to the bipolar parts treated above. Since $\mathcal{P}'_i$ satisfies Definition 8(ii) by assumption, all elements of $P^{-\gamma}$ are also contained in P', and must be partitioned according to their sums. Then we have for all $v \in P^{-\gamma}$ that $\sum_{w\in Q'} \widehat{M}_{v,w} = -\gamma_v \sum_{w\in Q'} G_{v,w} = -\alpha$, which shows that there exists some j' such that $P'_{j'} = P^{-\gamma}$. Thus, we have shown that Definition 8(i) holds for $\widehat{\mathcal{P}}_{i+1}$ in the case where P' is bipolar. Thus, we have shown that $(\widehat{\mathcal{P}}_{i+1}, \widehat{\mathcal{Q}}_i)$ is symmetric and refines $(\widehat{\mathcal{P}}_i, \widehat{\mathcal{Q}}_i)$.

Case 2: P' is unipolar.
If P' is unipolar, then there exists $P \subseteq V$ and $\gamma \in \{-1,1\}^V$ such that $P' = P^\gamma$ holds. By the assumption that $\widehat{\mathcal{P}}_i$ is symmetric, it also holds that $P^{-\gamma} \in \widehat{\mathcal{P}}_i$. Then, we show that for P^γ and $P^{-\gamma}$ the corresponding blocks $\widehat{M}_{P^\gamma,Q'}$ and $\widehat{M}_{P^{-\gamma},Q'}$ can be simultaneously refined to preserve symmetry in $\widehat{\mathcal{P}}_{i+1}$. For $x \in \{\gamma,-\gamma\}$, let $P^x_+ \coloneqq \{v \in P \mid v^+ \in P^x\}$ and $P^x_- \coloneqq \{v \in P \mid v^- \in P^x\}$ denote the positive and negative variables associated to P^x, respectively. Note that $P^\gamma_+ = P^{-\gamma}_-$ and $P^\gamma_- = P^{-\gamma}_+$ hold. Then, for the considered blocks P^γ and $P^{-\gamma}$, we have that:

$$\begin{bmatrix} \widehat{M}_{P^\gamma,Q'} \\ \widehat{M}_{P^{-\gamma},Q'} \end{bmatrix} = \begin{bmatrix} \widehat{M}_{P^\gamma,Q^\lambda} \\ \widehat{M}_{P^{-\gamma},Q^\lambda} \end{bmatrix} = \begin{bmatrix} A_{P^\gamma_+,Q^+} & -A_{P^\gamma_+,Q^-} \\ -A_{P^\gamma_-,Q^+} & A_{P^\gamma_-,Q^-} \\ -A_{P^{-\gamma}_-,Q^+} & A_{P^{-\gamma}_-,Q^-} \\ A_{P^{-\gamma}_+,Q^+} & A_{P^{-\gamma}_+,Q^-} \end{bmatrix} = \begin{bmatrix} A_{P^\gamma_+,Q^+} & -A_{P^\gamma_+,Q^-} \\ -A_{P^\gamma_-,Q^+} & A_{P^\gamma_-,Q^-} \\ -A_{P^\gamma_+,Q^+} & A_{P^\gamma_+,Q^-} \\ A_{P^\gamma_-,Q^+} & A_{P^\gamma_-,Q^-} \end{bmatrix} = \begin{bmatrix} G \\ -G \end{bmatrix},$$

where $G = \widehat{M}_{P^\gamma,Q'} = \begin{bmatrix} A_{P^\gamma_+,Q^+} & -A_{P^\gamma_+,Q^-} \\ -A_{P^\gamma_-,Q^+} & A_{P^\gamma_-,Q^-} \end{bmatrix}$. Thus, the block corresponding to $(P^{-\gamma}, Q')$ is the negated block of (P^γ, Q').

For $j = 1,\dots,k$, let P_j denote the partition of P into P_j so that the row sums of all $v \in P^\gamma_j$ are identical, and for $j \neq j'$ and $v \in P_j$ and $v \in P_{j'}$, the row sums of v and v' differ. Then, we similarly define the partition $P^{-\gamma}_j$ for $j = 1,\dots,k$. For $v, v' \in P^{-\gamma}$, note that

$$\sum_{w\in Q'} \widehat{M}_{v,w} = \sum_{w\in Q'} -G_{v,w} = -\sum_{w\in Q'} G_{v,w} \overset{\text{(a)}}{=} -\sum_{w\in Q'} G_{v',w} = \sum_{w\in Q'} -G_{v',w} = \sum_{w\in Q'} \widehat{M}_{v',w}$$

where (a) holds because P^γ_j has the same row sums by definition. Thus, the partitions P^γ_j and $P^{-\gamma}_j$ are both valid refinements. Note that for each $j = 1,\cdots,k$, the pair P^γ_j and $P^{-\gamma}_j$ is a unipolar pair as in Definition 8(i). Then, the partition defined by $\widehat{\mathcal{P}}_{i+1} \coloneqq \widehat{\mathcal{P}}_i \setminus \{P^\gamma, P^{-\gamma}\} \cup \bigcup_{j=1}^k (P^\gamma_j \cup P^{-\gamma}_j)$ satisfies Definition 8(i), and (ii) follows since we did not alter any bipolar part in $\widehat{\mathcal{P}}_i$. Thus, $(\widehat{\mathcal{P}}_{i+1}, \widehat{\mathcal{Q}}_i)$ is symmetric and refines $(\widehat{\mathcal{P}}_i, \widehat{\mathcal{Q}}_i)$.

Thus, we have shown that there exists a symmetric refinement $(\widehat{\mathcal{P}}_{i+1}, \widehat{\mathcal{Q}}_{i+1})$ in all cases, where $\widehat{\mathcal{Q}}_{i+1} = \widehat{\mathcal{Q}}_i$ holds. Using essentially the same proof, one can show that there exists a symmetric refinement $(\widehat{\mathcal{P}}_{i+1}, \widehat{\mathcal{Q}}_{i+1})$ with $\widehat{\mathcal{P}}_i = \widehat{\mathcal{P}}_{i+1}$ if there exists some $(P', Q') \in \widehat{\mathcal{P}}_i \times \widehat{\mathcal{Q}}_i$ such that (2) is not satisfied for $\widehat{M}$. This completes the proof. □

Theorem 14 *The coarsest bound-respecting and equitable partition $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ of $F_S(A, b, \ell, u, c)$ is symmetric.*

Proof Lemma 12 shows that the initial partition is symmetric, and it follows from Lemma 13 shows that applying color refinement on the constraint matrix produces a symmetric equitable partition, given the symmetric initial partition. □

Theorem 14 shows that symmetric equitable partitions of split reformulations can be obtained by using the assumption that the partition is bound-respecting. Although this assumption seems a bit arbitrary for now, we will motivate it in further detail later on.

A simple but crucial observation for symmetric partitions is that symmetric partition of the split reformulation induce partitions and signings for the original linear program.

Proposition 15 *Consider any symmetric partition $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$. For its unipolar parts $\widehat{\mathcal{P}}_U$ and $\widehat{\mathcal{Q}}_U$, there exist $\gamma \in \{-1,1\}^{V_U}$ and a partition $\mathcal{P}_U$ of V_U such that $\widehat{\mathcal{P}}_U = \bigcup_{P\in\mathcal{P}}\{P^\gamma, P^{-\gamma}\}$, and there exists $\lambda \in \{-1,1\}^{W_U}$ and a partition $\mathcal{Q}_U$ of W_U such that $\widehat{\mathcal{Q}}_U = \bigcup_{Q\in\mathcal{Q}}\{Q^\lambda, Q^{-\lambda}\}$. For its bipolar parts $\widehat{\mathcal{P}}_B$ and $\widehat{\mathcal{Q}}_B$, there exist partitions $\mathcal{P}_B$ of V_B and $\mathcal{Q}_B$ of W_B such that $\widehat{\mathcal{P}}_B = \{P^{\mathbb{1}} \cup P^{-\mathbb{1}} \mid P \in \mathcal{P}_B\}$ and $\widehat{\mathcal{Q}}_B = \{Q^{\mathbb{1}} \cup Q^{-\mathbb{1}} \mid Q \in \mathcal{Q}_B\}$ hold.*

Proof Consider any unipolar part $\widehat{P} \in \widehat{\mathcal{P}}$. By unipolarity, there exists $P \subseteq V$ such that $\widehat{P} = P^\gamma$. Then, since $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ is symmetric, Definition 8(i) holds and shows $P^{-\gamma} \in \widehat{\mathcal{P}}$ holds too. Since this holds for all elements and $\widehat{\mathcal{P}}$ is a partition, it follows that $\widehat{\mathcal{P}}$ consists of pairs P^γ, $P^{-\gamma}$. Then, $\mathcal{P}_U$ is generated by all such P and γ is defined by picking a representative part for each pair P^γ, $P^{-\gamma}$. A similar argument using Definition 8(iii)shows the result for λ and $\mathcal{Q}_U$. For the bipolar parts, the statement follows directly from Definition 8(ii)(iv). □

For any $\gamma \in \{-1,1\}^{V_U}$, $\lambda \in \{-1,1\}^{W_U}$, $\mathcal{P} \coloneqq \mathcal{P}_U \cup \mathcal{P}_B$ and $\mathcal{Q} \coloneqq \mathcal{Q}_U \cup \mathcal{Q}_B$ such as those in Proposition 15 we say that they are *generated by* $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$. We use $\widehat{P}_U^\gamma \coloneqq \{P^\gamma \mid P \in \mathcal{P}_U\}$ and $\widehat{Q}_U^\lambda \coloneqq \{Q^\lambda \mid Q \in \mathcal{Q}_U\}$, and define $\widehat{P}_U^{-\gamma}$ and $\widehat{Q}_U^{-\lambda}$ similarly. Note that the choice for γ is not unique as for each $P \in \mathcal{P}_U$ we can exchange P^γ and $P^{-\gamma}$ by negating γ_v for all $v \in P$, i.e. by using $\gamma' \coloneqq (-\gamma_P, \gamma_{V\setminus P})$. Similarly, one can also choose for each $Q \in \mathcal{Q}$ to simultaneously negate λ_w for all $w \in Q$. We note that these actions only affect the signs γ and λ and not the sets $\mathcal{P}_U$ and $\mathcal{Q}_U$.

3.4 Reducing the split reformulation

Now that we have a good understanding of the equitable partitions of split reformulations, we can consider how they can be used to reduce their dimension. In this section, we consider a linear program $F(A, b, \ell, u, c)$ and its split reformulation $F_S(A, b, \ell, u, c)$. We apply Proposition 3 to the split reformulation using a symmetric equitable partition $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ to obtain a smaller linear program.

To clarify the effect of the symmetric equitable partition on $F_S(A, b, \ell, u, c)$, we partition the row and variable sets into 3 sets each. Let $\gamma \in \{-1,1\}^{V_U}$, $\lambda \in \{-1,1\}^{W_U}$ and $\mathcal{P}_U$ and $\mathcal{Q}_U$ be generated by the symmetric equitable partition $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$. We consider the corresponding partition of $\widehat{V}$ and $\widehat{W}$ into $\widehat{V}_U^\gamma \coloneqq \bigcup_{P\in\widehat{\mathcal{P}}_U^\gamma} P$, $\widehat{V}_U^{-\gamma} \coloneqq \bigcup_{P\in\widehat{\mathcal{P}}_U^{-\gamma}} P$ and

$\widehat{V}_B$. Similarly, the column partition is defined by $\widehat{W}_U^\lambda \coloneqq \bigcup_{Q\in\widehat{\mathcal{Q}}_U^\lambda} Q$, $\widehat{W}_U^{-\lambda} \coloneqq \bigcup_{Q\in\widehat{\mathcal{Q}}_U^{-\lambda}} Q$ and $\widehat{W}_B$. Then, we consider the variable partition $(\widehat{x}_{\widehat{W}_U^\lambda}, \widehat{x}_{\widehat{W}_U^{-\lambda}}, \widehat{x}_{\widehat{W}_B})$ for the split reformulation $F_S(A,b,\ell,u,c)$. Due to the symmetric definition of $\widehat{M}$, rearranging of the variables $\widehat{W}$ into $\widehat{W}_U^\lambda$ and $\widehat{W}_U^{-\lambda}$ can be achieved by multiplying by $\Lambda \coloneqq \operatorname{diag}(\lambda)$ from the right. Similarly, rearranging $\widehat{V}$ into $\widehat{V}_U^\gamma$ and $\widehat{V}_U^{-\gamma}$ can be achieved by multiplying by $\Gamma \coloneqq \operatorname{diag}(\gamma)$ from the left. Furthermore, we remark that rows $v \in \widehat{V}_B$ have right-hand side 0 and that variables $w \in W_B$ have objective 0 and have identical bounds $u_w = -\ell_w$ for w^+ and w^- by Lemma 12.

To denote the variable bounds for the unipolar variables, we define the *bound selector vector* $\operatorname{bs}(\ell,u,\lambda) \in \mathbb{R}^W$ such that that $\operatorname{bs}(\ell,u,\lambda)_w \coloneqq \begin{cases} u_w & \text{if } \lambda_w = 1 \\ -\ell_w & \text{if } \lambda_w = -1. \end{cases}$ By permuting the variables of $F_S(A,b,\ell,u,c)$ and applying the mentioned substitutions, we obtain the following linear program:

$$\min\ c_{W_U}^\intercal \Lambda \widehat{x}_{\widehat{W}_U^\lambda} - c_{W_U}^\intercal \Lambda \widehat{x}_{\widehat{W}_U^{-\lambda}} \tag{7a}$$

$$\begin{bmatrix} \Gamma A^1 \Lambda & -\Gamma A^1 \Lambda & \widehat{M}^1 \\ -\Gamma A^1 \Lambda & \Gamma A^1 \Lambda & -\widehat{M}^1 \\ \widehat{M}^2 & -\widehat{M}^2 & \widehat{M}^3 \end{bmatrix} \begin{bmatrix} \widehat{x}_{\widehat{W}_U^\lambda} \\ \widehat{x}_{\widehat{W}_U^{-\lambda}} \\ \widehat{x}_{\widehat{W}_B} \end{bmatrix} = \begin{bmatrix} \Gamma b_{W_U} \\ -\Gamma b_{W_U} \\ \mathbb{O} \end{bmatrix} \tag{7b}$$

$$\mathbb{O} \le \widehat{x}_{\widehat{W}_U^\lambda} \le \operatorname{bs}(\ell,u,\lambda)_{W_U} \tag{7c}$$

$$\mathbb{O} \le \widehat{x}_{\widehat{W}_U^{-\lambda}} \le \operatorname{bs}(\ell,u,-\lambda)_{W_U} \tag{7d}$$

$$\mathbb{O} \le \widehat{x}_{\widehat{W}_B} \le \begin{bmatrix} u_{W_B} \\ u_{W_B} \end{bmatrix} \tag{7e}$$

where we use $A^1 \coloneqq A_{V_U,W_U}$, $\widehat{M}^1 \coloneqq \Gamma \begin{bmatrix} A_{V_U,W_B} & -A_{V_U,W_B} \end{bmatrix}$, $\widehat{M}^2 \coloneqq \begin{bmatrix} A_{V_B,W_U} \\ -A_{V_B,W_U} \end{bmatrix} \Lambda$ and $\widehat{M}^3 \coloneqq \begin{bmatrix} A_{V_B,W_B} & -A_{V_B,W_B} \\ -A_{V_B,W_B} & A_{V_B,W_B} \end{bmatrix}$.

We use $\widehat{y} \in \mathbb{R}^{\widehat{\mathcal{Q}}}$ to denote the transformed variables, where we let $(\widehat{y}_{\widehat{\mathcal{Q}}_U^\lambda}, \widehat{y}_{\widehat{\mathcal{Q}}_U^{-\lambda}}, \widehat{y}_{\widehat{\mathcal{Q}}_B})$ be the reduced variable sets corresponding to the variables $(\widehat{x}_{\widehat{W}_U^\lambda}, \widehat{x}_{\widehat{W}_U^{-\lambda}}, \widehat{x}_{\widehat{\mathcal{Q}}_B})$ in $F_S(A,b,\ell,u,c)$.

Then, we are almost ready to apply the equitable partition $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ to (7). Since we determined that $\widehat{\mathcal{P}}$ consisted of the partitions $\widehat{\mathcal{P}}_U^\gamma, \widehat{\mathcal{P}}_U^{-\gamma}$ and $\widehat{\mathcal{P}}_B$, $\Pi_{\widehat{\mathcal{P}}}$ is a block diagonal matrix consisting of the blocks $\Pi_{\widehat{\mathcal{P}}_U^\gamma}, \Pi_{\widehat{\mathcal{P}}_U^{-\gamma}}$ and $\Pi_{\widehat{\mathcal{P}}_B}$. Moreover, note that since $\widehat{\mathcal{P}}_U^\gamma$ and $\widehat{\mathcal{P}}_U^{-\gamma}$ arise from the same underlying partition $\mathcal{P}_U$ of the rows V_U, we have that $\Pi_{\widehat{\mathcal{P}}_U^\gamma} = \Pi_{\widehat{\mathcal{P}}_U^{-\gamma}} = \Pi_{\mathcal{P}_U}$. Similarly, $\Pi_{\widehat{\mathcal{Q}}}$ is a block diagonal matrix consisting of the blocks $\Pi_{\widehat{\mathcal{Q}}_U^\lambda}$, $\Pi_{\widehat{\mathcal{Q}}_U^{-\lambda}}$ and $\Pi_{\widehat{\mathcal{Q}}_{W_B}}$, where $\Pi_{\widehat{\mathcal{Q}}_U^\lambda} = \Pi_{\widehat{\mathcal{Q}}_U^{-\lambda}} = \Pi_{\mathcal{Q}_U}$ holds. In Corollary 10 we observed that equitable bipolar parts have zero row and column sums. In Lemma 16, we show that this implies that the matrices $\widehat{M}^i$ for $i = 1,2,3$ transform into all-zero matrices once we apply Proposition 3. Then, the reduction of $F_S(A,b,\ell,u,c)$ in the reordered form (7) is given by (8).

Lemma 16 *Let $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ be a symmetric equitable partition of $F_S(A, b, \ell, u, c)$, where $\widehat{M}$ is the constraint matrix of $F_S(A, b, \ell, u, c)$. Then, $\Pi_{\widehat{\mathcal{P}}_B}^\intercal \widehat{M}_{\widehat{V}_B,\star} = \mathbb{O}$ and $\widehat{M}_{\star,\widehat{W}_B} \widetilde{\Pi_{\widehat{\mathcal{Q}}_B}} = \mathbb{O}$ hold.*

Proof For any $P \in \widehat{\mathcal{P}}_B$ and any $w \in W$ we have that: $(\Pi_{\widehat{\mathcal{P}}_B}^\intercal \widehat{M}_{\widehat{V}_B,\star})_{P,w} = \sum_{v \in P} \widehat{M}_{v,w} = 0$, where the final equality follows from Corollary 10. Similarly, for any $Q \in \widehat{\mathcal{Q}}_B$ and any $v \in V$ we have that $(\widehat{M}_{\star,\widehat{W}_B} \widetilde{\Pi_{\widehat{\mathcal{Q}}_B}} = \frac{1}{Q} \sum_{w \in Q} \widehat{M}_{v,w} = 0$ holds by Corollary 10. □

$$\min (\widetilde{\Pi_{\mathcal{Q}_u}}^\intercal \Lambda c_{W_U})^\intercal \widehat{y}_{\widehat{\mathcal{Q}}_U^\lambda} - (\widetilde{\Pi_{\mathcal{Q}_u}}^\intercal \Lambda c_{W_U})^\intercal \widehat{y}_{\widehat{\mathcal{Q}}_U^{-\lambda}} \tag{8a}$$

$$\begin{bmatrix} \Pi_{\mathcal{P}_U}^\intercal \Gamma A^1 \Lambda \widetilde{\Pi_{\mathcal{Q}_U}} & -\Pi_{\mathcal{P}_U}^\intercal \Gamma A^1 \Lambda \widetilde{\Pi_{\mathcal{Q}_U}} & \mathbb{O} \\ -\Pi_{\mathcal{P}_U}^\intercal \Gamma A^1 \Lambda \widetilde{\Pi_{\mathcal{Q}_U}} & \Pi_{\mathcal{P}_U}^\intercal \Gamma A^1 \Lambda \widetilde{\Pi_{\mathcal{Q}_U}} & \mathbb{O} \\ \mathbb{O} & \mathbb{O} & \mathbb{O} \end{bmatrix} \begin{bmatrix} \widehat{y}_{\widehat{\mathcal{Q}}_U^\lambda} \\ \widehat{y}_{\widehat{\mathcal{Q}}_U^{-\lambda}} \\ \widehat{y}_{\widehat{\mathcal{Q}}_B} \end{bmatrix} = \begin{bmatrix} \Pi_{\mathcal{P}_U}^\intercal \Gamma b_{W_U} \\ -\Pi_{\mathcal{P}_U}^\intercal \Gamma b_{W_U} \\ \mathbb{O} \end{bmatrix} \tag{8b}$$

$$\mathbb{O} \leq \widehat{y}_{\widehat{\mathcal{Q}}_U^\lambda} \leq \Pi_{\mathcal{Q}_U}^\intercal \mathrm{bs}(\ell, u, \lambda)_{W_U} \tag{8c}$$

$$\mathbb{O} \leq \widehat{y}_{\widehat{\mathcal{Q}}_U^{-\lambda}} \leq \Pi_{\mathcal{Q}_U}^\intercal \mathrm{bs}(\ell, u, -\lambda)_{W_U} \tag{8d}$$

$$0 \leq \widehat{y}_{\widehat{\mathcal{Q}}_B} \leq 2\Pi_{\mathcal{Q}_B}^\intercal u_{W_B} \tag{8e}$$

There are a few things to note about the system (8). First of all, the rows corresponding to the bipolar rows clearly become redundant, and the columns $\widehat{y}_{\widehat{\mathcal{Q}}_B}$ corresponding to the bipolar variables become independent of the other variables. Indeed, (8a)-(8d) and (8e) are the Cartesian product of two polyhedra. Since $\widehat{y}_{\widehat{\mathcal{Q}}_B}$ does not appear in the objective, we may arbitrarily fix $\widehat{y}_{\widehat{\mathcal{Q}}_B} = 0$. Furthermore, the system (8a)-(8d) still has the redundancies introduced by the split reformulation of $F(A, b, \ell, u, c)$. In fact, the linear program on the variables $\widehat{\mathcal{Q}}_U^\lambda \cup \widehat{\mathcal{Q}}_U^{-\lambda}$ in (8a)-(8d) is exactly the split reformulation of another linear program! One can verify that (8a)-(8d) is equivalent to split reformulation G_S of the following linear program:

$$G \coloneqq F(\Pi_{\mathcal{P}_U}^\intercal \Gamma A^1 \Lambda \widetilde{\Pi_{\mathcal{Q}_U}}, \Pi_{\mathcal{P}_U}^\intercal \Gamma b_{W_U}, \Pi_{\mathcal{Q}_U}^\intercal \mathrm{bs}(\ell, u, \lambda)_{W_U}, -\Pi_{\mathcal{Q}_U}^\intercal \mathrm{bs}(\ell, u, -\lambda)_{W_U}, \widetilde{\Pi_{\mathcal{Q}_U}}^\intercal \Lambda c_{W_U}).$$

Proposition 17 *Let $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ be a symmetric equitable partition of $F_S(A, b, \ell, u, c)$, and let $\lambda, \gamma, \mathcal{P}$ and $\mathcal{Q}$ be generated by $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$. Then the reduced linear program can be defined as:*

$$R^{\widehat{\mathcal{P}},\widehat{\mathcal{Q}}}(F_S(A, b, \ell, u, c)) \coloneqq G_S \times \{\widehat{y}_{\widehat{\mathcal{Q}}_B} \in \mathbb{R}^{\widehat{\mathcal{Q}}_B} \mid \mathbb{O} \leq \widehat{y}_{\widehat{\mathcal{Q}}_B} \leq 2\Pi_{\mathcal{Q}_B}^\intercal u_{W_B}\}.$$

Proof By reordering the variables and rows of $F_S(A, b, \ell, u, c)$ according to the signs λ and γ for the unipolar variables we can obtain the form (7), where we implicitly use Lemma 12 to show that bipolar rows have zero right-hand side and bipolar variables have zero objective and identical upper and lower variable bounds.

Then, the reduced form $R^{\widehat{\mathcal{P}},\widehat{\mathcal{Q}}}(F_S(A, b, \ell, u, c))$ is given by (8), where we use Lemma 16 to show that the bipolar rows and variables have only zero coefficients. Then, note that the feasible region of the linear program (8) is exactly the Cartesian product of G_S and $\{\widehat{y}_{\widehat{\mathcal{Q}}_B} \in \mathbb{R}^{\widehat{\mathcal{Q}}_B} \mid \mathbb{O} \leq \widehat{y}_{\widehat{\mathcal{Q}}_B} \leq 2\Pi_{\mathcal{Q}_B}^\intercal u_{W_B}\}$ on the unipolar and bipolar parts, respectively, where we use that the all-zero rows with right-hand side zero in (8b) are redundant. □

Note that the linear program G is obtained from $F(A, b, \ell, u, c)$ by taking the linear program induced by V_U and W_U, scaling the rows and columns with Γ and Λ, applying the equitable partition $(\mathcal{P}_U, \mathcal{Q}_U)$ and then computing the split reformulation. Thus, application of the symmetric equitable partition to the split reformulation can also be viewed as taking the unipolar rows and variables in the original linear program, scaling the rows and variables by ± 1 factors, computing an equitable partition and then applying the split reformulation. The final step that computes the split reformulation is not helpful computationally, as G has less variables than G_S and is equivalent to G_S. Thus, we wish to reduce the linear programs into the form of G. We can obtain G from (8) by reversing the split reformulation as described in Lemma 7. We use $y_{\mathcal{Q}_U} \coloneqq \widehat{y}_{\widehat{\mathcal{Q}}_U^{\lambda}} - \widehat{y}_{\widehat{\mathcal{Q}}_U^{-\lambda}}$ to define the variables of G. Then, G is given by the following linear program.

$$\min (\widetilde{\Pi_{\mathcal{Q}_u}}^{\intercal} \Lambda c_{W_U})^{\intercal} y_{\mathcal{Q}_U} \tag{9a}$$

$$\Pi_{\mathcal{P}_U}^{\intercal} \Gamma A_{V_U, W_U} \Lambda \widetilde{\Pi_{\mathcal{Q}_U}} y_{\mathcal{Q}_U} = \Pi_{\mathcal{P}_U}^{\intercal} \Gamma b_{W_U} \tag{9b}$$

$$-\Pi_{\mathcal{Q}_U}^{\intercal} \operatorname{bs}(\ell, u, -\lambda)_{W_U} \leq y_{\mathcal{Q}_U} \leq \Pi_{\mathcal{Q}_U}^{\intercal} \operatorname{bs}(\ell, u, \lambda)_{W_U} \tag{9c}$$

Note that G is obtained only through transformations that preserve the feasibility and optimality of the linear program. In particular, both the split reformulation and the dimension reduction via color refinement preserve feasibility and optimality. Thus, we can reduce any linear program $F(A, b, \ell, u, c)$ to the form of G in (9) through a series of reformulations.

One small assumption in our analysis was that we required $\ell \leq 0$ and $u_w \geq 0$ to hold so that the split reformulations are well-defined. For any feasible linear program $\ell \leq u$ must hold, and it is then possible to choose $\delta \in \mathbb{R}^W$ such that $\ell \leq \delta \leq u$ holds. Then, one can apply an affine transformation $x = x' + \delta$ to the linear program $F(A, b, \ell, u, c)$. By doing so, one obtains the equivalent linear program $F(A, b - A\delta, \ell - \delta, u - \delta, c)$, where the objective function has an additional constant offset of $c^{\intercal}\delta$. Clearly, $\ell - \delta \leq 0$ and $u - \delta \geq 0$ hold, so the split reformulation is well defined.

In Definition 18, we formulate the concept of a *reflection reduction*, which represents the pair of linear programs given by $F(A, b, \ell, u, c)$ and its reduced form obtained by computing a symmetric equitable partition of $F_S(A, b - A\delta, \ell - \delta, u - \delta, c)$ for any $\ell \leq \delta \leq u$. In Theorem 19, we show that the symmetric equitable partitions of $F_S(A, b - A\delta, \ell - \delta, u - \delta, c)$ imply the existence of a reflection reduction.

In order to simplify the notation, we define $\operatorname{comp}(\lambda, \delta, \ell, u) \in \mathbb{R}^W$ to be a vector such that $\operatorname{comp}(\lambda, \delta, \ell, u)_w \coloneqq \begin{cases} \ell_w - \delta_w \text{ if } \lambda_w = 1 \\ \delta_w - u_w \text{ if } \lambda_w = -1 \end{cases}$ holds for all $w \in W$, where we define that $\infty + k = \infty$ for all finite $k \in \mathbb{R}$ and similarly that $-\infty + k = -\infty$. The vector $\operatorname{comp}(\lambda, \delta, \ell, u)$ corresponds to the complemented variable lower bounds. Note that swapping ℓ and u immediately also gives the complemented variable upper bounds using $\operatorname{comp}(\lambda, \delta, u, \ell)$. For the variable bound pair (ℓ, u), we will primarily use

the shorthand notation $\ell^{\delta,\lambda} \coloneqq \mathrm{comp}(\lambda, \delta, \ell, u)$ and $u^{\delta,\lambda} \coloneqq \mathrm{comp}(\lambda, \delta, u, \ell)$. One can verify that $\ell^{\delta,\lambda} = -\mathrm{bs}(\ell - \delta, u - \delta, -\lambda)$ and $u^{\delta,\lambda} = \mathrm{bs}(\ell - \delta, u - \delta, \lambda)$ hold.

Definition 18 (Reflection reduction) *Consider the linear program $F(A, b, \ell, u, c)$ with $A \in \mathbb{R}^{V \times W}$, $b \in \mathbb{R}^V$, $c \in \mathbb{R}^W$ and vectors ℓ, u with $\ell_w \in \mathbb{R} \cup \{-\infty\}$ and $u_w \in \mathbb{R} \cup \{\infty\}$ such that $\ell_w \leq u_w$ holds for all $w \in W$. Let $V_U \subseteq V$ and $W_U \subseteq W$, and define $W_B \coloneqq W \setminus W_U$. Consider a partition $(\mathcal{P}, \mathcal{Q})$ where $\mathcal{P}$ is a partition of V_U and $\mathcal{Q}$ is a partition of W_U, and let $\gamma \in \{-1, 1\}^{V_U}$, $\lambda \in \{-1, 1\}^{W_u}$ and let $\delta \in \mathbb{R}^W$ be such that $\ell \leq \delta \leq u$ holds. Then, define $A' \coloneqq \Pi_{\mathcal{P}}^{\intercal} \Gamma A_{V_U, W_U} \Lambda \widetilde{\Pi_{\mathcal{Q}}}$, $b' \coloneqq \Pi_{\mathcal{P}}^{\intercal} \Gamma (b - A\delta)_{V_U}$, $\ell' \coloneqq \Pi_{\mathcal{Q}}^{\intercal} \ell_{W_U}^{\lambda,\delta}$, $u' \coloneqq \Pi_{\mathcal{Q}}^{\intercal} u_{W_U}^{\lambda,\delta}$ and $c' \coloneqq \widetilde{\Pi_{\mathcal{Q}}}^{\intercal} \Lambda c_{W_U}$. We say that $(\mathcal{P}, \mathcal{Q}, \gamma, \lambda, \delta, V_U, W_U)$ is a* reflection reduction *of $F(A, b, \ell, u, c)$ if the following hold.*

(i) For every $x \in P(A, b, \ell, u)$, $y \coloneqq \Pi_{\mathcal{Q}}^{\intercal} \Lambda (x_{W_U} - \delta_{W_U})$ is feasible for $P(A', b', \ell', u')$ and $c^{\intercal} x = c'^{\intercal} y + c^{\intercal} \delta$ holds.

(ii) For every $y \in P(A', b', \ell', u')$, $x \coloneqq (\Lambda \widetilde{\Pi_{\mathcal{Q}}} y + \delta_{W_U}, \delta_{W_B})$ is feasible for $P(A, b, \ell, u)$ and $c'^{\intercal} y + c^{\intercal} \delta = c^{\intercal} x$ holds.

The main use of reflection reductions is that they show that one can equivalently solve the linear program $F(A', b', \ell', u', c')$ which has reduced dimension compared to the original linear program $F(A, b, \ell, u, c)$. Then, our main result shows that symmetric equitable partitions imply reflection reductions. Its proof relies strongly on the equivalent linear programming relations defined by the split reformulation and the DRCR reduction.

Theorem 19 *Consider the linear program $F(A, b, \ell, u, c)$ with $A \in \mathbb{R}^{V \times W}$, $b \in \mathbb{R}^V$, $c \in \mathbb{R}^W$ and vectors ℓ, u with $\ell_w \in \mathbb{R} \cup \{-\infty\}$ and $u_w \in \mathbb{R} \cup \{\infty\}$ such that $\ell_w \leq u_w$ holds for all $w \in W$, and let $\delta \in \mathbb{R}^W$ be given such that $\ell \leq \delta \leq u$ holds. Let $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ be a symmetric equitable partition of $F_S(A, b - A\delta, \ell - \delta, u - \delta, c)$, and let $\gamma \in \{-1, 1\}^{V_U}$, $\lambda \in \{-1, 1\}^{W_U}$, $\mathcal{P}$ and $\mathcal{Q}$ be generated by $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$. Then, $(\mathcal{P}, \mathcal{Q}, \gamma, \lambda, \delta, V_U, W_U)$ is a reflection reduction of $F(A, b, \ell, u, c)$.*

Proof We use $\Lambda \coloneqq \mathrm{diag}(\lambda)$, $\Gamma \coloneqq \mathrm{diag}(\gamma)$, $A' \coloneqq \Pi_{\mathcal{P}}^{\intercal} \Gamma A_{V_U, W_U} \Lambda \widetilde{\Pi_{\mathcal{Q}}}$, $b' \coloneqq \Pi_{\mathcal{P}}^{\intercal} \Gamma (b - A\delta)_{V_U}$, $\ell' \coloneqq \Pi_{\mathcal{Q}}^{\intercal} \ell_{W_U}^{\lambda,\delta}$, $u' \coloneqq \Pi_{\mathcal{Q}}^{\intercal} u_{W_U}^{\lambda,\delta}$ and $c' \coloneqq \widetilde{\Pi_{\mathcal{Q}}}^{\intercal} \Lambda c_{W_U}$ as in Definition 18.

First, let us prove Definition 18(i). Let x be a feasible solution for $F(A, b, \ell, u, c)$. Then, we have the following series of linear programs with feasible solutions:

x is feasible for $F(A, b, \ell, u, c)$

$\overset{\text{(a)}}{\Longrightarrow} x' \coloneqq x - \delta$ is feasible for $F(A, b - A\delta, \ell - \delta, u - \delta, c)$

$\overset{\text{(b)}}{\Longrightarrow} \widehat{x} \coloneqq (\max(x', \mathbb{O}), \max(-x', \mathbb{O}))$ is feasible for $F_S(A, b - A\delta, \ell - \delta, u - \delta, c)$

$\overset{\text{(c)}}{\Longrightarrow} \widehat{y} \coloneqq \Pi_{\widehat{\mathcal{Q}}}^{\intercal} \widehat{x}$ is feasible for $R^{\widehat{\mathcal{P}}, \widehat{\mathcal{Q}}}(F_S(A, b - A\delta, \ell - \delta, u - \delta, c))$

$\overset{\text{(d)}}{\Longrightarrow} (\widehat{y}_{\widehat{\mathcal{Q}}_U}, \widehat{y}_{\widehat{\mathcal{Q}}_B})$ is feasible for $P_S(A', b', \ell', u') \times \{\widehat{y}_{\widehat{\mathcal{Q}}_B} \in \mathbb{R}^{\widehat{\mathcal{Q}}_B} \mid \mathbb{O} \leq \widehat{y}_{\widehat{\mathcal{Q}}_B} \leq 2\Pi_{\mathcal{Q}_B}^{\intercal} (u - \delta)_{W_B}\}$

$\overset{\text{(e)}}{\Longrightarrow} y_{\mathcal{Q}_U} \coloneqq \widehat{y}_{\widehat{\mathcal{Q}}_U^{\lambda}} - \widehat{y}_{\widehat{\mathcal{Q}}_U^{-\lambda}}$ is feasible for $F(A', b', \ell', u', c')$

where (a) a holds by Proposition 5, (b) holds by Lemma 7(i) and (c) holds by Proposition 3(i). The implication in (d) holds by Proposition 17, and (e) follows from Lemma 7 and since the polyhedron for the bipolar variables is related only by the Cartesian product. For $Q \in \mathcal{Q}_U$, consider that

$$y_Q = (\widehat{y}_{\mathcal{Q}_U^\lambda})_Q - (\widehat{y}_{\mathcal{Q}_U^{-\lambda}})_Q = \sum_{w \in Q} \widehat{x}_{w^\lambda} - \widehat{x}_{w^{-\lambda}} \overset{\text{(f)}}{=} \sum_{w \in Q} \lambda_w x'_w,$$

where (f) follows since

$$\widehat{x}_{w^\lambda} - \widehat{x}_{w^{-\lambda}} = \begin{cases} \max(x'_w, 0) - \max(-x'_w, 0) \text{ if } \lambda_w = 1 \\ \max(-x'_w, 0) - \max(x'_w, 0) \text{ if } \lambda_w = -1 \end{cases} = \begin{cases} x'_w \text{ if } \lambda_w = 1 \\ -x'_w \text{ if } \lambda_w = 1 \end{cases} = \lambda_w x'_w.$$

Thus, $y_{\mathcal{Q}_U} = \Pi^\intercal_{\mathcal{Q}_U} \Lambda x'_{W_U} = \Pi^\intercal_{\mathcal{Q}_U} \Lambda (x_{W_U} - \delta_{W_U})$ is feasible for $F(A', b', \ell', u', c')$, which establishes the feasibility of Definition 18(i) for $(\mathcal{P}, \mathcal{Q}, \gamma, \lambda, \delta, V_U, W_U)$.
For the objective, we have

$$c^\intercal x - c^\intercal \delta \overset{\text{(g)}}{=} c^\intercal x' \overset{\text{(h)}}{=} c^\intercal (\widehat{x}^+_{W_U} - \widehat{x}^-_{W_U}) \overset{\text{(i)}}{=} c'^\intercal (\widehat{y}_{\widehat{\mathcal{Q}}_U^\lambda} - \widehat{y}_{\widehat{\mathcal{Q}}_U^{-\lambda}}) \overset{\text{(j)}}{=} c'^\intercal y$$

where (g) follows from Proposition 5, (h) follows from Lemma 7(i) and the bipolar variables having zero objective by Lemma 12, (i) follows from Proposition 3 and the reordering of the variables described in Proposition 17 and (j) follows from Lemma 7(ii). This establishes the objective in Definition 18(i).

To show Definition 18(ii), let $y \in P(A', b', \ell', u)$ be a feasible solution. Then, we have the following series of linear programs with feasible solutions.

y is feasible for $F(A', b', \ell', u', c')$

$\overset{\text{(k)}}{\Longrightarrow} \widehat{y}_{\widehat{\mathcal{Q}}_U} = (\widehat{y}_{\widehat{\mathcal{Q}}_U^\lambda}, \widehat{y}_{\widehat{\mathcal{Q}}_U^{-\lambda}}) \coloneqq (\max(y_{\mathcal{Q}_U}, \mathbb{O}), \max(-y_{\mathcal{Q}_U}, \mathbb{O}))$ is feasible for $F_S(A', b', \ell', u', c')$

$\overset{\text{(l)}}{\Longrightarrow} \widehat{y} \coloneqq (\widehat{y}_{\widehat{\mathcal{Q}}_U}, \mathbb{O}_{\widehat{\mathcal{Q}}_B})$ is feasible for $P_S(A', b', \ell', u') \times \{\widehat{y}_{\widehat{\mathcal{Q}}_B} \in \mathbb{R}^{\widehat{\mathcal{Q}}_B} \mid \mathbb{O} \leq \widehat{y}_{\widehat{\mathcal{Q}}_B} \leq 2\Pi^\intercal_{\mathcal{Q}_B} (u - \delta)_{W_B}\}$

$\overset{\text{(m)}}{\Longrightarrow} \widehat{y}$ is feasible for $R^{\widehat{\mathcal{P}}, \widehat{\mathcal{Q}}}(F_S(A, b - A\delta, \ell - \delta, u - \delta, c))$

$\overset{\text{(n)}}{\Longrightarrow} \widehat{x} \coloneqq \widetilde{\Pi_{\widehat{\mathcal{Q}}}} \widehat{y}$ is feasible for $F_S(A, b - A\delta, \ell - \delta, u - \delta, c)$

$\overset{\text{(o)}}{\Longrightarrow} x' \coloneqq \widehat{x}^+ - \widehat{x}^-$ is feasible for $F(A, b - A\delta, \ell - \delta, u - \delta, c)$

$\overset{\text{(p)}}{\Longrightarrow} x \coloneqq x' + \delta$ is feasible for $F(A, b, \ell, u, c)$

Here, (k) follows from Lemma 7(i) and (l) holds since taking the Cartesian product preserves feasibility and since $\mathbb{O}$ is clearly feasible. Moreover, (m) follows from Proposition 17, (n) follows from Proposition 3(ii), (o) follows from Lemma 7(ii) and (p) follows from Proposition 5.

Next, let us express y in terms of x. If $w \in W_U$ holds, then we have the following for $Q \in \mathcal{Q}_U$ such that $w \in Q$

$$x'_w = \widehat{x}_{w^+} - \widehat{x}_{w^-} = (\widetilde{\Pi_{\widehat{\mathcal{Q}}}} \widehat{y})_{w^+} - (\widetilde{\Pi_{\widehat{\mathcal{Q}}}} \widehat{y})_{w^+} \overset{\text{(q)}}{=} \frac{\lambda_w}{|Q|} (\widehat{y}_{Q^\lambda} - \widehat{y}_{Q^{-\lambda}}) = \frac{\lambda_w}{|Q|} y_Q = (\Lambda \widetilde{\Pi_{\mathcal{Q}_U}} y)_w$$

where (q) holds because we have by the reordering of the variables that

$$(\widetilde{\Pi_{\widehat{\mathcal{Q}}}} \widehat{y})_{w^+} - (\widetilde{\Pi_{\widehat{\mathcal{Q}}}} \widehat{y})_{w^-} = \begin{cases} \frac{1}{|Q|} (\widehat{y}_{Q^\lambda} - \widehat{y}_{Q^{-\lambda}}) & \text{if } \lambda_w = 1 \\ \frac{1}{|Q|} (\widehat{y}_{Q^{-\lambda}} - \widehat{y}_{Q^\lambda}) & \text{if } \lambda_w = -1 \end{cases} = \frac{\lambda_w}{|Q|} (\widehat{y}_{Q^\lambda} - \widehat{y}_{Q^{-\lambda}}).$$

For $w \in W_B$, then for $Q \in \widehat{\mathcal{Q}_B}$ such that $w \in Q$ we have the following:

$$x'_w = \widehat{x}_{w^+} - \widehat{x}_{w^-} = (\widetilde{\Pi_{\widehat{\mathcal{Q}}}}\widehat{y})_{w^+} - (\widetilde{\Pi_{\widehat{\mathcal{Q}}}}\widehat{y})_{w^-} = \frac{1}{|Q|}(\widehat{y}_Q - \widehat{y}_Q) = 0.$$

Note that this equation does not depend on the value of $\widehat{y}_Q$, so in fact the choice to assign $y_{\mathcal{Q}_B} = \mathbb{O}$ is arbitrary as its values always cancel when they are transformed into x'_{W_B}. Then, we have that $x = x' + \delta = (\Lambda\widetilde{\Pi_{\mathcal{Q}_U}}y, \mathbb{O}) + \delta = (\Lambda\widetilde{\Pi_{\mathcal{Q}_U}}y + \delta_{W_U}, \delta_{W_B})$, which shows that the feasibility in Definition 18(ii) is satisfied. For the objective, we have similar reductions as in the first part of the proof but in reverse order:

$$c'^{\mathsf{T}}y \overset{\text{(s)}}{=} c'^{\mathsf{T}}(\widehat{y}_{\widehat{\mathcal{Q}}_U^{\lambda}} - \widehat{y}_{\widehat{\mathcal{Q}}_U^{-\lambda}}) \overset{\text{(t)}}{=} c'^{\mathsf{T}}x'_{W_U} \overset{\text{(u)}}{=} c^{\mathsf{T}}x' \overset{\text{(v)}}{=} c^{\mathsf{T}}x - c^{\mathsf{T}}\delta,$$

where (s) follows from Lemma 7(i), (t) holds by the relation between $\widehat{x}$ and $\widehat{y}$ that we explained above, (u) holds since bipolar variables have zero objective by Lemma 12 and (v) holds by Proposition 5. Then, the objective in Definition 18(ii) is also satisfied.

Thus, we have established that $(\mathcal{P}, \mathcal{Q}, \gamma, \lambda, \delta, V_U, W_U)$ satisfies Definition 18, which shows that it is a reflection reduction of $F(A, b, \ell, u, c)$. □

3.5 Choosing the affine offset

In order to apply reflection reductions to arbitrary linear programs, we had to introduce the affine offset δ to ensure that the lower bounds were nonpositive and the upper bounds were nonnegative. Given δ, the linear transformation that is represented by the ± 1 scaling vectors λ and γ follows automatically from the symmetric equitable partition of the split reformulation. In this section, we argue how to choose δ. In previous work, Hojny [2] chooses δ to be the center of the variable domains such that $\delta_w \coloneqq \frac{\ell_w + u_w}{2}$. One issue is that such a center is not well-defined for variables with infinite lower or upper bounds. Similarly to Hojny, we use the offset $\delta^{\mathcal{C}}$ defined for all $w \in W$ as

$$\delta^{\mathcal{C}}_w \coloneqq \begin{cases} \frac{\ell_w + u_w}{2} & \text{if } \ell_w \text{ and } u_w \text{ are finite.} \\ 0 & \text{if } \ell_w = -\infty \text{ and } u_w = \infty \\ \ell_w & \text{if } \ell_w \text{ is finite and } u_w = \infty. \\ u_w & \text{if } \ell_w = -\infty \text{ and } u_w \text{ is finite.} \end{cases}$$

Next, let us argue why $\delta^{\mathcal{C}}$ is a good choice for the offset. First of all, we note that for the most common case of variables w with ℓ_w and u_w finite, they can only belong to bipolar parts if $-(\ell_w - \delta_w) = u_w - \delta_w$ holds, which holds exactly if $\delta_w = \frac{\ell_w + u_w}{2}$ holds. Thus, variables with finite bounds can only be detected as bipolar variables if $\delta_w = \delta^{\mathcal{C}}_w = \frac{\ell_w + u_w}{2}$ holds. Note that in the case exactly one of ℓ_w and u_w is finite, we can never detect bipolarity. If $\ell_w = -\infty$ and $u_w = \infty$ hold, then the choice of δ_w does not affect the transformed variable bounds and bipolarity may be detected in any case.

Second, recall that we assumed that the symmetric equitable partitions must be bound-respecting. For the linear program $F(A, b - A\delta, \ell - \delta, u - \delta, c)$, this condition can be stated as follows. The partition $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ of the split reformulation must satisfy for all $Q \in \widehat{\mathcal{Q}}$ and any $w_1^s, w_2^s \in Q$ that $\ell_{w_1} - \delta_{w_1} = \ell_{w_2} - \delta_{w_2}$ and $u_{w_1} - \delta_{w_1} = u_{w_2} - \delta_{w_2}$ hold if $s_1 = s_2$ holds, and if $s_1 \neq s_2$ holds that $\ell_{w_1} - \delta_{w_1} = -u_{w_2} + \delta_{w_2}$ and $u_{w_1} - \delta_{w_1} =$

$-\ell_{w_2} + \delta_{w_2}$ hold. Next, we show that for $\delta = \delta^{\mathcal{C}}$ only the *size* $u_w - \ell_w$ of the domain of w matters. Assume for all $Q \in \mathcal{Q}$ and any $w_1^s, w_2^s \in Q$ that $u_{w_1} - \ell_{w_1} = u_{w_2} - \ell_{w_2}$ holds. For the $s_1 = s_2$ case, we can use this assumption to show that:

$$\ell_{w_1} - \delta^{\mathcal{C}}_{w_1} = \ell_{w_1} - \frac{\ell_{w_1} + u_{w_1}}{2} = -\frac{u_{w_1} - \ell_{w_1}}{2} = \frac{u_{w_2} - \ell_{w_2}}{2} = \ell_{w_2} - \frac{\ell_{w_2} + u_{w_2}}{2} = \ell_{w_2} - \delta^{\mathcal{C}}_{w_2}$$

and similarly that

$$u_{w_1} - \delta^{\mathcal{C}}_{w_1} = u_{w_1} - \frac{\ell_{w_1} + u_{w_1}}{2} = \frac{u_{w_1} - \ell_{w_1}}{2} = \frac{u_{w_2} - \ell_{w_2}}{2} = u_{w_2} - \frac{\ell_{w_2} + u_{w_2}}{2} = u_{w_2} - \delta^{\mathcal{C}}_{w_2}.$$

Also for the $s_1 \neq s_2$ case, we can use the assumption to show:

$$\ell_{w_1} - \delta^{\mathcal{C}}_{w_1} = -\frac{u_{w_1} - \ell_{w_1}}{2} = -\frac{u_{w_2} - \ell_{w_2}}{2} = -u_{w_2} + \frac{u_{w_2} + \ell_{w_2}}{2} = -u_{w_2} + \delta^{\mathcal{C}}_{w_2}$$

and similarly:

$$u_{w_1} - \delta^{\mathcal{C}}_{w_1} = \frac{u_{w_1} - \ell_{w_1}}{2} = \frac{u_{w_2} - \ell_{w_2}}{2} = -\ell_{w_2} + \frac{u_{w_2} + \ell_{w_2}}{2} = -\ell_{w_2} + \delta^{\mathcal{C}}_{w_2}.$$

Thus, the simpler condition $u_{w_1} - \ell_{w_1} = u_{w_2} - \ell_{w_2}$ is sufficient in the case where $\delta = \delta^{\mathcal{C}}$ holds to show that a partition is bound-respecting for finite u_{w_1} and ℓ_{w_1}. For infinite values, the bound-respecting condition is still necessary. In particular, the above bound argumentation does not prevent that free variables may be aggregated with variables with one infinite bound, as in both cases the domain size is infinite. Thus, the domain-respecting assumption is still necessary for linear programs with infinite variable bounds.

Although $\delta^{\mathcal{C}}$ is a natural choice, it does have a few problematic aspects. First of all, the product $A\delta^{\mathcal{C}}$ that appears in the right-hand side $b - A\delta^{\mathcal{C}}$ of the linear program after applying the transformation $x = x' + \delta$ can be problematic. If A contains large nonzero-entries and/or $\delta^{\mathcal{C}}$ is large due to variables with large domains, then the product $A\delta^{\mathcal{C}}$ may result in large numerical values in the right-hand $b - A\delta^{\mathcal{C}}$ in the linear program obtained from the reflection reduction. This is undesirable, as it can lead to numerical problems that slow down the linear programming solver. Thus, we generally prefer offsets δ that are sparse and/or have small norms.

A second issue appears when we apply reflection reductions in the context of mixed-integer linear programs: the offset $\delta^{\mathcal{C}}$ is not necessarily integer. As a result, an integrality constraint $x_w \in \mathbb{Z}$ may be transformed into the form $x'_w + \delta^{\mathcal{C}}_w \in \mathbb{Z}$, which cannot be handled directly by MILP solvers if $\delta^{\mathcal{C}}_w$ is not integer. Next, we formulate a result that helps to derive a family of reflection reductions for different δ-values by detecting one for a single value of δ. This result will be helpful to construct an offset δ that does not cause numerical issues and that also works well for mixed-integer linear programs.

Theorem 20 *Let $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ be a symmetric equitable partition of $F_S(A, b - A\delta, \ell - \delta, u - \delta, c)$ and let γ, λ, $\mathcal{P}_U$ and $\mathcal{Q}_U$ be generated by $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$. Let $\nu \in \mathbb{R}^{W_U}$ be a vector that is equitably partitioned by $\mathcal{Q}_U$ and define $\delta' \coloneqq (\delta_{W_U} + \Lambda\nu, \delta_{W_B})$ such that $\ell \leq \delta' \leq u$. Then $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ is a symmetric equitable partition of $F_S(A, b - A\delta', \ell - \delta', u - \delta', c)$.*

Proof Clearly, $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ is also a symmetric partition of $F_S(A, b - A\delta', \ell - \delta', u - \delta', c)$. Since the objective and constraint matrix are unchanged compared to $F_S(A, b - A\delta, \ell - \delta, u - \delta, c)$ and remain so by the split reformulation, it is sufficient to show that $\widehat{\mathcal{P}}$ is a symmetric equitable partition of the right-hand side and that $\widehat{\mathcal{Q}}$ is an equitable partition of the variable upper bounds of $F_S(A, b - A\delta', \ell - \delta', u - \delta', c)$. Note that by definition of the split reformulation, all variables have lower bound zero, so we only consider the upper bounds.

Let us state the reordered linear program of $F_S(A, b - A\delta', \ell - \delta', u - \delta', c)$, which is similar to the form (7).

$$\min\ c^{\intercal}_{W_U}\Lambda\widehat{x}_{\widehat{W}_U^{\lambda}} - c^{\intercal}_{W_U}\Lambda\widehat{x}_{\widehat{W}_U^{-\lambda}} \tag{10a}$$

$$\begin{bmatrix} \Gamma A_{V_U,W_U}\Lambda & -\Gamma A_{V_U,W_U}\Lambda & \widehat{M}^1 \\ -\Gamma A_{V_U,W_U}\Lambda & \Gamma A_{V_U,W_U}\Lambda & -\widehat{M}^1 \\ \widehat{M}^2 & -\widehat{M}^2 & \widehat{M}^3 \end{bmatrix} \begin{bmatrix} \widehat{x}_{\widehat{W}_U^{\lambda}} \\ \widehat{x}_{\widehat{W}_U^{-\lambda}} \\ \widehat{x}_{\widehat{W}_B} \end{bmatrix} = \begin{bmatrix} \Gamma(b - A\delta')_{V_U} \\ -\Gamma(b - A\delta')_{V_U} \\ \widehat{b}_{\widehat{V}_B} \end{bmatrix} \tag{10b}$$

$$\mathbb{O} \leq \widehat{x}_{\widehat{W}_U^{\lambda}} \leq \mathrm{bs}(\ell - \delta', u - \delta', \lambda)_{W_U} \tag{10c}$$

$$\mathbb{O} \leq \widehat{x}_{\widehat{W}_U^{-\lambda}} \leq \mathrm{bs}(\ell - \delta', u - \delta', -\lambda)_{W_U} \tag{10d}$$

$$\mathbb{O} \leq \widehat{x}_{\widehat{W}_B} \leq \begin{bmatrix} -(\ell - \delta')_{W_B} \\ (u - \delta')_{W_B} \end{bmatrix} \tag{10e}$$

We use $\widehat{b}_{\widehat{V}_B} \coloneqq \begin{bmatrix} (b - A\delta')_{V_B} \\ -(b - A\delta')_{V_B} \end{bmatrix}$ to denote the variable bounds of the bipolar rows. For the variables $\widehat{W}_U^{\lambda}$, we can compute their upper bounds using:

$$\mathrm{bs}(\ell - \delta', u - \delta', \lambda)_w = \begin{cases} u_w - \delta'_w & \text{if } \lambda_w = 1 \\ -\ell_w + \delta'_w & \text{if } \lambda_w = -1 \end{cases} = \begin{cases} u_w - \delta_w - \lambda_w\nu_w & \text{if } \lambda_w = 1 \\ -\ell_w + \delta_w + \lambda_w\nu_w & \text{if } \lambda_w = -1 \end{cases}$$

$$= -\nu_w + \begin{cases} u_w - \delta_w & \text{if } \lambda_w = 1 \\ -\ell_w + \delta_w & \text{if } \lambda_w = -1 \end{cases} = -\nu_w + \mathrm{bs}(\ell - \delta, u - \delta, \lambda)_w.$$

Thus, we have that $\mathrm{bs}(\ell - \delta', u - \delta', \lambda) = -\nu + \mathrm{bs}(\ell - \delta, u - \delta, \lambda)$ holds. Then, since $\widehat{\mathcal{Q}}_U^{\lambda}$ is exactly the partition given by $\mathcal{Q}_U$ and ν is $\mathcal{Q}_U$ equitable, it follows that it follows that $\widehat{\mathcal{Q}}_U^{\lambda}$ is an equitable partition of $\mathrm{bs}(\ell - \delta', u - \delta', \lambda)$ since it is the sum of the equitably partitioned vectors $-\nu$ and $\mathrm{bs}(\ell - \delta, u - \delta, \lambda)$. For $\widehat{W}_U^{-\lambda}$ a similar argument shows that $\mathrm{bs}(\ell - \delta', u - \delta', -\lambda) = \nu + \mathrm{bs}(\ell - \delta, u - \delta, -\lambda)$ holds, which implies that the upper bounds $\mathrm{bs}(\ell - \delta', u - \delta', -\lambda)$ are equitably partitioned by $\widehat{\mathcal{Q}}_U^{\lambda}$. For the bipolar variables $\delta'_{W_B} = \delta_{W_B}$ holds so the variable bounds are unchanged, which directly implies that $\widehat{\mathcal{Q}}_B$ is an equitable partition of the variable bounds of $\widehat{W}_B$. Combining these statements, it follows that $\widehat{\mathcal{Q}}$ is an equitable partition of the variable upper bounds of $F_S(A, b - A\delta', \ell - \delta', u - \delta', c)$.

Next, let us show that the right-hand sides of (10b) are equitably partitioned by $\widehat{\mathcal{P}}$. Note that for any $v \in V$ that

$$(b - A\delta')_v = b_v - (A(\delta + (\Lambda\nu, \mathbb{O}_{W_B})))_v = (b - A\delta)_v - (A_{\star,W_U}\Lambda\nu)_v$$

holds. Thus, it follows that

$$\Gamma(b - A\delta')_{V_U} = \Gamma(b - A\delta - A_{\star,W_U}\Lambda\nu)_{V_U} = \Gamma(b - A\delta)_{V_U} - \Gamma A_{V_U,W_U}\Lambda\nu.$$

The term $\Gamma(b - A\delta)_{V_U}$ is equitably partitioned by the subpartition $\widehat{\mathcal{P}}_U^\gamma$ of $\widehat{\mathcal{P}}$. Then, we claim that $\Gamma A_{V_U,W_U}\Lambda\nu$ is also a $\widehat{\mathcal{P}}_U^\gamma$-equitable vector.

First, note that $(\widehat{\mathcal{P}}_U^\gamma, \mathcal{Q}_U)$ is an equitable partition of $\Gamma A_{V_U,W_U}\Lambda$, which follows from the fact that $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ is an equitable partition of $\widehat{M}$ in the linear program $F_S(A, b - A\delta, \ell - \delta, u - \delta, c)$, where $(\widehat{\mathcal{P}}_U^\gamma, \widehat{\mathcal{Q}}_U^\lambda) = (\widehat{\mathcal{P}}_U^\gamma, \mathcal{Q}_U)$ induces the subpartition.

Secondly, since ν is $\mathcal{Q}_U$-equitable, we claim that multiplication of any $(\widehat{\mathcal{P}}_U^\gamma, \mathcal{Q}_U)$-matrix with a $\mathcal{Q}_U$-equitable vector produces a $\widehat{\mathcal{P}}_U^\gamma$-equitable vector. In particular note that for all $P \in \widehat{\mathcal{P}}_U^\gamma$ and any $v \in P$, we have:

$$\begin{aligned}(\Gamma A_{V_U,W_U}\Lambda\nu)_v &= \sum_{w\in W} (\Gamma A_{V_U,W_U}\Lambda)_{v,w}\nu_w = \sum_{Q\in\mathcal{Q}_U}\sum_{w\in Q} (\Gamma A_{V_U,W_U}\Lambda)_{v,w}\nu_w \\ &\overset{(a)}{=} \sum_{Q\in\mathcal{Q}} \alpha_Q \sum_{w\in Q} (\Gamma A_{V_U,W_U}\Lambda)_{v,w} \overset{(b)}{=} \sum_{Q\in\mathcal{Q}} \alpha_Q\beta_{P,Q}.\end{aligned}$$

Here, (a) follows since $\mathcal{Q}_U$ is an equitable partition of ν, which implies that $\nu_w = \alpha_Q$ for $Q \in \mathcal{Q}_U$ such that $w \in Q$. Then, (b) follows since $(\widehat{\mathcal{P}}_U^\gamma, \mathcal{Q}_U)$ is an equitable partition of $\Gamma A\Lambda$, which implies that condition (1) holds. Thus, the row sums of $\Gamma A\Lambda$ are identical for all $v \in P$, which we express using $\beta_{P,Q}$. Since the final expression is independent of v for every part $P \in \widehat{\mathcal{P}}_U^\gamma$, this shows that $\Gamma A\Lambda\nu$ is $\widehat{\mathcal{P}}_U^\gamma$-equitable. Then, since the sum of two equitable vectors is equitable, it follows that $\widehat{\mathcal{P}}_U^\gamma$ is an equitable partition of $\Gamma(b - A\delta')_{V_U} = \Gamma(b - A\delta)_{V_U} - \Gamma A_{V_U,W_U}\Lambda\nu$. A very similar argument shows that $\widehat{\mathcal{P}}_U^{-\gamma}$ is an equitable partition of $-\Gamma(b - A\delta')_{V_U} = -\Gamma(b - A\delta)_{V_U} + \Gamma A_{V_U,W_U}\Lambda\nu$.

For the bipolar rows, we consider $(b - A\delta')_v$ for $v \in V_B$ such that $\{v^+, v^-\} \subseteq P$ for $P \in \widehat{\mathcal{P}}_B$. Then, we have that:

$$\begin{aligned}(b - A\delta')_v &= (b - A\delta)_v - (A_{\star,W_U}\Gamma\nu)_v \overset{(c)}{=} -(A_{\star,W_U}\Gamma\nu)_v = -\sum_{w\in W_U} A_{v,w}\lambda_w\nu_w \\ &\overset{(d)}{=} -\sum_{Q\in\mathcal{Q}_U} \alpha_Q \sum_{w\in Q} A_{v,w}\lambda_w \overset{(e)}{=} -\sum_{Q\in\mathcal{Q}_U} \alpha_Q \sum_{w^\lambda\in Q^\lambda} \widehat{M}_{v^+,w^\lambda} \overset{(f)}{=} 0\end{aligned}$$

Here, (c) follows since $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ being a symmetric equitable partition for $F_S(A, b - A\delta, \ell - \delta, u - \delta, c)$ implies that $(b - A\delta)_v = 0$ for all $v \in V_B$. The relation in (d) follows since ν is an $\mathcal{Q}_U$-equitable vector. In (e), we use that $(A\Lambda)_{v,Q} = \widehat{M}_{v^+,Q^\lambda}$ holds by construction of $\widehat{M}$. In (f), we use that $\sum_{w^\lambda\in Q^\lambda} \widehat{M}_{v^+,w^\lambda} = 0$ holds by Corollary 10, using that v is bipolar and that the given partition is equitable. Then, it follows that $\widehat{b}_{\widehat{V}_B} \coloneqq \begin{bmatrix}(b - A\delta')_{V_B} \\ -(b - A\delta')_{V_B}\end{bmatrix} = \begin{bmatrix}\mathbb{0}_{V_B} \\ \mathbb{0}_{V_B}\end{bmatrix}$, which shows that $\widehat{\mathcal{P}_B}$ is an equitable partition of $\widehat{b}$. Moreover, it implies that the rows $\widehat{V}_B$ remain bipolar, which implies that the symmetry of $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ is also preserved for $F_S(A, b - A\delta', \ell - \delta', u - \delta')$. Then, we have that $\widehat{\mathcal{P}}$ is an equitable partition of the right-hand side of $F_S(A, b - A\delta', \ell - \delta', u - \delta', c)$. Using the earlier derived equitability for the variable bounds, we conclude that $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ is a symmetric equitable partition of $F_S(A, b - A\delta', \ell - \delta', u - \delta', c)$. □

Usually, MIP solvers use *variable complementation*, which maps x_w to $x_w - \ell_w$ or $u_w - x_w$, in the context of reflections. Since the bounds of an integer variable are integral, these transformations have the convenient property of preserving integrality.

In Theorem 20, we showed that by computing symmetric equitable partitions for one δ value, the obtained partition is valid for a larger set of δ-values. Next, we use this result to show that computing the symmetric equitable partition for $\delta^{\mathcal{C}}$ can be used to derive variable complementations.

Corollary 21 Let $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ be a symmetric, equitable and bound-respecting partition of $F_S(A, b - A\delta^{\mathcal{C}}, \ell - \delta^{\mathcal{C}}, u - \delta^{\mathcal{C}}, c)$ and let γ, λ, $\mathcal{P}_U$ and $\mathcal{Q}_U$ be generated by $(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$. Then, for $\delta^{\mathcal{S}}$ defined via

$$\delta^{\mathcal{S}}_w := \begin{cases} \ell_w & \text{if } \ell_w \text{ and } u_w \text{ are finite, } w \in W_U \text{ and } \lambda_w = 1. \\ u_w & \text{if } \ell_w \text{ and } u_w \text{ are finite, } w \in W_U \text{ and } \lambda_w = -1. \\ \frac{\ell_w + u_w}{2} & \text{if } \ell_w \text{ and } u_w \text{ are finite and } w \in W_B. \\ 0 & \text{if } \ell_w = -\infty \text{ and } u_w = \infty. \\ \ell_w & \text{if } \ell_w \text{ is finite and } u_w = \infty. \\ u_w & \text{if } \ell_w = -\infty \text{ and } u_w \text{ is finite.} \end{cases},$$

$(\widehat{\mathcal{P}}, \widehat{\mathcal{Q}})$ is a symmetric equitable partition of $F_S(A, b - A\delta^{\mathcal{S}}, \ell - \delta^{\mathcal{S}}, u - \delta^{\mathcal{S}}, c)$.

Proof Let $\nu \in \mathbb{R}^{W_U}$ be such that $\nu := \begin{cases} -\frac{u_w - \ell_w}{2} & \text{if } \ell_w \text{ and } u_w \text{ are finite and } w \in W_U \\ 0 & \text{otherwise.} \end{cases}$

Since the symmetric equitable partition is bound-respecting, the domain size $u_w - \ell_w$ is identical for all variables $w \in Q$ for some part $Q \in \mathcal{Q}$ in the case where u_w and ℓ_w are both finite. Thus, ν is a $\mathcal{Q}_U$-equitable vector. Using $\frac{\ell_w + u_w}{2} - \frac{u_w - \ell_w}{2} = \ell_w$ and $\frac{\ell_w + u_w}{2} + \frac{u_w - \ell_w}{2} = u_w$ one can verify that $\delta^{\mathcal{S}} := \delta^{\mathcal{C}} + (\Lambda\nu, \mathbb{0}_{W_B})$ holds, and the result follows by Theorem 20. □

One important detail about Corollary 21 is that the variable bounds of bipolar variables are not adjusted and may still be fractional. Later on, we will see that this is not an issue in the context of Mixed-Integer Linear Programming as the bipolar variables are eliminated from the reduced model.

3.6 Comparison with reflection symmetries

Next, we will show that reflection reductions generalize reflection symmetries. First let us define reflection symmetries more formally. A *permutation matrix* B is a square binary matrix with exactly one nonzero element with value 1 in each row and column. It is well known that the inverse B^{-1} is again a permutation matrix, which corresponds to the inverse permutation of the permutation given by B. A *signed permutation matrix* is a square $\{-1, 0, 1\}$-matrix with exactly one nonzero element in each row and column. Note that each signed permutation matrix B can be decomposed as $\mathrm{diag}(s)C$ or $C\mathrm{diag}(s')$ where C is the nonzero support of B, which is a permutation matrix, and s and s' are suitably chosen $\{-1, 1\}$ vectors. In this section, we do not allow for arbitrary reordering of matrix columns and rows. In order to clarify the effect of the permutation symmetries, we denote each permutation explicitly using a permutation matrix.

Definition 22 *Consider a linear program $F(A, b, \ell, u, c)$, with $A \in \mathbb{R}^{V \times W}$. Let $\gamma \in \{-1, 1\}^V$, $\lambda \in \{-1, 1\}^W$ and define $\Gamma := \mathrm{diag}(\gamma)$ and $\Lambda := \mathrm{diag}(\lambda)$, and let $C \in \{0, 1\}^{V \times V}$ and*

$D \in \{0,1\}^{W\times W}$ *be permutation matrices. Then,* (γ, λ, C, D) *is a* reflection symmetry of $F(A, b, \ell, u, c)$ *if the following hold:*

(i) $C\Gamma A\Lambda D = A$

(ii) $C\Gamma b = b$

(iii) $D^{-1}\lambda c = c$

(iv) $-D^{-1}\mathrm{bs}(\ell, u, -\lambda) = \ell$ *and* $D^{-1}\mathrm{bs}(u, \ell, \lambda) = u$

Furthermore, we say that (C, D) *is a* permutation symmetry *of* $F(A, b, \ell, u, c)$ *if* $(\mathbb{1}, \mathbb{1}, C, D)$ *is a reflection symmetry of* $F(A, b, \ell, u, c)$.

Theorem 23 *Every reflection symmetry of* $F(A, b, \ell, u, c)$ *induces a permutation symmetry of* $F_S(A, b, \ell, u, c)$.

Proof Let (γ, λ, C, D) be a reflection symmetry of $F(A, b, \ell, u, c)$. Then, consider the split reformulation $F_S(A, b, \ell, u, c)$. We argued earlier that for the split permutation there exist permutation matrices $\widehat{C}_1$ and $\widehat{D}_1$ whose application to $F_S(A, b, \ell, u, c)$ yields the following linear program, see e.g. (7). Here, $\widehat{C}_1$ corresponds to the permutation that swaps v^+ and v^- if $\gamma_v = -1$ holds, and $\widehat{D}_1$ corresponds to the permutation that swaps w^+ and w^- if $\lambda_w = -1$ holds.

$$\min(\Lambda c)^\intercal \widehat{x}^\lambda - (\Lambda c)^\intercal \widehat{x}^{-\lambda} \tag{11a}$$

$$\begin{bmatrix} \Gamma A\Lambda & -\Gamma A\Lambda \\ -\Gamma A\Lambda & \Gamma A\Lambda \end{bmatrix} \begin{bmatrix} \widehat{x}^\lambda \\ \widehat{x}^{-\lambda} \end{bmatrix} = \begin{bmatrix} \Gamma b \\ -\Gamma b \end{bmatrix} \tag{11b}$$

$$\mathbb{O} \le \begin{bmatrix} \widehat{x}^\lambda \\ \widehat{x}^{-\lambda} \end{bmatrix} \le \begin{bmatrix} \mathrm{bs}(\ell, \mathrm{u}, \lambda) \\ \mathrm{bs}(\ell, \mathrm{u}, -\lambda) \end{bmatrix} \tag{11c}$$

Then, we define permutation matrices $\widehat{C}_2 \coloneqq \begin{bmatrix} C & \mathbb{O} \\ \mathbb{O} & C \end{bmatrix}$ and $\widehat{D}_2 \coloneqq \begin{bmatrix} D & \mathbb{O} \\ \mathbb{O} & D \end{bmatrix}$ and apply them to (11). Doing so, we obtain the linear program:

$$\min(D^{-1}\Lambda c)^\intercal \widehat{x}^\lambda - (D^{-1}\Lambda c)^\intercal \widehat{x}^{-\lambda} \tag{12a}$$

$$\begin{bmatrix} C\Gamma A\Lambda D & -C\Gamma A\Lambda D \\ -C\Gamma A\Lambda D & C\Gamma A\Lambda D \end{bmatrix} \begin{bmatrix} \widehat{x}^\lambda \\ \widehat{x}^{-\lambda} \end{bmatrix} = \begin{bmatrix} C\Gamma b \\ -C\Gamma b \end{bmatrix} \tag{12b}$$

$$\mathbb{O} \le \begin{bmatrix} \widehat{x}^\lambda \\ \widehat{x}^{-\lambda} \end{bmatrix} \le \begin{bmatrix} D^{-1}\mathrm{bs}(\ell, \mathrm{u}, \lambda) \\ D^{-1}\mathrm{bs}(\ell, \mathrm{u}, -\lambda) \end{bmatrix} \tag{12c}$$

Then, since (γ, λ, C, D) is a reflection symmetry, it follows by substitution that $C\Gamma A\Lambda D = A$, $C\Gamma b = b$, $D^{-1}\mathrm{bs}(\ell, u, \lambda) = u$, $D^{-1}\mathrm{bs}(\ell, u, -\lambda) = -\ell$. Using these substitutions, (12) is identical to $F_S(A, b, \ell, u, c,)$, without reordering the indices. Thus, the permutation matrices $\widehat{C} \coloneqq \widehat{C}_2\widehat{C}_1$ and $\widehat{D} \coloneqq \widehat{D}_1\widehat{D}_2$ form a permutation symmetry $(\widehat{C}, \widehat{D})$ of $F_S(A, b, \ell, u, c)$. □

By Theorem 23, the reflection symmetries of a linear program are captured completely by the permutation symmetries of its split reformulation. In section 7.2 in [11], it is argued that for any linear program, performing the reduction from Proposition 3 using equitable partitions generalizes the projection to the fixed space of the symmetry group formed by the permutation symmetries. Although we do not show it formally, the permutation symmetries that we exhibit in the proof

of Theorem 23 are bound-respecting, as they act similarly on the positive and negative variables of the split reformulation. As a consequence, the symmetry subgroup of the split reformulation corresponding to any permutation symmetries derived from reflection symmetries generates a bound-respecting equitable partition of the split reformulation. Thus, by computing bound-respecting equitable partitions of the split reformulation, our approach generalizes the projection to the fixed space of the symmetry group formed by reflections symmetries.

As is also remarked in [11], equitable partitions may also occur even if there is no symmetry. Next, we show an example where this occurs for the split reformulation of a general class of matrices that does not necessarily contain symmetries.

Example 24 Let $A \in \{0,1\}^{m \times n}$ be a matrix with two non-zero entries in each row. The matrix A does not necessarily contain any symmetries, and we can consider the following linear program with no objective.

$$\begin{aligned} Ax &= \mathbb{1} \\ \mathbb{O} \leq x &\leq \mathbb{1} \end{aligned}$$

Then, centering the linear program using $\delta_w \coloneqq \frac{1}{2}$ for all $w \in W$ yields the following linear program.

$$\begin{aligned} Ax &= \mathbb{O} && \\ -\frac{1}{2} \leq x_w &\leq \frac{1}{2} && \forall w \in W \end{aligned}$$

Its split reformulation is given by

$$\begin{aligned} Ax^+ - Ax^- &= \mathbb{O} && \\ -Ax^+ + Ax^- &= \mathbb{O} && \\ 0 \leq x_w^+ &\leq \frac{1}{2} && \forall w \in W \\ 0 \leq x_w^- &\leq \frac{1}{2} && \forall w \in W \end{aligned}$$

Then, since the objective and right-hand sides are all zero and the variable bounds are identical, the coarsest equitable partition of this linear program has a single bipolar variable part and a single bipolar row part, even though the original LP does not in general contain any symmetries.

3.7 Detecting reflection reductions

In order to detect reflection reductions, it suffices to compute symmetric equitable partitions of the split reformulation form by Theorem 19. Thus, we can directly reuse the algorithm formulated in [11], which relies on the fast color refinement algorithm by Paige and Tarjan [35]. Since the split reformulation consists of four copies of the constraint matrix, we can detect symmetric equitable partitions, and thus reflection reductions, in identical asymptotic worst-case time and space complexity as in [11].

Theorem 25 *For a linear program $F(A, b, \ell, u, c)$ with $A \in \mathbb{R}^{V \times W}$, let m be the total bitlength of the entries of $\begin{bmatrix} \ell^\intercal & 0 \\ u^\intercal & 0 \\ c^\intercal & 0 \\ A & b \end{bmatrix}$ and let $n = |V| + |W|$. For δ with total bitlength $\mathcal{O}(m)$, a reflection reduction $(\mathcal{P}, \mathcal{Q}, \gamma, \lambda, \delta, V_U, W_U)$ of $F(A, b, \ell, u, c)$ can be determined in $\mathcal{O}((n+m)\log n)$ time and $\mathcal{O}(n+m)$ space.*

Proof We assume that the constraint matrix is given in a sparse format such as the CSR and/or CSC format. First, $F(A, b - A\delta, \ell - \delta, u - \delta, c)$ can be computed in $\mathcal{O}(n+m)$ time and space and has bitlength $\mathcal{O}(m)$, where we use that δ has bitlength of order $\mathcal{O}(m)$. Then, the split reformulation $F_S(A, b - A\delta, \ell - \delta, u - \delta, c)$ can also be computed in $\mathcal{O}(n+m)$ time and space and has bitlength of order $\mathcal{O}(m)$. The initial symmetric equitable partition can be determined in and $\mathcal{O}(n+m)$ time and $\mathcal{O}(n+m)$ space by using standard hashing techniques. Then, computing the coarsest symmetric equitable partition of the split reformulation takes $\mathcal{O}((n+m)\log n)$ time and $\mathcal{O}(n+m)$ space [11, Theorem 3.1]. Using the symmetric equitable partition, we can directly derive a reflection reduction in $\mathcal{O}(n+m)$ time and space. Summing up all steps, we obtain a worst-case time complexity of $\mathcal{O}((n+m)\log n)$ time and a worst-case space complexity of $\mathcal{O}(n+m)$. □

In Theorem 25, we assumed that the offset δ has bitlength that is of the same order as $F(A, b, \ell, u, c)$. This is a reasonable choice, as both $\delta^{\mathcal{C}}$ and $\delta^{\mathcal{S}}$ have encoding lengths bounded by the encoding length of ℓ and u by their definition.

Although the running time is asymptotically equivalent to the 'ordinary' equitable partition algorithm presented in [11], computing equitable partitions of the split reformulation, which has 4 times as many nonzero entries, may still be undesirable in practice. To reduce the running time by a constant factor one can exploit the structure of symmetric equitable partitions. We formulate a refinement algorithm that computes γ and λ that runs on the original matrix but models the refinement algorithm that computes a symmetric partition on the split reformulation. This closely follows the argumentation presented in Lemma 12 and Lemma 13. The core idea to achieve this constant-factor improvement is to refine unipolar part pairs at the same time, as is also done in the proof of Lemma 13. The implemented algorithm is very similar to that in [11], who use the algorithm in [42] which provides a thorough description of the fast color refinement algorithm. Since the algorithm we present does not have a better asymptotic running time, we outline the main changes and adaptations compared to the description in [11] here without proof.

For the data structures, we add arrays to track λ, γ and the type (unipolar or bipolar) of each part, which takes $\mathcal{O}(n)$ space. The signs λ_w and γ_w are used during the computation of the column and row sums for equitability. Then, λ and γ are determined whenever a bipolar part P is turned into a unipolar one or at initialization.

Initially, we compute the affine transformation given by $\delta^{\mathcal{C}}$. Then, we compute an initial symmetric bound-respecting partition of the split reformulation as in Lemma 12, which can be achieved in linear time by using hashing techniques. For a row $v \in V$, we initialize it to be bipolar if $b_v = 0$ holds. Otherwise, the row is marked

as unipolar and grouped with other rows based on the absolute values $|b_v|$ of the right-hand side.

For a variable $w \in W$, we initialize them it to be bipolar if $c_w = 0$ and $-\ell_w = u_w$ hold. Otherwise, we let them be unipolar, and compute signs λ_w that so that variables whose objective and bounds can be mapped to each other by negation are put into the same initial part.

For the subsequent refinement of the constraint matrix we change the behavior slightly based on the type of each part. The changes are similar to those described in the proof of Lemma 13. When refined, unipolar parts retain their type, but bipolar parts may be refined into smaller parts that may be either unipolar or bipolar. When a bipolar part is refined into a unipolar part, the signs λ or γ are determined from its elements based on the column or row sums. In our refinement algorithm, we partition the columns (rows) of a bipolar part based on the absolute value of the sums, rather than the sums directly as in the refinement algorithm in [11]. Then, we set the signs λ_w so that the column sums are all positive when multiplied by λ_w. Finally, one improvement that is directly suggested by Corollary 10 is that if P is a bipolar type, that then $\widehat{M}_{P,Q}$ is an equitable block for any $Q \in \mathcal{Q}$. Thus, we do not need to add bipolar parts to the search stack, since checking them for refinements is pointless.

There are two cases where one needs to be careful with the adapation of the color refinement algorithm to the reduced matrix size. First, if a bipolar part contains row or columns sums with identical but nonzero values, then it must be converted to a unipolar part and added to the search stack. Second, we note that one has to be careful with the 'leave largest refined part out'-rule that is used in most color refinement algorithms and is key to the fast running time of color refinement [36]. This rule pushes all but the largest part obtained by a refinement of a part to the stack of parts for which refinements need to be checked. Since we are modeling the refinement of symmetric partitions of the split reformulation, we cannot leave out a unipolar part in the algorithm that runs on the original matrix since it represents two parts in the split reformulation, and leaving out both parts would break correctness. Luckily, this does not lead to a slower running time, since the running time is still guaranteed by Theorem 25 as we maintain the invariant that our partition models a symmetric partition of the split reformulation.

Choosing γ, λ and δ

One detail that is important for a practical implementation is the choice of δ. If δ has many nonzero entries and the matrix A has large values, then the product $A\delta$ that appears in the right-hand side of the reflection reduced model can become very large and lead to numerical issues.

Recall that the γ and λ vectors that were generated by the symmetric equitable partition were not unique, and could be optionally negated for each part $P \in \mathcal{P}_U$ and $Q \in \mathcal{Q}_U$. In order to minimize the number of complemented rows, we choose γ so that for each $P \in \mathcal{P}_U$ at least $\frac{|P|}{2}$ rows $v \in P$ have value $\gamma_v = 1$. Similarly, we minimize the number of complemented variables by picking λ such that for each $Q \in \mathcal{Q}_U$ at least $\frac{|Q|}{2}$ variables $w \in Q$ have value $\lambda_w = 1$.

As a starting point for our δ vector we consider $\delta^{\mathcal{S}}$ as defined in Corollary 21. However, this vector may still not be very sparse due to the ℓ_w and u_w values. By Theorem 20, we may modify $\delta^{\mathcal{S}}$ to $\delta \coloneqq \delta^{\mathcal{S}} + (\Lambda\nu, \mathbb{0})$ for any $\mathcal{Q}_U$-equitable vector ν. In our implementation, we choose ν so that δ becomes sparse. For every part $Q \in \mathcal{Q}_U$, we compute how often the value $\lambda_w \delta^{\mathcal{S}}_w$ appears for all $w \in Q$ and set ν_Q to be the negative of the most frequently appearing value. This way, most variables $w \in Q$ satisfy

$$\delta_w = \delta^{\mathcal{S}}_w + \lambda_w \nu_w = \delta^{\mathcal{S}}_w + \lambda_w(-\lambda_w \delta^{\mathcal{S}}_w) = \delta^{\mathcal{S}}_w - \delta^{\mathcal{S}}_w = 0.$$

Thus, the resulting δ vector is typically quite sparse. Furthermore, we remark that for a unipolar variable $w \in W$ with integral bounds, $\delta^{\mathcal{S}}_w$ is also integral. One important consequence is that if all variables in some part $Q \in \mathcal{Q}_U$ have integral bounds, then δ_Q is also integral. Although we cannot modify $\delta^{\mathcal{S}}_w$ for variables w in bipolar parts, these values do not lead to numerical issues as fixing the corresponding values does not affect the right-hand sides since the corresponding parts have zero sums due to bipolarity.

4 Dimension Reduction for Mixed-Integer Linear Programs

Throughout this section we consider a mixed-integer linear program given by a linear program $F(A, b, \ell, u, c)$ with $A \in \mathbb{R}^{V \times W}$, $b \in \mathbb{R}^V$, $\ell, u, c \in \mathbb{R}^W$ and integer variables $W_I \subseteq W$. For any partition $\mathcal{Q}$ of W that is compatible with W_I, we use $\mathcal{Q}_I \coloneqq \{Q \in \mathcal{Q} \mid Q \subseteq W_I\}$ to denote the corresponding partition over the integer variables.

Let us consider the effect of DRCR in context of mixed-integer linear programs. For the mixed-integer linear program given by $F(A, b, \ell, u, c)$ and integrality constraints $x \in \mathbb{MI}^{W_I}$, we consider the *reduced problem* $R^{\mathcal{P},\mathcal{Q}}(F(A, b, \ell, u, c))$ with integrality constraints $y \in \mathbb{MI}^{\mathcal{Q}_I}$ for $y \in R^{\mathcal{P},\mathcal{Q}}(F(A, b, \ell, u, c))$. For the linear programming relaxations, Proposition 3 establishes a correspondence between the original and the reduced problem. Let us consider how these mappings interact with integrality. For a solution $x \in F(A, b, \ell, u, c) \cap \mathbb{MI}^{W_I}$, we can observe that $y \coloneqq \Pi^{\intercal}_{\mathcal{Q}} x$ satisfies the integrality constraints for the reduced problem, since then $y_Q \coloneqq \sum_{w \in Q} x_w$ is integral for any $Q \in \mathcal{Q}_I$ as it is the sum of integral values. Linear feasibility and optimality are guaranteed already by Proposition 3. Thus, every mixed-integer solution for the original problem maps to a mixed-integer solution of the reduced problem. However, for the other direction integrality is not preserved, since for $y \in R^{\mathcal{P},\mathcal{Q}}(F(A, b, \ell, u, c)) \cap \mathbb{MI}^{\mathcal{Q}_I}$ the original solution is given for $w \in W$ with $w \in Q$ by $x_w \coloneqq \frac{1}{|Q|} y_Q$, which is not necessarily integral. In particular, this is an issue if the reduced problem contains integer solutions $y \in R^{\mathcal{P},\mathcal{Q}}(F(A, b, \ell, u, c)) \cap \mathbb{MI}^{\mathcal{Q}_I}$ that do not map to integer solutions of the original problem. Previous works on Orbital Shrinking [8–10] deal with this limitation by solving a feasibility subproblem, which checks if the provided mixed-integer solution for the reduced problem can be mapped to a mixed-integer solution of the original problem. For any fixed mixed-integer

y-solution, these subproblems can be formulated by adding the constraints $\Pi_{\mathcal{Q}}^{\intercal} x = y$ to the original MILP. This subproblem is a mixed-integer linear program of reduced size with no objective, and the DRCR procedures defines a point that is feasible for its LP relaxation. Frequently, this subproblem is solved using techniques such as constraint programming that exploit the specific structure of the subproblem [9, 25].

In this work, we take a different approach compared to previous works on Orbital Shrinking by characterizing when each solution of the reduced problem corresponds to a solution of the original model. In terms of orbital shrinking, one can think of our approach as *Exact Orbital Shrinking*, where the orbital shrinking subproblem is given by a perfect formulation of a mixed-integer linear program. In this case, the orbital shrinking subproblem can be solved via linear programming. One of the main advantages of our approach is that it guarantees a tight correspondence between the mixed-integer solutions of the original and the reduced problem, and that the subproblems can be solved easily via linear programming. This correspondence comes at a cost: the main downside of our approach is that it cannot handle all symmetries. Due to the requirement that the subproblem is a perfect formulation, our approach cannot in general handle the complete symmetry group of the MILP, and instead only handles a subgroup of the symmetry group that generates subproblems that are perfect formulations.

In Theorem 26, we characterize when reflection reductions, which generalize the DRCR procedure, can be used to reduce the dimension of a mixed-integer linear program.

Theorem 26 *Let $(\mathcal{P}, \mathcal{Q}, \gamma, \lambda, \delta, V_U, W_U)$ be a reflection reduction of $F(A, b, \ell, u, c)$ such that $\delta \in \mathbb{MI}^{W_I \cap W_U}$ holds and $W_I \cap W_U$ is compatible with $\mathcal{Q}$. We formulate the reduced mixed integer program using $A' \coloneqq \Pi_{\mathcal{P}}^{\intercal} \Gamma A_{V_U, W_U} \Lambda \widetilde{\Pi_{\mathcal{Q}}}$, $b' \coloneqq \Pi_{\mathcal{P}}^{\intercal} \Gamma (b_{V_U} - A_{V_U, W_U} \delta_{W_U})$, $\ell' \coloneqq \Pi_{\mathcal{Q}}^{\intercal} \ell_{W_U}^{\lambda, \delta}$, $u' \coloneqq \Pi_{\mathcal{Q}}^{\intercal} u_{W_U}^{\lambda, \delta}$ and $c' \coloneqq \widetilde{\Pi_{\mathcal{Q}}}^{\intercal} \Lambda c_{W_U}$. If the following equation holds*

$$\operatorname{conv}(\{x \in P(A, b, \ell, u) \mid (\Pi_{\mathcal{Q}}^{\intercal} \Lambda x)_{\mathcal{Q}_I} \in \mathbb{Z}^{\mathcal{Q}_I}\}) = \operatorname{conv}(P(A, b, \ell, u) \cap \mathbb{MI}^{W_I}) \tag{13}$$

then the following hold:

(i) For any $x \in P(A, b, \ell, u) \cap \mathbb{MI}^{W_I}$, then $y \coloneqq \Pi_{\mathcal{Q}}^{\intercal} \Lambda (x_{W_U} - \delta_{W_U}) \in P(A', b', \ell', u') \cap \mathbb{MI}^{\mathcal{Q}_I}$ and $c^{\intercal} x = c'^{\intercal} y + c^{\intercal} \delta$ hold.

(ii) For any $y \in P(A', b', \ell', u') \cap \mathbb{MI}^{\mathcal{Q}_I}$ there exists a solution $x \in P(A, b, \ell, u) \cap \mathbb{MI}^{W_I}$ such that $c^{\intercal} x \leq c'^{\intercal} y + c^{\intercal} \delta$ hold.

Proof First, consider the proof of the first point. By Definition 18(i), it follows that $y \in P(A', b', \ell', u')$ and $c^{\intercal} x = c'^{\intercal} y + c^{\intercal} \delta$ hold. Thus, it is sufficient to show that $y \in \mathbb{MI}^{\mathcal{Q}_I}$ holds. First, since $x \in \mathbb{MI}^{W_I}$ and $\delta \in \mathbb{MI}^{W_I \cap W_U}$ hold we have that $x_{W_U} - \delta_{W_U} \in \mathbb{MI}^{W_I \cap W_U}$. Then, for any $Q \in \mathcal{Q}_I$, we have that $y_Q = (\Pi_{\mathcal{Q}}^{\intercal} \Lambda (x_{W_U} - \delta_{W_U}))_Q = \sum_{w \in Q} \lambda_w (x_w - \delta_w)$. Since $Q \in \mathcal{Q}_I$ holds, it follows that all $w \in Q$ satisfy $w \in W_I$. Thus, this implies that x_w and δ_w must be integral by the given conditions. Then, since λ_w is also integral, it follows that y_Q is integral. Since this holds for any $Q \in \mathcal{Q}_I$, we have $y \in \mathbb{MI}^{\mathcal{Q}_I}$.

We proceed with the proof of the second point. Let $y \in P(A', b', \ell', u') \cap \mathbb{MI}^{\mathcal{Q}_I}$ be given. Then, by Definition 18(ii) we have for $x' := (\Lambda\widetilde{\Pi_{\mathcal{Q}}} y, \mathbb{O}) + \delta$ that $x' \in P(A, b, \ell, u)$ and $c'^{\intercal} y + c^{\intercal}\delta = c^{\intercal} x'$ hold. Then, we observe that for each $Q \in \mathcal{Q}$, we have the following for $\Pi_{\mathcal{Q}}^{\intercal}\Lambda x'$:

$$(\Pi_{\mathcal{Q}}^{\intercal}\Lambda x')_Q = \sum_{w \in Q} \lambda_w x'_w = \sum_{w \in Q} \lambda_w (\frac{\lambda_w y_Q}{|Q|} + \delta_w) = \sum_{w \in Q} \frac{y_Q}{|Q|} + \lambda_w \delta_w = y_Q + \sum_{w \in Q} \lambda_w \delta_w.$$

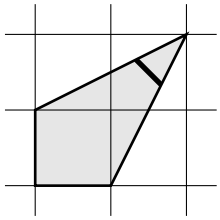
Then, for $Q \in \mathcal{Q}_I$, y_Q is integral by assumption, and all $w \in Q$ lie in W_I by definition, which shows that $\sum_{w \in Q} \lambda_w \delta_w$ is integral since $\delta \in \mathbb{MI}^{W_I \cap W_U}$ holds. Thus, we have that $x' \in \operatorname{conv}(\{x \in P(A, b, \ell, u) \mid (\Pi_{\mathcal{Q}}^{\intercal}\Lambda x)_{\mathcal{Q}_I} \in \mathbb{Z}^{\mathcal{Q}_I}\})$ holds. By the equality in (13) we have that $x' \in \operatorname{conv}(P(A, b, \ell, u) \cap \mathbb{MI}^{W_I})$ holds. Since $\operatorname{conv}(P(A, b, \ell, u) \cap \mathbb{MI}^{W_I})$ is a convex polyhedron, there exists some $x \in P(A, b, \ell, u) \cap \mathbb{MI}^{W_I}$ such that $c^{\intercal} x \leq c^{\intercal} x'$ holds. Then $c^{\intercal} x \leq c'^{\intercal} y + c^{\intercal}\delta$ follows, which completes the proof. □

Theorem 26 differs from reflection reductions in a few subtle but important ways. First of all, the new and crucial condition (13) in Theorem 26 states for each part $Q \in \mathcal{Q}_I$ that the integrality of all variables in Q must be implied by the integrality of the sum $\sum_{w \in Q} \lambda_w x_w \in \mathbb{Z}$. We will later elaborate on a few different methods to detect this fact. Additionally, note that the partition $\mathcal{Q}$ may not have parts that contain both integer and continuous variables and that $\delta_{W_I \cap W_U}$ must be integral.

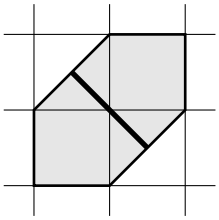
In the second point of Theorem 26 we only show existence of a better solution in the original problem, rather than proving that a direct mapping exists, as was done in Proposition 3. This is due to the fact that the mapped solution, x' in the proof, does not necessarily satisfy all the integrality constraints for W_I, even though it is feasible for the linear relaxation. This raises an important issue; it is not yet clear how, given arbitrary $x' \in P(A', b', \ell', u')$, if one can obtain in polynomial time a solution solution x to the original problem that also satisfies the integrality constraints. Intuitively, one may suspect that it is sufficient to consider the affine subspace generated by $x = \Pi_{\mathcal{Q}}^{\intercal}\Lambda(x_{W_U} - \delta_{W_U})$ in the original problem and recover an original solution. This approach is also taken in Orbital Shrinking [10]. However, we show in Example 27 that it may be the case that there exists no solution $x \in P(A, b, \ell, u) \cap \mathbb{MI}^{W_I}$ such that $x' = \Pi_{\mathcal{Q}}^{\intercal}\Lambda(x_{W_U} - \delta_{W_U})$ holds, even though the conditions of Theorem 26 and $x' \in \mathbb{MI}^{\mathcal{Q}_I}$ hold. This fact is an important limitation for Orbital Shrinking [10] as it may be difficult to recover integral solutions for the original integer program.

Example 27 Consider the integer program given by polyhedron $P = \{(x, y) \in \mathbb{R}^2 \mid -x + 2y \leq 2,\ 2x - y \leq 2, x \geq 0, y \geq 0\}$ and $x \in \mathbb{Z}, y \in \mathbb{Z}$ and note that $P = \operatorname{conv}(P \cap \mathbb{Z}^2) = \operatorname{conv}(\{(x, y) \in P \mid x + y \in \mathbb{Z}\})$ holds and that Theorem 26 is applicable to the standard representation of the integer program, where $\lambda = \gamma = \mathbb{1}$, $V_U = V$ and $W_U = W$ hold. See Figure 1a for the geometric representation of P. For the reduced integer program $P' = \{z \in \mathbb{R} \mid z \leq 4, z \geq 0\} \cap \mathbb{Z}$, $z = 3$ is a feasible integer solution. However, there exists no solution $(x, y) \in P$ such that $x + y = 3$, $x \in \mathbb{Z}$ and $y \in \mathbb{Z}$ hold.

Finally, Theorem 26 clarifies the need to prove the variant of Proposition 3 compared to that in [11]. In [11], the mapping from the original to the reduced problem is scaled, which makes it so that integrality of the original solution does not imply



(a) P from Example 27 (thin) and its subset induced by $x + y = 3$ (thick).



(b) A symmetric integral polyhedron P (thin) with a totally unimodular constraint matrix whose x and y fibers are integral, but for which the fiber defined by $x + y = 2$ (thick) is not integral.

Fig. 1: Examples of symmetric integral polyhedra with non-integral affine fibers

integrality of the mapped solution in the reduced problem. Instead, we require the scaling as in Proposition 3 in Theorem 26 to ensure that we can map integral solutions between the original and the reduced problem. A similar scaling is also used in Orbital Shrinking [10].

Next, we will consider several methods to detect that (13) is satisfied. For each method, we additionally show that a solution to the original problem can be recovered in polynomial time from a solution to the reduced problem.

First, we show in Lemma 28 that an equitable partition where each integer variable is contained in its own part of size 1 satisfies (13). This result seems to already be known as a folklore result within the integer programming solver community[1], but to the best of our knowledge it was never documented in public. This idea also extends naturally to reflection reductions.

Lemma 28 *If $\delta \in \mathbb{MI}^{W_I}$ and if for each $w \in W_I$, $\{w\} \in \mathcal{Q}$ holds, then (13) is satisfied. Additionally, for every $y \in P(A', b', \ell', u') \cap \mathbb{MI}^{\mathcal{Q}_I}$, $x' \coloneqq (\Lambda \widetilde{\Pi_{\mathcal{Q}}} y, \mathbb{O}) + \delta$ lies in $P(A, b, \ell, u) \cap \mathbb{MI}^{W_I}$.*

Proof For the first point, note that $(\Pi_{\mathcal{Q}}^{\intercal} \Lambda x)_{\mathcal{Q}_I} \in \mathbb{Z}^{\mathcal{Q}_I} \iff (\Lambda x)_{W_I} \in \mathbb{Z}^{W_I} \iff x_{W_I} \in \mathbb{Z}^{W_I}$ holds, where the first equivalence follows since every $w \in W_I$ is a part of size one. The second equivalence holds since Λ is a unimodular matrix. Thus, this directly establishes that (13) holds.

For the second point, note that for all $\{w\} = Q \in \mathcal{Q}_I$, $x_w = ((\Lambda \widetilde{\Pi_{\mathcal{Q}}} y, \mathbb{O}) + \delta)_w = \lambda_w y_Q + \delta_w$ holds since $|Q| = 1$ holds. The latter expression is integral since y_Q, δ_w and λ_w are all integral, which shows that $x \in \mathbb{MI}^{W_I}$. By Definition 18(ii), $x \in P(A, b, \ell, u)$ holds, which completes the proof. □

It is implicit in Lemma 28 that all bipolar parts must consist of continuous variables only by the condition that each $w \in W_I$ has $\{w\} \in \mathcal{Q}$. This condition is not required in Theorem 26.

[1] Developers of both Gurobi and FICO Xpress mentioned this idea in personal communication.

The condition in Lemma 28 that each integer variable is in its own partition class can be easily enforced during detection by modifying the initial partition $\mathcal{Q}_0$ of the refinement algorithm. The condition also implies a strong condition on the rows by the requirement that the partition is equitable for the constraint matrix. In particular, this implies that for any $P \in \mathcal{P}$ that $A_{v_1,w} = A_{v_2,w}$ holds for any $v_1, v_2 \in P$ and $w \in W_I$. In other words, for any two rows v_1, v_2 that are in the same row partition, the rows must be identical in the column submatrix $A_{\star,W_I}$. Another perspective is that we are considering DRCR for a parametric set of LPs, whose right-hand sides depend on the fixed integer values for the x_{W_I} variables. Since the right-hand sides for two rows in the same row-block need to be identical under all fixed integer values of x_{W_I}, their rows in $A_{\star,W_I}$ must be identical.

Although Lemma 28 is very useful, the detection of (13) is mostly seen as an afterthought and is done by making the partition much finer. Thus, we investigate more sophisticated methods to detect (13) which do not require every integer variable to be in its own part, so that we may also aggregate integer variables. In the context of detecting equivalent mixed-integer formulations as we have in (13), there are two works that are relevant to the current work. Recall that a matrix A is said to be *totally unimodular* if $\det(A') = \pm 1$ holds for every nonsingular square submatrix A' of A. A famous result by Hoffman and Kruskal [43] shows that $\{x \in \mathbb{R}^n \mid Ax \leq b, x \geq 0\}$ defines an integral polyhedron for all $b \in \mathbb{Z}^m$ if and only if A is totally unimodular. First of all, the author has previously investigated the phenomenon of *implied integrality* [29], which is said to occur when $\text{conv}(P \cap \mathbb{MI}^W) = \text{conv}(P \cap \mathbb{MI}^{W'})$ holds for some $W' \subseteq W$. There, implied integrality is detected by detecting totally unimodular submatrices within the constraint matrix.

In their work, Bader, Hildebrand, Weismantel and Zenklusen [30] investigate methods to detect matrices $T \in \mathbb{Z}^{k \times n}$ where $k < n$ such that $\text{conv}(\{x \in P \mid Tx \in \mathbb{Z}^k\}) = \text{conv}(P \cap \mathbb{Z}^n)$ holds. They investigate the case where T is totally unimodular and formulate the notion of an *affine totally unimodular decomposition* (affine TU decomposition) to detect such cases.

Definition 29 (Bader, Hildebrand, Weismantel & Zenklusen[30]) *A matrix $A \in \mathbb{Z}^{m \times n}$ admits a* k-row affine totally unimodular decomposition *if there exist matrices $S \in \mathbb{Z}^{m \times n}$, $U \in \mathbb{Z}^{m \times k}$ and $T \in \mathbb{Z}^{k \times n}$ such that $\begin{bmatrix} S \\ T \end{bmatrix}$ is totally unimodular and $A = S + UT$ holds.*

In particular, they show that affine TU decompositions can be used to remove and reformulate integrality constraints.

Proposition 30 ([30]) *Let $A = S + UT \in \mathbb{Z}^{m \times n}$ with $T \in \{-1, 0, 1\}^{k \times n}$ be a k-row affine TU decomposition and $b \in \mathbb{Z}^m$ and $\ell \in (\mathbb{Z} \cup \{-\infty\})$, $u \in (\mathbb{Z} \cup \{\infty\}$ with $\ell \leq u$. Then for $P = \{x \in \mathbb{R}^n \mid \ \ell \leq x \leq u, Ax \leq b\}$ it holds that $\text{conv}(\{x \in P \mid Tx \in \mathbb{Z}^k\}) = \text{conv}(P \cap \mathbb{Z}^n)$.*

Note that the matrix $\Pi_{\mathcal{Q}}^{\intercal}\Lambda$ in (13) is totally unimodular as it contains at most one ± 1 entry in every column. Thus, the notion of an affine TU decomposition is very suitable for our setting.

One idea which is central to the proof of Proposition 30 and the detection of implied integrality, is to show that every *fiber* defined by a linear subspace of P given by $Tx = d$ for some $d \in \mathbb{Z}^Q$ is (mixed)-integral. Although this is not a necessary condition for the possibility of a mixed-integer formulation (see Fig. 1a), it is practical to detect and exploit this condition. In our setting, imposing the requirement that the fibers have to be integral has the additional benefit that an integral solution to the original problem may be recovered in polynomial time, avoiding the issue in Figure 1a. We adapt Lemma 31 from the proof of Proposition 30 in [30] to work in a slightly more general mixed-integer setting.

Lemma 31 (Generalized from [30]) *Let $P \subseteq \mathbb{R}^W$ be a rational polyhedron, let $W_I \subseteq W$ denote the integer variables. Let $T \in \mathbb{Z}^{k \times W_I}$ be any integral matrix and consider the affine fiber $H_d \coloneqq \{x \in P \mid Tx_{W_I} = d\}$ for any fixed $d \in \mathbb{Z}^k$. If H_d is W_I-integral for every $d \in \mathbb{Z}^k$, then $\operatorname{conv}(\{x \in P \mid Tx_{W_I} \in \mathbb{Z}^k\}) = \operatorname{conv}(P \cap \mathbb{MI}^{W_I})$ holds.*

Proof Let $G \coloneqq \operatorname{conv}(\{x \in P \mid Tx_{W_I} \in \mathbb{Z}^k\})$. First, we show that G is a rational polyhedron. This is relatively easy to see, since it is the projection of the rational mixed integer program given by $G' \coloneqq \operatorname{conv}(\{(x, d) \in \mathbb{MI}^{W_I} \times \mathbb{Z}^k \mid x \in P,\ Tx_{W_I} = d\})$. By Meyer's theorem [44], G' is a rational polyhedron. Then, since the projection of a rational polyhedron is again a polyhedron and $G = \operatorname{proj}_x(G')$, G is also a rational polyhedron.

Then the following holds:

$$G = \operatorname{conv}(\{x \in P \mid Tx_{W_I} \in \mathbb{Z}^k\}) = \operatorname{conv}\Big(\bigcup_{d \in \mathbb{Z}^k} H_d\Big) \overset{\text{(a)}}{=} \operatorname{conv}\Big(\bigcup_{d \in \mathbb{Z}^k} \operatorname{conv}(H_d \cap \mathbb{MI}^{W_I})\Big)$$
$$= \operatorname{conv}\Big(\bigcup_{d \in \mathbb{Z}^k} H_d \cap \mathbb{MI}^{W_I}\Big) \overset{\text{(b)}}{=} \operatorname{conv}(P \cap \mathbb{MI}^{W_I}).$$

The equality in (a) follows from the fact that H_d is W_I-integral. For (b) note that T is an integral matrix, which implies for all $x \in \mathbb{MI}^{W_I}$ that $Tx_{W_I} = d$ holds for some $d \in \mathbb{Z}^k$. Then, the equality $\bigcup_{d \in \mathbb{Z}^k} H_d \cap \mathbb{MI}^{W_I} = P \cap \mathbb{MI}^{W_I}$ holds since any $x \in P \cap \mathbb{MI}^{W_I}$ must be contained in $H_d \cap \mathbb{MI}^{W_I}$ for some $d \in \mathbb{Z}^k$, which implies that $P \cap \mathbb{MI}^{W_I} \subseteq \bigcup_{d \in \mathbb{Z}^k} H_d \cap \mathbb{MI}^{W_I}$ holds. The reverse inclusion follows easily since $H_d \subseteq P$ holds. □

Then, using Lemma 31, we show that one can recover solutions to the original problem from solutions to the reduced problem in polynomial time.

Theorem 32 *Consider the setting of Theorem 26 and let $\mathcal{Q}' \subseteq \mathcal{Q}_I$. If for each $d \in \mathbb{Z}^{\mathcal{Q}'}$, $H_d \coloneqq \{x \in P(A, b, \ell, u) \mid (\Pi_{\mathcal{Q}}^{\intercal}\Lambda x)_{\mathcal{Q}'} = d\}$ is an $\mathbb{MI}^{W_I}$-integral polyhedron then (13) holds and $H'_{d'} \coloneqq \{y \in P(A', b', \ell', u') \mid y_{\mathcal{Q}'} = d'\}$ is a $\mathcal{Q}_I$-integral polyhedron for each $d' \in Z^{\mathcal{Q}'}$. Moreover, for every $y \in P(A', b', \ell', u') \cap \mathbb{MI}^{Q'}$ there exists a feasible solution $x \in H_{d^y} \cap \mathbb{MI}^{W_I}$ with $d^y \coloneqq (y + \Pi_{\mathcal{Q}}^{\intercal}\Lambda\delta_{W_U})_{\mathcal{Q}'}$ that can be found in polynomial time, where $c^{\intercal}x \leq c'^{\intercal}y + c^{\intercal}\delta$ holds.*

Proof For $G = P(A, b, \ell, u)$, we have that

$$\{x \in G \mid (\Pi_{\mathcal{Q}}^{\intercal} \Lambda x)_{\mathcal{Q}'} \in \mathbb{Z}^{\mathcal{Q}'}\} \subseteq \{x \in G \mid (\Pi_{\mathcal{Q}}^{\intercal} \Lambda x)_{\mathcal{Q}_I} \in \mathbb{Z}^{\mathcal{Q}_I}\} \subseteq G \cap \mathbb{MI}^{W_I},$$

where the first inclusion follows since $\mathcal{Q}' \subseteq \mathcal{Q}_I$ holds and the latter inclusion since $(\Pi_{\mathcal{Q}}^{\intercal} \Lambda x)_{\mathcal{Q}_I} \in \mathbb{Z}^{\mathcal{Q}_I}$ holds for all $x \in \mathbb{MI}^{W_I}$. Then, by monotonicity of the convex hull we have:

$$\operatorname{conv}(\{x \in G \mid (\Pi_{\mathcal{Q}}^{\intercal} \Lambda x)_{\mathcal{Q}'} \in \mathbb{Z}^{\mathcal{Q}'}\}) \subseteq \operatorname{conv}(\{x \in G \mid (\Pi_{\mathcal{Q}}^{\intercal} \Lambda x)_{\mathcal{Q}_I} \in \mathbb{Z}^{\mathcal{Q}_I}\}) \subseteq \operatorname{conv}(G \cap \mathbb{MI}^{W_I}).$$

Since the conditions of Lemma 31 are satisfied for $\mathcal{Q}'$, we have equality throughout, which shows that (13) holds.

Secondly, let $y \in P(A', b', \ell', u') \cap \mathbb{MI}^{\mathcal{Q}'}$ be a feasible solution to the reduced problem. By Definition 18(ii) it holds for $x' := (\widetilde{\Lambda \Pi_{\mathcal{Q}}} y, \mathbb{O}_{W_B}) + \delta$ that $x' \in P(A, b, \ell, u)$ and $c^{\intercal} x' = c'^{\intercal} y + c^{\intercal} \delta$. Then, we have for $Q \in \mathcal{Q}'$ that:

$$(\Pi_{\mathcal{Q}}^{\intercal} \Lambda x')_Q = \sum_{w \in Q} \lambda_w x'_w = \sum_{w \in Q} \lambda_w (\lambda_w \frac{y_Q}{|Q|} + \delta_w) = y_Q + \sum_{w \in Q} \lambda_w \delta_w = (y + \Pi_{\mathcal{Q}}^{\intercal} \Lambda \delta_{W_U})_Q = d^y_Q,$$

where d^y_Q is integral by integrality of y_Q, Λ and $\delta_{W_I \cap W_U}$. Note that we implicitly use that $Q \subseteq W_U$ holds, which is implied by $Q \in \mathcal{Q}'$. Since this holds for any $Q \in \mathcal{Q}'$, it follows that $x' \in H_{d^y}$ holds. Since H_{d^y} is W_I-integral and nonempty since $x' \in H_{d^y}$ holds, the linear program $\min\{c^{\intercal} x \mid x \in H_{d^y}\}$ contains a solution $x \in H_{d^y} \cap \mathbb{MI}^{W_I}$ in its minimal face such that $c^{\intercal} x \leq c^{\intercal} x'$. Such a solution can be found in polynomial time using the ellipsoid method [45], see [46] and [47] for an explanation on how the optimal minimal face can be identified. Proposition 6.1 in [29] provides a proof of how x can be determined from the minimal face in polynomial time. Furthermore, note that $c^{\intercal} x \leq c^{\intercal} x' = c'^{\intercal} y + c^{\intercal} \delta$ holds.

Finally, let us show that $H'_{d'}$ is an $\mathcal{Q}_I$-integral polyhedron for any $d' \in \mathbb{Z}^{\mathcal{Q}'}$. First, note that we obtained $P(A', b', \ell', u')$ through the affine map $\kappa : \mathbb{R}^{W_U} \to \mathbb{R}^{\mathcal{Q}}$ defined as $\kappa(x_{W_U}) := \Pi_{\mathcal{Q}}^{\intercal} \Lambda (x_{W_U} - \delta W_U)$ that was applied to $P(A, b, \ell, u)$. Thus, the preimage of $H'_{d'}$ in $P(A, b, \ell, u)$ consists of those $x \in P(A, b, \ell, u)$ such that $(\Pi_{\mathcal{Q}}^{\intercal} \Lambda (x_{W_U} - \delta_{W_U}))_{\mathcal{Q}'} = d'$. Then, for $d := d' + (\Pi_{\mathcal{Q}}^{\intercal} \Lambda \delta_{W_U})'_{\mathcal{Q}}$ the preimage of $H'_{d'}$ is exactly H_d. As we have $|\mathcal{Q}| \leq |W_U|$, κ is a surjective affine transformation and standard theory shows that the preimage of any face F' of $H'_{d'}$ contains some minimal face F of H_d (see e.g. [48, Lemma 7.10]). Since H_d is $\mathbb{MI}^{W_I}$-integral, there exists some $x \in F \cap \mathbb{MI}^{W_I}$. Then, note that $y := \Pi_{\mathcal{Q}}^{\intercal} \Lambda (x_{W_U} - \delta_{W_U}) \in \mathbb{MI}^{\mathcal{Q}_I}$ holds, which we show in the proof of Theorem 26(i). Then, since $y \in F'$ holds, it follows that any minimal face F' of $H'_{d'}$ contains a $\mathcal{Q}_I$-integral point, which shows that $H'_{d'}$ is $\mathcal{Q}_I$-integral. □

In Theorem 32, we also show that H'_d is integral. By Lemma 31, this directly implies that $\operatorname{conv}(P(A', b', \ell', u') \cap \mathbb{MI}^{\mathcal{Q}'}) = \operatorname{conv}(P(A', b', \ell', u') \cap \mathbb{MI}^{\mathcal{Q}_I})$ holds for the reduced problem. Thus, any variables $Q \in \mathcal{Q}_I \setminus \mathcal{Q}'$ are implied integer in the reduced problem. One may hypothesize that the condition $\mathcal{Q}' \subseteq \mathcal{Q}_I$ is unnecessary and that the integrality of the fibers for $\mathcal{Q}'$ implies the integrality for the fibers of $\mathcal{Q}_I$. The example in Figure 1b shows that this is not the case, and that the condition $\mathcal{Q}' \subseteq \mathcal{Q}_I$ can indeed be helpful when the the fibers obtained by fixing the matrix with respect to $\mathcal{Q}_I$ are not integral.

Given Theorem 32, we are motivated to investigate methods to determine when the affine fibers H_d are integral. Although Proposition 30 is useful for integer programs, we cannot apply it directly to mixed-integer programs, as they do not satisfy its conditions. Thus, we generalize affine TU decompositions to mixed-integer linear programs to deal with this limitation.

4.1 Affine TU decompositions for mixed-integer linear programs

We consider a mixed-integer linear program with variables $(x, y, z, \omega) \in \mathbb{Z}^{W^1} \times \mathbb{Z}^{W^2} \times \mathbb{R}^{W^3} \times \mathbb{MI}^{W_I^4}$, where $W_I^4 \subseteq W^4$ are the integer variables in W^4. We use $W_I = W^1 \cup W^2 \cup W_I^4$ to denote the integer variables and $W = W^1 \cup W^2 \cup W^3 \cup W^4$ to denote all variables. The linear relaxation of the integer program is of the form:

$$P \coloneqq \{Ax + By + Cz = b,\ Dx + Ey + F\omega = g,\ \ell \leq \begin{bmatrix} x & y & z & \omega \end{bmatrix}^\intercal \leq u\},$$

where $A \in \mathbb{Z}^{m_1 \times W^1}$, $B \in \mathbb{Z}^{m_1 \times W^2}$, $C \in \mathbb{Z}^{m_1 \times W^3}$, $b \in \mathbb{Z}^{m_1}$, $D \in \mathbb{Q}^{m_2 \times W^1}$, $E \in \mathbb{Q}^{m_2 \times W^2}$, $F \in \mathbb{Q}^{m_2 \times W^4}$ and $\ell \leq u$ hold, where ℓ and u are rational vectors where we consider infinite entries to be integral. Any MILP can be brought into this form if we allow $m_1 = 0$ and $m_2 = 0$ to hold.

Theorem 33 *Suppose that for $k \leq |W^2|$, there exist matrices $S \in \mathbb{Z}^{m_1 \times W^2}$, $U \in \mathbb{Z}^{m_1 \times k}$, $T \in \mathbb{Z}^{k \times W^2}$ and $R \in \mathbb{Q}^{m_2 \times k}$ such that $B = S + UT$, $E = RT$ and $\begin{bmatrix} S & C \\ T & \mathbb{O} \end{bmatrix}$ is totally unimodular and that $\ell, u \in \mathbb{MI}^{W_I \cup W^3}$ holds. Then, for any $d = (d_{W^1}, d_T, d_{W_I^4}) \in \mathbb{Z}^{W^1} \times \mathbb{Z}^k \times \mathbb{Z}^{W_I^4}$, the affine fiber $H_d \coloneqq \{(x, y, z, \omega) \in P \mid x = d_{W^1},\ Ty = d_T,\ \omega_{W_I^4} = d_{W_I^4}\}$ satisfies*

$$H_d = \operatorname{conv}(H_d \cap \mathbb{MI}^{W_I}) = \operatorname{conv}(H_d \cap \mathbb{MI}^{W_I \cup W^3}) \text{ and}$$

$$\operatorname{conv}(P \cap \mathbb{MI}^{W_I}) = \operatorname{conv}(P \cap \mathbb{MI}^{W_I \cup W^3}) = \operatorname{conv}(\{(x, y, z, \omega) \in P \mid x \in \mathbb{Z}^{W^1},\ Ty \in \mathbb{Z}^k,\ \omega \in \mathbb{MI}^{W_I^4}\})$$

holds.

Proof We consider the affine fiber H_d for any fixed $d \in \mathbb{Z}^{W^1} \times \mathbb{Z}^k \times \mathbb{Z}^{W_I^4}$ and rewrite it by substituting $B = S + UT$ and $D = RT$.

$$\begin{aligned}
H_d \coloneqq &\{(x, y, z, \omega) \in \mathbb{R}^W \mid Ax + By + Cz = b,\ Dx + Ey + F\omega = g,\ x = d_{W^1},\ Ty = d_T,\\
&\qquad \omega_{W_I^4} = d_{W_I^4},\ \ell \leq \begin{bmatrix} x & y & z & \omega \end{bmatrix}^\intercal \leq u\}\\
= &\{(x, y, z, \omega) \in \mathbb{R}^W \mid x = d_{W^1},\ Sy + Cz = b - Ad_{W^1} - Ud_T,\ Ty = d_T,\\
&\qquad F\omega = g - Dd_{W^1} - Rd_T,\ \omega_{W_I^4} = d_{W_I^4},\ \ell \leq \begin{bmatrix} x & y & z & \omega \end{bmatrix}^\intercal \leq u\}\\
= &\{x \in \mathbb{R}^{W^1} \mid x = d_{W^1},\ \ell_{W^1} \leq x \leq u_{W^1}\} \times\\
&\{\omega \in \mathbb{R}^{W^4} \mid F\omega = g - Dd_{W^1} - Rd_T,\ \omega_{W_I^4} = d_{W_I^4},\ \ell_{W^4} \leq \omega \leq u_{W^4}\} \times\\
&\{(y, z) \in \mathbb{R}^{W^2 \cup W^3} \mid Sy + Cz = b - Ad_{W^1} - Ud_T,\ Ty = d_T, \ell_{W^2 \cup W^3} \leq \begin{bmatrix} y & z \end{bmatrix}^\intercal \leq u_{W^2 \cup W^3}\}\\
\coloneqq &H_d^x \times H_d^\omega \times H_d^{y,z}
\end{aligned}$$

In the third step, we use that H_d can be rewritten as the Cartesian product of the three polyhedra $H_d^x \subseteq \mathbb{R}^{W^1}$, $H_d^{y,z} \subseteq \mathbb{R}^{W^2 \cup W^3}$ and $H_d^\omega \subseteq \mathbb{R}^{W^4}$. Then since $\begin{bmatrix} S & C \\ T & \mathbb{O} \end{bmatrix}$ is totally unimodular and $\begin{bmatrix} b - Ad_{W^1} - Ud_T \\ d_T \end{bmatrix}$ is integral by integrality of b, U, A and d, and

$\ell_{W^2 \cup W^3}$ and $u_{W^2 \cup W^3}$ are integral, $H_d^{y,z}$ is an integral polyhedron by Hoffman and Kruskal's theorem [43]. Thus, we have $H_d^{y,z} = \text{conv}(H_d^{y,z} \cap \mathbb{Z}^{W^2 \cup W^3}) = \text{conv}(H_d^{y,z} \cap (\mathbb{Z}^{W^2} \times \mathbb{R}^{W^3}))$, where the last equality follows since $H_d^{y,z} \cap \mathbb{Z}^{W^2 \cup W^3} \subseteq H_d^{y,z} \cap (\mathbb{Z}^{W^2} \times \mathbb{R}^{W^3}) \subseteq H_d^{y,z}$ holds. This implies that $H_d = \text{conv}(H_d \cap \mathbb{MI}^{W_I}) = \text{conv}(H_d \cap \mathbb{MI}^{W_I \cup W^3})$ holds.

Then, note that C is totally unimodular since it is a submatrix of $\begin{bmatrix} S & C \\ T & \mathbb{O} \end{bmatrix}$. Since A, B and b are integral and the variable bounds ℓ_{W^3} and u_{W^3} of z are integral we have by that the equation $\text{conv}(P \cap \mathbb{MI}^{W_I}) = \text{conv}(P \cap \mathbb{MI}^{W_I \cup W^3})$ holds by [29, Theorem 3]. Finally, using $\widehat{T} = \begin{bmatrix} I_{W^1} & \mathbb{O} & \mathbb{O} \\ \mathbb{O} & T & \mathbb{O} \\ \mathbb{O} & \mathbb{O} & I_{W_I^3} \end{bmatrix}$ as the matrix in Lemma 31 and the fact that $H_d = \text{conv}(H_d \cap \mathbb{MI}^{W_I})$ holds, we derive that $\text{conv}(P \cap \mathbb{MI}^{W_I})) = \text{conv}(\{(x, y, z, \omega) \in P \mid x \in \mathbb{Z}^{W^1},\ Ty \in \mathbb{Z}^k,\ \omega \in \mathbb{MI}^{W_I^4}\})$ holds. □

There are a few key differences between Proposition 30 and Theorem 33 that make the latter more convenient to use. First of all, we do not require integrality of all of the right-hand sides, and we only need to construct an affine TU decomposition of a block in the constraint matrix rather than the complete matrix. Additionally, the $E = RT$ condition in Theorem 33 is new. This condition broadens the scope of the theorem, as otherwise, $D = \mathbb{O}$ would have to hold to guarantee implied integrality in the affine fiber H_d in the proof. Furthermore, during the detection we infer implied integrality for the continuous variables and have $\text{conv}(P \cap \mathbb{MI}^{W_I}) = \text{conv}(P \cap \mathbb{MI}^{W_I \cup W^3})$.

Theorem 33 can be used directly to show that the condition (13) in Theorem 26 holds. For integer variables $W_I \subseteq W$ and an equitable partition $(\mathcal{P}, \mathcal{Q})$ such that either $Q \cap W_I = \emptyset$ or $Q \subseteq W_I$ holds, let $\mathcal{Q}^2 = \{Q \in \mathcal{Q}_I \mid |Q| > 1\}$ and let $W^2 = W_B \cup \bigcup_{Q \in \mathcal{Q}^2} Q$. Then, if the conditions of Theorem 33 hold for $T = (\Pi_{\mathcal{Q}}^{\intercal} \Lambda)_{\mathcal{Q}', \star}$ for some subset $\mathcal{Q}' \subseteq \mathcal{Q}^2$, this directly shows condition (13) in Theorem 26 by using Theorem 32, and Theorem 32 shows that we can recover integral solutions for the original model.

4.2 Example: Generalized Assignment problem

Consider the generalized assignment problem, where one computes an assignment of J items with sizes $a_j,\ j \in J$ to I knapsacks with capacity C_i such that we maximize the sum of profits $\sum_{j \in J} c_j$ such that the capacities on the knapsacks are not exceeded. This can be modeled as an integer program using the binary variables $x_{ij} = \begin{cases} 1 & \text{if } j \text{ is packed into knapsack } i \\ 0 & \text{otherwise} \end{cases}$ for all $i \in I, j \in J$.

$$\max \quad \sum_{i \in I} \sum_{j \in J} c_j x_{ij} \tag{14a}$$

$$\text{s.t.} \quad \sum_{j \in J} a_j x_{ij} \leq C_i \qquad \forall i \in I \tag{14b}$$

$$\sum_{i \in I} x_{ij} \leq 1 \qquad \forall j \in J \tag{14c}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I,\ \forall j \in J \tag{14d}$$

Now, suppose that there exist identical items $j \in J$ that have the same weight and cost. Let $\mathcal{K} = \{K_1, \ldots, K_n\}$ be a partition of J into n item sets that have identical cost and weight. Clearly, $n \leq |J|$ must hold, and by definition we have that $a_K = a_j$ and $c_K = c_j$ for any $j \in K$, for any $K \in \mathcal{K}$. Let the variable partition $\mathcal{Q}$ be given by the parts $Q_{iK} = \{x_{ij} \mid \forall j \in K\}$ for $i \in I$, $K \in \mathcal{K}$, where identical items are placed into the same part. Let $\mathcal{P}$ be a constraint partition where all constraints (14c) and (14d) belonging to identical items are placed into the same partition $P \in \mathcal{P}$ and all other constraints are given their own partition. Then, it can be verified that $(\mathcal{P}, \mathcal{Q})$ is an equitable partition of the generalized assignment problem in (14). Applying the reformulation suggested by the equitable partition we obtain a smaller formulation:

$$\max \quad \sum_{i \in I} \sum_{K \in \mathcal{K}} c_K y_{iK} \tag{15a}$$

$$\text{s.t.} \quad \sum_{K \in \mathcal{K}} a_K y_{iK} \leq C_i \qquad \forall i \in I \tag{15b}$$

$$\sum_{i \in I} y_{iK} \leq |K| \qquad \forall K \in \mathcal{K} \tag{15c}$$

$$y_{ik} \in \{0, 1, \cdots, |K|\} \qquad \forall i \in I,\ \forall K \in \mathcal{K} \tag{15d}$$

Now, suppose that we are given any integral solution $y^\star$ to the reduced problem. For these values, we wish to recover a solution in the x-space. Then, we can add the constraints (16d), that link the x variables to $y^\star$, and hope to recover a solution for x.

$$\max \quad \sum_{i \in I} \sum_{j \in J} c_j x_{ij} \tag{16a}$$

$$\text{s.t.} \quad \sum_{j \in J} a_j x_{ij} \leq C_i \qquad \forall i \in I \tag{16b}$$

$$\sum_{i \in I} x_{ij} \leq 1 \qquad \forall j \in J \tag{16c}$$

$$\sum_{j \in K} x_{ij} = y^\star_{iK} \qquad \forall i \in I, \forall K \in \mathcal{K} \tag{16d}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I,\ \forall j \in J \tag{16e}$$

We can substitute the equations (16d) into the constraints (16b) and the objective (16a).

$$
\begin{aligned}
\max \quad & \sum_{i \in I} \sum_{K \in K} c_K y^\star_{iK} && && (17a)\\
\text{s.t.} \quad & \sum_{K \in \mathcal{K}} a_K y^\star_{iK} \leq C_i && \forall i \in I && (17b)\\
& \sum_{i \in I} x_{ij} \leq 1 && \forall j \in J && (17c)\\
& \sum_{j \in K} x_{ij} = y^\star_{iK} && \forall i \in I, \forall K \in \mathcal{K} && (17d)\\
& x_{ij} \in \{0,1\} && \forall i \in I,\ \forall j \in J && (17e)
\end{aligned}
$$

Note that in (17), both the objective and the knapsack constraints are not dependent on x anymore. In fact, by feasibility of $y^\star$, all constraints in (17b) are feasible by feasibility of (15b). Then, we observe that the remaining problem is given by an assignment problem on the x variables. Since the constraints (17c), (17d) form a totally unimodular matrix and the right-hand side given by 1 and $y^\star$ is integral, the LP relaxation of (17) is integral. Moreover, the solution $\bar{x}_{ij} = \frac{y^\star_{iK}}{|K|}$ for all $i \in I, K \in \mathcal{K}$ such that $j \in K$ is feasible for the LP relaxation of (17) by Proposition 3. Thus, an integral solution x can be obtained by finding a vertex solution of the LP relaxation of (17).

The reformulation of (16) into (17) can be seen as exhibiting an affine TU decomposition. The rows of the constraints (17d) correspond to the T-matrix, and their substitution in (17b) corresponds to the U-matrix. The rows of (17c) form the nonzeros of the S-matrix. Additionally, note that the usage of the equations to cancel the nonzeros in the objective is possible due to the fact that the objective vector is equitably partitioned by $\mathcal{Q}$.

There are a few interesting observations that can be made about the generalized assignment problem. First of all, as also implied by Salvagnin [9], the used equitable partition is not necessarily the coarsest equitable partition. Suppose for example, that for all knapsacks $i \in I$ the capacity C_i is equal to some constant C. Then, the knapsack constraints (14c) have additional permutation symmetries which permute the items between the bins. These symmetries are also captured by the coarsest equitable partition since equitable partitions generalize permutation symmetries. Consequently, for the coarsest equitable partition $(\mathcal{P}, \mathcal{Q})$, $\mathcal{Q}$ contains variables x_{ij} belonging to multiple bins in I. In the case where one would reformulate (14) using orbital shrinking, the orbital shrinking subproblem would then become the NP-hard binpacking problem, in contrast to the polynomial-time solvable assignment problem that we derive here. Salvagnin shows in [9] that for the generalized assignment model, employing a constraint programming solver to solve the binpacking subproblem and additionally reformulating the symmetric knapsacks yields better performance than the reformulation with the assignment problem that we derive here.

Secondly, we note that there is a minor way to improve the postsolve subproblem in (17). Let $\mathcal{K}^1 := \{K \in \mathcal{K} \mid a_K = 1\}$ and $J^1 := \{j \in J \mid a_j = 1\}$ denote the set of items with weight 1. For these items, it is not necessary to fix them to their corresponding $y^\star_{iK}$ value in (17). In particular, if we change the equalities in (17d) to only be added for all $\mathcal{K} \setminus \mathcal{K}^1$, and substitute only all items with non-unit weight in (17b), we obtain the following reformulation.

$$\max \quad \sum_{i \in I} \sum_{J \in \mathcal{J}^1} c_j x_{ij} + \sum_{i \in I} \sum_{K \in \mathcal{K} \setminus \mathcal{K}^1} c_K y^\star_{iK} \tag{18a}$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{J}^1} x_{ij} \leq C_i - \sum_{K \in \mathcal{K} \setminus \mathcal{K}^1} a_K y^\star_{iK} \qquad \forall i \in I \tag{18b}$$

$$\sum_{i \in I} x_{ij} \leq 1 \qquad \forall j \in J \tag{18c}$$

$$\sum_{j \in K} x_{ij} = y^\star_{iK} \qquad \forall i \in I, \forall K \in \mathcal{K} \setminus \mathcal{K}^1 \tag{18d}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I,\ \forall j \in J \tag{18e}$$

It is not difficult to see that (18) is still a perfect formulation. The constraint matrix defined by (18b)–(18d) is totally unimodular as it is the node-edge incidence matrix of a bipartite graph and the right-hand sides are integral. The two parts of the bipartition are given by constraints (18c) and ((18b) plus (18d)), respectively. The fixings for $\mathcal{K}^1$ can be left out due to the fact that the variables in J^1 are implied integer. Let P be the linear relaxation of (14). Then total unimodularity of (18) shows by affine TU decomposition that $\operatorname{conv}(P \cap \mathbb{Z}^n) = \operatorname{conv}(\{x \in P \mid \sum_{j \in K} x_{ij} \in \mathbb{Z},\ \forall i \in I,\ \forall K \in \mathcal{K} \setminus \mathcal{K}^1\})$ holds. From this equality it also follows that y_{iK} is implied integer for all $i \in I$ and all $K \in \mathcal{K}^1$ by the other variables in (15). One way in which this additional power from implied integrality manifests is that in contrast to (17), solving (18) may give a stronger solution than the provided $y^\star$ solution, if $y^\star$ is suboptimal and one of the items $j \in J^1$ can still be assigned to some knapsack. This correspond to having the proper subset $\mathcal{Q}' \subset \mathcal{Q}_I$ in the statement of Theorem 32.

5 Detecting DRCR for mixed-integer linear programs

Next, let us formulate an algorithm to detect the reductions described in Theorem 26. To achieve such a reduction, we need to balance two conditions: we would like the partition $\mathcal{Q}$ to be as coarse as possible, but (13) must also hold. These two objectives naturally conflict with each other, as a coarse partition of the integer variables requires us to find an affine TU decomposition with fewer integrality constraints. Although one could consider the coarsest possible partition and simply try to detect an affine TU decomposition and apply Theorem 33 and Theorem 32, it is unlikely that (13) holds, either by the structure of the problem or by additional symmetries. For the example of the generalized assignment problem in Section 4.2, if two bins have identical capacities then the symmetries that permute the two bins imply that

(13) is not satisfied for the coarsest equitable partition.

In order to detect an affine TU decomposition that we can use, we attempt to detect the structure of Theorem 33 where T is given by $(\Pi_{\mathcal{Q}}^{\intercal} \Lambda x_{W_U})_{\mathcal{Q}',\star}$, where $\mathcal{Q}' \subseteq \{Q \in \mathcal{Q}_I \mid |Q| > 1\}$. Note that the columns of T consist of unit vectors, which may have been negated. For $w \in W^2$ we have for its column in T that $T_{\star,w} = \lambda_w \mathbb{e}_Q$ holds for $Q \in \mathcal{Q}'$ such that $w \in Q$, where $\mathbb{e}_Q$ is a unit vector. Then it follows that $B_{\star,w} = (S + UT)_{\star,w} = S_{\star,w} + UT_{\star,w} = S_{\star,w} + \lambda_w U_{\star,Q}$ holds. Since we want S to be part of the totally unimodular matrix, its entries are ± 1 values. Thus, if a column $B_{\star,w}$ contains large entries these must come from the $U_{\star,Q}$ vector. Furthermore, note that for all columns in the block Q, its entries must differ by at most 1 from $\lambda_w U_{\star,Q}$. Thus, columns in $B_{\star,Q}$ must differ by at most 1 up to ± 1 scaling. For example, if Q contains some block $(P, Q) \in \mathcal{P} \times \mathcal{Q}$ such that $B_{P,Q} = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}$, then it is impossible to construct U and S such that we can get $B = S + UT$, since $B_{P,w_1} - B_{P,w_2} \notin \{-1, 0, 1\}^P$ implies that there exists no choice of $U_{\star,Q}$ such that we can construct a totally unimodular matrix S.

For a matrix $A \in \mathbb{R}^{V \times W}$, we define the *nonternary mask* $\operatorname{ntmask}(A) \in \mathbb{R}^{V \times W}$ such that $\operatorname{ntmask}(A)_{v,w} \coloneqq \begin{cases} 0 & \text{if } |A_{v,w}| = 1 \\ A_{v,w} & \text{otherwise} \end{cases}$ holds for all $v \in V$ and $w \in W$. Then, if two columns of B have the same nonternary mask up to ± 1 scaling, they are candidates to belong to the same part $Q \in \mathcal{Q}$. In order to generate equitable partitions which satisfy this condition, we change the initial partition that we pass to the color refinement algorithm to only place two integer columns w_1 and w_2 in the same initial part $Q \in \mathcal{Q}_0$ if $\operatorname{ntmask}(A_{\star,w_1}) = \pm \operatorname{ntmask}(A_{\star,w_2})$ holds. Using standard hashing techniques, this can be done in linear time, see [28] for further details. The condition that the nonternary masks are equivalent requirement is a sufficient condition to ensure that S is a ternary matrix, but not a necessary one. For example, we have $\begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 2 & 2 \end{bmatrix}^{\intercal} \begin{bmatrix} 1 & 1 \end{bmatrix}$ where the $\begin{bmatrix} S & T \end{bmatrix}^{\intercal}$ matrix is given by the totally unimodular matrix $\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}^{\intercal}$ but the two nonternary masks of the original columns differ.

Then, consider the second condition that $E = RT$ holds. For $v \in V$ and $w \in W$ we have $E_{v,w} = (R\Pi_{\mathcal{Q}}^{\intercal} \lambda)_{v,w} = R_{v,Q} \lambda_w$ for Q such that $w \in Q$, which only holds if the columns in $E_{\star,Q}$ are identical up to ± 1 scaling. Thus, we require that $E_{\star,w} = \pm E_{\star,w'}$ holds for all $w, w' \in Q$.

Although totally unimodular matrices can be detected in polynomial time [49], current implementations still have quintic time complexity [50], which is too slow for practical purposes. Instead, we focus on *network matrices* and their transposes, which form large subclasses of totally unimodular matrices [33]. We use the fast column augmentation algorithms by Bixby and Wagner for network matrices [31] and the

algorithm by van der Hulst and Walter for transposed network matrices [32]. For a (transposed) network matrix $M \in \mathbb{R}^{V \times W}$ and a column $u \in \mathbb{R}^V$, both algorithms are given by a procedure AugmentNetwork(M, u) that returns whether the augmented matrix $\begin{bmatrix} M & u \end{bmatrix}$ is a (transposed) network matrix, growing the (transposed) network matrix by one column. The operation of *augmenting* a column to a matrix M turns M into $\begin{bmatrix} M & u \end{bmatrix}$.

Then, we are ready to describe an algorithm that detects the conditions of Theorem 26 by simultaneously computing a reflection reduction and an affine TU decomposition of the MILP. The algorithm is described in Algorithm 1 and consists of roughly five steps.

In the first step, we determine the row partition that is used for Theorem 33, by partitioning the rows according to whether they contain non-integral entries or right-hand side. Then, we determine for each of the the connected components of the submatrix formed by the continuous columns that is in the integral part whether it is (transposed) network. These components form the C-matrix in Theorem 33. This step is identical to part of the algorithm described in [29] to detect implied integrality using totally unimodular submatrices.

In the second step, we use the computed row partition for Theorem 33 to compute an initial partition that we pass to the color refinement algorithm. As discussed above, we put two columns in the same partition only if they are (up to ± 1 scaling) identical in the non-integral rows and if their nonternary masks are identical (up to ± 1 scaling) in the integral rows. In the third step, we compute a reflection reduction given this initial partition.

For the fourth step, we consider the computed reflection reduction and attempt to construct a matching affine TU decomposition using Theorem 33 that extends the computed reduction to integer variables. For every $Q \in \mathcal{Q}'$, where $\mathcal{Q}' \coloneqq \{Q \in \mathcal{Q}_I \mid |Q| > 1\}$, we consider adding a column u to the U matrix of the affine TU decomposition. For our approach in Algorithm 1, we simply check if $A_{v,Q} = \alpha \lambda_Q$ holds for some $\alpha \in \mathbb{R} \setminus \{0\}$. If so, then we set $u_v = \alpha_v$. Then, we use $S_{\star,Q} = A_{\star,Q} - u(\Pi_{\mathcal{Q}}^{\intercal} \Lambda)_{\star,Q}$ This effectively cancels the v'th row of $A_{\star,w}$ by using u. If $u = \mathbb{0}$, then we do not need the additional row of $\Pi_{\mathcal{Q}} \Lambda$ to cancel any rows, and the additional row in the T part of the affine TU decomposition can only harm detection of total unimodularity, as any submatrix of a TU matrix is TU. In this case, we consider the partition of the subset $\mathcal{Q}' \setminus \{Q\}$ rather than $\mathcal{Q}'$, which is sufficient by Theorem 32, and we do not add the corresponding row to the T matrix. Finally, we ensure that for the affine TU decomposition $E = RT$ holds by checking if $S_{V_R,Q} = \mathbb{0}$ holds, i.e. that all rows in V_R are properly canceled.

Algorithm 1: Detecting DRCR for MILP using affine TU decompositions.

Algorithm: DRCR(A, b, ℓ, u, c, W_I)

Input: Constraint matrix $A \in \mathbb{R}^{V \times W}$, right-hand side $b \in \mathbb{R}^V$, lower bounds $\ell \in \mathbb{R}^W$, upper bounds $u \in \mathbb{R}^W$, objective $c \in \mathbb{R}^W$ and integer variables $W_I \subseteq W$

Output: A reflection reduction $(\mathcal{P}, \mathcal{Q}, \lambda, \gamma, \delta, V_U, W_U)$ such that the conditions of Theorem 26 hold.

$V_R \leftarrow \{v \in V \mid b_v \notin \mathbb{Z} \text{ or } a_{v,w} \notin \mathbb{Z} \text{ for some } w \in W\}$;
Let $\mathcal{K}$ be the set of connected components of $A_{V,W \setminus W_I}$;
Let M denote the (transposed) network matrix that is constructed;
for $K \in \mathcal{K}$ **do**
  if $V_R \cap V_K \neq \emptyset$ **then**
    $V_R \leftarrow V_R \cup V_K$;
    continue
  end
  isNetwork $\leftarrow True$;
  for $w \in W_K$ **do**
    if *not* AugmentNetwork$(M, A_{\star,w})$ **then**
      isNetwork $\leftarrow False$;
      break
    else Augment M with $A_{\star,w}$;
  end
  if *not* isNetwork **then**
    $V_R \leftarrow V_R \cup V_K$;
    Remove from M any columns of $A_{\star,W_K}$ that were augmented;
  end
end
Let $\mathcal{Q}_0$ be a partition of W such that $Q \cap W_I = \emptyset$ or $Q \subseteq W_I$ holds for all $Q \in \mathcal{Q}_0$. Additionally, we impose the condition that for all $Q \subseteq W_I$, $w, w' \in Q$ holds if and only if $A_{V_R,w} = \pm A_{V_R,w'}$ and $\text{ntmask}(A_{V \setminus V_R,w}) = \pm\text{ntmask}(A_{V \setminus V_R,w'})$ hold;
Let $\mathcal{P}_0 \leftarrow \{V\}$;
$(\mathcal{P}^1, \mathcal{Q}^1, \lambda, \gamma, \delta, V_O^1, W_O^1) \leftarrow$ ReflectionRefinement$(\mathcal{P}_0, \mathcal{Q}_0, A, b, \ell, u)$;
Sort $\mathcal{Q}^1$ by descending size $|Q|$ of its parts $Q \in \mathcal{Q}^1$;
for $w \in W \setminus W_O^1$ **do**
  if *not* AugmentNetwork$(M, A_{\star,w})$ **then** $\mathcal{Q}^1 \leftarrow \mathcal{Q}^1 \cup \{w\}$;
  else Augment M with $A_{\star,w}$;
end
for $Q \in \mathcal{Q}^1$ *such that* $Q \subseteq W_I$ *and* $|Q| > 2$ **do**
  Let $V_{\text{cancel}} \coloneqq \{v \in V \mid A_{v,Q} = \alpha_v \lambda_Q \text{ for some } \alpha_v \in R \setminus \{0\}\}$ and let $u \in \mathbb{R}^V$ be such that $u_v = \alpha_v$ if $v \in V_{\text{cancel}}$ and $u_v = 0$ otherwise.;
  Let $S = A_{\star,Q} - u(\Pi_Q^\intercal \Lambda)_Q$ and let $M' \in \mathbb{R}^{(V \cup Q^1) \times Q}$ using $M' \leftarrow \begin{bmatrix} S \\ \mathbb{O}_{|\mathcal{Q}^1| \times Q} \end{bmatrix}$;
  if $S_{V_R,\star} \neq \mathbb{O}$ **then**
    Modify $\mathcal{Q}^1$ by splitting Q into $|Q|$ parts that contain its individual elements;
    continue
  end
  if $V_{\text{cancel}} \neq \emptyset$ **then** Update $M'_{\mathcal{Q}^1,Q} \leftarrow (\Pi_Q^\intercal \Lambda)_Q$;
  isNetwork $\leftarrow True$;
  for $w \in Q$ **do**
    if *not* AugmentNetwork$(M, M'_{\star,w})$ **then**
      isNetwork $\leftarrow False$;
      break
    else Augment M with $M'_{\star,w}$;
  end
  if *not* isNetwork **then**
    Remove from M any column that was augmented from M';
    Modify $\mathcal{Q}^1$ by splitting Q into $|Q|$ parts that contain its individual elements;
  end
end
$(\mathcal{P}^2, \mathcal{Q}^2, \lambda, \gamma, \delta, V_O^2, W_O^2) \leftarrow$ ReflectionRefinement$(\mathcal{P}^1, \mathcal{Q}^1, A, b, \ell, u)$;
return $(\mathcal{P}^2, \mathcal{Q}^2, \lambda, \gamma, \delta, V_O^2, W_O^2)$;

In the fifth and final step, we consider all $Q \in \mathcal{Q}$ that could not be added to the affine TU decomposition. For each such a part Q, we *individualize* it by splitting Q into $|Q|$ parts where each element is in its own part of size 1, and run the refinement algorithm again. Note that this final refinement may refine 'good' parts that we computed the affine TU decomposition for previously into several parts. This is unfortunate, as it may adjust the requirements for the T matrix, and break the affine TU decomposition. An example of this phenomenon is shown in Example 34. However, we show in Lemma 35 that, because the new partition $\mathcal{Q}$ is a refinement of the previous partition, (13) is still satisfied and we can still recover solutions in polynomial time.

Example 34 We consider a run of Algorithm 1 on the mixed-integer linear program $\operatorname{conv}(\{x \in \mathbb{R}^6 \mid Ax = b, x \geq 0\} \cap \mathbb{Z}^6)$ where

$$A = \begin{bmatrix} 2 & 2 & 2 & 2 & 3 & 3 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \in \mathbb{Z}^{9\times 6} \text{ and } b = \begin{bmatrix} 8 \\ 2 \\ 2 \\ 2 \\ 2 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \in \mathbb{Z}^9.$$

We use v_i to indicate the i'th row index and w_j to indicate the j'th column index. The partition returned by ReflectionReduction is given by $\mathcal{P}_1 = \{\{v_1\}, \{v_2, v_3, v_4, v_5\}, \{v_6, v_7, v_8, v_9\}\}$ and $\mathcal{Q}_1 = \{\{w_1, w_2, w_3, w_4\}, \{w_5, w_6\}\}$. Consider $Q := \{w_1, w_2, w_3, w_4\}$, and note that $A_{\star,Q}$ admits an affine TU decomposition using $A_{\star,Q} = \begin{bmatrix} \mathbb{O}_{1\times 4} \\ A_{V\setminus\{v_1\},Q} \end{bmatrix} + \begin{bmatrix} 2 \\ \mathbb{O}_{8\times 1} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}$, where it can be verified that the matrix $\begin{bmatrix} \mathbb{O}_{1\times 4} \\ A_{V\setminus\{v_1\},Q} \\ \mathbb{1}_{1\times,4} \end{bmatrix}$ is totally unimodular. Thus, initially, Algorithm 1 will reformulate the integrality constraints on Q. Then, Algorithm 1 attempts to reformulate the integrality constraints for $\{w_5, w_6\}$ and fails to do so. Thus, we refine $\{w_5, w_6\}$ into $\{w_5\}, \{w_6\}$, and run ReflectionReduction again, which results in the partition given by $\mathcal{P}_2 = \{\{v_1\}, \{v_2, v_3\}, \{v_4, v_5\}, \{v_6, v_7\}, \{v_8, v_9\}\}$ and $\mathcal{Q}_2 = \{\{w_1, w_2\}, \{w_3, w_4\}, \{w_5\}, \{w_6\}\}$. Note for $\mathcal{Q}_2$ that Q is now split up into the two parts $\{w_1, w_2\}$ and $\{w_3, w_4\}$. However, this also affects the affine TU decomposition that we previously computed, as the matrix $\Pi^{\intercal}_{Q}$ that represents the T matrix in the affine TU decomposition is refined from $\Pi^{\intercal}_{\mathcal{Q}_1}$ to $\Pi^{\intercal}_{\mathcal{Q}_2}$, where the nonzero submatrix on Q changes from $\mathbb{1}_{1\times 4}$ to $\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$. Unfortunately, the matrix $\begin{bmatrix} 0 & 0 & 0 & 0 \\ A_{V\setminus\{v_1\},Q} & & & \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$ is not totally unimodular, which invalidates the previously computed affine TU decomposition. Thus, it may be the case that after a run of Algorithm 1 that the affine TU decomposition that is computed no longer matches the equitable partition, due to the additional refinements that are computed in line 49 of Algorithm 1.

Lemma 35 *Let $(\mathcal{P}^1, \mathcal{Q}^1, \gamma^1, \lambda^1, \delta, V_O^1, W_O^1)$ be a reflection reduction of $G \coloneqq P(A, b, \ell, u)$ and let it be given that for some $\widehat{\mathcal{Q}^1} \subseteq \mathcal{Q}_I^1$ that $H_d \coloneqq \{x \in G \mid (\Pi_{\mathcal{Q}^1}^\intercal \Lambda^1 x_{W_O^1})_{\widehat{\mathcal{Q}^1}} = d\}$ is a W_I-integral polyhedron for all $d \in \mathbb{Z}^{\widehat{\mathcal{Q}^1}}$. Consider any refinement $(\mathcal{P}^2, \mathcal{Q}^2, \lambda^2, \gamma^2, \delta, V_O^2, W_O^2)$ of the given reflection reduction where $\widehat{\mathcal{Q}}^2 \subseteq \mathcal{Q}^2$ corresponds to the parts that were refined from $\widehat{\mathcal{Q}}^1$ such that $\bigcup_{Q \in \widehat{\mathcal{Q}}^2} Q = \bigcup_{Q \in \widehat{\mathcal{Q}}^1} Q$ holds. Then, (13) holds for the refined reduction, i.e. we have:*

$$\operatorname{conv}(G \cap \mathbb{MI}^{W_I}) = \operatorname{conv}(\{x \in G \mid (\Pi_{\mathcal{Q}^2}^\intercal \Lambda^2 x_{W_O^2})_{\mathcal{Q}_I^2} \in \mathbb{Z}^{\mathcal{Q}_I^2}\}) = \operatorname{conv}(\{x \in G \mid (\Pi_{\mathcal{Q}^2}^\intercal \Lambda^2 x_{W_O^2})_{\widehat{\mathcal{Q}^2}} \in \mathbb{Z}^{\widehat{\mathcal{Q}^2}}\}),$$

Proof First, let us show a few facts about the refined reduction $(\mathcal{P}^2, \mathcal{Q}^2, \lambda, \gamma, \delta, V_O^2, W_O^2)$. First of all, note that $\mathcal{P}^2$ and $\mathcal{Q}^2$ must refine $\mathcal{P}^1$ and $\mathcal{Q}^1$ respectively, and we must have $W_O^1 \subseteq W_O^2$ since additional refinements can only turn bipolar variables into unipolar variables, but not the other way around. Furthermore, note that additional refinements cannot change the value of λ_w^1 for $w \in W_O^1$, since the relative signs of unipolar parts are fixed. Thus, $\lambda^1 = \lambda^2_{W_O^1}$ holds. By design of our algorithm, δ is constant in both reductions. Then, we claim that the following holds:

$$G \cap \mathbb{MI}^{W_I} \subseteq \{x \in G \mid (\Pi_{\mathcal{Q}^2}^\intercal \Lambda^2 x_{W_O^2})_{\widehat{\mathcal{Q}^2}} \in \mathbb{Z}^{\widehat{\mathcal{Q}^2}}\} \subseteq \{x \in G \mid (\Pi_{\mathcal{Q}^1}^\intercal \Lambda^1 x_{W_O^1})_{\widehat{\mathcal{Q}^1}} \in \mathbb{Z}^{\widehat{\mathcal{Q}^1}}\}.$$

Here, the first inclusion follows since $\Pi_{\mathcal{Q}}^\intercal \Lambda^2$ is an integral matrix. For the second inclusion, consider any $x \in \{x \in G \mid (\Pi_{\mathcal{Q}^2}^\intercal \Lambda^2 x_{W_O^2})_{\widehat{\mathcal{Q}^2}} \in \mathbb{Z}^{\widehat{\mathcal{Q}^2}}\}$. Then, consider any $Q \in \widehat{\mathcal{Q}^1}$. Since $\mathcal{Q}^2$ refines $\mathcal{Q}^1$, there exist parts $Q_i' \in \widehat{\mathcal{Q}^2}$ for $i = 1, \cdots, k$ such that $\bigcup_{i=1}^k Q_i' = Q$ holds. Then, we can derive that:

$$(\Pi_{\mathcal{Q}^1}^\intercal \Lambda^1 x_{W_O^1})_Q = \sum_{w \in Q} \lambda_w^1 x_w = \sum_{i=1}^k \sum_{w \in Q_i'} \lambda_w^1 x_w \overset{(a)}{=} \sum_{i=1}^k \sum_{w \in Q_i'} \lambda_w^2 x_w = \sum_{i=1}^k (\Pi_{\mathcal{Q}^2}^\intercal \Lambda^2 x_{W_O^2})_{Q_i'},$$

where (a) holds since $\lambda^1 = \lambda^2_{W_O^1}$ holds, and the final expression is integral by integrality of $(\Pi_{\mathcal{Q}^2}^\intercal \Lambda^2 x_{W_O^2})_{Q_i'}$, using that $Q_i' \in \widehat{Q^2}$.

Taking convex hulls, we note that by applying Lemma 31, using W_I-integrality of H_d, we have equality throughout:

$$\operatorname{conv}(G \cap \mathbb{MI}^{W_I}) = \operatorname{conv}(\{x \in G \mid (\Pi_{\mathcal{Q}^2}^\intercal \Lambda^2 x_{W_O^2})_{\widehat{\mathcal{Q}^2}} \in \mathbb{Z}^{\widehat{\mathcal{Q}^2}}\}) = \operatorname{conv}(\{x \in G \mid (\Pi_{\mathcal{Q}^1}^\intercal \Lambda^1 x_{W_O^1})_{\widehat{\mathcal{Q}^1}} \in \mathbb{Z}^{\widehat{\mathcal{Q}^1}}\}).$$

Finally, we have the following inclusions:

$$G \cap \mathbb{MI}^{W_I} \subseteq \{x \in G \mid (\Pi_{\mathcal{Q}^2}^\intercal \Lambda^2 x_{W_O^1})_{\mathcal{Q}_I^2} \in \mathbb{Z}^{\mathcal{Q}_I^2}\} \subseteq \{x \in G \mid (\Pi_{\mathcal{Q}^2}^\intercal \Lambda^2 x_{W_O^2})_{\widehat{\mathcal{Q}^2}} \in \mathbb{Z}^{\widehat{\mathcal{Q}^2}}\},$$

taking convex hulls and noting that the first and last sets are equal, we again have equality throughout, which concludes the proof

$$\operatorname{conv}(G \cap \mathbb{MI}^{W_I}) = \operatorname{conv}(\{x \in G \mid (\Pi_{\mathcal{Q}^2}^\intercal \Lambda^2 x_{W_O^1})_{\mathcal{Q}_I^2} \in \mathbb{Z}^{\mathcal{Q}_I^2}\}) = \operatorname{conv}(\{x \in G \mid (\Pi_{\mathcal{Q}^2}^\intercal \Lambda^2 x_{W_O^2})_{\widehat{\mathcal{Q}^2}} \in \mathbb{Z}^{\widehat{\mathcal{Q}^2}}\}),$$

□

It is not too difficult to see that finding a solution to the original problem can also still be done in polynomial time by simply considering the fiber H_d defined where the affine subspaces are defined by $\Pi_{\mathcal{Q}}^1 \Lambda^1 x_{W_O^1} = d$. In Example 34, this corresponds to adding the equality $x_1 + x_2 + x_3 + x_4 = y_{1,2} + y_{3,4}$ in the postsolve procedure

rather than the two equalities $x_1 + x_2 = y_{1,2}$ and $x_3 + x_4 = y_{3,4}$ which are shown to be problematic in Example 34. Although the latter two equalities clearly imply the former equality, adding them to the linear program breaks the integrality guarantee.

Next, let us consider the total runtime. As for linear programs, we assume that the reflection reduction uses a δ-vector whose bitlength is of the same order in the worst-case, which is reasonable as the choice for δ that is described in Corollary 21 satisfies this assumption.

Theorem 36 *For a mixed-integer linear program $F(A, b, \ell, u, c)$ with $A \in \mathbb{R}^{V \times W}$ and integer variables $W_I \subseteq W$, let m be the total bitlength of the entries of $\begin{bmatrix} \ell & 0 \\ u & 0 \\ c & 0 \\ A & b \end{bmatrix}$ and let $n = |V| + |W|$. Assuming that* ReflectionRefinement *computes a δ with total bitlength $\mathcal{O}(m)$, Algorithm 1 computes a reflection reduction $(\mathcal{P}, \mathcal{Q}, \gamma, \lambda, \delta, V_U, W_U)$ of $F(A, b, \ell, u, c)$ such that (13) holds, i.e.:*

$$\operatorname{conv}(\{x \in P(A, b, \ell, u) \mid (\Pi_{\mathcal{Q}}^{\intercal} \Lambda x)_{\mathcal{Q}_I} \in \mathbb{Z}^{\mathcal{Q}_I}\}) = \operatorname{conv}(P(A, b, \ell, u) \cap \mathbb{MI}^{W_I})$$

in $\mathcal{O}(n + m)$ space and $\mathcal{O}((n + m) \log n)$ time using network matrices or $\mathcal{O}(n^2 \alpha(n, n) + m \log(n))$ time using transposed network matrices.

Proof We do not provide a complete proof of every step here, but instead emphasize the most relevant parts of the correctness proof and the running time. First let us prove correctness. In lines 1-20, we compute the connected components of the continuous variables that are (transposed) network, which is necessary to compute our affine TU decomposition. Then, we compute an initial reflection reduction in lines 21-23, where $\mathcal{Q}_1$ satisfies the following for all $Q \in \mathcal{Q}_1$:

1. Either $Q \cap W_I$ or $Q \subseteq W_I$ holds.
2. If $Q \subseteq W_I$, then $A_{V_R, w} = \pm A_{V_R, w'}$ and $\operatorname{ntmask}(A_{V \setminus V_R, w}) = \pm \operatorname{ntmask}(A_{V \setminus V_R, w'})$ hold for all $w, w' \in Q$.

In lines 24-48, we greedily construct an affine TU decomposition that satisfies the conditions of Theorem 33 where we keep the invariant that T is equal to $(\Pi_{\mathcal{Q}^1}^{\intercal} \Lambda)_Q$ for some $\mathcal{Q}' \subseteq \mathcal{Q}^1$. The construction uses condition 2. by testing if the vector $\begin{bmatrix} A_{V_R, w} \\ \operatorname{ntmask}(A_{V \setminus V_R, w}) \end{bmatrix}$, which is identical up to ± 1 scale for all $w \in Q$ by construction, can be used to augment the (transposed) network matrix in the affine TU decomposition. Our construction does not always add such a column to u: we determine dynamically if this is necessary by checking if $V_{\text{cancel}} \neq \emptyset$ holds. If an additional column in the U-matrix is needed, we add Q to $\mathcal{Q}'$ and add the row $(\Pi_{\mathcal{Q}^1}^{\intercal} \Lambda)_Q$ to T. Otherwise, we leave $\mathcal{Q}'$ and T unchanged. By doing so for all columns, we ensure that $T = \Pi_{\mathcal{Q}'}^{\intercal} \Lambda$ holds. For any $Q \in \mathcal{Q}^1$ that can not added to the affine TU decomposition, we simply split it into single-variable parts in $\mathcal{Q}^1$ to satisfy the conditions of the affine TU decomposition. Then, it follows from Theorem 33 and Theorem 32 that for $\mathcal{Q}_I^1 = \{Q \in \mathcal{Q}^1 \mid Q \subseteq W_I\}$ we have:

$$\operatorname{conv}(\{x \in P(A, b, \ell, u) \mid (\Pi_{\mathcal{Q}^1}^{\intercal} \Lambda x)_{\mathcal{Q}_I^1} \in \mathbb{Z}^{\mathcal{Q}_I^1}\}) = \operatorname{conv}(P(A, b, \ell, u) \cap \mathbb{MI}^{W_I}). \tag{19}$$

Finally, we refine $\mathcal{Q}^1$ to $\mathcal{Q}^2$ in 49 to ensure that $\mathcal{Q}^2$ corresponds to a reflection reduction. By (19) and the fact that $\mathcal{Q}^2$ refines $\mathcal{Q}^1$, Lemma 35 shows for $\mathcal{Q}^2_I \coloneqq \{Q \in \mathcal{Q}^2 \mid Q \subseteq W_I\}$ that

$$\operatorname{conv}(\{x \in P(A,b,\ell,u) \mid (\Pi^{\intercal}_{\mathcal{Q}^2}\Lambda x)_{\mathcal{Q}^1_I} \in \mathbb{Z}^{\mathcal{Q}^2_I}\}) = \operatorname{conv}(P(A,b,\ell,u) \cap \mathbb{MI}^{W_I}).$$

Then, since $(\mathcal{P}^2, \mathcal{Q}^2, \lambda, \gamma, \delta, V^2_O, W^2_O)$ is a reflection reduction of $F(A,b,\ell,u,c)$, it follows that the conditions of Theorem 26 hold. Thus, we have shown correctness of Algorithm 1.

Next, let us consider the running time and space-complexity of Algorithm 1. First of all, ReflectionRefinement runs in $\mathcal{O}((n+m)\log(n))$ and $\mathcal{O}(n+m)$ space by Theorem 25 and is called twice. Note that we use the fact that δ has bitlength at most $\mathcal{O}(m)$.

At the start of the Algorithm 1, we compute the connected components of the submatrix formed by the continuous columns. This can be done in $\mathcal{O}(n)$ time and space by using a depth-first search.

Computing the initial partition $\mathcal{Q}_0$ can be done in $\mathcal{O}(n+m)$ time and space by using standard hashing techniques. Sorting $\mathcal{Q}^1$ can be done in $\mathcal{O}(n\log(n))$ time.

Finally, we note that AugmentNetwork is called at most once for every column. The matrix M has size $(|V|+|\mathcal{Q}|)\times|W|$, which is of the order $\mathcal{O}(n)\times\mathcal{O}(n)$. Moreover, M has $\mathcal{O}(m)$ nonzeros as it has fewer nonzeros than A, since we only add the additional row of $(\Pi^{\intercal}_{\mathcal{Q}}\Lambda)_Q$ for all $Q \in \mathcal{Q}$ if it can be used to cancel one or more rows from $A_{\star,Q}$. We note that regardless of the size of the nonzero entries, one can check in constant time if a nonzero has value ± 1. For network matrices the column augmentation algorithm by Bixby and Wagner then runs in $\mathcal{O}(m\alpha(m,n))$ time, where α is the inverse Ackermann function, which grows extremely slowly and is dominated by the order $\mathcal{O}((n+m)\log(n))$ running time for the reflection reduction. The transposed network algorithm by van der Hulst and Walter runs in $\mathcal{O}(n^2\alpha(n,n))$ time, and dominates the total runtime in case it is used. Taking the bitlength m into account, we can simplify to $\mathcal{O}(n^2\alpha(n,n) + m\log(n))$ in this case. □

6 Computational Results

In order to validate the developed algorithms, we perform experiments on linear programming and mixed-integer linear programming problems.

Algorithm 1 and the reflection reduction detection algorithm are implemented in C++ and are publicly available [51]. The implementation of Algorithm 1 uses the network matrix augmentation algorithms described in [32] and [31] to recognize totally unimodular matrices, which are also publicly available [52]. All experiments are run on a AMD Ryzen 2600 CPU with 16GB of memory. Prior to their solution, the linear programming and mixed-integer linear programming instances are presolved using PaPILO 3.0.0 [53]. In the context of our study, presolving the instances with PaPILO has two benefits. First of all, well-known presolving reductions such as those described in [54] may provide additional opportunities for the DRCR algorithm, and provide a realistic setting for the application of DRCR. Secondly, PaPILO removes parallel rows and columns from the problem, which would also be detected by DRCR. This way, we can be sure that the reductions detected by R-DRCR and DRCR are not detected easily by other presolving techniques.

For our experiments on mixed-integer linear programs we use the SCIP 10 solver [34]. We note that currently parallel column reductions are not available in SCIP 10 due to technical limitations of its API. Thus, presolving once with PaPILO before

handing the problem to SCIP is necessary to provide a fair comparison of DRCR for MILP with state-of-the-art presolving techniques.

6.1 Folding reflection symmetries for Linear Programs

As a baseline, we compare our dimension reduction algorithm against DRCR as described in [11]. The improved DRCR algorithm as presented Section 3 is denoted by R-DRCR.

We use the linear programming relaxations of problems in the MIPLIB 2017 collection dataset [1], which contains 1065 instances. We exclude the 10 largest instances due to memory requirements, and any instances with the (nonlinear) indicator constraints. Furthermore, we exclude instances where DRCR and R-DRCR produce an identical reduced instance. This yields a subset of 83 instances that we use for the computational experiments. The resulting linear programs are solved using SoPlex 8.0.0 [34] with a time limit of 1 hour. An overview of the results can be found in Table 1. The complete results can be found in Appendix A.

Table 1: Performance comparison between DRCR and R-DRCR on the 83 linear programming relaxations of instances in the MIPLIB2017 collection set where R-DRCR finds additional reductions compared to DRCR. The subsets contain the instances whose solution time for the DRCR method (in seconds) lies in the indicated bracket. The times reported are shifted geometric means in seconds, with a shift of 1 second. The iterations column indicates the number of simplex iterations, and is reported as a shifted geometric mean with a shift of 10 iterations. The faster and slower columns indicate the number of instances where R-DRCR is faster/slower than DRCR.

		DRCR		R-DRCR			
subset	instances	time	iterations	time	iterations	faster	slower
all	83	1.27	1866.0	0.93	962.0	40	43
<0.1	34	0.02	142.9	0.02	60.4	13	21
[0.1,1]	20	0.35	2907.9	0.33	2441.6	9	11
[1,10]	9	3.26	11942.0	3.00	7783.7	5	4
>10	20	174.02	37205.0	58.50	13094.6	13	7

The R-DRCR algorithm was also tested on the well-known Netlib linear programming dataset. However, for all of the instances in this dataset, DRCR and R-DRCR produced identical reduced instances. Thus, we omit the results on Netlib in this report.

From the results in Table 1, it is clear that R-DRCR is more effective than DRCR on the tested subset. In all categories, the running time is smaller and the number of required simplex iterations is also reduced. The running time improvement is particularly impressive for some of the more difficult linear programs that take longer

than 10 seconds to solve, with the shifted geometric mean of the running times dropping from 174.0 to 58.5 seconds. Furthermore, we observe that on many of the easier instances there is a moderate reduction in running time and a clear reduction in the number of simplex iterations.

Let us highlight a few instances where reflection refinement is particularly effective. For the combinatorial instances `pythago7824` and `pythago7825`, the linear program is even reduced to a linear program with a single variable and row. Interestingly, these instances do not contain any (reflection) symmetries. Another impressive result is on the large instance `neos-3402454-bohle` where the solution time is reduced from 1353 to 6 seconds, where in the latter case, the majority of the running time is spent computing the reduced problem. The original problem has over 2.8 million rows, whereas the reduced problem has roughly 80 thousand rows.

6.2 Folding symmetries in Mixed-Integer Linear Programming

Next, we evaluate the impact of Algorithm 1 on the solution of mixed-integer linear programs. As before, we consider the MIPLIB 2017 collection dataset [1]. We exclude the 10 largest instances and any instances with indicator constraints.

For our experiments, we consider two configurations of Algorithm 1. We use R-DRCR-N to indicate the version that detects network matrices, and R-DRCR-T to indicate the version that detects transposed network matrices. For both configurations, we only test those instances where some reduction is found by the algorithms. R-DRCR-N find reductions on 207 of the 1065 MIPLIB 2017 collection instances, and R-DRCR-T finds reductions on 153 instances. In total, 208 of the instances admit a reduction by at least one of R-DRCR-N and R-DRCR-T.

The solving procedure is structured as follows. First, the problem is presolved using PaPILO. Then, we apply the DRCR algorithm to further reduce the presolved problem. Then, the presolved problem is solved using SCIP 10.0 [34] with a time limit of 1 hour. Finally, if SCIP finds a feasible integer solution to the reduced problem that is not integer when mapped to the original variables, we use SoPlex to postsolve the found integer solution to an integer solution. All running times reported include the runtime of the DRCR algorithm, SCIP's time to solve the problem, and time required to postsolve the final solution to obtain a feasible integral solution to the original problem. We compare the R-DRCR methods against the default SCIP 10 configuration, which is run on the presolved model that is output by PaPILO. In all cases, we use the default parameters and configuration of SCIP 10. One important detail is that, although the DRCR algorithm may also detect implied integrality in the reduced problem, we do not communicate this information to SCIP in order to measure the effect of the reduction detected by Algorithm 1 in isolation.

In Tables 2 and 3 we summarize the results of R-DRCR-N and R-DRCR-T compared to the SCIP 10 baseline on instances which were solved by at least one of the methods. In Table 4 we compare the performance on the instances that were not solved within the time limit by any method. Here, we use the Primal-Dual Integral measure introduced by Berthold [55], which measures the average gap between the primal and dual solution during the solving process. This gives an indication for the quality of the obtained primal and dual bounds.

Table 2: Performance comparison between SCIP 10 and R-DRCR-N on the solvable instances of the MIPLIB2017 collection set that are affected by R-DRCR-N. The subsets contain the instances whose solution time for SCIP 10 (in seconds) lies in the indicated bracket. The time and nodes column indicate the fraction of the shifted geometric mean of the R-DRCR method compared to the SCIP 10 baseline, where we use shifts of 1 second and 1 node respectively. The faster and slower columns indicate the number of instances where R-DRCR method is faster/slower than SCIP 10.

		SCIP 10	R-DRCR-N				
subset	instances	solved	solved	time	nodes	faster	slower
all	107	100	103	0.48	0.69	67	40
<10	25	25	25	1.18	1.40	15	10
[10,100]	24	24	23	0.86	0.98	12	12
[100,1000]	35	35	34	0.61	0.70	21	14
>1000	23	16	21	0.075	0.25	19	4

Table 3: Performance comparison between SCIP 10 and R-DRCR-T on the solvable instances of the MIPLIB2017 collection set that that are affected by R-DRCR-T.

		SCIP 10	R-DRCR-T				
subset	instances	solved	solved	time	nodes	faster	slower
all	77	75	76	0.75	0.86	47	30
<10	19	19	19	1.08	0.95	12	7
[10,100]	14	14	14	0.94	1.09	8	6
[100,1000]	30	30	30	0.80	0.92	17	13
>1000	14	12	13	0.35	0.49	10	4

In Tables 2 and 3, we observe that R-DRCR-N dominates R-DRCR-T in both the number of affected instances and the speedups on the affected instances. Of the 208 instances where some reduction was found by either of the two methods, R-DRCR-T found a smaller reduced problem on only 8 instances. One explanation for this observation is that assignment subproblems such as those in the generalized assignment problem discussed Section 4.2 consist of matrices that are in general network matrices but not transposed network matrices. Otherwise, both methods have similar characteristics. Although they do not seem to help with the solution of 'easy' problems that can be solved under 10 seconds, we observe large speedups for the more difficult problems.

For R-DRCR-N, seven new instances are newly solved, and four instances are no longer solved compared to SCIP 10's default. For the seven instances that are newly solved, the R-DRCR-N algorithm reduces the size of the original system significantly.

For example, for the newly solved instance `neos-631710` the size of the constraint matrix is reduced from 169576×167056 to 9448×7216.

For the four instances that are no longer solved after applying R-DRCR-N, there are various potential explanations for the loss of performance. For the two instances `supportcase34` and `neos-5129192-manaia` a strong primal solution is no longer found early in the search after the reduction is applied to the problem. For `cvs08r139-94`, it seems that the default symmetry handling methods in SCIP are more effective than DRCR. One potential explanation for this is that the symmetries and the equitable partition in this instance are relatively simple: all the nontrivial variable parts consist of two binary variables that are aggregated into one integer variable.

Table 4: Comparison of SCIP 10 and R-DRCR-N and R-DRCR-T on unsolved instances of MIPLIB 2017 collection set that are affected by the R-DRCR-N or R-DRCR-T. The two different rows of the table correspond to the two different instance sets for which R-DRCR-N and R-DRCR-T found one or more reductions. The 'nodes' column indicates the fractional of the shifted geometric mean of R-DRCR method compared to the SCIP 10 baseline. The SCIP 10 PDI and method PDI indicate the Primal-Dual Integral values of the SCIP 10 baseline and the R-DRCR method, respectively.

Method	instances	nodes	SCIP 10 PDI	method PDI
R-DRCR-N	100	1.19	$1.30 \cdot 10^5$	$1.22 \cdot 10^5$
R-DRCR-T	76	1.25	$1.23 \cdot 10^5$	$1.14 \cdot 10^5$

In Table 4 we evaluate the performance of DRCR on instances that were not solved within the time limit by any method. We observe that both DRCR methods help to reduce the Primal-Dual Integral, indicating that the DRCR helps to find strong primal and dual bounds more quickly than SCIP 10. Furthermore, we note that the DRCR-methods explore more branch-and-bound nodes within the time limit. This observation can be explained by the fact that the linear programming relaxations are easier to solve due to their reduced size.

Table 5 provides an overview of the distribution of presolve and postsolve times. We note that for the vast majority of the instances, the presolve and postsolve procedures are cheap. For a small number of instances, the postsolve LP solution using SoPlex took longer than 10 seconds.

One effect that we observe in a few larger instances is that DRCR helps to reduce the running time of further symmetry detection methods. For example for the instance `neos-3322547-alsek`, symmetry detection for the original model with 1001000 columns takes 378.2 seconds. R-DRCR-N reduces the model to have only 82000 columns in 6.3 seconds. Symmetry detection on the reduced model completes in only 0.24 seconds.

Table 5: Distribution of the time taken (in seconds) to detect and postsolve for R-DRCR-N and R-DRCR-T for the affected instances. For each bracket, the number of instances that fall into it is indicated.

	DRCR detection			postsolve			
Method	<0.1	[0.1,1]	[1,10]	<0.1	[0.1,1]	[1,10]	>10
R-DRCR-N	146	46	15	134	37	28	8
R-DRCR-T	114	28	11	108	25	15	5

To summarize our results, we observe that both R-DRCR methods are effective in reducing the solution time of general mixed-integer linear programs. The R-DRCR-N algorithm dominates the R-DRCR-T algorithm. Furthermore, R-DRCR-N solves three more instances, and both methods yield a smaller PDI on the set of instances that are not solved within the time limit.

7 Discussion

We will follow the structure of the paper, and first evaluate the extension to reflection symmetries for linear programming and then the extension to mixed-integer linear programs.

In Section 3, we extended DRCR to reflection symmetries of linear programs. There are several ways in which the current work can still be extended. First of all, we note that it is possible to extend the detection of reflection symmetries to not just use a primal affine offset δ but to also introduce a *dual offset*. Such a dual offset alters the objective by subtracting a weighted linear combination of the rows from the objective. One hopes that doing so will lead to a coarser equitable partition by having more similar entries in the objective. Similar to the choice of δ, there are infinitely many possibilities to do so. However, unlike the primal variables, the dual variables of linear programming models are typically not explicitly bounded by the user, so there is no 'obvious choice' at the center of the dual variable domains like for the primal offset. Thus, one would need to come up with some method to choose a sensible dual offset.

The choice of δ is somewhat justified by Corollary 21, which shows that it corresponds to the complementation of the variables at their bounds. However, it is still unclear whether other choices of δ may lead to better results. This is true in particular for free variables and variables with half-open domains, where any offset can be used and may change the right-hand sides, and there is no obvious center of the variable domain.

More generally, we believe it would be interesting to understand in further detail how one should apply linear and affine transformations to linear programs in order to detect arbitrary symmetries. One direction which naturally extends this research

would be to detect arbitrary row and column scalings that result in coarser equitable partitions, compared to the ± 1 scalings that we considered here.

In Section 6.1, we performed computational experiments on the linear programming relaxations of MIPLIB 2017 instances. As expected, reflection symmetries are somewhat uncommon as was also observed by Hojny [2], as for only 81 out of 1065 MIPLIB instances R-DRCR finds more reductions than DRCR. For these instances, we observed on average a running-time reduction for R-DRCR of roughly 27% on the affected models compared to DRCR, with most of the benefit coming from a few large models. One limitation of the current study is that we did not include a crossover procedure to obtain a basic solution in our experiments, which is crucial in the context of solving linear programming relaxations in MILP. However, we note that since we compare R-DRCR against DRCR, a fair comparison with crossover would affect the running time of both models, so the performance implications in this case are unclear. As R-DRCR still clearly improves over DRCR without crossover it can be considered a relevant improvement for the dimension reduction of linear programs. In particular, reflection reductions can be computed in the same worst-case time complexity as DRCR by Theorem 25. Also in practice, they can be computed with only minor computational overhead using the implementation details that we describe in Section 3.7.

As the computational effort for crossover is the most important limitation for applying DRCR to linear programs in practice, one direction for future research is to investigate if it is possible to exploit symmetry to speed up the crossover procedure. Since (R)-DRCR factors out symmetries, the crossover problem contains these symmetries. It is likely that these symmetries hinder the performance of the crossover procedure. In an unpublished work, Deakins, Knueven and Ostrowski [56] formulate a method that takes advantage of equitable partitions using an iterative lifting scheme and show that their method can reduce the time spent in crossover. It would be promising to further develop symmetry-exploiting crossover methods and incorporate them with our approach. More generally, one could investigate if knowledge of symmetries of a linear program can be used directly to improve the performance of the simplex algorithm without modifying the analyzed (symmetric) polytope.

In Section 4, we extended DRCR to mixed-integer linear programs and highlight that affine TU decompositions may be used to aggregate integer variables. In Section 5, we formulate an algorithm that detects an equitable partition that satisfies the necessary conditions for dimension reduction of mixed-integer linear programs. The computational results in Section 6.2 indicate that the formulated algorithm is effective in reducing the running time of an MILP solver: affected instances are solved more than twice as quickly on average.

The algorithm we propose for MILPs may have several additional benefits. First of all, if we are able to perform dimension reduction, we may also eliminate some of the symmetries in the instance. Although we may only eliminate a subgroup of the symmetries for MILP, this reduction may help to speed up the detection of the

remaining symmetries. Second, even if the dimension of the MILP is not reduced, our algorithm still computes an affine TU decomposition and/or implied integrality that can be used by the MIP solver. Thus, the computational effort to do so does not go to waste, regardless of whether a nontrivial equitable partition is detected.

One downside of our algorithm is that it completely rejects any equitable integer block $Q \in \mathcal{Q}_I$ that can not be taken into the reduced problem, by partitioning it into its individual columns. Although this is a simple approach, it may also reject blocks that have many columns that contain a network substructure and only a few that break it. One may hope that instead of a rejecting the block completely, refining the block to compute an equitable partition and then attempting to add certain subsets of it might be more effective. However, this complicates the algorithm and requires a way to decide how equitable parts must refined in case they can not be augmented to the network matrix.

Our detection algorithm may benefit from detecting and exploiting additional substructures. For example, if the equitable partition $\mathcal{Q}$ contains some $Q \in \mathcal{Q}$ such that the constraint $\sum_{w \in Q} \lambda_w x_w = b$ appears in the constraint matrix, then the corresponding variable is fixed in the reduced problem and may be ignored for the detection of the affine TU decomposition, as it is an integral constant in the reduced problem. Additionally, it would be interesting to combine our reduction with the aggregation of non-overlapping symmetric variables from Section 7.1 in [27]. Another possibility that may enhance DRCR for mixed-integer programs is to take the viewpoint that it corresponds to simultaneously performing DRCR on all affine fibers H_d that result from fixing integral expressions, where the vector $d \in \mathbb{Z}^k$ indicates the subspace. Our current analysis only performs reduces the dimension if DRCR can be applied for all $d \in \mathbb{Z}^k$. It may be possible to use the structure of the polyhedron to argue for some affine fibers H_d that they are irrelevant for the MIP because they are infeasible or suboptimal, and use this insight to derive a stronger size reduction that exploits these properties.

For our computational experiments in Section 6.2, we remark that we did not spend any time or effort for the postsolve procedure by solving it as a linear program over the affine fibers, without utilizing the given non-basic LP-feasible point. Although this is not a bottleneck of our approach, a crossover procedure that uses the solution as an initial feasible point should be more efficient. If one uses affine TU decompositions to aggregate symmetric integer variables, then a postsolve step to recover integrality of the original solutions is necessary. We emphasize that in our experiments in Section 6.2 we perform crossover only once for the best integer solution that is found. Most MILP solvers do not report just the optimal solution, but also provide a solution pool that contains the best found primal solutions. For each such a solution, one needs to call a crossover procedure. Thus, the computational costs of the crossover step may become burdensome as it is repeated for every integer solution that is reported to the user. Thus, this justifies research into fast crossover procedures that can be used in conjunction with R-DRCR-N and R-DRCR-T.

Since we detect if the affine fibers are given by network matrices, the affine fibers can be transformed into a network flow problem, see [57] for further information. In crossover, one could also exploit this network structure by using crossover algorithms that are specialized for network flow problems [58]. The authors of [58] show that exploiting the network structure and using ideas from the network simplex algorithm for crossover may help to reduce crossover time by a factor of 10 or more compared to Gurobi's default crossover procedure on large optimal transport instances. As the crossover problem that we consider is also symmetric, the methods from [58] may be further enhanced by formulating symmetry-aware crossover strategies.

Considering our results, we remark that R-DRCR-N was more effective on the tested instances than R-DRCR-T. One explanation for this observation is that the structure of the affine TU decomposition that we observe for the generalized assignment problem corresponds to an assignment problem, which is given by a constraint matrix that is a network matrix but not a transposed network matrix.

Our approach can also be interpreted as *exact orbital shrinking* where the orbital shrinking subproblem is a perfect formulation. Salvagnin [9] observed that orbital shrinking may be significantly more powerful than exact orbital shrinking if one aggregates a larger symmetry group, which leads to a more difficult subproblem that is not necessarily a perfect formulation. Further research could investigate how one can better navigate the trade-off between stronger bounds and the difficulty of the subproblem for orbital shrinking. A crucial component here is efficient solution strategies for the orbital shrinking subproblem, which still contains the symmetries that are factored out of the original problem. As also remarked by Salvagnin [9], one could also view orbital shrinking as a logic-based Benders' decomposition and exploit infeasible subproblems by providing cutting planes that cut off infeasible solutions.

Acknowledgements. I am grateful to Matthias Walter for his feedback that helped improve the quality of this work, and I would like to thank Christopher Hojny for recommending relevant literature.

The author acknowledges funding support from the Netherlands Organisation for Scientific Research (NWO) on grant number OCENW.M20.151: "Making Mixed-Integer Programming Solvers Smarter and Faster using Network Matrices".

References

[1] Gleixner, A., al.: Miplib 2017: data-driven compilation of the 6th mixed-integer programming library. Mathematical Programming Computation **13**(3), 443–490 (2021) https://doi.org/10.1007/s12532-020-00194-3

[2] Hojny, C.: Detecting and handling reflection symmetries in mixed-integer (nonlinear) programming and beyond. Mathematical Programming Computation (2025) https://doi.org/10.1007/s12532-025-00289-9

[3] Margot, F.: In: Jünger, M., Liebling, T.M., Naddef, D., Nemhauser, G.L., Pulleyblank, W.R., Reinelt, G., Rinaldi, G., Wolsey, L.A. (eds.) Symmetry in Integer Linear Programming, pp. 647–686. Springer, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-540-68279-0_17 . https://doi.org/10.1007/978-3-540-68279-0_17

[4] Bödi, R., Herr, K., Joswig, M.: Algorithms for highly symmetric linear and integer programs. Mathematical Programming **137**(1-2), 65–90 (2013) https://doi.org/10.1007/s10107-011-0487-6 arXiv:1012.4941

[5] Gatermann, K., Parrilo, P.A.: Symmetry Groups, Semidefinite Programs, and Sums of Squares vol. 192, pp. 95–128 (2004). https://doi.org/10.1016/j.jpaa.2003.12.011

[6] Ostrowski, J., Linderoth, J., Rossi, F., Smriglio, S.: Orbital branching. Mathematical Programming **126**(1), 147–178 (2011) https://doi.org/10.1007/s10107-009-0273-x

[7] Liberti, L.: Reformulations in mathematical programming: automatic symmetry detection and exploitation. Mathematical Programming **131**(1-2), 273–304 (2012) https://doi.org/10.1007/s10107-010-0351-0

[8] Fischetti, M., Liberti, L.: Orbital shrinking. In: Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) vol. 7422 LNCS, pp. 48–58. Springer, Berlin, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32147-4_6

[9] Salvagnin, D.: Orbital shrinking: A new tool for hybrid mip/cp methods. In: Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) vol. 7874 LNCS, pp. 204–215 (2013). https://doi.org/10.1007/978-3-642-38171-3_14

[10] Fischetti, M., Liberti, L., Salvagnin, D., Walsh, T.: Orbital shrinking: Theory and applications. Discrete Applied Mathematics **222**, 109–123 (2017) https://doi.org/10.1016/j.dam.2017.01.015

[11] Grohe, M., Kersting, K., Mladenov, M., Selman, E.: Dimension reduction via colour refinement. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **8737 LNCS**, 505–516 (2014) https://doi.org/10.1007/978-3-662-44777-2_42 arXiv:1307.5697

[12] Bachoc, C., Gijswijt, D.C., Schrijver, A., Vallentin, F.: Invariant semidefinite programs. In: Handbook of Semidefinite, Conic and Polynomial Optimization vol. 166, pp. 219–270. Springer, New York (2012). https://doi.org/10.1007/978-1-4614-0769-0_9

[13] Permenter, F., Parrilo, P.A.: Dimension reduction for semidefinite programs via jordan algebras. Mathematical Programming **181**(1), 51–84 (2020) https://doi.org/10.1007/s10107-019-01372-5 arXiv:1608.02090

[14] Read, R.C., Corneil, D.G.: The graph isomorphism disease. Journal of Graph Theory **1**(4), 339–363 (1977) https://doi.org/10.1002/jgt.3190010410

[15] Junttila, T., Kaski, P.: bliss: A Tool for Computing Automorphism Groups and Canonical Labelings of Graphs. http://www.tcs.hut.fi/Software/bliss/ Accessed 6 November 2025

[16] McKay, B.D., Piperno, A.: Nauty User's Guide (version 2.7) (2025). https://users.cecs.anu.edu.au/$\sim$bdm/nauty/ Accessed 6 November 2025

[17] Darga, P.T., Katebi, H., Liffiton, M., Markov, I.L., Sakallah, K.: Saucy3: Fast Symmetry Discovery in Graphs. http://vlsicad.eecs.umich.edu/BK/SAUCY/ Accessed 6 November 2025

[18] Berthold, T.: How to Fold a linear programming Problem (2018). https://community.fico.com/s/blog-post/a5Q80000000Drp2EAC/fico1299 Accessed 13 October 2025

[19] Optimization, G.: What's new in the Gurobi Optimizer 9.1? (2020). https://www.youtube.com/watch?v=EwuEPu-nBmg Accessed 2025-10-13

[20] Liberti, L., Ostrowski, J.: Stabilizer-based symmetry breaking constraints for mathematical programs. Journal of Global Optimization **60**(2), 183–194 (2014) https://doi.org/10.1007/s10898-013-0106-6

[21] Margot, F.: Pruning by isomorphism in branch-and-cut. Mathematical Programming **94**(1), 71–90 (2002) https://doi.org/10.1007/s10107-002-0358-2

[22] Margot, F.: Exploiting orbits in symmetric ilp. Mathematical Programming **98**(1-3), 3–21 (2003) https://doi.org/10.1007/s10107-003-0394-6

[23] Ostrowski, J., Linderoth, J., Rossi, F., Smriglio, S.: Constraint orbital branching. In: Integer Programming and Combinatorial Optimization vol. 5035 LNCS, pp. 225–239. Springer, Berlin, Heidelberg (2008). https://doi.org/10.1007/978-3-540-68891-4_16

[24] Pfetsch, M.E., Rehn, T.: A computational comparison of symmetry handling methods for mixed integer programs. Mathematical Programming Computation **11**(1), 37–93 (2019) https://doi.org/10.1007/s12532-018-0140-y

[25] Salvagnin, D., Walsh, T.: A hybrid mip/cp approach for multi-activity shift scheduling. In: Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 7514

LNCS, pp. 633–646 (2012). https://doi.org/10.1007/978-3-642-33558-7_46

[26] Herr, K., Rehn, T., Schürmann, A.: Exploiting symmetry in integer convex optimization using core points. Operations Research Letters **41**(3), 298–304 (2013) https://doi.org/10.1016/j.orl.2013.02.007 arXiv:1202.0435

[27] Achterberg, T., Bixby, R.E., Gu, Z., Rothberg, E., Weninger, D.: Presolve reductions in mixed integer programming. Technical report, Zuse Institute Berlin, Berlin (2016)

[28] Gemander, P., Chen, W.-K., Weninger, D., Gottwald, L., Gleixner, A., Martin, A.: Two-row and two-column mixed-integer presolve using hashing-based pairing methods. EURO Journal on Computational Optimization **8**(3-4), 205–240 (2020) https://doi.org/10.1007/s13675-020-00129-6

[29] van der Hulst, R., Walter, M.: Implied integrality in mixed-integer optimization. In: Integer Programming and Combinatorial Optimization, pp. 452–465. Springer, Cham (2025). https://doi.org/10.1007/978-3-031-93112-3_33

[30] Bader, J., Hildebrand, R., Weismantel, R., Zenklusen, R.: Mixed integer reformulations of integer programs and the affine tu-dimension of a matrix. Mathematical Programming **169**(2), 565–584 (2018) https://doi.org/10.1007/s10107-017-1147-2 1508.02940

[31] Bixby, R.E., Wagner, D.K.: An almost linear-time algorithm for graph realization. Mathematics of Operations research **13**(1) (1988) https://doi.org/10.1287/moor.13.1.99

[32] van der Hulst, R., Walter, M.: A row-wise algorithm for graph realization **1**, 1–44 (2025) arXiv:2408.12869

[33] Schrijver, A.: Theory of Linear and Integer Programming. John Wiley & Sons, Inc., New York, NY, USA (1986)

[34] Hojny, C., Besançon, M., Bestuzheva, K., Borst, S., Dionísio, J., Ehls, J., Eifler, L., Ghannam, M., Gleixner, A., Göß, A., Hoen, A., Holly-Ponientzietz, J., van der Hulst, R., Kamp, D., Koch, T., Kofler, K., Lentz, J., Lübbecke, M., Maher, S.J., Meinhold, P.M., Mexi, G., Mohr, T., Mühmer, E., Patel, K.K., Pfetsch, M.E., Pokutta, S., Groba, C.R., Serrano, F., Shinano, Y., Turner, M., Vigerske, S., Walter, M., Weninger, D., Xu, L.: The SCIP Optimization Suite 10.0 (2025). https://arxiv.org/abs/2511.18580

[35] Paige, R., Tarjan, R.E.: Three partition refinement algorithms. SIAM Journal on Computing **16**(6), 973–989 (1987) https://doi.org/10.1137/0216062

[36] Hopcroft, J.: An n log n algorithm for minimizing states in a finite automaton. In: Theory of Machines and Computations vol. 7874 LNCS, pp. 189–196. Elsevier,

Haifa, Israel (1971). https://doi.org/10.1016/B978-0-12-417750-5.50022-1

[37] Geyer, A.J., Bulutoglu, D.A., Ryan, K.J.: Finding the symmetry group of an lp with equality constraints and its application to classifying orthogonal arrays. Discrete Optimization **32**, 93–119 (2019) https://doi.org/10.1016/j.disopt.2019.01.001

[38] Gomory, R.E.: An algorithm for the mixed-integer problem. Technical Report RM-2597, RAND Corporation (1960)

[39] Megiddo, N.: On finding primal- and dual-optimal bases. ORSA Journal on Computing **3**(1), 63–65 (1991) https://doi.org/10.1287/ijoc.3.1.63

[40] Bolusani, S., Besançon, M., Bestuzheva, K., Chmiela, A., Dionísio, J., Donkiewicz, T., Doornmalen, J., Eifler, L., Ghannam, M., Gleixner, A., Graczyk, C., Halbig, K., Hedtke, I., Hoen, A., Hojny, C., Hulst, R., Kamp, D., Koch, T., Kofler, K., Lentz, J., Manns, J., Mexi, G., Mühmer, E., Pfetsch, M.E., Schlösser, F., Serrano, F., Shinano, Y., Turner, M., Vigerske, S., Weninger, D., Xu, L.: The scip optimization suite 9.0 (05), 1–36 (2024) arXiv:2402.17702

[41] Christophel, P.M., Güzelsoy, M., Pólik, I.: New symmetries in mixed-integer linear optimization: Symmetry heuristics and complement-based symmetries (2014)

[42] Berkholz, C., Bonsma, P., Grohe, M.: Tight lower and upper bounds for the complexity of canonical colour refinement. Theory of Computing Systems **60**(4), 581–614 (2017) https://doi.org/10.1007/s00224-016-9686-0 arXiv:1509.08251

[43] Hoffman, A.J., Kruskal, J.B.: Integral boundary points of convex polyhedra. In: Linear Inequalities and Related Systems vol. 38, pp. 223–246. Princeton University Press, Princeton, New Jersey (1956). https://doi.org/10.1515/9781400881987-014

[44] Meyer, R.R.: On the existence of optimal solutions to integer and mixed-integer programming problems. Mathematical Programming **7**, 223–235 (1974) https://doi.org/10.1007/BF01585518

[45] Khachiyan, L.G.: Polynomial algorithms in linear programming. USSR Computational Mathematics and Mathematical Physics **20**(1), 53–72 (1980) https://doi.org/10.1016/0041-5553(80)90061-0

[46] Grötschel, M., Lovász, L., Schrijver, A.: The ellipsoid method and its consequences in combinatorial optimization. Combinatorica **1**(2), 169–197 (1981) https://doi.org/10.1007/BF02579273

[47] Bland, R.G., Goldfarb, D., Todd, M.J.: The ellipsoid method: A survey. Operations Research **29**(6), 1039–1091 (1981) https://doi.org/10.1287/opre.29.6.1039

[48] Ziegler, G.M.: Lectures on Polytopes (Graduate Texts in Mathematics). Springer, New York (2001)

[49] Truemper, K.: A decomposition theory for matroids. v. testing of matrix total unimodularity. Journal of Combinatorial Theory, Series B **49**(2), 241–281 (1990) https://doi.org/10.1016/0095-8956(90)90030-4

[50] Walter, M., Truemper, K.: Implementation of a unimodularity test. Mathematical Programming Computation **5**(1), 57–73 (2013) https://doi.org/10.1007/s12532-012-0048-x

[51] van der Hulst, R.: Folding Mixed-Integer Linear Programs and Reflection Symmetries. GitHub (2026). https://github.com/rolfvdhulst/folding

[52] van der Hulst, R.: MATREC: Matrix recognition algorithms. GitHub (2024). https://github.com/rolfvdhulst/matrec

[53] Gleixner, A., Gottwald, L., Hoen, A.: PaPILO: A parallel presolving library for integer and linear programming with multiprecision support. INFORMS Journal on Computing (2023) https://doi.org/10.1287/ijoc.2022.0171.cd

[54] Achterberg, T., Bixby, R.E., Gu, Z., Rothberg, E., Weninger, D.: Presolve reductions in mixed integer programming. INFORMS Journal on Computing **32**(2), 473–506 (2020) https://doi.org/10.1287/ijoc.2018.0857

[55] 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

[56] Deakins, E., Knueven, B., Ostrowski, J.: Orbital Crossover (2022). https://optimization-online.org/2022/12/orbital-crossover/

[57] Bixby, R.E., Cunningham, W.H.: Converting linear programs to network problems. Mathematics of Operations Research **5**(3), 321–357 (1980) https://doi.org/10.1287/moor.5.3.321

[58] Ge, D., Wang, C., Xiong, Z., Ye, Y.: From an interior point to a corner point: Smart crossover. INFORMS Journal on Computing (December) (2025) https://doi.org/10.1287/ijoc.2022.0291 arXiv:2102.09420

Appendix A Experimental results

Table A1: Comparison of DRCR and R-DRCR on the linear relaxation of MIPLIB2017 instances.

	Original		DRCR				R-DRCR			
instance	rows	columns	time	iters	rows	columns	time	iters	rows	columns
academictimetablebig	146304	154686	155.04	36162	130415	121047	259.45	38111	121759	111881
academictimetablesmall	17258	25550	0.59	2969	13727	9096	0.88	3030	13442	8845
bab3	22500	393401	1085.09	353260	22500	393401	1133.28	350161	22578	393401
brazil3	3876	13002	7.46	24655	3278	5604	3.25	15098	2842	3642
bts4-cta	34301	74259	0.62	5578	34013	74115	0.63	5579	34013	74115
cryptanalysiskb128n5obj14	68508	33549	11.53	20225	50252	24417	12.77	21498	50220	24401
cryptanalysiskb128n5obj16	68508	33549	12.34	21756	50252	24417	10.28	19876	50220	24401
eva1aprime5x5opt	11737	1520	12.01	28688	9623	1520	4.24	17575	7270	1141
eva1aprime6x6opt	34464	3106	78.44	63347	23945	3106	34.74	50149	19227	2359
fastxgemm-n2r6s0t2	4462	784	0.00	7	24	10	0.00	3	20	8
fastxgemm-n2r7s4t1	5180	904	0.00	7	24	10	0.00	3	20	8
fastxgemm-n3r21s3t6	158132	18684	0.04	7	24	10	0.04	3	20	8
fastxgemm-n3r22s4t6	165590	19539	0.05	7	24	10	0.04	3	20	8
fastxgemm-n3r23s5t6	173048	20394	0.05	7	24	10	0.04	3	20	8
fhnw-binpack4-48	4480	3710	0.04	3062	4480	3710	0.02	0	3885	3083
fhnw-binpack4-4	620	520	0.00	549	620	520	0.00	0	542	431
fiball	2387	32899	0.11	2570	878	4948	0.06	1831	878	4878
graphdraw-grafo2	203455	9258	0.33	1087	40068	2813	0.45	1515	40067	2813
highschool1-aigio	67773	299378	3602.72	203703	54501	154250	3600.71	229931	54482	154227
ic97_tension	202	334	0.00	27	201	333	0.00	31	379	332
icir97_potential	1657	1945	0.01	687	1657	1945	0.01	586	2698	1637
icir97_tension	880	1119	0.01	16	820	1057	0.01	18	1574	1039
in	1504669	1444973	3606.62	144868	1504365	1444707	3607.59	130656	1503062	1443561
irish-electricity	70342	38611	292.93	85415	66825	36122	210.96	72675	60894	32221
kottenpark09	119513	1315854	3603.47	53892	118541	1299889	3604.04	66638	118538	1299793
lectsched-1	9206	9350	0.04	0	9108	9252	0.05	0	17935	9123
lectsched-2	4894	5013	0.02	0	4852	4971	0.03	0	9557	4903
lectsched-3	8185	8321	0.03	0	8104	8240	0.04	0	15979	8137
lectsched-4-obj	3102	3187	0.00	115	2950	3035	0.02	116	5285	2989
lectsched-5-obj	9453	9582	0.05	421	8822	8951	0.06	432	15848	8833
liu	2178	1154	0.02	560	2178	1154	0.00	1	1089	563
maxgasflow	5786	5974	0.11	4463	5473	5658	0.11	4015	5219	5366
neos-1324574	5763	5220	0.01	364	1247	1093	0.01	523	1036	910
neos-1330346	4177	2628	0.18	2628	4177	2628	0.06	1340	2794	1747
neos-2974461-ibar	209853	210174	3600.76	112419	209853	210174	3601.85	158103	208249	209304
neos-3009394-lami	1677	1353	0.00	169	419	341	0.00	93	220	179
neos-3068746-nene	4409	4520	0.24	2482	4228	4415	0.21	2178	4123	4356
neos-3211096-shag	5818	4271	0.00	35	250	163	0.00	19	193	82
neos-3355323-arnon	11010	10128	0.07	3564	8106	7224	0.01	21	131	62
neos-3402294-bobin	32932	780	0.69	978	32932	780	0.04	96	4259	132
neos-3402454-bohle	2881228	2496	1353.86	25314	2881228	2496	5.78	796	81413	990
neos-3656078-kumeu	9614	10255	0.14	4269	8266	8831	0.24	5338	10850	8477
neos-4306827-ravan	120429	67539	0.28	2680	29549	14519	0.28	1415	25145	10790
neos-4338804-snowy	1470	1323	0.01	340	1470	1323	0.00	0	1048	888
neos-4754521-awarau	128319	16140	14.31	0	117152	15099	10.49	0	84952	15086
neos-4797081-pakoka	6402	11777	49.80	21872	6402	11777	44.89	21164	6664	11777
neos-5075914-elvire	2629	2602	0.05	1310	2531	2504	0.05	1360	2532	2388
neos-5076235-embley	6336	23618	0.64	4391	6238	23569	0.61	4005	5783	22239
neos-5078479-escaut	2783	2602	0.05	1340	2765	2584	0.05	1386	2912	2559
neos-5079731-flyers	6335	23618	0.52	3968	6237	23569	0.71	5123	5691	21973
neos-5093327-huahum	5988	19252	0.34	4276	5922	19220	0.44	4446	5808	18992
neos-5100895-inster	4802	14000	0.20	2325	4746	13972	0.24	2278	4460	13004
neos-5102383-irwell	6629	24500	0.51	3877	6531	24451	1.07	6802	5972	22559
neos-5107597-kakapo	3249	3045	0.01	200	3249	3045	0.00	158	3513	1606
neos-5115478-kaveri	3249	3045	0.01	198	3249	3045	0.01	140	3513	1606
neos-5125849-lopori	397	8074	0.02	132	327	6170	0.02	77	205	3470
neos-5129192-manaia	530672	161799	2.31	1115	530672	161799	2.08	40	529939	159533
neos-780889	25866	112634	1.60	8074	17032	74453	6.93	13524	16873	73785
neos-826650	2245	5024	0.00	94	137	157	0.00	83	128	144
neos-983171	6669	7269	2.22	14068	3556	3871	3.15	17258	3556	3870
neos9	31600	81408	0.03	82	332	672	0.04	38	166	336
nh97_potential	958	1100	0.00	227	958	1100	0.00	130	564	354
ns1905797	51600	18180	0.39	1097	38715	13635	0.19	669	25830	9090
ns4-pr6	1589	4991	0.00	261	395	1146	0.00	243	384	1108
nu120-pr12	1515	4444	0.02	1127	1363	3939	0.04	1402	1360	3927
nu120-pr9	1823	6780	0.50	4785	1803	6702	0.50	4991	1803	6702
nu25-pr12	1516	4445	0.03	1289	1347	3889	0.04	1343	1344	3877
nu4-pr9	1823	6780	0.76	7556	1803	6702	0.64	6286	1803	6702
p0201	110	183	0.00	44	80	102	0.00	26	50	52
physicianshed6-1	38870	26248	5.60	15821	38104	25478	7.92	16836	38139	25452
physicianshed6-2	31064	17093	10.48	25047	30830	16963	9.72	22787	30850	16943
piperout-27	17225	11403	1.37	11423	17225	11403	0.81	10283	17226	11403
pythago7824	7326	3740	34.37	39996	7326	3740	0.00	0	1	1
pythago7825	7336	3745	33.88	39996	7336	3745	0.00	0	1	1
supportcase33	9862	13641	0.21	549	9862	13641	0.29	555	9901	13631
supportcase37	39192	9674	3.44	15069	36075	8800	3.62	15156	36084	8800
tokyometro	6835	3606	0.06	1792	6791	3575	0.07	1852	6919	3567
transportmoment	7312	7268	0.60	8299	6989	6946	0.55	7757	6719	6646
uccase10	140361	73892	145.33	94940	79074	39006	12.57	30401	26022	12518

uccase12	88635	40242	9.46	35027	60606	26822	2.53	14668	36541	15758
uccase7	31928	21289	90.57	47420	31093	20786	83.19	44585	31581	20786
unitcal_7	41890	21979	2.56	16471	32001	16779	2.15	15560	33834	16779
woodlands09	36150	227094	3600.49	165913	26988	110657	3600.55	364827	26988	110657