Outer approximations of core points for integer programming

Naghmeh Shahverdi* David Bremner†

July 14, 2023

Abstract

For several decades the dominant techniques for integer linear programming have been branching and cutting planes. Recently, several authors have developed core point methods for solving symmetric integer linear programs (ILPs). An integer point is called a core point if its orbit polytope is lattice-free. It has been shown that for symmetric ILPs, optimizing over the set of core points gives the same answer as considering the entire space. Existing core point techniques rely on the number of core points (or equivalence classes) being finite, which requires special symmetry groups. In this paper we develop some new methods for solving symmetric ILPs — based on outer approximations of core points — that do not depend on finiteness but are more efficient if the group has large disjoint cycles in its set of generators.

1 Introduction

Formulation symmetries occur in practice when relabellings yield equivalent problem structure; this causes repeated work for branching solvers, and state of the art commercial and research solvers make efforts to break symmetries [14]. Let $G \leqslant S_n$ be a permutation group acting on $\mathbb{R}^n$ by permuting coordinates. For any integer point $z \in \mathbb{Z}^n$, the *orbit polytope* of z is the convex hull of the G-orbit of z. If the vertices of an orbit polytope are the only integer points in the polytope we call it *lattice-free* and call z a *core point*. Instead of seeing symmetry as a problem, core point techniques seek to exploit it to solve integer linear programs (ILPs) faster. In the most direct approach, when the number of core points is finite (which only holds for certain special groups), core points are enumerated and tested individually [2]. It should be noted that core point techniques are not useful for binary problems since every $\{0, 1\}$-point is a core point; [9] considers an alternative approach based on lexicographical order.

*naghmehshahverdi2@gmail.com
†bremner@unb.ca

Computation of symmetries in the MIPLIB 2010 and 2017 instances has been done in [13] and this study shows that many instances are affected by symmetry. Symmetric Integer Linear Programming appears in many problems such as scheduling on identical machines and code construction. For solving these problems, Artificial Intelligence approaches have been investigated in [10, 12].

A (not necessarily polyhedral) outer approximation is a set of constraints that is feasible for all of the points in the set one wishes to approximate. A well known example of an outer approximation is an ILP, where the (initial) linear constraints define an outer approximation of the feasible integer points. Outer approximations lead naturally to a hybrid approach where synthesized constraints are added to an existing formulation and then solved with a traditional solver. Outer approximations are implicit in previous results bounding the distance of core points to certain linear subspaces (see e.g. Theorem 3.24 in [15]). The distance bounds do not themselves seem to be tight enough to provide a practical improvement for solving ILPs. In this paper we develop some new constraints for problems with formulation symmetries. While these constraints are nonlinear and non-convex, initial experiments with nonlinear solvers seem promising.

In Section 2 we give some basic definitions. In Section 3 we consider integer linear programs with cyclic symmetry groups and provide some new constraints to determine outer approximations of their core points. We also provide an algorithm that uses these constraints to solve an ILP. In Section 4 we generalize this algorithm for ILPs where only some of their variables have cyclic symmetry. In Section 5 we generalize the algorithm for direct products of cyclic groups. In Section 6 we classify permutation groups based on their generators and explain how the algorithms of the previous sections can be applied to ILPs having arbitrary permutation groups as symmetry groups. Finally in Section 7 we use our algorithms to solve some hard symmetric integer linear feasibility problems.

2 Basic Definitions

Symmetries of geometric objects (e.g. polyhedra and integer lattices) in integer linear programming can be viewed as the action of some underlying group.

Definition 2.1. *If G is a group and X is a set, then a (left)* group action *ϕ of G on X is a function*

$$\begin{aligned}\phi : G \times X &\to X,\\ (g, x) &\to gx,\end{aligned}$$

that satisfies the following two axioms.

1. *$ex = x$ for all $x \in X$, where $e \in G$ is the identity element.*
2. *$(gh)x = g(hx)$ for all $g, h \in G$ and all $x \in X$.*

In this paper we assume that all groups considered are permutation groups, and that they act on a family $\mathbb{X}^n$ of n-tuples (e.g. $\mathbb{C}^n$, $\mathbb{R}^n$, or $\mathbb{Z}^n$ in the usual coordinates) as follows.

Remark 2.2. *The permutation group $G \leqslant S_n$ acts on $\mathbb{X}^n$ by permuting coordinates $\{0, \dots n-1\}$: for $g \in G$ and $x = (x_0, \dots, x_{n-1}) \in \mathbb{X}^n$*

$$\phi_g(x) = (x_{g^{-1}(0)}, x_{g^{-1}(1)}, \dots, x_{g^{-1}(n-1)}).$$

There are certainly more general notions of symmetry possible, but permuting coordinates is the most widely studied class of symmetries in integer linear programming [14].

The techniques in this paper rely on restricting the search for feasible integral points to certain special integral points called *core points*. Core points are defined relative to a given group as follows.

Definition 2.3 (Core Points)**.**

1. *Let $P \subset \mathbb{R}^n$ be a convex polytope with integral vertices. We call P* lattice-free *if $P \cap \mathbb{Z}^n = \mathrm{vert}(P)$ where $\mathrm{vert}(P)$ is the set of vertices of P.*

2. *Let $G \leqslant GL_n(\mathbb{R})$ be a finite group and let G_z be the G-orbit of some point $z \in \mathbb{R}^n$. We call the convex hull of this orbit an* orbit polytope *and denote it by $\mathrm{conv}(G_z)$.*

3. *Let $G \leqslant GL_n(\mathbb{Z})$ be a finite group of unimodular matrices. A point $z \in \mathbb{Z}^n$ is called a* core point *for G if and only if the orbit polytope $\mathrm{conv}(G_z)$ is lattice-free.*

Let $C_n = \langle \sigma \rangle$ denote the *cyclic group* generated by σ, a cyclic permutation of coordinates. In other words, $\sigma : \mathbb{R}^n \to \mathbb{R}^n$ is given by

$$\sigma(c_0, c_1, \dots, c_{n-1}) = (c_{n-1}, c_0, \dots, c_{n-2})$$

We represent points in $\mathbb{R}^n$ by column vectors but for convenience, we write such vectors here in a transpose way. The map σ is easily iterated:

$$\sigma^k(c_0, c_1, \dots, c_{n-1}) = (c_{n-k}, c_{n-k+1}, \dots, c_{n-1}, c_0, \dots, c_{n-k-1})$$

Consider a cyclic group $C_4 = \langle (1, 2, 3, 4) \rangle$. The orbit of $x = (2, 0, 4, 1)$ consists of the four vectors $x_1 = (2, 0, 4, 1)$, $x_2 = (1, 2, 0, 4)$, $x_3 = (4, 1, 2, 0)$, $x_4 = (0, 4, 1, 2)$. The orbit polytope of x is not lattice-free because

$$(1, 2, 1, 3) = \frac{1}{5}x_1 + \frac{3}{5}x_2 + 0x_3 + \frac{1}{5}x_4,$$

so that x is not a core point.

Definition 2.4. *Let G act on $\mathbb{C}^n$. The subset of $\mathbb{C}^n$ preserved by all elements of G is called the* fixed space*:*

$$\mathrm{Fix}(G) := \{x \in \mathbb{C}^n \mid gx = x, \forall g \in G\}.$$

We denote by $\mathrm{Fix}_{\mathbb{Z}}(G)$ (resp. $\mathrm{Fix}_{\mathbb{R}}(G)$), the intersection $\mathrm{Fix}(G) \cap \mathbb{Z}^n$ (resp. $\mathrm{Fix}(G) \cap \mathbb{R}^n$).

We use $\mathbf{1}^k$ to denote the k-dimensional vector of all ones (or just $\mathbf{1}$ where the dimension is clear from context). It is easy to see that $\mathbf{1}^n$ is contained in $\mathrm{Fix}_{\mathbb{Z}}(G)$ for any group G acting on $\mathbb{R}^n$ by permuting coordinates. If G is further transitive, then $\mathrm{Fix}_{\mathbb{R}}(G) = \mathrm{Span}(\mathbf{1})$.

Definition 2.5. *We define the k-th* layer *to be the set*

$$\mathbb{Z}^n_{(k)} := \{z \in \mathbb{Z}^n \mid \langle z, \mathbf{1}\rangle = k\}.$$

Note that the set $\mathbb{Z}^n_{(k)}$ is G-invariant because G acts by permuting coordinates. Consider the cyclic group $C_4 = \langle(1,2,3,4)\rangle$. Let $X = \mathbb{Z}^4_{(2)} \cap \{0,1\}^4$ be the set of $\{0,1\}$-points in $\mathbb{R}^4$ in layer two, i.e., each point has two 1s and two 0s, and suppose the action is the same as in Remark 2.2. This action has two orbits:

$$\begin{aligned} O_G((1,1,0,0)) &= \{\,(1,1,0,0),(0,1,1,0),(0,0,1,1),(1,0,0,1)\,\}\,,\\ O_G((1,0,1,0)) &= \{\,(1,0,1,0),(0,1,0,1)\,\}\,. \end{aligned}$$

The fixed space $\mathrm{Fix}_{\mathbb{Z}}(C_4)$ contains only two $0,1$ points, namely $(1,1,1,1)$ and $(0,0,0,0)$, and neither is in X.

In the remainder of the paper we will need several different notions of equivalence for integral points. Potentially larger equivalence classes (based on normalizers) of core points are studied in [11].

Definition 2.6 (Equivalence relations)**.**

1. *Two points $x, y \in \mathbb{Z}^n$ are called* equivalent *if there exists $g \in G$ such that $x = gy$. It follows from the group axioms that this is an equivalence relation.*

2. *Two points $x, y \in \mathbb{Z}^n$ are called* isomorphic *if there exists $g \in G$ such that $x - gy \in \mathrm{Fix}_{\mathbb{Z}}(G)$. This is an equivalence relation because $\mathrm{Fix}_{\mathbb{Z}}(G)$ is a lattice.*

3. *Two integer points z_1 and z_2 in $\mathbb{Z}^n$ are called* co-projective *if there exists an integer $k \in \mathbb{Z}$ such that $z_1 = z_2 + k\mathbf{1}$. Equivalently, if the group is transitive, z_2 is a translation of z_1 through the fixed space.*

Each of the equivalence relations in Definition 2.6 has equivalence classes that are either entirely core-points or entirely non-core integer points. Along with the observation that $\{0,1\}$-vectors are core points for any permutation group [15, Lemma 3.7], we can define a family of *universal* core points.

Definition 2.7. *A point $u \in \mathbb{Z}^n$ is called a* universal core point *if it is isomorphic to a $\{0,1\}$-vector.*

As a heuristic for identifying more points contained in the orbit polytopes of many (non-core) points, we consider integer points near universal core points.

Definition 2.8. *An integer point z is called an* atom *if there is a universal core point u in the layer containing z such that the distance between z and u is $\sqrt{2}$.*

For example if $G = \langle(1,2,3,4,5)\rangle$ then the fixed space is spanned by $\mathbf{1}$ and $(2,2,2,2,1) = (1,1,1,1,0) + (1,1,1,1,1)$ is a universal core point. The point $(3,2,2,1,1)$ is an atom since $(3,2,2,1,1) - (2,2,2,2,1) = (1,0,0,-1,0)$ hence its distance to the universal core point $(2,2,2,2,1)$ is $\sqrt{2}$.

Definition 2.9. *Suppose the cyclic group $G = \langle(g_1,\ldots,g_k)\rangle \leqslant S_n$, $g_i \in \{0,\ldots,n-1\}$ acts on $\mathbb{R}^n$ by permuting coordinates. Then coordinate i is called* active *if there is $j \in \{1,\ldots,k\}$ such that $g_j = i$.*

For example if $G = \langle(0,1,3,4)\rangle$ acts on $\mathbb{R}^5$ then x_0, x_1, x_3, x_4 are active but x_2 is non-active.

3 Circulant Matrices

Circulant matrices play an important role in finding our constraints because any point x in the orbit polytope of the integer point $c \in \mathbb{Z}^n$ under the cyclic group C_n can be written as $x = C\lambda$, where $x \in \mathbb{R}^n$, $\lambda \in [0,1]^n$ and C is the *circulant matrix* of c.

Definition 3.1. *A* circulant matrix *is a matrix where each column vector is rotated one element down relative to the preceding column vector. An $n \times n$ circulant matrix* $\mathrm{Cir}(c)$ *takes the form*

$$C = \begin{bmatrix} c_0 & c_{n-1} & \cdots & c_2 & c_1 \\ c_1 & c_0 & c_{n-1} & \cdots & c_2 \\ \vdots & c_1 & c_0 & \ddots & \vdots \\ c_{n-2} & & \ddots & \ddots & c_{n-1} \\ c_{n-1} & c_{n-2} & \cdots & c_1 & c_0 \end{bmatrix}.$$

One amazing property of circulant matrices is that the eigenvectors are always the same for all $n \times n$ circulant matrices. The eigenvalues are different for each matrix, but since we know the eigenvectors a circulant matrix can be diagonalized easily. For more detailed background on circulant matrices see [7].

The m-th eigenvector y^m for any $n \times n$ circulant matrix $\mathrm{Cir}(c)$ is given by:

$$y^m = \frac{1}{\sqrt{n}}(1, w_n^{-m}, \ldots, w_n^{-(n-1)m})^T, \tag{1}$$

where $w_n^m = \exp(2\pi mi/n)$. Suppose $c = (c_0,\ldots,c_{n-1}) \in \mathbb{R}^n$ (usually for us c will be an integer point in $\mathbb{Z}^n$). By Euler's formula we have $\sqrt{n}y^m = V_m - iU_m$, $m = 0,\ldots,n-1$, where

$$V_m = \left(1, \cos\left(\frac{2\pi m}{n}\right), \ldots, \cos\left(\frac{2\pi(n-1)m}{n}\right)\right), \tag{2}$$

$$U_m = \left(0, \sin\left(\frac{2\pi m}{n}\right), \ldots, \sin\left(\frac{2\pi(n-1)m}{n}\right)\right). \tag{3}$$

The eigenvalue of the m-th eigenvector of the circulant matrix $\mathrm{Cir}(c)$ is

$$\psi_m = \sum_{k=0}^{n-1} c_k w_n^{km} = \langle V_m, c\rangle + i\langle U_m, c\rangle.$$

So we have

$$\mathrm{Cir}(c)Y = Y\Psi \quad \Longrightarrow \quad \mathrm{Cir}(c) = Y\Psi Y^*,$$

where

$$Y = [y^0 \mid \ldots \mid y^{n-1}],$$

is the unitary matrix composed of the eigenvectors as columns, and Ψ is the diagonal matrix with diagonal elements $\psi_0, \ldots, \psi_{n-1}$.

The inverse of a circulant matrix is circulant [4] and its inverse is given by

$$\mathrm{Cir}(c)^{-1} = Y\Psi^{-1}Y^*. \tag{4}$$

Since Ψ is a diagonal matrix its inverse is also a diagonal matrix with diagonal elements ψ_m^{-1}, $m = 0, \ldots, n-1$ where

$$\frac{1}{\psi_m} = \frac{1}{\sum_{k=0}^{n-1} c_k w_n^{km}} = \frac{1}{\langle V_m, c\rangle + i\langle U_m, c\rangle} = \frac{\langle V_m, c\rangle - i\langle U_m, c\rangle}{\langle V_m, c\rangle^2 + \langle U_m, c\rangle^2}. \tag{5}$$

Remark 3.2. *Note that the length of the projection of a vector $c \in \mathbb{R}^n$ onto a complex vector $v = a + ib \in \mathbb{C}^n$ is defined as*

$$\|Proj_v^c\|^2 = \frac{\|\langle c, a\rangle - i\langle c, b\rangle\|^2}{\|v\|^2} = \frac{\langle c, a\rangle^2 + \langle c, b\rangle^2}{\|v\|^2}. \tag{6}$$

Furthermore, the term $\langle V_m, c\rangle^2 + \langle U_m, c\rangle^2$ in (5) is the length of the projection of c onto invariant subspace y^m.

Lemma 3.3. *Let $c \in \mathbb{R}^n$ and suppose $\mathrm{Cir}(c)$ is invertible. Then its inverse is $\mathrm{Cir}(\widehat{T}(c))$, where $\widehat{T}(c)$ is defined as follows:*

$$\widehat{T}(c) = \frac{1}{n}\begin{bmatrix} \frac{1}{\langle c, \mathbf{1}\rangle} + T_0(c) \\ \frac{1}{\langle c, \mathbf{1}\rangle} + T_1(c) \\ \vdots \\ \frac{1}{\langle c, \mathbf{1}\rangle} + T_{n-1}(c) \end{bmatrix} = \frac{1}{n}\begin{bmatrix} \frac{1}{\langle c, \mathbf{1}\rangle} + \psi_1^{-1} + \ldots + \psi_{n-1}^{-1} \\ \frac{1}{\langle c, \mathbf{1}\rangle} + w_n^{-1}\psi_1^{-1} + \ldots + w_n^{-(n-1)}\psi_{n-1}^{-1} \\ \vdots \\ \frac{1}{\langle c, \mathbf{1}\rangle} + w_n^{-(n-1)}\psi_1^{-1} + .. + w_n^{-(n-1)^2}\psi_{n-1}^{-1} \end{bmatrix}.$$

Note that actually $\frac{1}{\langle c, \mathbf{1}\rangle} = \psi_0^{-1}$.

Proof. By (4) we have $\mathrm{Cir}(c)^{-1} = Y\Psi^{-1}Y^*$. Now suppose k_{hj} is the (h,j)-th component of $Y\Psi^{-1}$. We have

$$k_{hj} = Y_{hj}\Psi^{-1}_{jj} = \frac{1}{\sqrt{n}} w_n^{-(h-1)(j-1)} \Psi^{-1}_{(j-1)(j-1)}.$$

So, k^l, the l-th row of $Y\Psi^{-1}$, is equal

$$k^l = \frac{1}{\sqrt{n}} \left[\Psi^{-1}_{00}, w_n^{-(l-1)}\Psi^{-1}_{11}, w_n^{-2(l-1)}\Psi^{-1}_{22}, \ldots, w_n^{-(n-1)(l-1)}\Psi^{-1}_{(n-1)(n-1)} \right].$$

Now since $\mathrm{Cir}(c)^{-1}$ is a circulant matrix [4] it is enough to find the first column of $\mathrm{Cir}(c)^{-1}$ (the other columns can be found by permutation of the first column). Notice that the first row and column of Y and Y^* is $\frac{1}{\sqrt{n}}\mathbf{1} = \frac{1}{\sqrt{n}}(1,1,\ldots,1)$ so multiplying each row of $Y\Psi^{-1}$ with vector $\frac{1}{\sqrt{n}}\mathbf{1}$ gives us the first column of $Y\Psi^{-1}Y^*$, which is

$$\langle k^l, \frac{1}{\sqrt{n}}\mathbf{1}\rangle = \frac{1}{n}(\Psi^{-1}_{00} + w_n^{-(l-1)}\Psi^{-1}_{11} + w_n^{-2(l-1)}\Psi^{-1}_{22} + \ldots + w_n^{-(n-1)(l-1)}\Psi^{-1}_{(n-1)(n-1)}),$$

where $\Psi^{-1}_{00} = 1/\langle c, \mathbf{1}\rangle$. $\square$

The following theorem plays an important role in this paper, and is a useful formula for computing the inverse of an invertible circulant matrix.

Theorem 3.4. *For $c \in \mathbb{R}^n$, the entry $T_k(c)$ in the column vector $\widehat{T}(c)$ in* Lemma 3.3 *can be written as*

$$T_k(c) = 2\sum_{m=1}^{(n-1)/2} \frac{1}{\langle V_m, c\rangle^2 + \langle U_m, c\rangle^2}\langle \sigma^{-k}(V_m), c\rangle \quad \text{if } n \text{ is odd},$$

$$T_k(c) = 2\sum_{m=1}^{(n-2)/2} \frac{1}{\langle V_m, c\rangle^2 + \langle U_m, c\rangle^2}\langle \sigma^{-k}(V_m), c\rangle + \frac{(-1)^k}{\langle V_{\frac{n}{2}}, c\rangle} \quad \text{if } n \text{ is even}.$$

In particular each $T_k(c)$ is a real number.

Proof. As shown in Lemma 3.3 we have

$$T_k(c) = w_n^{-k}\psi_1^{-1} + \ldots + w_n^{-(n-1)k}\psi_{n-1}^{-1}.$$

Since ψ_l and ψ_{n-l} are complex conjugates of each other we have

$$\begin{aligned} w_n^{-lk}\psi_l^{-1} + w_n^{-(n-l)k}\psi_{n-l}^{-1} = &\\ &(\cos(\frac{2lk\pi}{n}) - i\sin(\frac{2lk\pi}{n}))\frac{\langle c, V_l\rangle - i\langle c, U_l\rangle}{\langle c, V_l\rangle^2 + \langle c, U_l\rangle^2} \\ &+ (\cos(\frac{2lk\pi}{n}) + i\sin(\frac{2lk\pi}{n}))\frac{\langle c, V_l\rangle + i\langle c, U_l\rangle}{\langle c, V_l\rangle^2 + \langle c, U_l\rangle^2} = \end{aligned}$$

$$\frac{2\cos(\frac{2lk\pi}{n})\langle c, V_l\rangle - 2\sin(\frac{2lk\pi}{n})\langle c, U_l\rangle}{\langle c, V_l\rangle^2 + \langle c, U_l\rangle^2} = 2\frac{\langle c, \sigma^{-k}(V_l)\rangle}{\langle c, V_l\rangle^2 + \langle c, U_l\rangle^2}.$$

Recall that $w_n^m = \cos\left(\frac{2\pi m}{n}\right) + i\sin\left(\frac{2\pi m}{n}\right)$ so the last equality holds because $\cos A\cos B - \sin A\sin B = \cos(A+B)$ and V_m is in terms of cos and U_m is in terms of sin (see (2) and (3)).

If n is even then for $l = \frac{n}{2}$ we have $\psi_{\frac{n}{2}} = \psi_{n-l}$ and $y^{\frac{n}{2}} = y^{n-l}$ so $w_n^{-\frac{n}{2}k}\psi_{\frac{n}{2}}^{-1}$ does not give a complex conjugate pair. But since the imaginary part of it is zero, we have

$$w_n^{-\frac{n}{2}k}\psi_{\frac{n}{2}}^{-1} = \cos(-k\pi)\frac{\langle c, V_{\frac{n}{2}}\rangle}{\langle c, V_{\frac{n}{2}}\rangle^2} = \frac{\cos(k\pi)}{\langle c, V_{\frac{n}{2}}\rangle} = \frac{(-1)^k}{\langle c, V_{\frac{n}{2}}\rangle}. \qquad \square$$

Lemma 3.5. *Suppose for $c \in \mathbb{R}^n$ that* $\mathrm{Cir}(c)$ *is invertible. Then* $\sum_{k=0}^{n-1} T_k(c) = 0$.

Proof. From Lemma 3.3 we have

$$\sum_{k=0}^{n-1} T_k(c) = \sum_{k=0}^{n-1}\sum_{m=1}^{n-1} w_n^{-km}\psi_m^{-1} = \sum_{m=1}^{n-1}\psi_m^{-1}\sum_{k=0}^{n-1} w_n^{-km}.$$

Since the inner sum is geometric, for $m < n$

$$\sum_{k=0}^{n-1} w_n^{-km} = 0. \qquad \square$$

Remark 3.6. *Note that if $z, c \in \mathbb{Z}^n_{(k)}$, $k \neq 0$, and $z = \mathrm{Cir}(c)\lambda$ then*

$$k = \langle \mathbf{1}, z\rangle = \langle \mathbf{1}, \mathrm{Cir}(c)\lambda\rangle = \langle \mathbf{1}, c\rangle\langle \mathbf{1}, \lambda\rangle = k\langle \mathbf{1}, \lambda\rangle.$$

This implies that $\langle \mathbf{1}, \lambda\rangle = 1$. Now suppose $\mathrm{Cir}(c)$ *is invertible, so $\lambda = \mathrm{Cir}(c)^{-1}z$. To check if $z \in \mathrm{conv}(G_c)$ or not, we only need to check if $\lambda = \mathrm{Cir}(c)^{-1}z \geqslant \mathbf{0}$. The constraint $\sum_{i=0}^{n-1}\lambda_i = 1$ is redundant.*

Let us denote by $\overline{T}(c)$ the first row of $\mathrm{Cir}(\widehat{T}(c))$ which is

$$\overline{T}(c) = \frac{1}{n}\left[\frac{1}{\langle c, \mathbf{1}\rangle} + T_0(c), \frac{1}{\langle c, \mathbf{1}\rangle} + T_{n-1}(c), \ldots, \frac{1}{\langle c, \mathbf{1}\rangle} + T_1(c)\right].$$

This will simplify the notation in following sections.

3.1 New Constraints for Singular Circulant Matrices

The circulant matrix $\mathrm{Cir}(c)$ corresponding to an integer point $c \in \mathbb{Z}^n$ is not invertible if and only if the determinant of $\mathrm{Cir}(c)$ is zero. Since the determinant of a square matrix is equal to the product of its n eigenvalues we have

$$\det(\mathrm{Cir}(c)) = \prod_{j=0}^{n-1}\left(\langle V_j, c\rangle + i\langle U_j, c\rangle\right).$$

Furthermore, since for $k = 1, \ldots, n-1$, $V_{n-k} + iU_{n-k}$ is the complex conjugate of $V_k + iU_k$ we have

$$\det(\mathrm{Cir}(c)) = \begin{cases} \langle V_0, c\rangle \prod_{j=1}^{(n-1)/2} (\langle V_j, c\rangle^2 + \langle U_j, c\rangle^2) & n \text{ odd}, \\ \langle V_0, c\rangle \langle V_{\frac{n}{2}}, c\rangle \prod_{j=1}^{(n-2)/2} (\langle V_j, c\rangle^2 + \langle U_j, c\rangle^2) & n \text{ even}. \end{cases}$$

So in order to check if an ILP has an integer solution whose circulant matrix is singular, we could add the following constraints to the problem

$$\langle V_0, c\rangle \prod_{j=1}^{(n-1)/2} (\langle V_j, c\rangle^2 + \langle U_j, c\rangle^2) = 0 \qquad \text{if } n \text{ is odd}, \tag{7}$$

$$\langle V_0, c\rangle \langle V_{\frac{n}{2}}, c\rangle \prod_{j=1}^{(n-2)/2} (\langle V_j, c\rangle^2 + \langle U_j, c\rangle^2) = 0 \qquad \text{if } n \text{ is even}. \tag{8}$$

Equations (7) and (8) involve polynomials in c of degree n. We can reformulate them in a way that make them easier to solve in practice by introducing a large positive constant M and $0,1$-valued variables r_m. Let P_j be a term in the products (7) and (8), namely, $P_j = \langle V_j, c\rangle$ when $j = 0$ or (for n even) when $j = \frac{n}{2}$. Otherwise, $P_j = \langle V_j, c\rangle^2 + \langle U_j, c\rangle^2 \geqslant 0$. For $j = 0, \ldots, \lceil (n-1)/2 \rceil$, add constraints

$$\begin{aligned} P_j &\leqslant r_j P_j & \qquad j \notin \{0, \tfrac{n}{2}\} \cap \mathbb{Z}, \\ -r_j M \leqslant P_j &\leqslant r_j M & \qquad \text{otherwise}. \end{aligned} \tag{9}$$

Finally add the following constraint to force at least one r_j to 0:

$$\sum_{j=0}^{\lceil (n-1)/2 \rceil} r_j \leqslant \left\lceil \frac{n-1}{2} \right\rceil - 1. \tag{10}$$

Note that constraints (9) and (10) forces at least one of the P_j to be zero and so the determinant will be zero. For maximization problems the constant M can be chosen as the absolute value of the objective value of the LP relaxation (assuming the objective function $f(x) = \langle \mathbf{1}, x\rangle$).

Remark 3.7. *We can formulate* (7) *and* (8) *in different ways. For example we can make* $\lceil \frac{n-1}{2} \rceil$ *subproblems by adding each* P_m *separately. The corresponding constraints in each subproblem are linear. For example in the* h*-th subproblem,* $h \in \{1, \ldots, \lceil \frac{n-1}{2} \rceil\}$*, we have*

$$P_h = \langle V_h, c\rangle^2 + \langle U_h, c\rangle^2 = 0,$$

which can be simplified as

$$\langle V_h, c\rangle = 0 \text{ and } \langle w_h, c\rangle = 0.$$

The weakness of this formulation is that adding these constraints does not simplify the problem enough because in each step we are searching in $n-2$ *dimensions which for large* n *is not a sufficient dimension reduction.*

3.2 New Constraints for Non-Singular Circulant Matrices

In this section we develop some new constraints to find an outer approximation of core points. These new constraints depend only on the symmetry group and not the ILP.

Suppose c is an arbitrary integer point in $\mathbb{Z}^n_{(k)},\ k \neq 0$, and G is the cyclic group with order n which acts on coordinates as usual. By Remark 3.6 a point $z \in \mathbb{Z}^n_{(k)}$ is in $\operatorname{conv}(G_c)$ if and only if there exist $\lambda = (\lambda_0, \ldots, \lambda_{n-1})$ such that

$$z = \operatorname{Cir}(c)\lambda, \qquad \lambda_i \geqslant 0.$$

If $\operatorname{Cir}(c)$ is invertible we have

$$\lambda = \operatorname{Cir}(c)^{-1} z, \tag{11}$$

and by Lemma 3.3

$$\lambda_j = \langle (\sigma^j(\overline{T}(c)), z\rangle.$$

The orbit polytope of a core point is lattice-free. Still assuming that $\operatorname{Cir}(c)$ is invertible, we conclude that $c \in \mathbb{Z}^n_{(k)}$ is a core point if and only if for each $z \in \mathbb{Z}^n_{(k)}$ which is not in the G-orbit of c, there exists a j such that

$$\lambda_j = \langle \overline{T}(c), \sigma^{-j}(z)\rangle < 0, \tag{12}$$

where λ is defined in (11). Since c and z lie in the same layer $\langle c, \mathbf{1}\rangle = k$, the last inequality is equivalent to (interpreting indices modulo n)

$$z_{n+j-0}T_0(c) + z_{n+j-1}T_1(c) + \ldots + z_{n+j-(n-1)}T_{n-1}(c) + 1 < 0.$$

Informally, we may say that for at least one permutation of subscripts, we have

$$z_0 T_0(c) + z_1 T_1(c) + \ldots + z_{n-1}T_{n-1}(c) + 1 < 0. \tag{13}$$

It should be mentioned that, for almost all $z \in \mathbb{Z}^n$ the constraint $\langle z, \overline{T}(c)\rangle < 0$ is nonlinear and non-convex with respect to c.

Define

$$H(z) = z_0 T_0(c) + z_1 T_1(c) + \ldots + z_{n-1}T_{n-1}(c) + 1,$$

so that $H(z) = 0$ is the equation of a hyperplane in $\mathbb{R}^n$ with non-zero normal vector $(T_0(c), T_1(c), \ldots, T_{n-1}(c))$ which is perpendicular to the fixed space (Lemma 3.5). (If all $T_j(c) = 0$, then $\widehat{T}(c)$ in Lemma 3.3 is a multiple of $\mathbf{1}$. But this makes $\operatorname{Cir}(c)^{-1}$ non-invertible). More precisely, $c \in \mathbb{Z}^n_{(k)}$ is a core point if for all integer points $z \in \mathbb{Z}^n_{(k)}$ there is an index $j \in [n]$ such that $H(\sigma^j(z)) < 0$.

Suppose Q is the orbit polytope of a non-core point and R is the orbit polytope of a universal core point in the same layer. Intuitively, we expect that many integer points whose orbit polytope contains Q, also have orbit polytope containing R. The idea for making new constraints is to remove from the feasible region all integer points whose orbit polytopes contain atoms or universal core points. This process can be done by searching layer by layer in the feasible region.

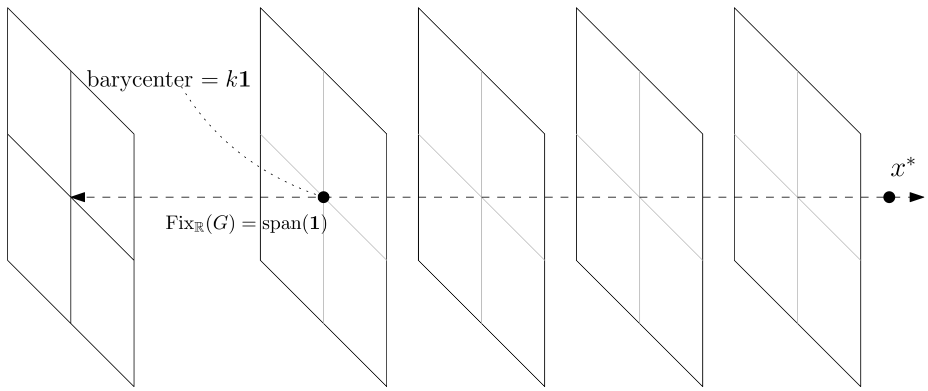

Figure 1: Illustrating the proof of Lemma 3.9.

Lemma 3.8. *For two co-projective integer points c,$c' \in \mathbb{Z}^n$ we have*

$$T_j(c) = T_j(c') \text{ for all } j = 0, \dots, n-1.$$

Moreover, the constraints (13) *are invariant under translation in the fixed space. That is, if z and z' are two co-projective integer points in the same non-zero layers as c and c' respectively, we have*

$$\langle z, \overline{T}(c)\rangle = \langle z', \overline{T}(c')\rangle.$$

Proof. Recall from Definition 2.6 that $c' = c + k\mathbf{1}$ for some $k \in \mathbb{Z}$. Since $\mathbf{1}$ is orthogonal to the other eigenvectors y^m, $m = 1, \dots, n-1$, we have

$$\begin{aligned} \psi_m &= \sqrt{n}\langle c, y^m\rangle = \sqrt{n}\langle c + k\mathbf{1}, y^m\rangle = \sqrt{n}\langle c', y^m\rangle \\ &= \psi'_m \quad \forall m = 1, \dots, n-1 \\ \psi_0 &= \langle c, \mathbf{1}\rangle, \quad \psi'_0 = \langle c', \mathbf{1}\rangle = \langle c, \mathbf{1}\rangle + \langle k\mathbf{1}, \mathbf{1}\rangle. \end{aligned}$$

Furthermore $\mathrm{Cir}(c)$ and $\mathrm{Cir}(c')$ have different inverses. But by Lemma 3.3 $T_j(c) = \sum_{m=1}^{n-1} w_n^{-jm}\psi_m^{-1}$. Since $\psi_m = \psi'_m$ we get $T_j(c) = T_j(c')$, $j = 0, \dots, n-1$. Now let $z \in \mathbb{Z}^n$ and $z' = z + k\mathbf{1}$ be in the same layers as c and c' respectively. With the help of Lemma 3.5 we have

$$\begin{aligned} n\langle z, \overline{T}(c)\rangle &= z_0T_0(c) + z_1T_{n-1}(c) + \dots + z_{n-1}T_1(c) + 1 \\ &= z_0T_0(c) + \dots + z_{n-1}T_1(c) + k(T_0(c) + \dots + T_{n-1}(c)) + 1 \\ &= (z_0 + k)T_0(c) + (z_1 + k)T_{n-1}(c) + \dots + (z_{n-1} + k)T_1(c) + 1 \\ &= z'_0T_0(c) + z'_1T_{n-1}(c) + \dots + z'_{n-1}T_1(c) + 1 = n\langle z', \overline{T}(c')\rangle. \qquad \square \end{aligned}$$

The main idea of our algorithm for the transitive case (all variables active) is as follows.

Lemma 3.9. *Let P be an integer linear maximization problem with n variables, objective function $f(x) = \langle \mathbf{1}, x \rangle$ and transitive symmetry group G. Let the LP relaxation of P have feasible region R and optimal solution x^*. Let P_j be the integer feasibility problem obtained by intersecting R with layer j. Let $l = \lfloor f(x^*) \rfloor$. Define $F = \{l - n < j \leqslant l \mid P_j$ is feasible$\}$. If F is empty, then P is infeasible; otherwise the optimal solution is in layer $k = \max(F)$.*

Proof. In the following discussion, we use *above* and *below* to refer to the natural ordering of layers by index. Since we assume G acts by permuting coordinates, there is no real loss of generality in fixing the objective function to $f(x) = \langle \mathbf{1}, x \rangle$. As we already observed, since G is transitive $\mathrm{Fix}_{\mathbb{R}}(G) = \mathrm{Span}(\mathbf{1})$. It is clear that layer l is the highest layer which could intersect R; furthermore if F is non-empty the optimal solution of P must lie in the highest layer with non-empty intersection with R. Suppose F is empty, so no layer above $l - n$ is feasible.

Define the *orbit-barycenter* $b(x)$ for $x \in \mathbb{R}^n$ as

$$b(x) = \frac{1}{|G|} \sum_{g \in G} gx.$$

Note that this point lies in $\mathrm{Fix}_{\mathbb{R}}(G)$ and $\langle \mathbf{1}, x \rangle = \langle \mathbf{1}, b(x) \rangle$; furthermore if $x \in R$ then $b(x) \in R$. This allows us to assume w.l.o.g. that $x^* \in \mathrm{Fix}_{\mathbb{R}}(G)$.

There exists integer k satisfying $l - n < l' = kn \leqslant l$. The integer point $k\mathbf{1}$ is the unique intersection of layer l' and $\mathrm{Fix}_{\mathbb{Z}}(G)$, so since $P_{l'}$ is infeasible, R does not intersect layer l'. Convexity then tells us that no layer below l' intersects R (as the segment of $\mathrm{Fix}_{\mathbb{R}}(G)$ between such a layer and x^* goes through $k\mathbf{1}$). □

Note that in order to check if an integer point $c \in \mathbb{Z}^n_{(k)}$ is a core point or not it is impossible to add constraints of the form of (13) for all $z \in \mathbb{Z}^n_{(k)}$ because there are infinitely many integer points in each layer. But we can add constraints for a finite subset of integer points in $\mathbb{Z}^n_{(k)}$. As mentioned earlier, it makes sense to choose atom points and universal core points since they are the closest integer points to the barycenter. For this finite set we have the following definition.

Definition 3.10. *In each layer k, any choice of set of atoms and universal core points for making inequalities* (13) *is called an* essential set *of the layer k and is denoted by E^k.*

Note that since vectors in $\{0, 1\}^n$ are core points, a non-empty choice of essential set always exists.

Definition 3.11. *We say that layers l and l' in $\mathbb{Z}^n$ are* congruent *if $l \equiv l' \pmod{n}$.*

Lemma 3.5 plays an important role in defining the specific essential set for any layer.

Remark 3.12. *We can translate each integer point $z \in E^k$ through the fixed space to get an integer point with entries in $\{-1, -2, 0, 1, 2\}$ and use inequality* (13)*, which can be written as*

$$1 + \langle z, (T_0(c), T_{n-1}(c), \dots, T_1(c)) \rangle < 0. \tag{14}$$

Since $T_0(c) + T_1(c) + \dots + T_{n-1}(c) = 0$, the inequality (14) *holds for both or neither co-projective points z, z' in congruent layers.*

For example, for an ILP in $\mathbb{R}^6$, we can define an essential set in layer 21 by:

$$E^{21} = \{(4, 4, 4, 3, 3, 3), (4, 3, 4, 4, 3, 3), (4, 3, 4, 3, 4, 3), (5, 3, 4, 3, 3, 3)\}.$$

The corresponding constraint for $z = (4, 4, 4, 3, 3, 3)$ is

$$1 + 4T_0(c) + 4T_5(c) + 4T_4(c) + 3T_3(c) + 3T_2(c) + 3T_1(c) < 0. \tag{15}$$

Since by Lemma 3.5 we have $T_0(c) + T_1(c) + T_2(c) + T_3(c) + T_4(c) + T_5(c) = 0$, inequality (15) can be written as

$$1 + T_0(c) + T_5(c) + T_4(c) < 0.$$

Furthermore, we can define the essential set as follows and use inequality (14) for making new constraints.

$$E^3 = \{(1, 1, 1, 0, 0, 0), (1, 0, 1, 1, 0, 0), (1, 0, 1, 0, 1, 0), (2, 0, 1, 0, 0, 0)\}.$$

If we consider layers $1, \dots, n$ as representatives for all congruence classes, we can define the essential set for layers $1, \dots, n$ and use universal or atom points with entries $\{-2, -1, 0, 1, 2\}$.

Definition 3.13. *The essential set in any layers $1, \dots, n$ is called the* projected essential set *and it is denoted by $\widehat{E}^k$ for $k = 1, \dots, n$.*

The idea is that first we search for integer points whose circulant matrix is singular. Next we consider the case where the circulant matrix is non-singular: by adding new constraints in each layer k we search for an integer point c whose orbit polytope does not contain the integer points of the essential set E^k. Since $\mathrm{Cir}(c)$ is non-singular in this step, we add the following constraints (cf. Theorem 3.4 and (7), (8)):

$$\langle V_m, c \rangle^2 + \langle U_m, c \rangle^2 > 0 \qquad \forall m = 1, \dots, \lceil (n-1)/2 \rceil. \tag{16}$$

We will present variations on this idea for different kinds of symmetry groups. In all of these variations there are three different types of subproblems as follows:

(Q_1) Add constraints (13) and (16) for each point in the projected essential set.

(Q_2) Add constraints (9) and (10) for the case of singular circulant matrices.

Algorithm 1 Maximize with all variables active

Input Maximization ILP P with bounded relaxation, transitive symmetry group, and objective function f.

Output Optimal objective value, or $-\infty$ if infeasible.

1: Construct and solve a subproblem P_0 of type Q_2 by adding constraints (9) and (10) to P.

2: If P_0 has integer optimal solution $\hat{z}_1$, set $f_1 = f(\hat{z}_1)$ otherwise set $f_1 = -\infty$.

3: Solve the LP relaxation of P, let x^* be the optimal solution and let $l = \lfloor f(x^*) \rfloor$, $l' = \lfloor l/n \rfloor n$ (cf. Figure 1).

4: **for** $i = l$ down to $l' + 1$ **do**

5:     **for** $j = 1, \ldots, m_i$ **do**

6:         Construct and solve subproblem P_i^j of type Q_3 for checking the feasibility of $z^j \in E^i = \{z^1, \ldots, z^{m_i}\}$. ▷ *Unprojected essential set.*

7:         **if** P_i^j has integer feasible solution $\hat{z}_2$ **then**

8:             **return** $\max\{f_1, f(\hat{z}_2)\}$

9:     Construct and solve a subproblem P_i of type Q_1 by adding constraints (13) for each $z^j \in \widehat{E}^i$ along with constraints (16) for $0 \leqslant m \leqslant \lceil (n-1)/2 \rceil$. ▷ *Projected essential set*

10:     **if** P_i has integer feasible solution $\hat{z}_3$ **then**

11:         **return** $\max\{f_1, f(\hat{z}_3)\}$

12: **return** $-\infty$

(Q_3) Check the feasibility of integer points in the essential sets. For Algorithm 1 they can be tested individually, for Algorithms 2 and 3 see Remark 4.5.

Algorithm 1 follows the proof of Lemma 3.9 for the case where all variables are active. Here are some observations related to Algorithm 1:

1. It is enough to make just λ_0 negative (see (12)), because if for a core point c, λ_0 is not negative, there is a permutation of c that has negative λ_0.

2. There might be some non-core integer points which satisfy constraints (13) for the given essential set.

3. By Lemma 3.5, in each layer L we need at most $\lceil (n-1)/2 \rceil$ terms T_k, to make constraints (13). For example if $n = 6$, in layer 5 for universal core point $(1,1,1,1,1,0)$ we can use $1 - T_5 < 0$ rather than $1 + T_0 + T_1 + T_2 + T_3 + T_4 < 0$.

The subproblems in steps 1 and 9 of Algorithm 1 are NP-hard, so there are no known worst case efficient algorithms. On the other hand, the cost of generating constraints, solving the LPs, and testing points for feasibility is all negligible. Actually, solving a symmetric LP can be done faster since it is enough to search through the fixed space (cf. [2, Theorem 1]). Since adding new constraints in steps 1 and 9 creates a nonlinear and non-convex mixed integer program, we should use a solver that can handle these constraints. In theory solving a nonlinear integer program is harder than the linear one, but in practice we see that (cf. Section 7), decreasing the feasible region by adding our constraints make it worthwhile to do so. It should also be mentioned that there is a trade off between the number of integer points in the essential set and running time of solving an ILP. Adding too many constraints might make the problem hard to solve. Some universal or atom points might be more effective for a specific problem. However, in our experiments there is no significant difference between different selected integer points in the essential sets. We summarize this discussion as follows.

Remark 3.14.

1. *The main cost of Algorithm 1 is in steps 1 and 9 where nonlinear integer programs must be solved.*
2. *The choice of essential set (as long as not too large) seems not to have a large impact.*

3.3 Example: solving a symmetric ILP with Algorithm 1

In this section, we illustrate using Algorithm 1 to solve a small ILP. Each of the subproblems discussed here is solved on commodity hardware in at most two seconds using Knitro [3]; for more details on the experimental setup see Section 7. The integer linear program *P1* with cyclic symmetry group C_5 is defined [11] as follows.

$$\text{minimize } x_0 + x_1 + x_2 + x_3 + x_4$$

subj. to

$$\begin{aligned}
515161x_0 + 18376x_1 - 503804x_2 - 329744x_3 + 300011x_4 &\leqslant 60\\
300011x_0 + 515161x_1 + 18376x_2 - 503804x_3 - 329744x_4 &\leqslant 60\\
-329744x_0 + 300011x_1 + 515161x_2 + 18376x_3 - 503804x_4 &\leqslant 60\\
-503804x_0 - 329744x_1 + 300011x_2 + 515161x_3 + 18376x_4 &\leqslant 60\\
515161x_4 + 18376x_0 - 503804x_1 - 329744x_2 + 300011x_3 &\leqslant 60\\
x_0 + x_1 + x_2 + x_3 + x_4 &= 2\\
x_0, x_1, x_2, x_3, x_4 &\in \mathbb{Z}
\end{aligned}$$

The group C_5 has a 1-dimensional fixed space generated by $\mathbf{1} = (1, 1, 1, 1, 1)$ and two 2-dimensional real invariant subspaces $\{V_1, U_1\}$ and $\{V_2, U_2\}$ where

$$\begin{aligned}
V_1 &= (1, \cos(\frac{-2\pi}{5}), \cos(\frac{-4\pi}{5}), \cos(\frac{-6\pi}{5}), \cos(\frac{-8\pi}{5}))\,,\\
V_2 &= (1, \cos(\frac{-4\pi}{5}), \cos(\frac{-8\pi}{5}), \cos(\frac{-12\pi}{5}), \cos(\frac{-16\pi}{5}))\,,\\
U_1 &= (0, \sin(\frac{-2\pi}{5}), \sin(\frac{-4\pi}{5}), \sin(\frac{-6\pi}{5}), \sin(\frac{-8\pi}{5}))\,,
\end{aligned}$$

$$U_2 = (0, \sin(\frac{-4\pi}{5}), \sin(\frac{-8\pi}{5}), \sin(\frac{-12\pi}{5}), \sin(\frac{-16\pi}{5}))\,.$$

Now if $c = (c_0, c_1, c_2, c_3, c_4)$ is an integer solution of the above ILP then the corresponding T_i in Theorem 3.4 are as below:

$$\begin{aligned}
T_0 &= \frac{\langle V_1, c\rangle}{\langle V_1, c\rangle^2 + \langle U_1, c\rangle^2} + \frac{\langle V_2, c\rangle}{\langle V_2, c\rangle^2 + \langle U_2, c\rangle^2}\,,\\
T_1 &= \frac{\langle \sigma^{-1}(V_1), c\rangle}{\langle V_1, c\rangle^2 + \langle U_1, c\rangle^2} + \frac{\langle \sigma^{-1}(V_2), c\rangle}{\langle V_2, c\rangle^2 + \langle U_2, c\rangle^2}\,,\\
T_2 &= \frac{\langle \sigma^{-2}(V_1), c\rangle}{\langle V_1, c\rangle^2 + \langle U_1, c\rangle^2} + \frac{\langle \sigma^{-2}(V_2), c\rangle}{\langle V_2, c\rangle^2 + \langle U_2, c\rangle^2}\,,\\
T_3 &= \frac{\langle \sigma^{-3}(V_1), c\rangle}{\langle V_1, c\rangle^2 + \langle U_1, c\rangle^2} + \frac{\langle \sigma^{-3}(V_2), c\rangle}{\langle V_2, c\rangle^2 + \langle U_2, c\rangle^2}\,,\\
T_4 &= \frac{\langle \sigma^{-4}(V_1), c\rangle}{\langle V_1, c\rangle^2 + \langle U_1, c\rangle^2} + \frac{\langle \sigma^{-4}(V_2), c\rangle}{\langle V_2, c\rangle^2 + \langle U_2, c\rangle^2}\,.
\end{aligned}$$

Since all variables are active we can use Algorithm 1 to solve the problem. Note that since the problem is in layer 2, we do not need to solve an LP relaxation problem to determine layers.

First we choose the essential set in layer 2 as $E^2 = \{(1, 1, 0, 0, 0)\}$. By adding the following nonlinear constraint,

$$1 + T_0(c) + T_1(c) < 0,$$

we quickly (via Knitro) see the subproblem is infeasible. Checking the feasibility of (1,1,0,0,0) is trivial. By adding the following constraints (see (9) and (10)) we check if there is an integer solution that has a singular circulant matrix.

$$\begin{aligned}
\langle V_1, c\rangle^2 + \langle U_1, c\rangle^2 &\leqslant r_1(\langle V_1, c\rangle^2 + \langle U_1, c\rangle^2)\\
\langle V_2, c\rangle^2 + \langle U_2, c\rangle^2 &\leqslant r_2(\langle V_2, c\rangle^2 + \langle U_2, c\rangle^2)\\
-2r_3 \leqslant c_0 + c_1 + c_2 &+ c_3 + c_4 \leqslant 2r_3\\
r_1 + r_2 &+ r_3 \leqslant 2
\end{aligned}$$

The final subproblem was also solved quickly by Knitro, and no integer solution was found.

4 Partial-Circulant Matrices

In the previous section we assumed that the order of the cyclic permutation group of an ILP was equal to the dimension of the problem. In this section, we generalize the algorithm of the previous section for some ILP where not all variables are active.

For the remainder of this section, we denote by $x(1)$ the coordinates of x that are active in the cyclic group C_k. We extend this notation below to direct products of several cyclic groups. Without loss of generality, assume $x(1)$ is the first $k < n$ coordinates of x. Notice that in this case the group action is not transitive. Indeed, the dimension of the fixed space is $n - k + 1$ and it is spanned by the orthogonal vectors $e_0 + \cdots + e_{k-1}, e_k, \ldots, e_{n-1}$ (using the usual basis vectors). Moreover, invariant subspaces of the action of C_k on $\mathbb{R}^k$ embed naturally into invariant subspaces for the action of C_k on $\mathbb{R}^n$. The following definition is a generalization of the circulant matrix that will be useful in generalizing the constraints of the previous section.

Definition 4.1. *For $c \in \mathbb{R}^n$, an $n \times k$* Partial-Circulant Matrix, $\mathrm{PCir}(c)$, *is a matrix where the first k rows are* $\mathrm{Cir}(c(1))$ *and the remaining rows are $\mathbf{1}^T$ scaled by the last $n - k$ elements of c.*

Example 4.2. *Let $c = (1, 2, 3, 4, 5) \in \mathbb{Z}^5$. Then the 5×3 partial circulant matrix of c takes the following form*

$$\begin{bmatrix} 1 & 3 & 2 \\ 2 & 1 & 3 \\ 3 & 2 & 1 \\ 4 & 4 & 4 \\ 5 & 5 & 5 \end{bmatrix}.$$

Let $c \in \mathbb{Z}^n$ and suppose the circulant part of $\mathrm{PCir}(c)$ is invertible. Then the orbit polytope of c has dimension $k - 1$ and any point $x \in \mathbb{R}^n$ in $\mathrm{conv}(G_c)$ can be written as

$$\mathrm{PCir}(c)_{n\times k}\lambda = x \quad \text{for some} \quad \lambda \in \mathbb{R}^k. \tag{17}$$

The rank of this partial-circulant matrix is k and so the solution λ of the system (17) can be determined by the circulant part.

Suppose the permutation $g \in S_n$ factors as the product $g = h_1 \cdot h_2 \cdots h_d$ of disjoint cycles h_j. Let $X_1, \ldots, X_d$ be the canonically associated subspaces of $\mathbb{R}^n$. Thus the cyclic group $H_j = \langle h_j \rangle$ permutes standard basis vectors for X_j. Furthermore, each integer point $z \in \mathbb{Z}^n$ has a unique decomposition $z = \oplus_{j=1}^{d} z(j)$ with $z(j) \in X_j \cap \mathbb{Z}^n$. Keeping this notation, we have that the next lemma follows from the definition of a convex combination.

Lemma 4.3. *Suppose $G = \langle g \rangle \leqslant S_n$, where $g = h_1 \cdot h_2 \cdots h_d$, is a product of disjoint cycles. Let $c \in \mathbb{Z}^n$. If the integer point $z = \oplus_{j=1}^{d} z(j)$ is in* $\mathrm{conv}(G_c)$*, then for $j \in 1, \ldots, d$, we have $z(j)$ in the orbit polytope of $c(j)$ under H_j.*

In this section we are only need the restricted case of $d = 1$ of Lemma 4.3: if the orbit polytope of $c(1) = (c_0, \ldots, c_{k-1})$ in $\mathbb{Z}^k$ is lattice-free then $c = (c_0, \ldots, c_{n-1})$ is a core point in $\mathbb{Z}^n$. Moreover, since the sum of weights is 1 in a convex combination, if $z \in \mathbb{Z}^n$ is in $\mathrm{conv}(G_c)$ then $z_{k+1} = c_{k+1}$, ..., $z_n = c_n$. Furthermore, applying inequality (13), if z does not lie in $\mathrm{conv}(G_c)$, then for at least one cyclic permutation of $z(1)$ the following constraint must be satisfied

$$z_0 T_0(c) + z_1 T_{k-1}(c) + \ldots + z_{k-1} T_1(c) + 1 < 0. \tag{18}$$

Recall that the above constraint is nonlinear and non-convex with respect to c.

For a given symmetric (maximization) ILP where not all variables are active in the cycle, let the optimal solution of its relaxation be x^*. In this case the symmetry group does not act transitively. So, searching in k layers is not sufficient and the optimal objective value can be in any layer less than or equal to $\lfloor f(x^*) \rfloor$. Nonetheless we can define a finite set of subproblems where we can apply the constraints developed in Section 3. In the following discussion, we use *sub-layer* j to refer to the set of integer points whose first k coordinates sum to j.

Lemma 4.4. *Given an ILP P with a cyclic symmetry group $G = \langle (1, 2, \dots, k) \rangle \leqslant S_n$ acting on the first k coordinates,*

1. *P can be solved by solving k subproblems created by adding the constraint*

$$\sum_{j=0}^{k-1} x_j = i \pmod{k} \quad \textit{for } i \in \{1 \dots k\} \,. \tag{19}$$

2. *We can use the constraints* (13) *in each subproblem.*

3. *Each of the subproblems can be modelled by adding a single integer variable and a single integer linear constraint to P.*

Proof. Let $L = l + l^*$ be the layer of an integer point $z \in \mathbb{Z}^n$ in P where l is the layer of active variables and l^* is the layer of non-active variables. The point z must fall into one of the k congruence classes defined by (19), which establishes the first point.

By Lemma 3.8 all congruent sub-layers l have the same constraints for co-projective integer points (see Definition 2.6 and Definition 3.11). Indeed congruence is not needed here, only that co-projective points form equivalence classes for (13).

Although the constraint (19) is nonlinear, it is easy to model using integer-linear constraints. Consider the congruence relation between layers (see Definition 3.11). Let $i = 1, \dots, k$ be the representative of each class. Let q_i, be an integer solver variable. Constraint (19) can be reformulated as

$$\sum_{j=0}^{k-1} x_j = q_i k + i \,. \qquad \square$$

Algorithm 1 in the previous section can be modified in this case as Algorithm 2 below. The difference is we should check sub-layers $l_i = q_i k + i$ where $q_1, \dots, q_k$ are some new integer variables (rather than a fixed layer). For example, if $k = 3$, then all layers in $\mathbb{Z}^3$ can be classified as

$$l_1 = 3q_1 + 1, \; l_2 = 3q_2 + 2, \text{ or } l_3 = 3q_3 + 3.$$

Note that since the algorithm for this case is not searching layer by layer, checking the feasibility of integer points in the essential set is different. In other words, we know the layers of points we have (since by Remark 3.12 the integer points in each projected essential set are in the layers $1, \ldots, k$) but they are only representatives for (many) pre-images, and we don't know the layers of the pre-images. If $z(1) = \sigma^i(c(1))$ then the solution of $\mathrm{Cir}(c(1))\lambda = z(1)$ is e_a, that is:

$$\lambda_a = 1, \quad \lambda_i = 0 \quad \forall i = 0, \ldots, k-1,\ i \neq a,$$

where $a \in \{0, \ldots, k-1\}$. Furthermore, by Theorem 3.4, to check if at least one of the pre-images of $z(1)$ in the projected essential set is a feasible point or not, for $a = 0$, we can add the following constraints:

$$\begin{aligned} \langle \sigma^0(z(1)), \overline{T}(c(1)) \rangle &= 1, \\ \langle \sigma^1(z(1)), \overline{T}(c(1)) \rangle &= 0, \\ &\vdots \\ \langle \sigma^{k-1}(z(1)), \overline{T}(c(1)) \rangle &= 0. \end{aligned}$$

Remark 4.5. *Since the coordinates of integer points in the projected essential set are 0,1,-1,2,-2, another way to check the feasibility of all translates along the fixed space (which uses only linear constraints with small coefficients) is to chose one of the coordinates c_i as a base and write other coordinates with respect to that coordinate.*

For example if $z = (1, -2, 0, 0, 0, 0, 0) \in \widehat{E}^1$, then by choosing c_2 as a base we can add the following constraints to check the feasibility of all integer points $(1, -2, 0, 0, 0, 0, 0) + t(1, 1, 1, 1, 1, 1, 1), t \in \mathbb{Z}$, in the sub-layers $7t + 1$:

$$\begin{aligned} c_0 &= c_2 + 1, \\ c_1 &= c_2 - 2, \\ c_3 &= c_2, \\ c_4 &= c_2, \\ c_5 &= c_2, \\ c_6 &= c_2 . \end{aligned}$$

Algorithm 2 on the next page generalizes Algorithm 1 to the case where $k < n$ coordinates are active in the symmetry group. For conciseness, we write $\mathrm{OPT}_\infty(P)$ for the optimal objective value of maximization problem P. If P is infeasible, $\mathrm{OPT}_\infty(P)$ is defined as $-\infty$. In practice this can be implemented by a Boolean flag, and does not require any specialized arithmetic.

As discussed in Remark 3.14, giving precise complexity bounds for Algorithm 2 is difficult. But it is worthwhile to compare the complexity of Algorithm 1 and Algorithm 2. In step 1 of of both algorithms the same constraints are used but the difference is that, in the second algorithm there are some non-active coordinates. So the subproblem in this case is harder to solve since

Algorithm 2 Maximize with first k variables active.

Input Maximization ILP P with bounded relaxation, cyclic symmetry group acting on the first k coordinates, and objective function f.

Output Optimum objective value, or $-\infty$ if P is infeasible.

1: Construct and solve a subproblem P_0 of type Q_2 by adding constraints (9) and (10) to P. Set f^* to $\mathrm{OPT}_\infty(P_0)$.
2: **for** $i = 1 \dots k$ **do**
3: Construct a subproblem L_i by adding constraint $\sum_{j=0}^{k-1} x_j = q_i k + i$ where q_i is a new integer variable.
4: Choose projected essential set $\widehat{E}^i = \{z^1, \dots, z^{m_i}\}$.
5: **for** $j = 1, \dots, m_i$ **do**
6: Construct and solve subproblem P_i^j of type Q_3 by adding constraints described in Remark 4.5 to L_i for point in $z^j \in \widehat{E}^i$
7: $f^* \leftarrow \max\{f^*, \mathrm{OPT}_\infty(P_i^j)\}$.
8: Construct a subproblem P_i of type Q_1 by adding to L_i constraint (16) for all $m = 0, \dots, \lceil (n-1)/2 \rceil$ and constraints (13) for each $z^j \in \widehat{E}^i$.
9: $f^* \leftarrow \max\{f^*, \mathrm{OPT}_\infty(P_i)\}$.
10: **return** f^*

constraints are effective on a small dimension of the problem. The type of constraints in step 8 of Algorithm 2 and step 9 of Algorithm 1 is the same but there are two differences: 1) less coordinates are again active in the second algorithm, and 2) in the first algorithm in each subproblem is in a single layer, but in the second algorithm in each subproblem is solved over an equivalence class of sub-layers. Both of these factors make the type Q_1 subproblems harder to solve in Algorithm 2. The other important fact is that, in the first algorithm since we are searching layer by layer, the first integer solution (if there is any) is an optimal solution. But in the second algorithm since all coordinates are not active if an integer solution is found it might not be optimal. On the other hand, when using Algorithm 2 for feasibility problems the subproblems can also be solved as feasibility problems, using local solvers if desired. Finally, checking the feasibility of universal and atom core points in these two algorithms is done differently. In the first algorithm in each subproblem, it is just testing an integer point against a set of linear inequalities. But in the second algorithm since not all coordinates are active, in each subproblem an ILP should be solved after adding constraints described in Remark 4.5.

5 New Constraints for Direct Products of Cyclic Groups

Lemma 4.4 shows that Algorithm 2 can be used for a cyclic subgroup of the symmetric group of an ILP. In some cases the cyclic subgroup is small and Algorithms 1 and 2 are not very practical. In this section we generalize Algorithm 2 for direct products of cyclic groups. Like Algorithm 2, the algorithm in this section also does not search layer by layer in the feasible region. Instead, we search in all equivalence classes of sub-layers.

Recall that the Cartesian product of the d sets $X_1, \dots, X_d$ is

$$X_1 \times \dots \times X_d = \prod_{i=1}^{d} X_i = \{(x_1, \dots, x_d) : \; x_i \in X_i \text{ for every } i \in \{1, \dots, d\}\}.$$

Definition 5.1. *Let $G_i, i = 1, \dots, d$ be some finite groups. The* direct product *$G_1 \times \dots \times G_d$ is defined as follows.*

1. *The underlying set is $G_1 \times \dots \times G_d$.*
2. *Multiplication is defined coordinate-wise:*

$$(g_1, \dots, g_d) \cdot (g'_1, \dots, g'_d) = (g_1.g'_1, \dots, g_d.g'_d).$$

3. *The identity element of this group is defined as $(e_1, \dots, e_d)$ where e_j is the identity element of G_j for $j = 1, \cdots, d$.*

It is routine to check that with the above operation, $\prod_{i=1}^{d} G_i$ is a group. Consider a product $g = h_1 \cdot h_2 \cdots h_d$ of disjoint cycles in S_n, and let $H_i = \langle h_i \rangle$.

Lemma 5.2. *The direct product $G = H_1 \times \cdots \times H_d$ is isomorphic to $G' = \langle h_1, \dots, h_d \rangle$.*

Proof. Let $g_i = h_i^{t_i}$ denote a general element of the cyclic group H_i. Consider the following map between G and G'

$$\phi : G \to G'$$
$$(g_1, \dots, g_d) \to g_1 g_2 \cdots g_d,$$

It is straightforward to check ϕ is a bijection since the cycles are disjoint (g_i's commute since the h_i's are disjoint). Also we have

$$\phi((g_1, \dots, g_d) \cdot (g'_1, \dots, g'_d)) = \phi(g_1 g'_1, \dots, g_d g'_d) = g_1 g'_1 \dots g_d g'_d =$$
$$g_1 \dots g_d g'_1 \dots g'_d = \phi(g_1, \dots, g_d)\phi(g'_1, \dots, g'_d).$$

So ϕ is a group isomorphism. Note that the second to last equality holds since the cycles are disjoint. $\square$

Example 5.3. *The permutation group* $G = \langle(1,2,3,4),(5,6,7)\rangle$ *on the set* $\{1,2,3,4,5,6,7\}$ *is isomorphic to the direct product of two cyclic groups* $H_1 = \langle(1,2,3,4)\rangle$ *and* $H_2 = \langle(5,6,7)\rangle$

In this section we denote by $z(i)$ the coordinates of z that are active in the i-th cyclic group in the direct product $H_1 \times \ldots \times H_d$.

The following theorem states that if a permutation group $G \leqslant S_n$ is a direct product of other permutation groups G^i, (that is, permutation groups G^i acting on disjoint subsets of coordinates in the usual way), then the core set of G is also a Cartesian product.

Theorem 5.4. [8, Theorem 8] *Let* $G = \prod_{i=1}^{d} G^i$, $G^i \leqslant S_n$. *Then*

$$\operatorname{core}(G) = \prod_{i=1}^{d} \operatorname{core}(G^i).$$

Proof. The product structure of G induces a decomposition of $\mathbb{R}^n$ into a Cartesian product of pairwise orthogonal coordinate subspaces $\bigoplus_{i=1}^{d} X_i = \mathbb{R}^n$. Thus, we can write every point $z \in \mathbb{R}^n$ as $z = \bigoplus_{i=1}^{d} z(i)$. The claim of the theorem follows immediately from $\operatorname{conv}(G_z) = \prod_{i=1}^{d} \operatorname{conv}(G^i_{z(i)})$. □

We are concerned with various subgroups of the direct product $\prod_{j=1}^{d} H_j$, where the cyclic groups $H_j = \langle h_j \rangle$ are generated by disjoint cycles $h_1, \ldots, h_d$ in S_n. Let k_j denote the length (period) of the cycle h_j. Thus, if $g = h_1 \cdots h_d$ is the complete factorization of the permutation $g \in S_n$ into disjoint (possibly trivial) cycles, then $k_1 + \cdots k_d = n$.

Note that we can use the same $\widehat{T}(c)$ as in Lemma 3.3 for each cycle of the direct product groups. In other words, let $c = \bigoplus_{i=1}^{d} c(i) \in \mathbb{Z}^n$. When required, we will suppose that the vector c has the property that the matrices $\operatorname{Cir}(c(j))$ are invertible, $1 \leqslant j \leqslant d$. Lemma 3.3 then tells us the corresponding inverse matrices are $\operatorname{Cir}(\widehat{T}(c(j)))$. Then a point $x = \bigoplus_{i=1}^{d} x(i) \in \mathbb{R}^n$ is in $\operatorname{conv}(G_c)$ if and only if

$$\begin{aligned} x(1) = \operatorname{Cir}(c(1))\lambda^1 &\Rightarrow \lambda^1 = \operatorname{Cir}(\widehat{T}(c(1)))x(1)\,, \\ &\vdots \\ x(d) = \operatorname{Cir}(c(d))\lambda^d &\Rightarrow \lambda^d = \operatorname{Cir}(\widehat{T}(c(d)))x(d)\,. \end{aligned}$$

Furthermore, by Theorem 5.4 we can apply our new constraints on each subspace.

We denote by l_i^j the sub-layer of active variables of an integer point in the j-th cyclic group and in the i-th equivalence class, and denote by $\widehat{E}_i^j$ the corresponding projected essential set of each sub-layer l_i^j.

Similarly to the previous section, all equivalence classes of all sub-layers must be checked. By Lemma 3.8, Remark 3.12 and Lemma 5.2, in each cyclic group H_i we need to search in k_i sub-layers. Then there are $k_1 \times \ldots \times k_d$ possibilities for sub-layers of an integer point. For example suppose $G = \langle(1,2,3)\rangle \times \langle(4,5)\rangle \leqslant S_5$. Then we have decomposition $\mathbb{Z}^n = X_1 \times X_2$ where $z(1) = (z_0, z_1, z_2)$ and $z(2) = (z_3, z_4)$. So we have the following possibilities for sub-layers of a feasible solution z

$$\begin{aligned}
l_1^1 &= 3q_1 + 1, \;\; l_1^2 = 2q_2 + 1,\\
l_2^1 &= 3q_1 + 2, \;\; l_1^2 = 2q_2 + 1,\\
l_3^1 &= 3q_1 + 3, \;\; l_1^2 = 2q_2 + 1,\\
l_1^1 &= 3q_1 + 1, \;\; l_2^2 = 2q_2 + 2,\\
l_2^1 &= 3q_1 + 2, \;\; l_2^2 = 2q_2 + 2,\\
l_3^1 &= 3q_1 + 3, \;\; l_2^2 = 2q_2 + 2,
\end{aligned}$$

where q_1 and q_2 are arbitrary integers.

Consider a cycle h_j. Since its length is k_j, a complete set of residues modulo k_j is the set $[k_j] := \{1, \ldots, k_j\}$. We require their Cartesian product

$$K = [k_1] \times \ldots \times [k_d] = \{(t_1, \ldots, t_d) : \; t_j \in [k_j] \; 1 \leqslant j \leqslant d\}.$$

For each element $(t_1, \ldots, t_d)$ of $K \subset \mathbb{Z}^n$, a subproblem of type Q_1 is solved, where $t_j, \; j = 1, \ldots, d$, is the sub-layer of active variables of cycle h_j.

Remark 5.5. *In the direct product group $G = \prod_{i=1}^{d} H_i$, each sub-layer has its own projected essential set. In step 6 of Algorithm 2 the feasibility of pre-images of integer points of the projected essential sets is checked. In the direct product case we can check the feasibility of these pre-images corresponding to each cycle separately. If $s_j := \sum_{i=1}^{k_j} |\widehat{E}_i^j|$, then $\sum_{j=1}^{d} s_j$ subproblems must be solved.*

On the other hand if adding constraints for one cycle is not enough, we can check for simultaneous feasibility. Let $(t_1, \ldots, t_d) \in K$ be a vector of residues (where K is defined above). Given points $z(j) \in \widehat{E}_{t_j}^j$, $j = 1 \ldots d$, we can check simultaneous feasibility of $z(1), \ldots, z(d)$. In this case, $\Pi_{j=1}^{d} s_j$ subproblems must be solved.

For example if $G = C_5 \times C_8$ and a universal core point in the projected essential set $\widehat{E}_2^1$ is $(1,1,0,0,0)$, by Remark 4.5, we add the following constraints to the problem:

$$\begin{aligned}
x_1 &= x_0\,,\\
x_2 &= x_0 - 1\,,\\
x_3 &= x_0 - 1\,,\\
x_4 &= x_0 - 1\,.
\end{aligned}$$

We can also check all possible combinations of integer points in the projected essential sets. For example if $G = \langle(1,2,3,4)\rangle \times \langle(5,6,7)\rangle \leqslant S_7$, for l_2^1 and l_1^2 let $\widehat{E}_2^1 = \{(1,1,0,0), (1,0,1,0)\}$ and $\widehat{E}_1^2 = \{(1,0,0), (2,-1,0)\}$. Then all possible combinations of $\widehat{E}_2^1$ and $\widehat{E}_1^2$ are as below:

$$(1,1,0,0,1,0,0), (1,1,0,0,2,-1,0), (1,0,1,0,1,0,0), (1,0,1,0,2,-1,0).$$

Now we can generalize Algorithm 2 for direct product groups as shown in Algorithm 3. For simplicity we present the version which tests the essential points individually for feasibility; the modification of the loop at step 5 for simultaneous testing is straightforward.

Algorithm 3 Maximization for ILPs with direct product cyclic symmetry

Input maximization ILP P with symmetry group $G = \Pi_{j=1}^{d} H_j$, bounded relaxation, objective function f.

Output The optimal objective value, or $-\infty$ if P is infeasible.

1: $f^* \leftarrow -\infty$
2: Let k_j be the order of cyclic group H_j.
3: **for** $t = (t_1, \ldots, t_d) \in [k_1] \times \ldots \times [k_d]$ **do**
4:     Choose projected essential sets $\widehat{E}_t = \bigcup\{\, \widehat{E}^1_{t_1}, \ldots, \widehat{E}^d_{t_d} \,\}$.
5:     **for** $z \in \widehat{E}_t$ **do**
6:         Construct and solve subproblem P^z_t of type Q_3 by adding constraints described in Remark 4.5 to P for point z.
7:         $f^* \leftarrow \max\{f^*, \mathrm{OPT}_\infty(P^z_t)\}$

    Construct and solve a subproblem P_t by adding to P:
8:     **for** $a = 1, \ldots, d$, **do**
9:         **if** $t_a = k_a$ **then**
10:             Constraints (9) and (10). ▻ *constraint type* Q_2.
11:         **else**
12:             Constraints (16) for $0 \leqslant m \leqslant \lceil (n-1)/2 \rceil$ and constraints (13) for each z in $\widehat{E}_t$. ▻ *constraint type* Q_1.
13:     $f^* \leftarrow \max\{f^*, \mathrm{OPT}_\infty(P_t)\}$
14: **return** f^*

Remark 5.6. *Compared to Algorithm 2, both the number of subproblems and their difficulty are potential bottlenecks in Algorithm 3. While the steps are similar to the Algorithm 2, the running time of each subproblem is longer. In particular in each subproblem of step 6 we either have constraints acting on a small set of coordinates (if we check each subgroup individually), or a very large number of subproblems (if we check for simultaneous feasibility).*

6 New Constraints For Permutation Groups

In this section we use Algorithms 3, 2 or 1 for a symmetric ILP with some permutation group as its symmetry group. First we define subdirect product groups and then classify the permutation groups with respect to their generators.

Definition 6.1. *Let H_1 and H_2 be groups. A* subdirect product *of H_1 and H_2 is a subgroup H of the external direct product $H_1 \times H_2$ such that the projection from H to either direct factor is surjective. In other words, if $p_1 : H_1 \times H_2 \to H_1$ is given by $(h_1, h_2) \mapsto h_1$ and $p_2 : H_1 \times H_2 \to H_2$ is given by $(h_1, h_2) \mapsto h_2$, then $p_1(H) = H_1$ and $p_2(H) = H_2$. Subdirect products of more than two groups are obtained naturally by iterating this construction.*

Example 6.2. *The group $\langle(1,2)(3,4,5,6)\rangle$ is a subdirect product of $\langle(1,2)\rangle$ and $\langle(3,4,5,6)\rangle$ and is a subgroup of the direct product $\langle(1,2)\rangle \times \langle(3,4,5,6)\rangle$.*

Remark 6.3. *Suppose $g \in G \leqslant S_n$ is factored as a product of disjoint cycles. These cycles themselves may or may not be elements of G. But Lemma 4.3 shows that for exploring feasible integer points of a symmetric polyhedron considering some of the cycles is enough. In other words, we can make our constraints for a few cycles and keep other variables non-active.*

Note that our constraints are still valid for a subdirect product of disjoint cycles, since in each sub-problem we remove universal or atom points (by generating constraints from their projections) from the orbit polytope of active variables of each cycle. So Algorithms 2 and 3 work for this case as well.

Now we classify permutation groups with respect to their generators.

Disjoint Cycles If G is a group generated by disjoint cycles $h_1, \ldots, h_d$, then G is the direct product of $H_1, .., H_d$ (Lemma 5.2). In this case Algorithm 3 can be applied.

Example 6.4. *The group $G = \langle(1,2,3,4),(6,8,9),(5,7,12,11)\rangle$ is the direct product of the three cyclic groups $H_1 = \langle(1,2,3,4)\rangle$, $H_2 = \langle(6,8,9)\rangle$, $H_3 = \langle(5,7,12,11)\rangle$.*

Product of disjoint cycles If the given set of generators of a group G is a single permutation which is the product of two or more disjoint cycles, then G is a subdirect product of the corresponding cyclic groups. In this case by Remark 6.3, Algorithm 3 can be applied.

Example 6.5. *The subdirect product group $G = \langle(1,2,5,3)(6,10,11)(9,7,12,8)\rangle$ is a subgroup of the direct product of three cyclic groups $H_1 = \langle(1,2,5,3)\rangle$, $H_2 = \langle(6,10,11)\rangle$, $H_3 = \langle(9,7,12,8)\rangle$.*

Combination of two of the above cases If the generators of G are products of various numbers of disjoint cycles, then Algorithm 3 can be applied.

Example 6.6. *Let $G = \langle(1,2,3,4)(5,6,12,13),(7,8,9,10)\rangle$, then G is a subgroup of the direct product of three cycles $H_1 = \langle(1,2,3,4)\rangle$, $H_2 = \langle(5,6,12,13)\rangle$, $H_3 = \langle(7,8,9,10)\rangle$.*

Non-disjoint cycles If the given set of generators of G are not disjoint cycles, we find a subgroup of G where all generators are disjoint. Then it falls into one of the previous three cases. Note that for finding the biggest cyclic subgroup we can find representatives of the conjugacy classes (e.g. by using GAP [6]) and choose the biggest cycle.

Example 6.7. *Let* $G = \langle(1, 2, 3, 4, 5), (7, 5, 8, 10, 11)(12, 13, 2, 14)\rangle$ *then* $H_1 = \langle(1, 14, 12, 13, 2, 3, 4, 8, 10, 11, 7, 5)\rangle$ *is a cyclic subgroup and since 9 and 6 are fixed, Algorithm 2 can be applied.*

7 Computational Experiments

In order to test the efficiency of our algorithms, in this section we create some symmetric integer linear programs that are hard to solve with standard solvers. For this purpose, we can use the orbit polytopes of core points. As the orbit polytope of a core point contains no integer points aside from the vertices, if we can cut off these vertices then the integer program corresponding to this polytope will be infeasible.

Infeasible problems are typically hard for branch-and-bound algorithms because there is no chance of early success. The goal of the *integer feasibility* problems is to find an integer point in a polyhedron P or decide that no such point exists. Aaarden and Lenstra presented some difficult integer feasibility problems [1, 5] whose relaxations are simplices. Our techniques are not suitable for the examples of [1, Table 1] because these simplices lack symmetries in the sense of Remark 2.2. Symmetric instances are interesting in their own right because equivalent partial solutions can blow up the branch-and-bound-tree (cf. [14]). In order to construct symmetric examples that are still integer infeasible, we start with lattice-free symmetric simplices and cut off the (integer) vertices; a similar technique is used in [15].

The convex hull of the orbit of a core point under a cyclic group with an invertible $n \times n$ circulant matrix is a simplex with dimension $n - 1$. We say an integer point has a *globally minimal projection* (with respect to some invariant subspace V) if

$$\|z|_V\| \leqslant \|z'|_V\| \text{ for all } z' \in \operatorname{aff}(G_z) \bigcap \mathbb{Z}^n.$$

For a primitive group the following theorem shows that the corresponding orbit polytope is often a simplex.

Theorem 7.1. [15, Theorem 5.37] *Let* $G \leqslant S_n$ *be primitive and let* $V \leqslant \mathbb{R}^n$ *be a rational invariant subspace. If* $e_0 = (1, 0, 0, \ldots, 0)$ *has globally minimal projection onto* V*, then there are infinitely many core points in layer one. The corresponding orbit polytopes are simplices.*

The groups used for these experiments are the primitive groups with GAP identifiers $(15, 2)$, $(21, 2)$, $(45, 1)$ (all three of them have rational invariant subspaces) along with the cyclic group C_5. C_5 has only trivial rational invariant

Table 1: ILP-feasibility problems.

Name	Symmetry group	Generators of the group
P1	Cyclic group C_5	$(1,2,3,4,5)$
P2	Primitive group (15,2)	$(1,15,7,5,12)(2,9,13,14,8)(3,6,10,11,4)$, $(1,4,5)(2,8,10)(3,12,15)(6,13,11)(7,9,14)$
P3	Primitive group (21,2)	$(1,7,12,16,19,21,6)(2,8,13,17,20,5,11)\cdot$ $(3,9,14,18,4,10,15)$, $(4,6,5)(9,11,10)(13,15,14)(16,18,17)(19,20,21)$
P4	Primitive group (45,1)	$(1,2,7)(3,11,27)(4,14,31)(5,18,32)(6,20,36)\cdot$ $(8,24,39)(9,25,28)(10,26,42)(12,15,16)(13,30,40)\cdot$ $(17,19,21)(22,35,44)(23,33,29)(34,43,37)\cdot$ $(38,45,41)$, $(1,3,5,6,7,22,13,23)(2,8,9,10)\cdot$ $(4,15,16,17,14,21,19,12)\cdot$ $(11,28,29,38,44,25,20,37)\cdot$ $(18,33,34,24,40,36,41,39)\cdot$ $(26,43,35,32,42,45,27,30)$, $(1,4)(3,12)(5,19)(6,21)(7,14)(8,10)(11,20)\cdot$ $(13,16)(15,23)(17,22)(18,33)(24,41)(25,28)\cdot$ $(26,43)(27,32)(29,44)(30,35)(34,39)(36,40)(42,45)$

subspaces; it is nonetheless easy to find integer points whose orbit under C_5 is a 4-simplex. Table 1 gives more details about symmetry groups of these instances.

As it was mentioned before, these problems are hard to solve with standard solvers. They take more than 3600 seconds to be solved in Gurobi 8.1, CPLEX 12.10 and GLPK 4.6 on an Intel Core-i5 machine with CPUs at 1.4 GHz and 8 GB RAM. Since adding new constraints creates a nonlinear and non-convex MILP, we should use a solver that can handle these constraints. We used Knitro to solve these instances. Knitro has three algorithms for mixed-integer nonlinear programming (MINLP):

1. Nonlinear branch-and-bound method.
2. A mixed-integer Sequential Quadratic Programming (MISQP) method.
3. The hybrid Quesada-Grossman method (for convex problems).

We used the first method above (branch-and-bound method) to solve these problems. This method involves solving a relaxed, continuous nonlinear optimization subproblem at every node of the branch-and-bound tree. It can also be applied to non-convex models. However it is a *local* solver and hence may sometimes get stuck at integer feasible points that are not globally optimal solutions when the model is non-convex. In addition, the integrality gap measure may not be accurate since this measure is based on the assumption that the nonlinear optimization subproblems are always solved to global optimality (which may not be the case when the model is non-convex). But local solvers can be used for solving feasibility problems (checking if the problem has a feasible integer solution or not).

For some instances we used both Algorithms 2 and 3 to compare results. We solved all of them on an Intel Core-i5 machine with CPUs at 1.4 GHz and 8 GB RAM. As Table 2 shows the total time of Algorithm 2 is less than the total time of Algorithm 3 in all instances.

Table 2: Total times in seconds and number of subproblems of Algorithm 1, 2 and 3

Name	Al	Cycle	TTs	NSP
P1	1	$(1,2,3,4,5)$	3.1	3
P2	2	(1,15,7,5,12)	17	11
P2	3	$(1,15,7,5,12)$ $(2,9,13,14,8)$	97	36
P3	2	$(1,7,12,16,19,21,6)$,	15	14
P4	2	$(11,28,29,38,44,25,20,37)$	335	36
P4	3	$(11,28,29,38,44,25,20,37)$, $(18,33,34,24,40,36,41,39)$	566	91

Al: Algorithm; NSP: Number of subproblems
TTs: Total time in seconds.

Table 3 shows the total time and number of subproblems of type Q_1, Q_2 and Q_3. The number of subproblems of type Q_1 corresponding to cycle h_i in Algorithm 2 is k_i (we have to check k_i sub-layers) and in Algorithm 3 is $k_1 \times \ldots \times k_d$. The number of subproblems of type Q_3 is the total number of integer points of the projected essential sets. Recall that a projected essential set contains a small subset of universal or atom points in each layer.

Table 3: Total times in seconds and number of subproblems of type Q_j

Name	Al	TT(s) of SP's Q_1	TT(s) of SP's Q_2	TT(s) of SP's Q_3	# Q_1	# Q_2	# Q_3
P1	1	1.987	1.113	0.008	1	1	1
P2	2	5	3	9	4	1	6
P2	3	62	20	15	16	8	12
P3	2	10	3	2	6	1	7
P4	2	50	5	280	7	1	28
P4	3	540	23	280	49	14	28

Al: Algorithm; TT(s): Total time in seconds; SP's : subproblems

As Table 3 shows Algorithm 2 is more efficient for our instances. Algorithm 3 might work better in the case that the symmetry group does not have a cycle which is big enough to make the problem easier to solve. All of our instances could be solved by using one cycle. Problem P4 (primitive group 45-2) we used at most four integer points in each sub-layer in Algorithm 2. For other problems we used at most two integer points in each sub-layer.

8 Conclusions

In this paper we introduced some new techniques for solving symmetric linear programs based on deriving nonlinear constraints from the symmetry group of the formulation. Since these constraints depend only on the symmetry group, the same constraints can be re-used for many problems. The practical benefits of using a nonlinear solver in order have a smaller search space need more evaluation, but at least on the artificial instances in Section 7 they show some promise, allowing the fast solution of instances not solvable within 1 hour on the same hardware using commercial MILP solvers. We think our methods may be useful for problems with a large enough cyclic subgroup in their symmetry group, particularly those where determining integer feasibility is a challenge. From a theoretical point of view, these techniques show that core point techniques are not limited to groups where the number of (non-equivalent) core-points is finite.

Acknowledgements

The authors wish to thank two anonymous referees for many helpful suggestions, including the simplified proof of Lemma 3.5 that is given here. They would also like to thank Barry Monson, Branimir Ćaćic and Nicholas Touikan for helpful feedback on earlier versions of the paper.

References

[1] Karen Aardal and Arjen K. Lenstra. Hard equality constrained integer knapsacks. *Math. Oper. Res.*, 29(3):724–738, 2004.

[2] Richard Bödi, Katrin Herr, and Michael Joswig. Algorithms for highly symmetric linear and integer programs. *Math. Prog.*, 137(1-2):65–90, 2013.

[3] Richard H Byrd, Jorge Nocedal, and Richard A Waltz. KNITRO: An integrated package for nonlinear optimization. In *Large-scale nonlinear optimization*, pages 35–59. Springer, 2006.

[4] Philip J. Davis. *Circulant Matrices*. Wiley, New York, 1979.

[5] Liyan Gao and Yin Zhang. Computational experience with Lenstra's algorithm. Technical Report TR02-12, Rice University, 2002. `https://hdl.handle.net/1911/101992`.

[6] The GAP Group. *GAP – Groups, Algorithms, and Programming, Version 4.11.1*, 2021.

[7] Robert M. Gray. Toeplitz and circulant matrices: A review. *Foundations and Trends® in Communications and Information Theory*, 2(3):155–239, 2006.

[8] Katrin Herr, Thomas Rehn, and Achill Schürmann. Exploiting symmetry in integer convex optimization using core points. *Operations Research Letters*, 41(3):298–304, 2013.

[9] Christopher Hojny and Marc E Pfetsch. Polytopes associated with symmetry handling. *Math. Prog.*, 175(1-2):197–240, 2019.

[10] Lingchen Huang, Huazi Zhang, Rong Li, Yiqun Ge, and Jun Wang. AI coding: Learning to construct error correction codes, 2019. Preprint. `https://arxiv.org/abs/1901.05719`.

[11] Frieder Ladisch and Achill Schürmann. Equivalence of lattice orbit polytopes. *SIAM J. Applied Algebra and Geom.*, 2(2):259–280, 2018.

[12] Sanja Petrovic and Carole Fayad. A genetic algorithm for job shop scheduling with load balancing. In Shichao Zhang and Ray Jarvis, editors, *AI 2005: Advances in Artificial Intelligence*, pages 339–348, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[13] Marc Pfetsch. Symmetry handling in MIPs using SCIP. `https://www2.mathematik.tu-darmstadt.de/~pfetsch/symmetries.html`, visited July 2020.

[14] Marc E Pfetsch and Thomas Rehn. A computational comparison of symmetry handling methods for mixed integer programs. *Math. Prog. Comput.*, 11(1):37–93, 2019.

[15] Thomas Rehn. *Exploring Core Points for Fun and Profit — A Study Of Lattice-free Orbit Polytopes*. PhD thesis, Universität Rostock, 2014.