On the Power of Symmetric Linear Programs

Albert Atserias[1] Anuj Dawar[2] Joanna Ochremiak[3]

Universitat Politècnica de Catalunya[1]
University of Cambridge[2]
University of Bordeaux, CNRS, LaBRI[3]

March 12, 2021

Abstract

We consider families of symmetric linear programs (LPs) that decide a property of graphs (or other relational structures) in the sense that, for each size of graph, there is an LP defining a polyhedral lift that separates the integer points corresponding to graphs with the property from those corresponding to graphs without the property. We show that this is equivalent, with at most polynomial blow-up in size, to families of symmetric Boolean circuits with threshold gates. In particular, when we consider polynomial-size LPs, the model is equivalent to definability in a non-uniform version of fixed-point logic with counting (FPC). Known upper and lower bounds for FPC apply to the non-uniform version. In particular, this implies that the class of graphs with perfect matchings has polynomial-size symmetric LPs while we obtain an exponential lower bound for symmetric LPs for the class of Hamiltonian graphs. We compare and contrast this with previous results (Yannakakis 1991) showing that any symmetric LPs for the matching and TSP polytopes have exponential size. As an application, we establish that for random, uniformly distributed graphs, polynomial-size symmetric LPs are as powerful as general Boolean circuits. We illustrate the effect of this on the well-studied planted-clique problem.

1 Introduction

The theory of linear programming is a powerful and widely-used tool for studying combinatorial optimization problems. By the same token, the limitations of such methods are an important object of study in complexity theory. A major step in this line of work was the seminal paper of Yannakakis [47] that initiated the study of *symmetric* linear programs for combinatorial problems.

A polytope in $\mathbb{R}^n$ is the convex hull of a finite set of points in $\mathbb{R}^n$. Dually, it is the intersection of the finite number of half-spaces that define its facets. Consider a language $S \subseteq \{0,1\}^*$ and let $S_n \subseteq \{0,1\}^n$ be the collection of strings in S of length n. We can associate with S_n the polytope $P_n \subseteq \mathbb{R}^n$ that is the convex hull of the points $\mathbf{x} \in \mathbb{R}^n$ with 0-1 coordinates that correspond to the strings in S_n. If this polytope has a succinct representation as a system of linear inequalities, we can use linear programming methods to optimize linear functions over S_n. In general, a succinct representation might mean that its size grows polynomially with n. Thus, the size of the polytope P_n, say measured by the number of its facets, is an important measure of the complexity of S.

In general, even when P_n has a large number of facets, it may admit a succinct representation as the projection onto $\mathbb{R}^n$ of a polytope $Q \subseteq \mathbb{R}^{n+m}$ of higher dimension. In this situation, we call Q a *lift* of P_n and P_n a *shadow* of Q. This is the basis for so-called *extended formulations* of combinatorial optimization problems. It allows us to optimize over S_n using linear programs with auxiliary variables. A classic example is the convex hull of all strings in $\{0,1\}^n$ of odd Hamming weight, known as the *parity polytope* which has exponentially many facets but has an extended formulation using only polynomially many inequalities. An interesting feature of many such examples of small extended formulations is that they are strongly symmetric, i.e., any basic automorphism of the shadow polytope extends to an automorphism of its lift.

Yannakakis [47] established lower bounds on the size of symmetric lifts for the perfect matching polytope and the travelling salesman polytope. The *perfect matching polytope* on $2n$ vertices is the convex hull of points in $\{0,1\}^E$ where $E = \binom{[2n]}{2}$ which represent the edge sets of a perfect matching on $2n$ vertices. Yannakakis shows that any symmetric lift Q of this polytope necessarily has a number of facets that is exponential in n. Here "symmetric" means that any permutation of the n vertices extends to an automorphism of Q. This lower bound is then used to show a similar lower bound for symmetric lifts of the Hamilton cycle polytope (also known as the travelling salesman polytope). This is the convex hull of points in $\{0,1\}^E$ where $E = \binom{[n]}{2}$ which are the edge sets of Hamilton cycles of length n. The conclusion is that any attempt to solve the travelling salesman problem by representing it as a linear program in a natural way (i.e. respecting the symmetries of the graph) is doomed to be exponential. These results launched a long study of extended formulations of combinatorial problems. Relatively recently, exponential lower bounds have been established even without the assumption of symmetry [39].

There is another way of representing a language $S \subseteq \{0,1\}^*$ by a family of polytopes that is also considered by Yannakakis. Say that S_n is *recognized* by a polytope P_n if $S_n \subseteq P_n$ and $\{0,1\}^n \setminus S_n$ is disjoint from P_n. In particular, the convex hull of S_n recognizes S_n, but it may well be that there are more succinct polytopes that also do. Indeed, Yannakakis

shows that for any language S decidable in polynomial time, there is a family of polynomial-size polytopes whose shadows recognize S_n. Thus, we cannot expect to prove exponential lower bounds on such polytopes without separating P from NP. Note that the assumption of symmetry has been dropped here. What can we say about symmetric lifts of polytopes recognizing S_n? Yannakakis does not consider this question and it does not appear to have been studied in the literature. This is the question that we take up in this paper.

We consider families of symmetric polytopes for recognizing classes of graphs (or other relational structures). This gives an interesting contrast with the results of Yannakakis. Our results show that there *is* a polynomial-size family of symmetric polytopes whose shadows recognize the class of graphs that contain a perfect matching. On the other hand, there is no family of symmetric polytopes of sub-exponential size whose shadows recognize the class of graphs with a Hamiltonian cycle.

We obtain these specific upper and lower bounds by relating the power of symmetric linear programs to two other natural models of symmetric computation, based respectively on logic and circuits. To be precise, we show that families of symmetric polytope lifts for recognizing a class of structures are equivalent to families of *symmetric* Boolean circuits with threshold gates, in the sense that there are translations between them with at most a polynomial blow-up in size in either direction. This places symmetric linear programs squarely in the context of a fairly robust notion of symmetric computation that has recently emerged. It was shown in [3] that P-uniform families of symmetric circuits with threshold gates are equivalent to fixed-point logic with counting (FPC), a well-studied logic in descriptive complexity theory (see [16]).

Our translation from circuits to linear programs is based on that given by Yannakakis, but we need to preserve symmetry. To construct symmetric linear programs that enforce the values of threshold gates we use the so-called *cardinality indicating polytopes*. Small and symmetric LP lifts for these polytopes were constructed by Pashkovich [38]. In the other direction, we make a detour through logic. That is, we show how a family of symmetric polytopes can be translated into a family of formulas of first-order logic with counting, with the number of variables and the size being tightly bounded based on the size of the polytopes. The translation is based on a support theorem, which allows us to interpret in the logic, given a linear program P as advice, a version of P for a particular input structure. This then allows us to use the result of [4] to the effect that solvability of linear programs is definable in FPC.

It is interesting to compare our results with the equivalence between FPC and P-uniform symmetric threshold circuits established in [3]. Our results are stated for the non-uniform model and it is not clear that they can be made uniform. In particular, our translation from linear programs to formulas of counting logic, while it preserves size, is not necessarily computable in polynomial time. It involves symmetry checks that are as hard as the graph isomorphism problem. On the other hand, the results in [3] were stated for polynomial-size families of circuits and we are able to extend them to sizes up to weakly exponential. The translation from circuits to formulas given in [3] was based on a support theorem proved there which only worked for circuit sizes bounded by $O(2^{n^{1/3}})$. We use a stronger support theorem (proved in Section 4.2) which enables us to prove the translation from families of symmetric linear programs to formulas of counting logic for sizes up to $O(2^{n^{1-\epsilon}})$ for arbitrarily small ϵ.

The upper and lower bounds for symmetric linear programs that we obtain (such as for the perfect matching and the Hamilton cycle problem, respectively) are direct consequences of the equivalence with non-uniform counting logic. For instance, it is known [4] that perfect matching is definable in FPC and it follows that it is recognized by a polynomial-size family of symmetric polytope lifts. Inexpressibility results for FPC are usually established by showing lower bounds on the number of variables required to express a property in counting logic, and they yield lower bounds even in the non-uniform setting. In particular, we tighten known lower bounds on Hamiltonicity to show that it cannot be expressed with a sub-linear number of variables hence with weakly exponential size symmetric polytope lifts. Indeed, if we use the strongest form of our translation from symmetric LPs to logic formulas, then the lower bound we get for Hamiltonicity is even exponential, i.e., of type $2^{\Omega(n)}$ where n is the number of vertices of the graph. Similar exponential lower bounds for other NP-complete problems (such as graph 3-colourability and Boolean satisfiability) follow from known bounds in counting logic. Indeed, exponential lower bounds for some problems in P (such as solving systems of linear equations over finite fields) also follow. It should be noted that this establishes exponential lower bounds also on symmetric threshold circuits, a problem left open in [3], where superpolynomial lower bounds were established.

Another consequence can be derived from the connection with FPC. We know that FPC can express all polynomial-time properties of *almost all structures* under a uniform distribution (see [25]). This can be used to show that FPC can solve the planted clique problem if, and only if, the problem is solvable in polynomial time. The planted clique problem is that of distinguishing a random graph from one in which a clique has been planted. It is a widely studied problem in the context of lower bounds on linear programming methods (see e.g. [1, 21, 9, 28]). It is a consequence of our results that if this problem can be solved in polynomial time, then it is solvable by polynomial sized symmetric linear programs. This is significant because a number of lower bounds have been established for the planted clique problem for a variety of models of linear and semidefinite programming, notably the well-studied Lovász-Schrijver, Sherali-Adams and Lasserre hierarchies. It is noteworthy that all of these hierarchies yield symmetric linear or semidefinite programs. Our results show that these lower bounds cannot be extended to general symmetric linear programs without separating P from NP.

In Section 2 we establish some preliminary definitions and notation. Section 3 gives the translation of circuits to linear programs. This translation is carried out for a very general notion of symmetry. For the reverse translation, from linear programs to logic given in Section 4, we restrict to the natural symmetries on graphs and relational structures. The main result and its consequences, including upper and lower bounds are presented in Section 5.

2 Preliminaries

For a natural number $n \in \mathbb{N}$, we write $[n]$ for $\{1, \ldots, n\}$ with the understanding that $[0] = \emptyset$. For any set X, by Sym_X we denote the *symmetric group on* X, that is, the group of all permutations of the set X, and by Alt_X we denote the *alternating group on* X, that is, the group of all even permutations of the set X. In the special case of $X = [n]$ we write Sym_n

and Alt_n, respectively. Logarithms are base 2 with the convention that $\log(0) = 0$.

2.1 Group actions

By $H \leq G$ we denote the fact that H is a subgroup of G. If $H \leq G$ then, for any $g \in G$, the subset $gH = \{gh : h \in H\}$ of G is called a *coset* of H in G. The number of such cosets is called the *index* of H in G and is denoted by $[G : H]$.

Recall that for any group G, a *G-set* is a set X with an action of the group G, where by an *action* we mean a mapping $\cdot : G \times X \to X$ such that for any $\pi, \sigma \in G$ and any $x \in X$ we have $\pi \cdot (\sigma \cdot x) = \pi\sigma \cdot x$ and $\mathrm{id} \cdot x = x$. Equivalently, X is a G-set if it comes with a homomorphism from the group G to the symmetric group Sym_X on X. A *homomorphism* from a G-set X to a G-set Y is a function f from X to Y such that for any $\pi \in G$ and $x \in X$, it holds that $\pi \cdot f(x) = f(\pi \cdot x)$. For a G-set X, the *stabilizer* of an element $x \in X$ in G consists of all $\pi \in G$ such that $\pi \cdot x = x$. It is easy to see that a stabilizer is a subgroup of G. We sometimes denote it by G_x.

For any set X, by $|X|$ we denote the number of elements in X, by $\mathcal{P}(X)$ we denote the power set of X and by X^n we denote the set of n-tuples of elements of X. Moreover, if $n \leq |X|$, by $X^{(n)}$ we denote the set of all n-tuples of *distinct* elements of X. In particular, for $n = 0$, both X^n and $X^{(n)}$ are one-element sets consisting of the empty tuple. If X is a G-set, then the action of G on X induces an action of G on each of the sets $\mathcal{P}(X)$, X^n and $X^{(n)}$ in the natural way: for any $\pi \in G$, we have $\pi \cdot T = \{\pi \cdot x : x \in T\}$, where $T \subseteq X$, and $\pi \cdot \mathbf{s} = (\pi \cdot s_1, \ldots, \pi \cdot s_n)$, where $\mathbf{s} = (s_1, \ldots, s_n)$ is a tuple from X^n or $X^{(n)}$. We refer to the latter group action as the *componentwise* action of the group G.

An action of a group G on a set of indices U defines an action of G on the set of indexed variables $\{x_u\}_{u \in U}$ in the natural way: $\pi \cdot x_u = x_{\pi \cdot u}$, for any $\pi \in G$ and $u \in U$. This extends to vectors of indexed variables, as discussed in the paragraph above. For instance, if the set of indices $[n]^2$ comes with the componentwise action of the group Sym_n, then for any $\pi \in \mathrm{Sym}_n$ and $\mathbf{x} = (x_{ij})_{i,j \in [n]}$, we have $\pi \cdot \mathbf{x} = (x_{\pi(i)\pi(j)})_{i,j \in [n]}$. From now on, in the case of vectors of variables we use the notation $\mathbf{x}^\pi$ instead of $\pi \cdot \mathbf{x}$.

For any G-set U, we define an action of G on the real vector space $\mathbb{R}^U$ in the following way. First, the action of G on the standard basis $\{\mathbf{e}_u\}_{u \in U}$, where $\mathbf{e}_u$ is the vector whose u-th coordinate is 1 and all other coordinates are 0, is given by $\pi \cdot \mathbf{e}_u = \mathbf{e}_{\pi \cdot u}$, for any $\pi \in G$ and $u \in U$. This way each $\pi \in G$ defines a mapping from $\{\mathbf{e}_u\}_{u \in U}$ to $\{\mathbf{e}_u\}_{u \in U}$. The action of G on the vector space spanned by $\{\mathbf{e}_u\}_{u \in U}$ can be seen as the linear extension of those mappings: for any $\pi \in G$ and any real vector $\mathbf{a} = \sum_{u \in U} a_u \mathbf{e}_u$, we have $\pi \cdot \mathbf{a} = \sum_{u \in U} a_u (\pi \cdot \mathbf{e}_u) = \sum_{u \in U} a_u \mathbf{e}_{\pi \cdot u}$. For instance, if the set of indices $[n]$ comes with the natural action of the group Sym_n, then for any $\pi \in \mathrm{Sym}_n$ and for any vector $\mathbf{a} = (a_1, \ldots, a_n)$, we have $\pi \cdot \mathbf{a} = \sum_{i \in [n]} a_i \mathbf{e}_{\pi(i)} = \sum_{i \in [n]} a_{\pi^{-1}(i)} \mathbf{e}_i = (a_{\pi^{-1}(i)}, \ldots, a_{\pi^{-1}(n)})$. Here again we use the notation $\mathbf{a}^\pi$ instead of $\pi \cdot \mathbf{a}$. This notational convention extends to subsets of real vector spaces: for $P \subseteq \mathbb{R}^U$ we write P^π instead of $\pi \cdot P$.

If a group G acts on a set U and a group H acts on a set W, then the product group $G \times H$ acts on the disjoint union $U \dot\cup W$: given $\pi \in G$ and $\sigma \in H$, we have $(\pi, \sigma) \cdot u = \pi \cdot u$, for $u \in U$, and $(\pi, \sigma) \cdot w = \sigma \cdot w$, for $w \in W$. Given such an action of the product group $G \times H$, of particular interest to us is its induced action on $\mathbb{R}^U \times \mathbb{R}^W$ and on sets of variables indexed

by $U \dot{\cup} W$.

2.2 Logic and structures

A (many-sorted relational) vocabulary consists of a finite set of sort symbols and a finite set of relation symbols. Each relation symbol R comes with an associated natural number $\mathrm{ar}(R)$ called its *arity* and with an associated *type* which is a product of $\mathrm{ar}(R)$-many sort symbols $U_{i_1} \times \ldots \times U_{i_{\mathrm{ar}(R)}}$. The vocabulary L_G of (directed) graphs is single-sorted, its single sort is denoted V, and has one relation symbol E of arity two. If L is a vocabulary, then an L-structure $\mathbb{A}$ is given by disjoint sets $U_1^{\mathbb{A}}, \ldots, U_s^{\mathbb{A}}$, called *domains*, one for each sort symbol U_i in L, and a relation $R^{\mathbb{A}} \subseteq U_{i_1}^{\mathbb{A}} \times \ldots \times U_{i_r}^{\mathbb{A}}$ for each $R \in L$ of arity r and type $U_{i_1} \times \ldots \times U_{i_r}$. The set $U^{\mathbb{A}}$ is called the domain of U in $\mathbb{A}$, and the relation $R^{\mathbb{A}}$ is called the interpretation of R in $\mathbb{A}$. Whenever this does not lead to confusion we use U to denote $U^{\mathbb{A}}$. Similarly, when $\mathbb{A}$ is clear from the context, we omit the superscript in $R^{\mathbb{A}}$. All our structures are finite: their domains are finite sets. A directed graph is an L_G-structure; the graph is undirected if its interpretation of E is symmetric and irreflexive. In the case of graphs, we write $V(G)$ for the domain of the single sort V of G, i.e., its set of vertices, and $E(G)$ for the interpretation of E, i.e., its set of edges.

In a logic for a many-sorted vocabulary L the variables are typed, that is, each different sort has its own set of individual variables. When an L-formula is interpreted on an L-structure, the variables range over the domain of their sort. The atomic L-formulas are equalities between variables of the same type, and formulas of the form $R(x_1, \ldots, x_r)$, where R is a relation symbol of arity r in L, and $x_1, \ldots, x_r$ are variables of appropriate types.

The class of formulas of first-order logic (FO) is the smallest class of formulas that contains all atomic formulas and is closed under negation, conjunction, and existential quantification. We consider an extension of first-order logic with *counting quantifiers*. For each natural number q, we have a quantifier $\exists^{\geq q}$ where $\mathbb{A} \models \exists^{\geq q} x\, \phi$ if, and only if, there are at least q distinct elements $a \in A$ such that $\mathbb{A} \models \phi[a/x]$. While the extension of first-order logic with counting quantifiers is no more expressive than FO itself, the presence of these quantifiers does affect the number of variables that are necessary to express a query. Let C^k denote the k-variable fragment of this logic, i.e. those formulas in which no more than k variables appear, free or bound. FPC is the extension of first-order logic with fixed-points and counting. We do not give a full definition here as it can be found in standard texts such as [36]. We note that formulas of FPC have two sorts of variables, ranging respectively over the elements of the domain of interpretation and over natural numbers (restricted to the size of the domain), and allow for terms of the form $\#x\phi$ which denotes the number of elements that satisfy $\phi(x)$. The syntax of FPC allows also the formation of *inflationary fixed-point* formulas $\mathrm{ifp}_{x,X}\varphi(x, X)$. On a structure $\mathbb{A}$ such formulas are interpreted as defining the fixed-point obtained by iterating the operator $A \mapsto A \cup \{a \in U_{i_1} \times \cdots \times U_{i_r} : \mathbb{A} \models \varphi(a, A)\}$, where $U_{i_1} \times \cdots \times U_{i_r}$ is the type of the relation symbol X in $\varphi(x, X)$. We write FOC to denote the fragment of FPC without fixed-point operators, but where we do allow arithmetic operations ($+$ and $\times$) on the number sort. The *size* of a formula is defined as the number of its subformulas. For each formula ϕ of FPC (and, per force, FOC), if the formula uses k variables then, for every n, there is a formula θ_n of C^{2k} such that ϕ is equivalent to θ_n on all structures of size at most n.

Moreover, θ_n has size that is polynomial in the size of ϕ, in k, and in n^k. For more on FPC and its relation to the bounded variable fragments C^k we refer to [36].

Rational numbers $q \in \mathbb{Q}$ are represented by structures of a single-sorted vocabulary $L_{\mathbb{Q}}$ with three monadic relation symbols and one binary relation symbol $\leq$. If $q = (-1)^b n/d$, where $n, d \in \mathbb{N}$ and $b \in \{0,1\}$, then the domain of an $L_{\mathbb{Q}}$-structure that represents q is $\{0,\ldots,N\}$ where $N \in \mathbb{N}$ is large enough to represent both the numerator and denominator with N bits. The binary relation $\leq$ is interpreted by the natural linear order on $\{0,\ldots,N\}$. The first of the monadic relation symbols of $L_{\mathbb{Q}}$ is used to represent the sign b of q by having it empty if, and only if, $b = 0$. The other two monadic relation symbols of $L_{\mathbb{Q}}$ are used to represent the bit positions on which the numerator n and the denominator d have a one. We use zero denominator to represent $\pm\infty$.

If $I_1,\ldots,I_d$ denote index sets, tensors $u \in \mathbb{Q}^{I_1\times\cdots\times I_d}$ are represented by many-sorted structures, with one sort $\bar{I}$ for each index set I on the list $I_1,\ldots,I_d$, and one sort $\bar{B}$ for a domain $\{0,\ldots,N\}$ of bit positions. The vocabulary $L_{\mathrm{vec},d}$ of these structures has a binary relation symbol $\leq$ for the natural linear order on $\{0,\ldots,N\}$ and three $d+1$-ary relation symbols P_s, P_n and P_d for encoding the signs and the bits of the numerators and the denominators of the entries of the tensor. Matrices $\mathbf{A} \in \mathbb{Q}^{I\times J}$ and vectors $\mathbf{a} \in \mathbb{Q}^I$ are special cases of these. Indexed sets of vectors $\{\mathbf{a}_i : i \in K\} \subseteq \mathbb{Q}^I$ and index sets of rationals $\{b_i : i \in K\} \subseteq \mathbb{Q}$ too.

2.3 Polytopes, lifts, and shadows

A polytope is a set of the form $P = \{\mathbf{x} \in \mathbb{R}^V : \mathbf{A}\mathbf{x} \leq \mathbf{b}\}$, where U and V are abstract non-empty index sets, $\mathbf{A} \in \mathbb{R}^{U\times V}$ is a constraint matrix, and $\mathbf{b} \in \mathbb{R}^U$ is an offset vector. If we think of $\mathbf{x} = (x_v)_{v\in V}$ as a sequence of variables, then the defining system of inequality constraints $\mathbf{A}\mathbf{x} \leq \mathbf{b}$ is called a *linear program* (LP) for P. Note that the defining LPs for polytopes are by no means unique. Typically $\mathbf{A}$ and $\mathbf{b}$ can be chosen to have rational entries, in which case P is represented by a sequence of linear constraints $(\gamma_u)_{u\in U}$ of one of its defining LPs with rational entries; i.e., each γ_u is of the form $\mathbf{a}_u^{\mathrm{T}}\mathbf{x} \leq b_u$, with $\mathbf{a}_u \in \mathbb{Q}^V$ and $b_u \in \mathbb{Q}$. The *size* of such an LP is $|U|(|V|+1)b$, where b is the maximum number of bits it takes to write all the numerators and all the denominators of the entries of the $\mathbf{a}_u$ and b_u in binary.

If $\mathbf{x} \in \mathbb{R}^V$ and $\mathbf{y}_1,\ldots,\mathbf{y}_m \in \mathbb{R}^V$, and $\alpha_1,\ldots,\alpha_m \in \mathbb{R}$ are such that $\mathbf{x} = \sum_{i=1}^m \alpha_i\mathbf{y}_i$, with $\alpha_i \geq 0$ and $\sum_{i=1}^m \alpha_i = 1$, then we say that $\mathbf{x}$ is a convex combination of $\mathbf{y}_1,\ldots,\mathbf{y}_m$. When $0 < \alpha_i < 1$ for some $i \in [m]$, the convex combination is called non-trivial. The convex hull $\mathrm{conv}(\mathbf{y}_1,\ldots,\mathbf{y}_m)$ is the set of all convex combinations of $\mathbf{y}_1,\ldots,\mathbf{y}_m$. A point $\mathbf{x}$ of a polytope P is called a *vertex* if it cannot be expressed as a non-trivial convex combination of any two other points of P. If P is a polytope and $P \subseteq \{\mathbf{x} \in \mathbb{R}^V : \mathbf{a}^{\mathrm{T}}\mathbf{x} \leq b\}$, then the set $\{\mathbf{x} \in P : \mathbf{a}^{\mathrm{T}}\mathbf{x} = b\}$ is called a *face* of P. The faces of dimension 0 are the vertices of P; the faces of dimension 1 are called *edges*; the faces of dimension $\dim(P)-1$ are called *facets*, where $\dim(P)$ is the dimension of P. Each polytope has finitely many faces of each dimension; in particular finitely many vertices (see [40]). A polytope is bounded if, and only if, it is the convex hull of its vertices.

If $P \subseteq \mathbb{R}^V \times \mathbb{R}^W$ is a polytope, its projection into $\mathbb{R}^V$ is the set of points $\mathbf{x} \in \mathbb{R}^V$ for

which there exists a point $\mathbf{y} \in \mathbb{R}^W$ with $(\mathbf{x}, \mathbf{y}) \in P$. The projection of a polytope is again a polytope. If Q is the projection of P into $\mathbb{R}^V$, then we say that Q is a *shadow* of P, and that P is a *lift* of Q. If $A, B \subseteq \{0,1\}^V$ are disjoint, then we say that Q is a polytope that separates A from B if $A \subseteq Q$ and $B \subseteq \mathbb{R}^V \setminus Q$. We also say that P is a polytope lift that separates A from B. If Q separates A from its complement $\overline{A} = \{0,1\}^V \setminus A$, then we say that Q is a polytope that recognizes A, and that P is a polytope lift that recognizes A. Since no point in $\{0,1\}^V$ is in the convex hull of any set of points in $\{0,1\}^V$ that does not contain it, the convex hull of $A \subseteq \{0,1\}^V$ always recognizes A. The converse is not true: a polytope could recognize A and not be the convex hull of A.

Let $P \subseteq \mathbb{R}^V$ be given by a sequence of linear constraints $(\gamma_u)_{u \in U}$. If a group G acts on the set V, then for any γ_u of the form $\mathbf{a}_u^{\mathrm{T}}\mathbf{x} \leq b_u$, we write γ_u^π for the linear constraint $\mathbf{a}_u^{\mathrm{T}}\mathbf{x}^\pi \leq b_u$. Note that the sequence $(\gamma_u^\pi)_{u \in U}$ defines $P^\pi \subseteq \mathbb{R}^V$, which is again a polytope. As long as this does not lead to confusion, we identify polytopes with sequences of linear constraints that represent them. In particular, if we assume the polytope $P \subseteq \mathbb{R}^V$ to be represented by the sequence of constraints $(\gamma_u)_{u \in U}$, then by P^π we mean both the permuted polytope itself, and its representation by the sequence of constraints $(\gamma_u^\pi)_{u \in U}$.

Let $P \subseteq \mathbb{R}^V \times \mathbb{R}^W$ be a polytope lift given by a sequence of linear constraints $(\gamma_u)_{u \in U}$ where each γ_u is of the form $\mathbf{a}^{\mathrm{T}}\mathbf{x} + \mathbf{b}^{\mathrm{T}}\mathbf{y} \leq c$, with $\mathbf{x} = (x_v)_{v \in V}$ and $\mathbf{y} = (y_w)_{w \in W}$. A pair of permutations $(\pi, \sigma) \in \mathrm{Sym}_V \times \mathrm{Sym}_W$ is called an *automorphism* of P if $\{\gamma_u^{(\pi,\sigma)}\}_{u \in U} = \{\gamma_u\}_{u \in U}$. If V is a G-set, then the polytope P is said to be *G-symmetric* if for every $\pi \in G$ there exists a permutation $\sigma \in \mathrm{Sym}_W$ such that $P^{(\pi,\sigma)} = P$. If additionally we are given an action of the group G on W such that $P^{(\pi,\pi)} = P$, for every $\pi \in G$, then we say that the polytope P is G-symmetric *with respect to this action*. Similarly, if for every $\pi \in G$ there exists a permutation $\sigma \in \mathrm{Sym}_W$ such that the sets of constraints $\{\gamma_u\}_{u \in U}$ and $\{\gamma_u^{(\pi,\sigma)}\}_{u \in U}$ are the same, then the linear program $(\gamma_u)_{u \in U}$ is called *G-symmetric*, and if we are given an action of the group G on W such that $\{\gamma_u^{(\pi,\pi)}\}_{u \in U} = \{\gamma_u\}_{u \in U}$, for every $\pi \in G$, then we say that the linear program $(\gamma_u)_{u \in U}$ is G-symmetric *with respect to this action*.

Note that a polytope represented by a G-symmetric linear program is G-symmetric. On the other hand, a G-symmetric polytope can be represented by a sequence of constraints that is not G-symmetric, however, for any G-symmetric polytope, there exists a G-symmetric linear program that defines it. In the following, whenever we talk about a G-symmetric polytope represented by some system of linear constraints, we implicitly assume that the system is G-symmetric. We define the *size* of a G-symmetric polytope to be the size of the smallest G-symmetric LP that represents it. We remark that this notion of size is different than the one used by Yannakakis in [47], where the size of a polytope is defined to be the size of its smallest LP description.

For any $n \in \mathbb{N}$, if the set $[n]^2$ comes with the natural action of the symmetric group Sym_n, then any Sym_n-symmetric polytope $P \subseteq \mathbb{R}^{[n]^2} \times \mathbb{R}^W$ is said to be *graph-symmetric*. It is not difficult to see that any set $A \subseteq \{0,1\}^{[n]^2}$ recognized by a graph-symmetric polytope lift $P \subseteq \mathbb{R}^{[n]^2} \times \mathbb{R}^W$ is invariant with respect to the action of the group Sym_n, i.e., for any $\mathbf{a} \in A$ and any $\pi \in \mathrm{Sym}_n$, we have $\mathbf{a}^\pi \in A$. The polytope lift P can be therefore seen as recognising a class of graphs with n-vertices. If we take a graph G with the set of vertices $V(G)$ of size n, fix a bijection f from $[n]$ to $V(G)$, and define a vector $\mathbf{a} = (a_{ij})_{i,j \in [n]} \in \{0,1\}^{[n]^2}$ by: $a_{ij} = 1$ if and only if there is an edge from $f(i)$ to $f(j)$ in G, then G belongs to the class recognised

by P if and only if $\mathbf{a} \in A$. Since A is a Sym_n-set, this definition does not depend on the choice of f.

More generally, we consider Sym_n-symmetric polytope lifts recognising properties of arbitrary structures. For any $n \in \mathbb{N}$ and any single-sorted vocabulary L, let $L(n)$ be the disjoint union of $[n]^{\mathrm{ar}(R)}$ over all relation symbols R in L. Since $L(n)$ comes with the natural action of the group $\mathrm{Sym}(n)$, we can talk about Sym_n-symmetric polytopes over $\mathbb{R}^{L(n)} \times \mathbb{R}^W$. Any such polytope is called *L-symmetric*. As a straightforward generalisation of the discussion in the previous paragraph, L-symmetric polytope lifts over $\mathbb{R}^{L(n)} \times \mathbb{R}^W$ are defined to recognise classes of L-structures with n-element domains.

2.4 Boolean circuits

A circuit with inputs $(x_v)_{v \in V}$ is a directed acyclic graph whose vertices of zero in-degree are labelled by some input x_v, and every other vertex is labelled by a function from some fixed basis of symmetric Boolean functions, with the constraint that the function takes the same number of inputs as the in-degree of the vertex. The vertices of zero out-degree are called outputs. If $(y_u)_{u \in U}$ is a fixed naming of the outputs, then a circuit computes a Boolean function $f : \{0,1\}^V \to \{0,1\}^U$ in the obvious way. When $m = 1$ we say that it recognizes $f^{-1}(1)$. The vertices of a circuit are also called *gates*. The *size* of a circuit is its number of gates and the number of *wires* of a circuit is the sum of the in-degrees of all the gates. A *Boolean threshold circuit* is one whose gates are labelled by NOTs, unbounded degree ANDs, unbounded degree ORs, or unbounded degree thresholds $\mathrm{TH}_{n,k}$, where $\mathrm{TH}_{n,k}(z_1, \ldots, z_n)$ outputs 1 if, and only if, the number of 1's in the input $z_1, \ldots, z_n$ is at least k.

If V is a G-set and W denotes the set of gates of C, we say that C is *G-symmetric* if for every $\pi \in G$ there exists $\sigma \in \mathrm{Sym}_W$ such that $C^{(\pi,\sigma)} = C$, where by $C^{(\pi,\sigma)}$ we mean that the gates of the circuit are permuted according to σ, the labels from $\{x_v\}_{v \in V}$ are permuted according to π, and none of the other labels is moved. A circuit with $V = L(n)$ is called L-symmetric if it is Sym_n-symmetric, with the natural action of Sym_n on $L(n)$. As for polytopes, we consider L-symmetric circuits as recognizing classes of L-structures $\mathbb{A}$ on abstract sets $V^{\mathbb{A}}$ of vertices through bijections $f : [n] \to V^{\mathbb{A}}$. In the case of graphs, for example, in which $L(n) = [n]^2$, we say that such a circuit accepts a graph G with the set of vertices $V(G)$ of size n if for some, and hence every, bijection $f : [n] \to V(G)$ it holds that $C(\mathbf{a}) = 1$, where $\mathbf{a}$ is the vector that describes the image of G under f^{-1}, as in the previous section.

3 From Circuits to LPs

In this section we prove the half of the equivalence that takes symmetric circuits with threshold gates into symmetric LPs. That is:

Lemma 1. *If $\mathscr{C}$ is a class of L-structures that is recognized by a family of L-symmetric Boolean threshold circuits of size $s(n)$, then $\mathscr{C}$ is recognized by a family of L-symmetric LP lifts of size polynomial in $s(n)$. More concretely, if the circuits have $w(n)$ many wires, then*

the LP lifts have $O(w(n)^2)$ many variables and inequalities with coefficients of $O(\log(w(n)))$ many bits. Moreover, if the Boolean circuits do not have threshold gates, then the LP lifts have $O(w(n))$ many variables and inequalities.

Since AND, OR and NOT gates can be very easily represented by small symmetric LPs, the main step in the construction is the simulation of the threshold gates. The naïve approach by which each threshold gate is replaced by an equivalent AND-OR-NOT circuit will not work: it is known that any symmetric such circuit that computes the majority function must have superpolynomial size. This follows from Theorem 2 in [3] and a standard Ehrenfeucht-Fraissé argument. We need an alternative approach and for that we use the so-called *cardinality indicating polytopes*. Small and symmetric lifts for these polytopes were constructed in [38]. First we need to discuss the "gate behaviour" that is required of them.

3.1 Gate behaviour: and, or, parity, and threshold

Let us start by recalling how AND, OR and NOT gates are represented by LPs. The LPs for these three types of gates do not require auxiliary variables. Define:

$$\begin{array}{lll}
\underline{\mathrm{AND}(x_1,\ldots,x_n,y)} & \underline{\mathrm{OR}(x_1,\ldots,x_n,y)} & \underline{\mathrm{NOT}(x,y)} \\
y \geq \sum_{i=1}^n x_i - n + 1 & 1 - y \geq \sum_{i=1}^n (1 - x_i) - n + 1 & y = 1 - x \\
y \leq x_i & 1 - y \leq 1 - x_i & 0 \leq x \leq 1 \\
0 \leq x_i \leq 1 & 0 \leq x_i \leq 1 & 0 \leq y \leq 1. \\
0 \leq y \leq 1. & 0 \leq y \leq 1. &
\end{array}$$

It is not hard to see that these LPs have the following additional useful property: If $x_1, \ldots, x_n$ are all in $\{0,1\}$, with $n = 1$ for NOT, then there is a unique $y \in \mathbb{R}$ that makes $(x_1, \ldots, x_n, y)$ a feasible solution for the LP, and this y is the unique output bit of the corresponding gate evaluated on inputs $x_1, \ldots, x_n$. We say that the LPs have the corresponding gate behaviour. More generally, if $f : \{0,1\}^n \to \{0,1\}$ is a Boolean function and P is a polytope in $\mathbb{R}^{n+1}$, then we say that P has *f-gate behaviour* if the following property holds:

> if $x_1, \ldots, x_n$ are in $\{0,1\}$, then there exists a unique y in $\mathbb{R}$ such that $(x_1, \ldots, x_n, y)$ is in P, and moreover this y is $f(x_1, \ldots, x_n)$. (1)

If we write and_n and or_n to denote the n-input Boolean AND and OR functions, respectively, and *not* to denote the Boolean NOT function, then the main properties of the LPs are summarized in the following lemma.

Lemma 2. *The linear programs* $\mathrm{AND}(x_1, \ldots, x_n, y)$, $\mathrm{OR}(x_1, \ldots, x_n, y)$, *and* $\mathrm{NOT}(x, y)$ *are symmetric with respect to the group of permutations of their variables that fix y, have $O(n)$ many variables and inequalities with coefficients of $O(\log n)$ many bits, and define polytopes that have and_n, or_n, and not-gate behaviour, respectively.*

Proof. For NOT this is totally obvious. For AND it is easy to check, and for OR it follows from the corresponding properties of AND and NOT. □

In what follows we want to show that the Boolean functions $xor_n : \{0,1\}^n \to \{0,1\}$, which is 1 on inputs $(x_1, \ldots, x_n)$ of odd Hamming weight, and $th_{n,k} : \{0,1\}^n \to \{0,1\}$, which is 1 on inputs $(x_1, \ldots, x_n)$ of Hamming weight at least k, also have such LPs. Unlike the LPs for AND, OR and NOT gates, for these two cases we will need lifts; i.e., auxiliary variables.

Let us start with the parity gates. Consider the parity polytope:

$$\mathrm{PP}(n) \ := \mathrm{conv}\{(x_1, \ldots, x_n) \in \{0,1\}^n : \textstyle\sum_i x_i \equiv 1 \pmod 2\}.$$

Yannakakis [47] showed that the parity polytopes $\mathrm{PP}(n)$ have symmetric LP lifts with $O(n^2)$ many variables and inequalities with integer coefficients bounded by $n+1$ in absolute value. Besides the input variables $x_1, \ldots, x_n$, the program has variables w_t and $z_{t,i}$ for each $t \in \{0, \ldots, \lfloor n/2 \rfloor\}$ and $i \in \{1, \ldots, n\}$. Writing T for $\{0, \ldots, \lfloor n/2 \rfloor\}$ and N for $\{1, \ldots, n\}$, the linear program that achieves this is the following (see page 444 in [47]):

$$\begin{array}{ll}
\sum_{t \in T} w_t = 1 & \\
0 \le w_t \le 1 & \text{for each } t \in T \\
\sum_{t \in T} z_{t,i} = x_i & \text{for each } i \in N \\
\sum_{i \in N} z_{t,i} = (2t+1) w_t & \text{for each } t \in T \\
0 \le z_{t,i} \le w_t & \text{for each } t \in T \text{ and } i \in N
\end{array} \tag{2}$$

Writing z and w for the set of its auxiliary variables, we write $\mathrm{XOR}(x_1, \ldots, x_n, y, z, w)$ to denote the LP for $\mathrm{PP}(n+1)$ with x_{n+1} replaced by $1 - y$. We claim that the shadow of the polytope defined by this LP on the variables $x_1, \ldots, x_n, y$ has xor_n-gate behaviour. Before we check this, it may be worth highlighting the following subtle point. While a polytope that has f-gate behaviour for a certain $f : \{0,1\}^n \to \{0,1\}$ must in particular recognize the graph $\mathrm{G}(f) := \{(x_1, \ldots, x_n, f(x_1, \ldots, x_n)) : x_1, \ldots, x_n \in \{0,1\}\}$ of f, the converse need not in general be true as some trivial examples can show; take for example the convex hull of the points $(x_1, \ldots, x_n, y) \in \{0,1\}^n \times \{0, 1/2, 1\}$ such that $|y - f(x_1, \ldots, x_n)| \le 1/2$. In other words, the gate-behaviour of a polytope is not a property that depends only on the set of Boolean vectors that it recognizes. The proof below relies on the fact that the set of points $(x_1, \ldots, x_n, y)$ for which $(x_1, \ldots, x_n, 1-y)$ belongs to $\mathrm{PP}(n+1)$ is actually the convex hull of the graph $\mathrm{G}(xor_n)$ of xor_n.

Lemma 3. *The linear program* $\mathrm{XOR}(x_1, \ldots, x_n, y, z, w)$ *is symmetric with respect to the group of permutations of the variables* $x_1, \ldots, x_n, y$ *that fix* y*, has* $O(n^2)$ *many variables and inequalities with coefficients of* $O(\log n)$ *many bits, and defines the lift of a polytope that has* xor_n*-gate behaviour.*

Proof. Let P be the shadow of the polytope defined by the LP for $\mathrm{PP}(n+1)$ with x_{n+1} replaced by $1 - y$. We show that P has xor_n-gate behaviour. For the existence of $y \in \mathbb{R}$ for fixed $x_1, \ldots, x_n \in \{0,1\}$ as required by definition (1) just take $y = 1 - x_{n+1} \in \{0,1\}$ so that $\sum_{k=1}^{n+1} x_k$ is odd. The uniqueness will follow once we show that any x_{n+1} for which the extension vector $(x_1, \ldots, x_n, 1 - x_{n+1})$ belongs to $\mathrm{PP}(n+1)$ is in $\{0,1\}$. In turn, this follows from the fact that all extremal points of $\mathrm{PP}(n+1)$ are in $\{0,1\}^{n+1}$, all have the same parity, and a single bit-flip flips their parity. Indeed, if $(x_1, \ldots, x_n, 1 - x_{n+1})$ is in $\mathrm{PP}(n+1)$ but is not an extremal point, then it must be a non-trivial convex combination of at least

two extremal points and, whenever $x_1, \ldots, x_n$ are all in $\{0,1\}$, only two candidate extremal points remain: $(x_1, \ldots, x_n, 0)$ and $(x_1, \ldots, x_n, 1)$. Otherwise some x_i with $1 \leq i \leq n$ would be strictly between 0 and 1. However, among these two candidate extremal points, at least one does not have the right parity, and hence is not even in $\mathrm{PP}(n+1)$.

The symmetry of $\mathrm{XOR}_n(x_1, \ldots, x_n, y, z, w)$ with respect to the permutations of the variables $x_1, \ldots, x_n$ and y that fix y is obvious: given a permutation $\pi \in \mathrm{Sym}_n$, let σ be the permutation that maps $z_{t,i}$ to $z_{t,\pi(i)}$ and leaves each w_t in place. □

We turn next to threshold gates. Consider the cardinality indicating polytope:

$$\mathrm{CP}(n) := \mathrm{conv}\{(x_1, \ldots, x_n, z_0, \ldots, z_n) \in \{0,1\}^{2n+1} : \textstyle\sum_i x_i = \sum_j j z_j,\ \sum_j z_j = 1\}. \tag{3}$$

It was shown by Pashkovich [38] that the cardinality indicating polytope $\mathrm{CP}(n)$ has a symmetric LP lift with $O(n^2)$ many variables and inequalities with integer coefficients bounded by n in absolute value. Besides the input variables $x_1, \ldots, x_n, z_0, \ldots, z_n$, the program has variables $w_{i,j}$ for each $i \in \{1, \ldots, n\}$ and $j \in \{0, \ldots, n\}$ and the following constraints:

$$\begin{array}{ll} \sum_{j=0}^{n} z_j = 1 & \\ \sum_{i=1}^{n} w_{i,j} = j z_j & \text{for each } j \in \{0, \ldots, n\} \\ \sum_{j=0}^{n} w_{i,j} = x_i & \text{for each } i \in \{1, \ldots, n\} \\ 0 \leq w_{i,j} \leq z_j & \text{for each } i \in \{1, \ldots, n\} \text{ and } j \in \{0, \ldots, n\} \end{array} \tag{4}$$

We write $\mathrm{CP}(x_1, \ldots, x_n, z, w)$ for this LP, and $\mathrm{TH}_k(x_1, \ldots, x_n, y, z, w)$ for the linear program defined by

$$\begin{array}{l} \mathrm{CP}(x_1, \ldots, x_n, z, w) \\ \mathrm{OR}(z_k, \ldots, z_n, y) \end{array} \tag{5}$$

We show that this LP has the required symmetry and defines a polytope whose shadow has $th_{n,k}$-gate behaviour.

Lemma 4. *The linear program $\mathrm{TH}_k(x_1, \ldots, x_n, y, z, w)$ is symmetric with respect to the group of permutations of the variables $x_1, \ldots, x_n, y$ that fix y, has $O(n^2)$ many variables and inequalities with coefficients of $O(\log n)$ many bits, and it defines the lift of a polytope that has $th_{n,k}$-gate behaviour.*

Proof. Let P be the shadow of the polytope defined by $\mathrm{TH}_k(x_1, \ldots, x_n, y, z, w)$. We need to show that P has $th_{n,k}$-gate behaviour. Fix $x_1, \ldots, x_n \in \{0,1\}$ and let $t = \sum_i x_i$. For the existence of $y \in \mathbb{R}$ as required by (1), take $y = 1$ if $t \geq k$ and $y = 0$ if $t < k$, and let $z = (z_0, \ldots, z_n) \in \{0,1\}^{n+1}$ and w be such that $(x_1, \ldots, x_n, z, w)$ belongs to $\mathrm{CP}(n)$. By construction the point $(x_1, \ldots, x_n, y, z, w)$ is feasible for the LP, so $(x_1, \ldots, x_n, y)$ is in P. For the uniqueness, suppose that y is such that $(x_1, \ldots, x_n, y)$ is in P. In particular, there exist z and w such that the point $(x_1, \ldots, x_n, y, z, w)$ is feasible for the LP. Therefore, $(x_1, \ldots, x_n, z)$ is in $\mathrm{CP}(n)$, which means that it is a convex combination of its extremal points. Since $x_1, \ldots, x_n$ are in $\{0,1\}$ and all extremal points of $\mathrm{CP}(n)$ are in $\{0,1\}^{2n+1}$, the only possibility for this is that $z = (z_0, \ldots, z_n)$ is also in $\{0,1\}^{n+1}$ and $z_j = 1$ if and only if $j = t$. But also $(z_k, \ldots, z_n, y)$ is in the polytope defined by $\mathrm{OR}(z_k, \ldots, z_n, y)$,

and since $z_k, \ldots, z_n$ are in $\{0,1\}$ we have that $y = 1$ if $t \geq k$ and $y = 0$ if $t < k$. Thus, $y = th_{n,k}(x_1, \ldots, x_n)$ as was to be shown.

The symmetry of $\mathrm{TH}_k(x_1, \ldots, x_n, y, z, w)$ with respect to the permutations of the variables $x_1, \ldots, x_n$ and y that fix y is clear: given a permutation $\pi \in \mathrm{Sym}_n$, let σ be the permutation that maps $w_{i,j}$ to $w_{\pi(i),j}$ and leaves each z_j in place. □

Lemmas 2 and 4 give the symmetric LPs that simulate the types of gates for our circuits. While the parity gates of Lemma 3 are not strictly needed for what follows, we included them here for their relevance to circuit complexity [46]. In the conference version of this paper [7] we gave a different construction for the threshold gates in terms of so-called *truncated parity polytopes* that were introduced there. We thank one of the referees of this paper for pointing out that the simpler and arguably better construction presented here was possible by using the symmetric LP lifts of the cardinality indicating polytopes from [38].

3.2 LPs for threshold circuits

Let V be a set and let C be a circuit with inputs $(x_v)_{v \in V}$ that is made of AND, OR, NOT and TH gates. We define the LP that simulates C; we call it $\mathrm{LP}(C)$.

For each gate o in C, let y_o be a variable constrained by the inequalities

$$0 \leq y_o \leq 1. \tag{6}$$

Add to these the constraints and the auxiliary variables, when necessary, that express their computation:

$$\begin{array}{ll}
y_o = x_v & \text{if } o \text{ is an input gate labelled by } x_v, \\
\mathrm{NOT}(y_i, y_o) & \text{if } o \text{ is a } \textit{not} \text{ gate with input } i, \\
\mathrm{AND}(y_{i_1}, \ldots, y_{i_m}, y_o) & \text{if } o \text{ is an } \textit{and}_m \text{ gate with inputs } i_1, \ldots, i_m, \\
\mathrm{OR}(y_{i_1}, \ldots, y_{i_m}, y_o) & \text{if } o \text{ is an } \textit{or}_m \text{ gate with inputs } i_1, \ldots, i_m, \\
\mathrm{TH}_k(y_{i_1}, \ldots, y_{i_m}, y_o, z, w) & \text{if } o \text{ is an } th_{m,k} \text{ gate with inputs } i_1, \ldots, i_m, \\
y_o = 1 & \text{if } o \text{ is the output gate of the circuit.}
\end{array} \tag{7}$$

By Lemmas 2 and 4, all six cases have size polynomial in their number of input wires, hence the total size is polynomial in the size of C. More precisely, each LP has integer coefficients whose absolute value is at most linear in the number of input wires and a number of variables and inequalities that is at most quadratic in the number of input wires. This means that the total number of variables and inequalities of the LP is at most quadratic in the number of wires in the circuit, and the coefficients have bit-size at most logarithmic in the number of wires in the circuit. In case C does not have TH gates, all gates are AND, OR, NOT, and then the number of variables and inequalities of the LP is even linear in the number of wires of C.

Lemma 5. *If V is a G-set and C is G-symmetric, then $\mathrm{LP}(C)$ is G-symmetric and recognizes the same subset of $\{0,1\}^V$ as C.*

Proof. The claim that $\mathrm{LP}(C)$ recognizes the same subset of $\{0,1\}^V$ as C follows from Lemmas 2 and 4. We show that the LP is also G-symmetric. Fix some $\pi \in G$. Let σ be a

permutation of the gates of C so that the pair (π, σ) leaves C in place. In particular, for each gate o of C with inputs $i_1, \ldots, i_m$, if $p = \sigma(o)$, then p is the same type of gate as o, has the same fan-in m, and if o is an input gate labelled by x_u, then p is an input gate labelled by $x_{\pi(u)}$. Moreover, if $j_1, \ldots, j_m$ are the inputs of gate p, then there is a permutation $\tau_o \in \mathrm{Sym}_m$ so that $\sigma(i_k) = j_{\tau_o(k)}$ for every $k \in [m]$. If we think of σ as mapping the output variable y_o of the linear program $P_o = P(y_{i_1}, \ldots, y_{i_m}, y_o)$ for gate o to the output variable y_p of the linear program $P_p = P(y_{j_1}, \ldots, y_{j_m}, y_p)$ for gate p, then, by the above, this map takes the input variables of P_o to the input variables of P_p. We want to show that this σ can be extended to also map the auxiliary variables of P_o to the auxiliary variables of P_p in such a way that the resulting extension of π is an automorphism of the linear program P for C. We define this extension automorphism gate by gate.

We start with the internal gates of C. By the symmetry claims in Lemmas 2 and 4, the permutation that agrees with τ_o on the input variables $y_{j_1}, \ldots, y_{j_m}$ of P_p and that fixes the output variable y_p, extends to an automorphism ρ_o of P_p. With the automorphisms ρ_o in hand, we are ready to define the automorphism of P that extends π: for each gate o, map the variable y_o to $y_{\sigma(o)}$, and map each auxiliary variable of P_o to the image under ρ_o of the corresponding auxiliary variable of P_p, where $p = \sigma(o)$. For all internal gates, this has the required properties by construction. For the gates o that are labelled by a variable x_v, the gate $\sigma(o)$ must be labelled by the variable $x_{\pi(v)}$, and therefore $y_o = x_v$ gets mapped to $y_{\sigma(o)} = x_{\pi(v)}$, as required. For the output gate o of C we have $\sigma(o) = o$, and the constraint $y_o = 1$ gets mapped to itself. □

Proof of Lemma 1. This is a consequence of Lemma 5: for the n-th LP we let $V = L(n)$, let $G = \mathrm{Sym}_n$ with the natural action on $L(n)$, and use $\mathrm{LP}(C_n)$, where C_n is the n-th circuit. The size of the LP was analyzed immediately following its definition. □

4 From LPs to Logic

We say that a function $s(n)$ is at most weakly exponential if there exists a positive real ϵ such that $s(n) \leq 2^{n^{1-\epsilon}}$ for every sufficiently large n. In this section we prove the second half of the equivalence which takes families of symmetric linear programs to families of formulas of counting logic. That is:

Lemma 6. *If $\mathscr{C}$ is a class of L-structures that is recognized by a family of L-symmetric LP lifts of size $s(n)$, then $\mathscr{C}$ is recognized by a family of $\mathrm{C}^{k(n)}$ formulas, where $k(n) = O(\log(s(n))/(\log(n) - \log\log(s(n))))$. Moreover, if $s(n)$ is at most weakly exponential, then the formulas have size $s(n)^{O(1)}$.*

The key technical tool is the notion of a *bounded support*. The existence of bounded supports implies that a property of L-structures recognized by a family of L-symmetric LP lifts is recognized by a family of *manageable* such LP lifts. Here we illustrate the notion of a manageable LP lift by an example and defer its formal definition to Subsection 4.3.

Consider a graph-symmetric polytope P over $\mathbb{R}^{[2]^2} \times \mathbb{R}^2$ given by the following system of

linear constraints:

$$\begin{aligned} x_{11} &\leq 1 \\ x_{22} &\leq 1 \\ x_{12} - y_{12} &\leq 0 \\ x_{21} - y_{21} &\leq 0. \end{aligned}$$

In this example $n = 2$. Two properties of P are central to the notion of a manageable linear program. Firstly, each of the auxiliary variables, here y_{12} and y_{21}, is essentially a tuple of integers from $[n]$ of a bounded length, here length two. Secondly, P is graph-symmetric with respect to the natural action of the group Sym_n on the set of auxiliary variables, here $\{y_{12}, y_{21}\}$. Indeed, for any permutation $\pi \in \mathrm{Sym}_2$ the system of constraints:

$$\begin{aligned} x_{\pi(1)\pi(1)} &\leq 1 \\ x_{\pi(2)\pi(2)} &\leq 1 \\ x_{\pi(1)\pi(2)} - y_{\pi(1)\pi(2)} &\leq 0 \\ x_{\pi(2)\pi(1)} - y_{\pi(2)\pi(1)} &\leq 0 \end{aligned}$$

defines the same polytope.

For recognizing classes of structures, the main features of manageable linear programs are that, given an L-structure over an n-element domain, a manageable LP lift over $\mathbb{R}^{L(n)} \times \mathbb{R}^W$ can be turned into a linear program whose variables and constraints are indexed by tuples of elements of the structure of bounded length. Moreover, the symmetry condition guarantees that this can be done without referring to any concrete bijection between $[n]$ and the domain of the structure. Thus, as we show, it can be performed by means of a logical interpretation applied to the input structure.

In what follows, from a family $(P_n)_{n\in\mathbb{N}}$ of L-symmetric LP lifts we obtain a family $(\bar{P}_n)_{n\in\mathbb{N}}$ of manageable LP lifts recognizing the same family of structures. We do this in a sequence of steps. Lemma 7 below implies that from each P_n one can construct a polytope lift $\widehat{P}_n$ which recognizes the same subset of $\{0,1\}^{L(n)}$ but comes with an action of the group Sym_n witnessing its symmetry. Further, in Subsection 4.2 we show that the action of Sym_n on each of the constraints and auxiliary variables of $\widehat{P}_n$ depends on a subset of $[n]$ of bounded size called its support. In the second part of Subsection 4.2 we analyse properties of sets whose elements have bounded supports in order to show that they are essentially sets of tuples of integers from $[n]$. This implies that, after a small modification, each $\widehat{P}_n$ becomes a manageable LP lift $\bar{P}_n$. We do this in Subsection 4.3. Finally, in Subsection 4.4 based on $\bar{P}_n$ we construct a FOC-interpretation that given an L-structure $\mathbb{A}$ over an n-element domain outputs a linear program which has a solution if, and only if, $\mathbb{A}$ belongs to the class of interest (for a definition of an interpretation see e.g. [26]). Since solving linear programs is expressible in FPC [4], we are able to conclude the proof.

4.1 Rigid polytopes

In this subsection we consider general G-symmetric LPs, i.e., not necessarily L-symmetric.

Let $P \subseteq \mathbb{R}^V \times \mathbb{R}^W$ be a G-symmetric polytope given by a G-symmetric sequence of linear constraints $(\gamma_u)_{u\in U}$ where each γ_u is of the form $\mathbf{a}^{\mathrm{T}}\mathbf{x} + \mathbf{b}^{\mathrm{T}}\mathbf{y} \leq c$, with $\mathbf{x} = (x_v)_{v\in V}$ and $\mathbf{y} = (y_w)_{w\in W}$. We say that the polytope P is *rigid* if for every $\pi \in G$ there exists a unique element of Sym_W, let us denote it by σ_π, such that $\{\gamma_u^{(\pi,\sigma_\pi)}\}_{u\in U} = \{\gamma_u\}_{u\in U}$.

Assume that P is rigid. It is easy see that the mapping from G to Sym_W given by $\pi \mapsto \sigma_\pi$ is a group homomorphism. Hence, there is a natural action of the group G on the set of auxiliary variables $\{y_w\}_{w\in W}$ such that for any $\pi \in G$ and $w \in W$ applying π to y_w gives $y_{\sigma_\pi(w)}$, and the polytope P is G-symmetric with respect to this action. Moreover, this induces an action of the group G on the set of linear constraints $\{\gamma_u\}_{u\in U}$ in the obvious way: for any $\pi \in G$ and any $u \in U$ applying π to γ_u of the form $\mathbf{a}^{\mathrm{T}}\mathbf{x}+\mathbf{b}^{\mathrm{T}}\mathbf{y} \leq c$ gives $\mathbf{a}^{\mathrm{T}}\mathbf{x}^\pi+\mathbf{b}^{\mathrm{T}}\mathbf{y}^{\sigma_\pi} \leq c$, and the symmetry of $(\gamma_u)_{u\in U}$ guarantees that this is also a constraint in $\{\gamma_u\}_{u\in U}$. For rigid G-symmetric polytopes, we write $\mathbf{y}^\pi$ to mean $\mathbf{y}^{\sigma_\pi}$, we use γ_u^π to denote $\mathbf{a}^{\mathrm{T}}\mathbf{x}^\pi + \mathbf{b}^{\mathrm{T}}\mathbf{y}^{\sigma_\pi} \leq c$, and P^π to denote $P^{(\pi,\sigma_\pi)}$.

Suppose that a subset A of $\{0,1\}^V$ is recognized by a G-symmetric polytope lift P. We show that there exists a rigid G-symmetric polytope lift $\widehat{P}$ of size polynomial in the size of P recognizing A. More precisely, the number of auxiliary variables and the number of constraints of $\widehat{P}$ are at most, respectively, the number of auxiliary variables and the number of constraints of P, and the bit-size of the coefficients which appear in the linear constraints defining $\widehat{P}$ is polynomial in the bit-size of the coefficients which appear in the linear constraints defining P.

The construction of $\widehat{P}$ goes as follows. For the subgroup of Sym_W consisting of all permutations σ such that $\{\gamma_u^{(\mathrm{id},\sigma)}\}_{u\in U} = \{\gamma_u\}_{u\in U}$, consider the orbits of the set of auxiliary variables $\{y_w\}_{w\in W}$ under the action of this subgroup. By identifying the variables in each of those orbits we obtain a new G-symmetric polytope lift recognizing A with potentially smaller number of auxiliary variables. This procedure needs to be iterated until the obtained polytope is rigid. Let us provide more details.

We need a few pieces of notation. For every $\pi \in G$, by $\mathrm{ext}(\pi)$ we denote the set of all $\sigma \in \mathrm{Sym}_W$ satisfying $\{\gamma_u^{(\pi,\sigma)}\}_{u\in U} = \{\gamma_u\}_{u\in U}$ and by H we denote the union of $\mathrm{ext}(\pi)$ over all the elements π of G, i.e., $H = \{\sigma \in \mathrm{Sym}_W :$ there exists $\pi \in G$ such that $\{\gamma_u^{(\pi,\sigma)}\}_{u\in U} = \{\gamma_u\}_{u\in U}\}$. It is easy to see that both $\mathrm{ext}(\mathrm{id})$ and H are subgroups of Sym_W. Moreover, for any $\pi \in G$ the set $\mathrm{ext}(\pi)$ is a coset of $\mathrm{ext}(\mathrm{id})$ in H. Now let $K = \sum_{\pi\in G}\{\pi\} \times \mathrm{ext}(\pi)$. Observe that K is a subgroup of the direct product $G \times H$ and consider the projection homomorphism $f_G : K \to G$ given by $f_G(\pi,\sigma) = \pi$. The kernel of this homomorphism is the group $\{\mathrm{id}\} \times \mathrm{ext}(\mathrm{id})$. Let us denote it by J. Since the homomorphism f_G is surjective, the quotient K/J is isomorphic to G.

Notice that the polytope P is rigid if, and only if, the group $\mathrm{ext}(\mathrm{id})$ is trivial. Assume therefore that this is not the case and let $\mathcal{O}$ denote the set of orbits of the set of auxiliary variables $\{y_w\}_{w\in W}$ under the action of $\mathrm{ext}(\mathrm{id})$. Recall that $\mathcal{O}$ forms a partition of $\{y_w\}_{w\in W}$. For every orbit $O \in \mathcal{O}$, we introduce a new variable y_O and for any polytope R, let $\widehat{R}$ denote the polytope obtained from R by substituting, for each $O \in \mathcal{O}$, every variable in O (if present) by y_O. We aim to show that $\widehat{P}$ is a G-symmetric polytope recognizing A.

The projection homomorphism $f_H : K \to H$ given by $f_H(\pi,\sigma) = \sigma$ can be seen as a homomorphism from K to Sym_W and hence defines an action of the group K on the set of

variables $\{y_w\}_{w\in W}$. Since J is a normal subgroup of K, the quotient group K/J acts on the set of orbits of $\{y_w\}_{w\in W}$ under the (induced) action of J: for any $\pi \in G$, $\sigma \in \mathrm{ext}(\pi)$ and $w \in W$, applying $(\pi,\sigma)J$ to the orbit of y_w maps it to the orbit of $y_{\sigma(w)}$. Since the groups J and ext(id) as well as the groups K/J and G are isomorphic, this gives us an action of G on $\mathcal{O}$ which, as we argue below, witnesses the fact that the polytope $\widehat{P}$ is G-symmetric. Unfolding the abstract definition of this group action, there is a homomorphism h from the group G to the symmetric group $\mathrm{Sym}_{\mathcal{O}}$ such that for every $\pi \in G$, $h(\pi)$ is the permutation of $\mathcal{O}$, which for any $w \in W$, maps the orbit of the variable y_w to the orbit of the variable $y_{\sigma(w)}$, where σ is any permutation from $\mathrm{ext}(\pi)$. Finally, let us observe that, for any $\pi \in G$ and $\sigma \in \mathrm{ext}(\pi)$, it holds that

$$\widehat{P}^{(\pi,h(\pi))} = \widehat{P^{(\pi,\sigma)}} = \widehat{P},$$

which means that indeed $\widehat{P}$ is G-symmetric. It remains to show that $\widehat{P}$ recognizes A.

Let $\mathbf{x} \in A$ and take some $\mathbf{y} \in \mathbb{R}^W$ such that $(\mathbf{x},\mathbf{y}) \in P$. Note that for every $\sigma \in \mathrm{ext}(\mathrm{id})$ it holds that $(\mathbf{x},\mathbf{y}^\sigma) \in P$. Hence, we have $(\mathbf{x},\mathbf{y}') \in P$ where $\mathbf{y}' = \sum_{\sigma\in\mathrm{ext}(\mathrm{id})} \mathbf{y}^\sigma / |\mathrm{ext}(\mathrm{id})|$. Now for every $O \in \mathcal{O}$ let p_O be the sum of the values taken by the variables from O in the solution $(\mathbf{x},\mathbf{y})$. In the solution $(\mathbf{x},\mathbf{y}')$ every variable y in $\mathcal{O}$ takes value $p_O|\mathrm{ext}(\mathrm{id})_y|/|\mathrm{ext}(\mathrm{id})| = p_O/|\mathcal{O}|$, where $\mathrm{ext}(\mathrm{id})_y$ denotes the stabilizer of y in ext(id). Hence, by assigning for every $O \in \mathcal{O}$ the value p_O to the variable y_O we obtain a point $(\mathbf{x},\widehat{\mathbf{y}})$ in $\widehat{P}$. This implies that A is contained in the subset of $\{0,1\}^V$ recognized by $\widehat{P}$. The other inclusion is clear.

We are now ready to state the main conclusion of this subsection.

Lemma 7. *For every G-symmetric polytope P of size s, there is a rigid G-symmetric polytope Q of size not more than $s\log(s)$ which recognizes the same set.*

Proof. First note that for any G-symmetric polytope R, if R is not rigid, then $\widehat{R}$ has strictly fewer variables than R. Thus, starting at P, if we iterate the process, we must, in a finite number of steps reach a rigid polytope Q. For the size bound, note that the number of variables and constraints in Q is at most the corresponding number in P. Moreover, each coefficient in Q is the sum of distinct coefficients in P, of which there are at most s. Thus, if each coefficient in P can be expressed with b bits, each coefficient in Q requires at most $b\log(s)$ bits and the bound follows. □

4.2 Bounded supports

For a Sym_n-set Y, a subset S of $[n]$ is said to be a *support* of an element $y \in Y$ if for every $\pi \in \mathrm{Sym}_n$ that fixes S pointwise, it holds that $\pi \cdot y = y$. Intuitively, this means that the action of the group Sym_n on y depends only on the set S. We get even supports by replacing the symmetric group Sym_n by the alternating group Alt_n. A subset S of $[n]$ is said to be an *even support* of $y \in Y$ if for every $\pi \in \mathrm{Alt}_n$ that fixes S pointwise, we have $\pi \cdot y = y$. Clearly, any support of y is also an even support of y. Note also that every element of a Sym_n-set is supported by the whole set $[n]$.

For a non-negative integer k, an (even) support S is called *k-bounded* if $|S| \leq k$. A Sym_n-set Y is called *k-supported* if each element of Y has a k-bounded support. A rigid L-

symmetric polytope P is called *k-supported* if the set of auxiliary variables and the set of constraints of P are k-supported. We now show the following:

Lemma 8. *There exists a positive integer n_0 such that for any positive integers s and n satisfying $s \geq n \geq n_0$, the following holds: If P is a rigid L-symmetric LP lift of size s for structures with n elements, then P is k-supported, where $k = O(\log(s)/(\log(n) - \log\log(s)))$. Moreover, if $s \leq 2^{n/3}$, then the size of P is at most n^k.*

Proof. For the sake of simplicity we give the proof for the case when L consists of a single binary symbol, that is for the case of graphs. The general case is completely analogous.

Consider a rigid graph-symmetric polytope lift $P \subseteq \mathbb{R}^{[n]^2} \times \mathbb{R}^W$ which recognizes some property of graphs with n vertices, and let P be given by a linear program $(\gamma_u)_{u \in U}$ of size s, where each γ_u is of the form $\mathbf{a}^{\mathrm{T}}\mathbf{x} + \mathbf{b}^{\mathrm{T}}\mathbf{y} \leq c$, with $\mathbf{x} = (x_{ij})_{i,j \in [n]}$ and $\mathbf{y} = (y_w)_{w \in W}$. Since P is rigid, it comes with an action of the group Sym_n on the set of auxiliary variables $\{y_w\}_{w \in W}$ such that for every $\pi \in \mathrm{Sym}_n$ we have $P^\pi = P$, and with an induced action of the group Sym_n on the set of constraints $\{\gamma_u\}_{u \in U}$.

If the size s of the polytope P is greater than $2^{n/3}$, we can take $k = n$. Indeed, observe that in case $s > 2^{n/3}$ we have

$$\frac{\log(s(n))}{\log(n) - \log\log(s(n))} \geq \frac{n/3}{\log(n) - \log(n) + \log(3)} = \frac{n}{3\log(3)}. \tag{8}$$

It follows that $k = n = O(\log(s)/(\log(n) - \log\log(s)))$. Since any element of a Sym_n-set is supported by $[n]$, for this choice of k, every auxiliary variable and every constraint of P has a k-bounded support.

In the case $s \leq 2^{n/3}$ the argument is more involved. First we obtain bounded even supports. Take $t = \log(s)/(\log(n) - \log\log(s))$ and $k = \lceil t \rceil$. Observe that the denominator in the definition of t is non-zero, since by assumption we have $s \leq 2^{n/3} < 2^n$. Also, we have $0 < t \leq k \leq n/3\log(3) < n/4 < n/e$, to be used later in the proof. Let us start by noting that

$$t \log\left(\frac{n}{t}\right) = \log(s) \frac{\log(n) - \log\log(s) + \log(\log(n) - \log\log(s))}{\log(n) - \log\log(s)} > \log(s). \tag{9}$$

The inequality follows from the fact that the big fraction in the middle is strictly bigger than 1 since $s \leq 2^{n/3}$.

For any $S \subseteq [n]$, let $\mathrm{Alt}_{(S)}$ denote the group of all even permutations of $[n]$ that fix the set S pointwise. We use the following fact which guarantees that subgroups of the symmetric group with small index contain as subgroups large alternating groups:

Lemma 9 (see Theorem 5.2B in [18])**.** *If $n > 8$ and $1 \leq k \leq n/4$, and G is a subgroup of Sym_n with $[\mathrm{Sym}_n : G] < \binom{n}{k}$, then there is a set $S \subseteq [n]$ with $|S| < k$ such that $\mathrm{Alt}_{(S)} \leq G$.*

For $w \in W$, let St_w denote the stabilizer of y_w in Sym_n. Since $[\mathrm{Sym}_n : \mathrm{St}_w]$ is the size of the orbit of y_w under the action of Sym_n and the total number of auxiliary variables is bounded by the size of P, we have

$$[\mathrm{Sym}_n : \mathrm{St}_w] \leq s < \left(\frac{n}{t}\right)^t \leq \left(\frac{n}{k}\right)^k \leq \binom{n}{k} \tag{10}$$

with the second following from (9), and the third from $0 < t \leq k < n/4 < n/e$ and the fact that $f(x) = (n/x)^x$ is an increasing function of x in the interval $(0, n/e)$. Lemma 9 implies that, if n is large enough, there exists $S \subseteq [n]$ with $|S| < k$ and $\mathrm{Alt}_{(S)} \leq \mathrm{St}_w$. This is a k-bounded even support of y_w in the way we defined. An entirely analogous argument yields a k-bounded even support for each constraint in $\{\gamma_u\}_{u \in U}$.

In order to obtain supports in place of even supports we need to introduce a way of looking at automorphism groups of polytope lifts as automorphism groups of graphs. In the following, whenever we talk about coloured vertices, by a colour we mean a graph gadget that forces the vertices of a colour to be mapped to vertices of the same colour by every possible automorphism.

Let C be the (multi)set of all numerical coefficients which appear in the linear constraints representing P. In particular, if some rational number appears more than once in the LP defining P, it appears the same number of times in C. A *graph representation* of the polytope P is an undirected graph $\mathbb{P}$ with:

1. six disjoint sets of vertices: $[n]$, $\{x_{ij}\}_{i,j \in [n]}$, $\{x'_{ij}\}_{i,j \in [n]}$, $\{y_w\}_{w \in W}$, $\{\gamma_u\}_{u \in U}$ and C,
2. the vertices in $[n]$, $\{x_{ij}\}_{i,j \in [n]}$, $\{x'_{ij}\}_{i,j \in [n]}$, $\{y_w\}_{w \in W}$ and $\{\gamma_u\}_{u \in U}$ coloured with five different colours depending on the set they belong to,
3. each vertex in C coloured with a colour depending on the value of the coefficient it corresponds to,
4. for any $(i,j) \in [n]^2$, an edge between i an x_{ij}, between j and x'_{ij}, and between x'_{ij} and x_{ij},
5. for any $u \in U$ and any variable which appears in the constraint γ_u, a pair of edges: between the variable and its coefficient in C, and between this coefficient and γ_u,
6. for any $u \in U$, an edge between γ_u and the vertex in C corresponding to its constant term.

Observe that the automorphism group of the graph $\mathbb{P}$ is isomorphic to the automorphism group of P. Also, the number of vertices in $\mathbb{P}$ can be bounded by $O(s^2)$. To see this, note that the number of vertices introduced in item (1) is $O(s)$. Moreover, we need $O(s)$ colour gadgets and each of them can be chosen to have $O(s)$ vertices. One way of achieving this is using as colour gadgets cycles of different length.

For any $S \subseteq [n]$, let $\mathrm{Sym}_{(S)}$ denote the group of all permutations of $[n]$ that fix the set S pointwise. Take some $w \in W$ and let S be a k-bounded even support of y_w. Since $\mathrm{Alt}_{(S)} \leq \mathrm{St}_w$, we have $\mathrm{Alt}_{(S)} \leq \mathrm{St}_w \cap \mathrm{Sym}_{(S)} \leq \mathrm{Sym}_{(S)}$. Hence, $\mathrm{St}_w \cap \mathrm{Sym}_{(S)} = \mathrm{Alt}_{(S)}$ or $\mathrm{St}_w \cap \mathrm{Sym}_{(S)} = \mathrm{Sym}_{(S)}$. We argue it is the latter case that holds using the following theorem which states that a graph whose automorphism group is the alternating group on an n-element set must be of size exponential in n:

Lemma 10 (Theorem A in [34])**.** *If $n > 22$, then the number of vertices of any graph whose full automorphism group is isomorphic to* Alt_n *is at least* $\frac{1}{2}\binom{n}{\lfloor n/2 \rfloor} \sim 2^n/\sqrt{2\pi n}$ *.*

Assume that $\mathrm{St}_w \cap \mathrm{Sym}_{(S)} = \mathrm{Alt}_{(S)}$. We use Lemma 10 to arrive at a contradiction. Consider the graph representation $\mathbb{P}$ of the polytope P modified in the following way: the

vertices in $S \subseteq [n]$ and the vertex y_w are coloured each with a different colour which did not appear in the graph before. Observe that the automorphism group of the graph $\mathbb{P}_w$ obtained this way is isomorphic to $\mathrm{St}_w \cap \mathrm{Sym}_{(S)}$, and therefore isomorphic to $\mathrm{Alt}_{(S)}$, which in turn is isomorphic to the alternating group on the set $[n - |S|]$. And, once again, the number of vertices of $\mathbb{P}_w$ is $O(s^2)$. Thus, if n is large enough, we have

$$s^2 < 2^{2n/3} < \frac{1}{2}\binom{n}{\lfloor n/2 \rfloor}. \tag{11}$$

Hence, by Lemma 10, we obtain the desired contradiction.

From $\mathrm{St}_w \cap \mathrm{Sym}_{(S)} = \mathrm{Sym}_{(S)}$ it follows that $\mathrm{Sym}_{(S)} \leq \mathrm{St}_w$, which means that S is a k-bounded support of y_w in the way we defined. An analogous argument yields a k-bounded support for each linear constraint in $\{\gamma_u\}_{u \in U}$. Recall also that $s < \binom{n}{k} \leq n^k$. In particular, P has at most n^k auxiliary variables, at most n^k constraints, and all its coefficients and constant terms can be written down using at most n^k bits. This finishes the proof of Lemma 8. □

We now show that it is possible to (non-uniquely) represent the auxiliary variables and constraints of k-supported polytopes by tuples of integers from $[n]$ of length k in a way that is consistent with the group action. In order for the representation to be uniform across all n, we extend the definition of the set $[n]^{(k)}$ to the case when $k > n$. For a set X with n elements and an integer $k > n$, the set $X^{(k)}$ consists of k-tuples of elements of X with the first n components pairwise distinct and the last $k - n$ components equal to the n-th component. In particular, if $k \geq n$, then every tuple in $[n]^{(k)}$ contains all the elements of $[n]$.

For any positive integer n and any non-negative integer k, we consider the set $[n]^{(k)}$ as a Sym_n-set with the natural action of the group Sym_n. Note that this Sym_n-set has one orbit. Indeed, if we take any k-tuples $\mathbf{s}_1$ and $\mathbf{s}_2$ from $[n]^{(k)}$, then there exists a permutation π such that $\pi \cdot \mathbf{s}_1 = \mathbf{s}_2$. This is because the equality types of any two elements of $[n]^{(k)}$ are the same.

Lemma 11. *Let Y be a single-orbit k-supported Sym_n-set. There is a surjective homomorphism of Sym_n-sets from $[n]^{(k)}$ to Y.*

Proof. Take any $y \in Y$ and let S be a k-bounded support of y. Since a superset of a support is a support itself, without loss of generality we can assume that $|S| = \min\{k, n\}$. Now, pick a tuple $\mathbf{s} \in S^{(k)}$. If $k \leq n$, then $|S| = k$ and every element of S appears exactly once in the tuple $\mathbf{s}$, otherwise the tuple $\mathbf{s}$ contains all the elements of $[n]$.

We define a homomorphism f from the Sym_n-set $[n]^{(k)}$ to the Sym_n-set Y which for any $\pi \in \mathrm{Sym}_n$ maps $\pi \cdot \mathbf{s}$ to $\pi \cdot y$. The only thing that needs to be verified is whether the function f is well defined. To this end, suppose that for some permutations $\pi_1, \pi_2 \in \mathrm{Sym}_n$ it holds that $\pi_1 \cdot \mathbf{s} = \pi_2 \cdot \mathbf{s}$. Then $\pi_2^{-1}\pi_1 \cdot \mathbf{s} = \mathbf{s}$, that is, the permutation $\pi_2^{-1}\pi_1$ fixes the support S of y pointwise. Therefore, $\pi_2^{-1}\pi_1 \cdot y = y$ which implies that $\pi_1 \cdot y = \pi_2 \cdot y$. Since Y has one orbit, the homomorphism f is surjective. □

Once a surjective homomorphism f from a Sym_n-set $[n]^{(k)}$ to a Sym_n-set Y is fixed, the family $\{f^{-1}(y)\}_{y \in Y}$ forms a partition of $[n]^{(k)}$. Hence, for any $y \in Y$, each tuple $(i_1, \ldots, i_k)$ from $f^{-1}(y)$ uniquely identifies y, and is called an *identifier* of y (with respect to the homomorphism f). In most cases each element of Y has several identifiers.

Let us illustrate Lemma 11 by an example. For $n \geq 2$, consider the set Y of two-element subsets of $[n]$ with the natural action of the group Sym_n, i.e., for any $\pi \in \mathrm{Sym}_n$ and any distinct $i, j \in [n]$, we have $\pi \cdot \{i, j\} = \{\pi(i), \pi(j)\}$. This single-orbit Sym_n-set is k-supported, for any $k \geq 2$. Let us take $k = 2$. The homomorphism from Lemma 11 is then unique and given by $(i, j) \mapsto \{i, j\}$. Note that the inverse image of any $\{i, j\} \in Y$ has two elements, and hence, each element $\{i, j\}$ of Y has two identifiers: (i, j) and (j, i). For $k = 3$, applying Lemma 11 yields several different homomorphisms. If $n \geq 3$, one of them is given by $(i, j, *) \mapsto \{i, j\}$, where by $*$ we mean any element of $[n]$ distinct from both i and j. In this case, the inverse image of any $\{i, j\} \in Y$ has $2n - 4$ elements. For $n = 2$ and $k = 3$, the homomorphism is again unique and given by $(i, j, j) \mapsto \{i, j\}$ yielding two identifiers, for every element of Y.

To give one more example, consider a single-orbit set which consists of a single element y with the trivial action of the symmetric group Sym_n. This set is k-supported, for any non-negative integer k. For any $k \geq 0$, the homomorphism from Lemma 11 maps each tuple from $[n]^{(k)}$ to y. In particular, for $k = 0$, we have $\epsilon \mapsto y$, where ϵ is the only element of $[n]^{(0)}$, that is, the empty tuple.

To represent elements of k-supported Sym_n-sets with potentially more than one orbit, we need to introduce several copies of the set $[n]^{(k)}$, one for each orbit.

Corollary 1. *Let Y be a k-supported Sym_n-set with l orbits, and let Q be an l-element set with the trivial action of the group Sym_n. There is a surjective homomorphism of Sym_n-sets from $Q \times [n]^{(k)}$ to Y.*

It is clear how to extend the definition of an identifier to the general case discussed in the corollary above. The crucial property of identifiers is that they behave well with respect to the group action as summarized in the following simple lemma:

Lemma 12. *Let f be a surjective homomorphism of Sym_n-sets from $Q \times [n]^{(k)}$ to Y. For any $\pi \in \mathrm{Sym}_n$ and any $y \in Y$, the mapping given by $(q, i_1, \ldots, i_k) \mapsto (q, \pi(i_1), \ldots, \pi(i_k))$ is a bijection from the set of identifiers of y to the set of identifiers of $\pi \cdot y$.*

Proof. Clearly, for any $\pi \in \mathrm{Sym}_n$, the mapping given by $(q, i_1, \ldots, i_k) \mapsto (q, \pi(i_1), \ldots, \pi(i_k))$ defines a permutation of $Q \times [n]^{(k)}$. Moreover, for any $y \in Y$, this permutation maps the identifiers of y to the identifiers of $\pi \cdot y$. Indeed, by the definition of a homomorphism, if $f(q, i_1, \ldots, i_k) = y$, then $f(q, \pi(i_1), \ldots, \pi(i_k)) = \pi \cdot y$. It remains to observe that for any $\pi \in \mathrm{Sym}_n$ and any $y \in Y$, the sets of identifiers of y and $\pi \cdot y$ have the same number of elements. □

4.3 Manageable polytopes

For a non-negative integer k, a polytope P over $\mathbb{R}^{L(n)} \times \mathbb{R}^W$ is called *k-manageable* if:

1. the constraints of P are indexed by $U = Q \times [n]^{(k)}$,
2. the primary variables of P are indexed by $V = L(n)$,
3. the auxiliary variables of P are indexed by $W = T \times [n]^{(k)}$,

4. the sets Q and T come with the trivial action of the group Sym_n,
5. P is L-symmetric with respect to the natural action of Sym_n on W, and the induced action of Sym_n on the set of constraints is exactly the natural action of Sym_n on U.

In the example of a manageable polytope given at the very beginning of this section, indices of the constraints are missing. There $n = k = 2$ and the set of constraints has two orbits $\{x_{11} \leq 1, x_{22} \leq 1\}$ and $\{x_{12} - y_{12} \leq 0, x_{21} - y_{21} \leq 0\}$. We reproduce the example here twice, choosing two different ways of indexing the linear constraints by elements of the Sym_2-set $\{q, q'\} \times [2]^{(2)}$.

$$\begin{array}{ll} \gamma_{q12} & x_{11} \leq 1 \\ \gamma_{q21} & x_{22} \leq 1 \\ \gamma_{q'12} & x_{12} - y_{12} \leq 0 \\ \gamma_{q'21} & x_{21} - y_{21} \leq 0 \end{array} \qquad \begin{array}{ll} \gamma_{q'12} & x_{11} \leq 1 \\ \gamma_{q'21} & x_{22} \leq 1 \\ \gamma_{q21} & x_{12} - y_{12} \leq 0 \\ \gamma_{q12} & x_{21} - y_{21} \leq 0. \end{array}$$

Note that the set of auxiliary variables has only one orbit $\{y_{12}, y_{21}\}$ so introducing T is not necessary.

The key property of k-manageable polytopes, which allows us to use them in the translation from families of linear programs to logic, is the following:

Lemma 13. *If P is a k-manageable polytope with constraints indexed by $Q \times [n]^{(k)}$ and auxiliary variables indexed by $T \times [n]^{(k)}$, then for any relation symbol $R \in L$, and each $q \in Q$, $t \in T$, $\mathbf{i}, \mathbf{i}', \mathbf{j}, \mathbf{j}' \in [n]^{(k)}$, $\mathbf{k}, \mathbf{k}' \in [n]^{\mathrm{ar}(R)}$:*

1. *the constant terms of the linear constraints $\gamma_{(q,\mathbf{i})}$ and $\gamma_{(q,\mathbf{i}')}$ are the same,*
2. *if the equality types of the tuples $(\mathbf{j}, \mathbf{i})$ and $(\mathbf{j}', \mathbf{i}')$ are the same, then the coefficient of the variable $y_{(t,\mathbf{j})}$ in the linear constraint $\gamma_{(q,\mathbf{i})}$ is the same as the coefficient of the variable $y_{(t,\mathbf{j}')}$ in the linear constraint $\gamma_{(q,\mathbf{i}')}$,*
3. *if the equality types of the tuples $(\mathbf{k}, \mathbf{i})$ and $(\mathbf{k}', \mathbf{i}')$ are the same, then the coefficient of the variable $x_{(R,\mathbf{k})}$ in the linear constraint $\gamma_{(q,\mathbf{i})}$ is the same as the coefficient of the variable $x_{(R,\mathbf{k}')}$ in the linear constraint $\gamma_{(q,\mathbf{i}')}$.*

Proof. 1. Recall that every tuple in the set $[n]^{(k)}$ has the same equality type. Therefore, there exists a permutation $\pi \in \mathrm{Sym}_n$ which maps the tuple $\mathbf{i}$ to $\mathbf{i}'$. Since $\gamma^{\pi}_{(q,\mathbf{i})} = \gamma_{(q,\mathbf{i}')}$, the constant terms in the linear constraints $\gamma_{(q,\mathbf{i})}$ and $\gamma_{(q,\mathbf{i}')}$ are the same.

2. If the equality types of the tuples $(\mathbf{j}, \mathbf{i})$ and $(\mathbf{j}', \mathbf{i}')$ are the same, then there exists a permutation $\pi \in \mathrm{Sym}_n$, which maps $\mathbf{j}$ to $\mathbf{j}'$, and $\mathbf{i}$ to $\mathbf{i}'$. Let a be the coefficient of the variable $y_{(t,\mathbf{j})}$ in the linear constraint $\gamma_{(q,\mathbf{i})}$. By applying the permutation π to the constraint $\gamma_{(q,\mathbf{i})}$, we get the constraint $\gamma_{(q,\mathbf{i}')}$. Moreover, since $\pi \cdot y_{(t,\mathbf{j})} = y_{(t,\mathbf{j}')}$, the coefficient of the variable $y_{(t,\mathbf{j}')}$ in $\gamma_{(q,\mathbf{i}')}$ is a. The proof of 3. is analogous. □

Now, suppose that a k-supported rigid L-symmetric LP lift $P \subseteq \mathbb{R}^{L(n)} \times \mathbb{R}^W$ given by a sequence of linear constraints $(\gamma_u)_{u \in U}$ recognizes some property of L-structures, that is, a subset A of $\{0,1\}^{L(n)}$. We argue that there exists a k-manageable polytope lift $\bar{P}$ recognizing A. Since the polytope P is k-supported, by applying Corollary 1 we obtain two sets of identifiers: $Q \times [n]^{(k)}$ for the constraints, and $T \times [n]^{(k)}$ for the auxiliary variables. Let

us introduce a new variable of the form $y_{(t,\mathbf{j})}$, for any identifier $(t, \mathbf{j}) \in T \times [n]^{(k)}$. We obtain a manageable polytope $\bar{P}$ from the polytope P by first, replacing each auxiliary variable y_w in P by the sum of variables $y_{(t,\mathbf{j})}$ over the set of all identifiers $(t, \mathbf{j})$ of y_w. After this first step, the obtained polytope is not yet manageable, but it is L-symmetric with respect to the natural action of Sym_n on $T \times [n]^{(k)}$. Moreover, the induced action of Sym_n on the set of constraints remains unchanged. Both those properties follow directly from Lemma 12. In the second step, we replace every constraint γ_u by several copies of this constraint, one for every identifier $(q, \mathbf{i})$ of γ_u. Another application of Lemma 12 implies that the induced action of Sym_n on the set of constraints is now exactly the natural action of Sym_n on $Q \times [n]^{(k)}$. It follows that the obtained polytope lift $\bar{P}$ is k-manageable.

Before arguing that $\bar{P}$ recognizes the same property of L-structures as the original polytope P, we illustrate the construction by an example. Consider a rigid graph-symmetric polytope P over $\mathbb{R}^{[3]^2} \times \mathbb{R}$ given by the following system of linear constraints:

$$\begin{aligned} 2x_{12} + 2x_{21} + y &\leq 1 \\ 2x_{13} + 2x_{31} + y &\leq 1 \\ 2x_{23} + 2x_{32} + y &\leq 1. \end{aligned}$$

This polytope is 2-supported. Indeed, the set of auxiliary variables $\{y\}$ comes with the trivial action of Sym_3, therefore, y is supported by any subset of $[3]$, in particular, by the empty set. Moreover, the first, second and third constraints in the system are supported by the sets $\{1, 2\}$, $\{1, 3\}$, $\{2, 3\}$, respectively. Hence, by applying Lemma 12 for $k = 2$, we obtain two sets of identifiers: $[3]^{(2)}$ for the constraints and $[3]^{(2)}$ for the auxiliary variables (introducing Q and T is not necessary, since both sets have only one orbit). More precisely, each element of $[3]^{(2)}$ is an identifier of y, and the sets of identifiers of the first, second and third constraints in the system are $\{(1, 2), (2, 1)\}$, $\{(1, 3), (3, 1)\}$ and $\{(2, 3), (3, 2)\}$, respectively. Following the construction described above, we first introduce a new variable of the form $y_{\mathbf{j}}$, for any identifier $\mathbf{j} \in [3]^{(2)}$ and replace each occurrence of y by the sum of the newly introduced variables. The obtained system is the following:

$$\begin{aligned} 2x_{12} + 2x_{21} + y_{12} + y_{21} + y_{13} + y_{31} + y_{23} + y_{32} &\leq 1 \\ 2x_{13} + 2x_{31} + y_{12} + y_{21} + y_{13} + y_{31} + y_{23} + y_{32} &\leq 1 \\ 2x_{23} + 2x_{32} + y_{12} + y_{21} + y_{13} + y_{31} + y_{23} + y_{32} &\leq 1. \end{aligned}$$

Next, we replace every constraint by two copies of this constraint, one for each of its identifiers, and obtain a polytope that is 2-manageable:

$$\begin{aligned} \gamma_{12} \quad & 2x_{12} + 2x_{21} + y_{12} + y_{21} + y_{13} + y_{31} + y_{23} + y_{32} \leq 1 \\ \gamma_{21} \quad & 2x_{12} + 2x_{21} + y_{12} + y_{21} + y_{13} + y_{31} + y_{23} + y_{32} \leq 1 \\ \gamma_{13} \quad & 2x_{13} + 2x_{31} + y_{12} + y_{21} + y_{13} + y_{31} + y_{23} + y_{32} \leq 1 \\ \gamma_{31} \quad & 2x_{13} + 2x_{31} + y_{12} + y_{21} + y_{13} + y_{31} + y_{23} + y_{32} \leq 1 \\ \gamma_{23} \quad & 2x_{23} + 2x_{32} + y_{12} + y_{21} + y_{13} + y_{31} + y_{23} + y_{32} \leq 1 \\ \gamma_{32} \quad & 2x_{23} + 2x_{32} + y_{12} + y_{21} + y_{13} + y_{31} + y_{23} + y_{32} \leq 1. \end{aligned}$$

Coming back to the general case, we argue that $\bar{P}$ recognizes the same subset of $\{0, 1\}^{L(n)}$ as P. Indeed, take $\mathbf{x} \in \{0, 1\}^{L(n)}$ and $\mathbf{y} \in \mathbb{R}^W$ such that $(\mathbf{x}, \mathbf{y}) \in P$. To obtain $(\mathbf{x}, \mathbf{y}') \in \bar{P}$, for

every $w \in W$, consider the set of all identifiers $(t, \mathbf{j})$ of y_w and assign to the variables indexed by the elements of this set arbitrary values that sum up to the value that the variable y_w takes in the solution $(\mathbf{x}, \mathbf{y})$. For the other direction, take $\mathbf{x} \in \{0,1\}^{L(n)}$ and $\mathbf{y} \in \mathbb{R}^{T \times [n]^{(k)}}$, such that $(\mathbf{x}, \mathbf{y}) \in \bar{P}$. To obtain $(\mathbf{x}, \mathbf{y}') \in P$, for every $w \in W$, assign to the variable y_w the sum over the set of all identifiers $(t, \mathbf{j})$ of y_w of the values taken by the variables $y_{(t,\mathbf{j})}$ in the solution $(\mathbf{x}, \mathbf{y})$.

To summarize, the constructions in Subsections 4.1, 4.2 and 4.3 imply that if a property of L-structures is recognized by a family of L-symmetric LP lifts of size $s(n)$, then the same property is recognized by a family of $k(n)$-manageable LP lifts, where $k(n) = O(\log(s(n))/(\log(n) - \log\log(s(n))))$.

4.4 From manageable polytopes to counting logic

We now put everything together in the proof of Lemma 6. For the sake of simplicity we give the proof for the case when L consists of a single binary symbol, that is for the case of graphs. The general case is completely analogous.

Let $P \subseteq \mathbb{R}^{[n]^2} \times \mathbb{R}^W$ be a graph-symmetric LP lift of size s recognizing some property of graphs with n vertices, that is, a subset A of $\{0,1\}^{[n]^2}$. We show that the same property of graphs is definable by a C^k formula, where $k = O(\log(s)/(\log(n) - \log\log(s)))$.

Let $\widehat{P}$ be a rigid graph-symmetric LP lift recognizing A, as constructed in Subsection 4.1. Recall that its size s' is at most $s\log(s)$ where s is the size of P. In particular, $s' \leq s^2$. If $s > 2^{n/6}$, by a calculation analogous to (8) at the beginning of Subsection 4.2, we have $n = O(\log(s)/(\log(n) - \log\log(s)))$. Since every class of graphs with n vertices is definable in C^n, we complete the proof of the lemma in this case by taking $k = n$. If $s \leq 2^{n/6}$, then $s' \leq s^2 \leq 2^{n/3}$. Hence, by Lemma 8, for some $k = O(\log(s')/(\log(n) - \log\log(s')))$, $\widehat{P}$ is k-supported, has at most n^k auxiliary variables, at most n^k constraints, and all its coefficients and constant terms can be encoded using at most n^k bits. Moreover, any such k clearly satisfies $k = O(\log(s)/(\log(n) - \log\log(s)))$.

Let $\bar{P}$ be a k-manageable polytope lift recognizing A (as described in Subsection 4.3) with the set of constraints indexed by $Q \times [n]^{(k)}$ and the set of auxiliary variables indexed by $T \times [n]^{(k)}$. Note that it follows from the construction of $\bar{P}$ that the number of elements in the sets T and Q is bounded, respectively, by the number of auxiliary variables and the number of constraints in $\widehat{P}$. Hence, $|Q|, |T| \leq n^k$.

Suppose now that we are given a graph G with the set of vertices $V(G)$ of size n and the set of edges $E(G)$. Intuitively, if we could fix a bijection between $[n]$ and $V(G)$, we could then compute from $\bar{P}$ and G a linear program $\bar{P}_G$ with the set of constraints I and the set of variables J as follows:

$$I = \{\gamma_{(q,\mathbf{v})} : q \in Q, \mathbf{v} \in V(G)^{(k)}\},$$
$$J = \{x_{vw} : v, w \in V(G)\} \cup \{y_{(t,\mathbf{v})} : t \in T, \mathbf{v} \in V(G)^{(k)}\}.$$

In order to decide whether G has the property of interest we would then check if the partial valuation: $x_{vw} = 1$ if $(v, w) \in E(G)$, and $x_{vw} = 0$ otherwise, can be extended to a full solution. This in turn can be easily done in logic using the following straightforward consequence of the results in [4]:

Lemma 14. *There exists an* FPC *formula ϕ which given a matrix $\mathbf{A} \in \mathbb{Q}^{I \times J}$ and a pair of vectors $\mathbf{b} \in \mathbb{Q}^I$, and $\mathbf{a} \in \mathbb{Q}^{J'}$, where $J' \subseteq J$, decides if $\mathbf{a}$ can be extended to a solution of the linear program $\mathbf{Ax} \leq \mathbf{b}$.*

Our goal is to use Lemma 13 to show that the linear program $\bar{P}_G$ can be computed without fixing a bijection between $[n]$ and $V(G)$. We define an FOC-interpretation Ψ which takes as input a graph G with n vertices and outputs a relational encoding of the linear program $\bar{P}_G$ together with the partial valuation discussed above. More precisely, Ψ outputs a matrix $\mathbf{A} \in \mathbb{Q}^{I \times J}$ and a pair of vectors $\mathbf{b} \in \mathbb{Q}^I$, and $\mathbf{a} \in \mathbb{Q}^{J'}$, where $J' \subseteq J$, such that $\mathbf{a}$ can be extended to a solution of $\mathbf{Ax} \leq \mathbf{b}$ if and only if G has the property of interest. To encode the fact that $J' \subseteq J$ we introduce an extra binary relation symbol F for an injective function from the index set J' to the index set J. Hence, Ψ outputs a structure over the vocabulary $L_{\text{vec},2} \dot{\cup} L_{\text{vec},1} \dot{\cup} L_{\text{vec},1} \dot{\cup} \{F\}$ modified, for simplicity, in such a way that there is a single sort symbol $\bar{B}$ for a domain of bit positions.

Given a graph G with n vertices the FOC-interpretation Ψ has access to the domain $V(G)$ of the graph, and the naturally ordered number domain $\{0, \ldots, n\}$. To represent the bit encodings of the numerical coefficients we use tuples from $[n]^k \subseteq \{0, \ldots, n\}^k$. Let $o : [n]^k \to \{0, 1, \ldots, n^k - 1\}$ be the order-preserving bijection from the set $[n]^k$ ordered lexicographically to the set $\{0, 1, \ldots, n^k - 1\}$ with the natural order. For any $\mathbf{s} \in [n]^k$, by $[\mathbf{s}]$ we denote the natural number $o(\mathbf{s})$. Tuples from $[n]^k \subseteq \{0, \ldots, n\}^k$ are also used to represent elements of Q and T. For simplicity, assume that $|Q| = |T| = n^k$, and let us fix a bijection f from Q to $[n]^k$ and a bijection g from T to $[n]^k$. The linear program $\mathbf{Ax} \leq \mathbf{b}$ in the output of Ψ has the set of constraints indexed by $[n]^k \times V(G)^{(k)}$ and the set of variables indexed by $V(G)^2 \cup [n]^k \times V(G)^{(k)}$.

The only non-trivial part in constructing the interpretation Ψ is showing that the relational encodings of the coefficients and constant terms of $\bar{P}_G$ are FOC-definable from G.

Recall from Lemma 13 that in the linear program $\bar{P}$ for any $(q, \mathbf{i}) \in Q \times [n]^{(k)}$ and $(t, \mathbf{j}) \in T \times [n]^{(k)}$, the coefficient of the auxiliary variable $y_{(t,\mathbf{j})}$ in the constraint $\gamma_{(q,\mathbf{i})}$ depends only on t, q and the equality type of the tuple $(\mathbf{i}, \mathbf{j}) \in [n]^{2k}$. This implies that in the linear program $\bar{P}_G$ for any $(q, \mathbf{v}_1) \in Q \times V(G)^{(k)}$ and $(t, \mathbf{v}_2) \in T \times V(G)^{(k)}$, the coefficient of the variable $y_{(t,\mathbf{v}_2)}$ in the constraint $\gamma_{(q,\mathbf{v}_1)}$ is determined by t, q and the equality type of the tuple $(\mathbf{v}_1, \mathbf{v}_2) \in V(G)^{2k}$. Hence, once the bijections from Q and T to $[n]^k$ are fixed, for any $(\mathbf{s}_1, \mathbf{v}_1) \in [n]^k \times V(G)^{(k)}$ and $(\mathbf{s}_2, \mathbf{v}_2) \in [n]^k \times V(G)^{(k)}$, the interpretation Ψ can define the coefficient of the variable $(\mathbf{s}_2, \mathbf{v}_2)$ in the constraint $(\mathbf{s}_1, \mathbf{v}_1)$. This follows from the fact that any element of $[n]^k$ as well as an equality type of a $2k$-tuple can be defined in first-order logic. An analogous reasoning shows that all the other coefficients and constant terms of $\bar{P}_G$ are FOC-definable.

More formally, based on Lemma 13 we define sets T_s^y, T_n^y, T_d^y, T_s^x, T_n^x, T_d^x, and C_s, C_n, C_d to carry all the information about the signs and the bits of the numerators and the denominators of: the coefficients of the auxiliary variables, the coefficients of the variables in $\{x_{ij}\}_{1 \leq i,j \leq n}$, and the constant terms in $\bar{P}$, respectively. We use them in the forthcoming definition of Ψ.

The set T_d^y is designed to carry all information about the denominators of the coefficients of the auxiliary variables of $\bar{P}$. It consists of tuples of the form $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \rho)$, where $\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3 \in [n]^k$, and ρ is a quantifier-free formula defining an equality type of $2k$-

tuples. Take such a $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \rho)$. The tuple $\mathbf{z}_1$ corresponds via f to some element $q \in Q$. The tuple $\mathbf{z}_2$ corresponds via g to some element $t \in T$. If now for any (or, equivalently, every) tuple $(\mathbf{i}, \mathbf{j}) \in [n]^{2k}$ which satisfies ρ, the $[\mathbf{z}_3]$-th least significant bit in the binary encoding of the denominator of the coefficient of the variable $y_{(t,\mathbf{j})}$ in the constraint $\gamma_{(q,\mathbf{i})}$ is 1, then $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \rho)$ belongs to T_d^y. Otherwise, it does not. The sets T_s^y, T_n^y, T_s^x, T_n^x, T_d^x, C_s, C_n and C_d are defined similarly.

We are now ready to define the FOC-interpretation Ψ. Given a graph G with the set of vertices $V(G)$ of size n and the set of edges $E(G)$ it does the following:

1. defines the domain of $\bar{I}$ as $[n]^k \times V(G)^{(k)}$, the domain of $\bar{J}$ as $V(G)^2 \cup [n]^k \times V(G)^{(k)}$, the domain of $\bar{J}'$ as $V(G)^2$, and the domain of $\bar{B}$ as $[n]^k$,
2. defines the relation $\leq$ for the linear order on $\bar{B}$ as the lexicographic order with respect to the natural order of the number domain,
3. defines the relation F of type $\bar{J}' \times \bar{J}$ as the equality relation on $V(G)^2$,
4. defines the ternary relation $P_d^{\mathbf{A}}$ of type $\bar{I} \times \bar{J} \times \bar{B}$ for encoding the denominators of the entries of the matrix $\mathbf{A}$ as a union of two relations. The first one encodes the denominators of the coefficients of the auxiliary variables in $\bar{P}_G$ and is defined as a subset of $([n]^k \times V(G)^{(k)}) \times ([n]^k \times V(G)^{(k)}) \times [n]^k$ consisting of tuples $(\mathbf{s}_1, \mathbf{v}_1, \mathbf{s}_2, \mathbf{v}_2, \mathbf{s}_3)$ for which there exists $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \rho)$ in T_d^y such that the tuple $(\mathbf{v}_1, \mathbf{v}_2)$ satisfies ρ, and for every $i \in [3]$ it holds $\mathbf{s}_i = \mathbf{z}_i$. The second one encodes the denominators of the coefficients of the variables $\{x_{vw}\}_{v,w \in V(G)}$ in $\bar{P}_G$ and is defined as a subset of $([n]^k \times V(G)^{(k)}) \times V(G)^2 \times [n]^k$ consisting of tuples $(\mathbf{s}_1, \mathbf{v}_1, v, w, \mathbf{s}_2)$ for which there exists $(\mathbf{z}_1, \mathbf{z}_2, \rho)$ in T_d^x such that the tuple $(\mathbf{v}_1, v, w)$ satisfies ρ and $\mathbf{s}_1 = \mathbf{z}_1$, and $\mathbf{s}_2 = \mathbf{z}_2$,
5. defines the relations $P_s^{\mathbf{A}}$, $P_n^{\mathbf{A}}$, $P_s^{\mathbf{b}}$, $P_n^{\mathbf{b}}$, $P_d^{\mathbf{b}}$ in a similar way as $P_d^{\mathbf{A}}$,
6. defines the binary relations $P_s^{\mathbf{a}}$, $P_n^{\mathbf{a}}$, $P_d^{\mathbf{a}}$ of type $\bar{J}' \times \bar{B}$ for encoding the entries of the vector $\mathbf{a}$ in the following way: the entries $((v, w), \mathbf{s})$ of $P_s^{\mathbf{a}}$, $P_n^{\mathbf{a}}$, $P_d^{\mathbf{a}}$ are defined to encode $1 = (-1)^0$ 1/1 or $0 = (-1)^0$ 0/1 depending on whether $(v, w) \in E(G)$ or not.

Note that by existential quantification over the sets T_d^y and T_d^x we really mean a disjunction. And by $\mathbf{s}_i = \mathbf{z}_i$ we mean the 2-variable FO-formula of size $O(kn)$ which, for every $j \in [k]$, says that the j-th component $s_{i,j}$ of $\mathbf{s}_i$ is the $z_{i,j}$-th smallest element of $[n]$, using the order on the number domain. Observe also that Ψ, as described, is not rigorously an FOC-interpretation, but it is not difficult to see that it can be easily turned into such.

The interpretation Ψ has $O(k)$ variables. Its size is polynomial in n^k, in k, and in the number of equality types of $2k$ tuples, that is, polynomial in n^k, k, and $(2k)^{2k}$. Since in our case $k = O(n)$, the size of Ψ is simply $n^{O(k)}$.

Now by composing Ψ with the FPC formula ϕ from Lemma 14 we obtain an FPC formula ψ which given a graph G with n vertices decides if G has the property of interest. The formula ψ has $l = O(k)$ variables and size $n^{O(k)}$. We translate it into a formula θ of C^{2l} such that ψ is equivalent to θ on all structures of size at most n and θ is of size polynomial in the size of ψ, in l, and in n^l (cf. Subsection 2.2). Hence, in terms of k and n, the formula θ has $O(k)$ variables and size $n^{O(k)}$.

We have therefore shown that a property of graphs with n vertices recognized by a graph-symmetric polytope lift of size s is defined by a C^k formula, where $k = O(\log(s)/(\log(n) -$

$\log\log(s)))$. Moreover, if s is at most weakly exponential, then for some positive real ϵ we have $k = O(\log(s)/(\log(n) - \log\log(s))) = O(\log(s)/(\epsilon\log(n))) = O(\log(s)/\log(n))$. Hence, in this case the size of θ is $n^{O(k)} = s^{O(1)}$. This finishes the proof of Lemma 6 and this section.

5 Results and Applications

In this section we develop the main consequences of our results. We start by establishing the main theorem of the paper, which characterizes the expressive power of symmetric linear programs. We continue with the applications to upper and lower bounds. And end with the observation that for random graphs over appropriate distributions symmetric LP lifts are as powerful as general Boolean circuits.

5.1 Equivalence of Models

If $\mathscr{C}$ is a class of finite L-structures of some single-sorted vocabulary L, and n is a positive integer, we write $\mathscr{C}_n$ for the set of all structures in $\mathscr{C}$ of cardinality n. We write $\mathrm{s}_{\mathscr{C}}(n)$ for the size of a smallest L-symmetric Boolean circuit that recognizes $\mathscr{C}_n$, and $\mathrm{lp}_{\mathscr{C}}(n)$ for the size of a smallest L-symmetric LP lift that recognizes $\mathscr{C}_n$. Similarly, we write $\mathrm{w}_{\mathscr{C}}(n)$ for the *counting-width* of $\mathscr{C}_n$, i.e., the smallest number of variables k of a C^k-formula that defines $\mathscr{C}_n$ on L-structures of cardinality n, and $\mathrm{sw}_{\mathscr{C}}(n)$ for the *counting size-width* of $\mathscr{C}_n$, i.e., the smallest k such that there is a C^k-formula of size at most n^k that defines $\mathscr{C}_n$ on L-structures of cardinality n.

Theorem 1. *Let $\mathscr{C}$ be a class of finite L-structures of some vocabulary L. If $\mathrm{lp}_{\mathscr{C}}(n)$ is at most weakly exponential, then*

1. $\mathrm{s}_{\mathscr{C}}(n)^{\Omega(1)} \leq \mathrm{lp}_{\mathscr{C}}(n) \leq \mathrm{s}_{\mathscr{C}}(n)^{O(1)}$,
2. $\Omega(\mathrm{sw}_{\mathscr{C}}(n)) \leq \log(\mathrm{lp}_{\mathscr{C}}(n))/\log(n) \leq O(\mathrm{sw}_{\mathscr{C}}(n))$.

Proof. The upper bound in *1* is a direct consequence of Lemma 1, and this holds without any assumption on the growth rate of $\mathrm{lp}_{\mathscr{C}}(n)$. The lower bound in *2* follows from Lemma 6: Write $s = \mathrm{lp}_{\mathscr{C}}(n)$ and choose $k = c\log(s)/(\log(n) - \log\log(s))$ for a large c to be specified later. Since $\mathrm{lp}_{\mathscr{C}}(n)$ is at most weakly exponential we have $s \leq 2^{n^{1-\epsilon}}$ for some $\epsilon > 0$ and large enough n. Hence $k = O(\log(s)/\log(n))$ with the hidden constant in the big-oh notation dependent on ϵ as in $1/\epsilon$. Now, for the appropriate constant in the big-oh in $k = O(\log(s)/\log(n))$, Lemma 6 says that there is a C^k-formula that defines $\mathscr{C}$ and has size polynomial in s, since again $\mathrm{lp}_{\mathscr{C}}(n)$ is at most weakly exponential. If the constant in the big-oh in $k = O(\log(s)/\log(n))$ is chosen big enough, we get that the size polynomial in s is bounded even by n^k, so $\mathrm{sw}_{\mathscr{C}}(n) = O(\log(s)/\log(n))$ as was to be proved. In turn, these two imply the lower bound in *1* and the upper bound in *2* through the well-known relationship $\mathrm{s}_{\mathscr{C}}(n) \leq n^{O(\mathrm{sw}_{\mathscr{C}}(n))}$ (see [36]). □

5.2 Upper and Lower Bounds

By Theorem 1, any class of graphs of unbounded counting width cannot be recognized by polynomial-size symmetric LP lifts. More strongly, in combination with the strongest known lower bounds on counting width, Theorem 1 gives weakly exponential lower bounds of the type $2^{\Omega(n^{1-\epsilon})}$. We show that the strongest forms of Lemmas 1 and 6 give even larger lower bounds.

Lower bounds on symmetric lifts and circuits In the sequel, let 3-XOR refer to the constraint satisfaction problem of deciding whether a system of 3-variable parity constraints on $\{0,1\}$-valued variables is satisfiable, and let 3-SAT refer to the satisfiability problem for 3-CNF formulas. In both cases, an instance is presented as a finite structure that encodes the incidence structure of the constraints: the domain is the disjoint union of the set of variables and the set of constraints, there is one monadic relation for each type of constraint that indicates which constraints are of that type, and three binary relations that indicate the three variables that participate in each constraint. Note that the instances for these problems are not plain graphs but graphs with coloured vertices and edges.

Theorem 2. *Every graph-symmetric LP lift or Boolean threshold circuit that recognizes the class of Hamiltonian graphs with n vertices, or the class of 3-colourable graphs with n vertices, or the class of satisfiable 3-SAT instances with n variables, or the class of satisfiable 3-XOR instances with n variables, has size $2^{\Omega(n)}$. Moreover, for 3-colouring, 3-SAT, and 3-XOR, the lower bound holds even on the class of instances with $O(n)$ edges, $O(n)$ clauses, and $O(n)$ constraints, respectively.*

Before we enter the proof let us note that these $2^{\Omega(n)}$ lower bounds for 3-colouring, 3-XOR and 3-SAT are optimal up to the multiplicative constant in the exponent. We discuss this later in this section. Now we turn to the proof of Theorem 2. First we handle 3-colourability, then Hamiltonicity. As intermediate steps towards both we do 3-XOR and 3-SAT.

By Lemma 6, for obtaining the lower bound for LP lifts it suffices to show that any C^k-sentence that defines the class of n-vertex 3-colourable graphs has $k = \Omega(n)$: indeed, whenever $s \leq 2^{n/d}$, we have

$$\log(s)/(\log(n) - \log\log(s)) \leq n/(d\log(d)). \tag{12}$$

By Lemma 1, the claim then follows for Boolean threshold circuits. A result from the literature that is quite close to the $k = \Omega(n)$ that we need can be found in Section 4.2 in [15], but the analysis in there gives $k = \Omega(\sqrt{n})$, and not $k = \Omega(n)$. While it should be possible to modify the construction in [15] to get what we need, we refer to a more recent construction that achieves what we want for the problems 3-XOR and 3-SAT, and then proceed by reduction. These intermediate steps will also be useful when we discuss Hamiltonicity. In what follows, the degree of a variable in a 3-XOR or 3-SAT instance is the number of times that the variable appears in some constraint. It is clear that an instance that has all its degrees bounded by a constant has $O(n)$ many constraints, where n is the number of variables.

Theorem 3 (see Theorem 3.7 and 3.8 in [6] and Lemmas 22 and 23 in [17]). *There exist $c > 0$ and $d > 0$ such that, for every k and every sufficiently large n, every C^k-sentence that separates the class of satisfiable 3-XOR (resp. 3-SAT) instances with n variables and degrees bounded by c from the class of unsatisfiable ones has $k \geq dn$.*

Neither [6] nor [17] state the constant bound on the degrees, but it easily follows from both proofs. Concretely, it follows from the proof of Lemma 3.7 in [6] by taking the bipartite unique-neighbour expander featuring there to have bounded degree on both sides. Such expanders exist by the comments in the first and last bullets in page 83 of [43] and the known relationship between lossless expanders and unique-neighbour expanders (see Problem 4.10 in [43]). It also follows from the proof of Lemma 22 in [17] since the graphs of large treewidth that feature there are indeed 3-regular, and the construction from [5] on which the proof of Lemma 22 is based turns graphs of bounded degree into instances of bounded degree. Now we proceed by reduction in order to get the same result for 3-colouring:

Lemma 15. *There exist $c > 0$ and $d > 0$ such that, for every k and every sufficiently large n, every C^k-sentence that separates the class of 3-colourable graphs with n vertices and cn edges from the class of non-3-colourable ones has $k \geq dn$.*

Proof. In the textbook reduction from 3-SAT to the problem of deciding whether a graph is 3-colourable (see, e.g., [37]), the output graph has one gadget with two vertices for each variable in the input formula, one gadget with six vertices for each constraint of the input formula (the reduction in [37] uses only three vertices for each constraint because the reduction starts at NAE-SAT; starting at 3-SAT we need six vertices), and one special vertex. The edges are local to each gadget, plus two edges from each variable gadget to the special vertex, and a constant number of edges for each variable occurrence in the input formula between the constraint gadget where the variable appears, and the corresponding variable gadget. There are no other vertices or edges in the graph. It is clear from the construction that if the input formula has n vertices and m constraints, then this graph has $O(n) + O(m)$ vertices and $O(n)+O(m)+O(m)$ edges, since each of the m constraints contributes three occurrences. And it is not difficult to see that any C^k formula that separates the 3-colourable graphs that are output by the reduction from the non-3-colourable ones can be converted into a $\mathrm{C}^{O(k)}$ formula that separate the satisfiable 3-SAT instances that are input to the reduction from the unsatisfiable ones. Another way to see this is by noting that the reduction is definable by a uniform quantifer-free interpretation (without the need for any ordering, or parameters, on the input structure), from which the claim on $\mathrm{C}^{O(k)}$-definability follows from the closure of the logic under quantifier-free interpretations (see Lemma 2.1 in [6]). Now the claim follows from Theorem 3. □

For Hamiltonicity we follow the same path. A well-known result of Dahlhaus [14] gives a first-order definable reduction from SAT to Hamiltonicity that does not require any linear order on the input. However, that reduction is quadratic and would only achieve a lower bound of the form $k = \Omega(\sqrt{n})$ for n-vertex graphs. We work out a linear reduction from 3-SAT:

Lemma 16. *There exists $d > 0$ such that, for every k and every sufficiently large n, every C^k-formula that defines the class of Hamiltonian graphs with n vertices has $k \geq dn$.*

Proof. In this case we need a small modification of the textbook reduction from 3-SAT to the problem of deciding whether a directed graph contains a Hamilton cycle. Composing this with the straightforward reduction to undirected graphs will give the result. For the first reduction we modify the construction in Theorem 7.35 of [42]. For the second we can use the construction in Theorem 7.36 of [42] without change.

Given a 3-CNF formula with n variables and m clauses, the output directed graph of the textbook reduction has one vertex for each clause, one variable-gadget of size $O(m)$ for each variable, and $O(n)$ additional vertices. See Figures 7.14–7.17 in [42]. The first minor modification we do is that the variable-gadget of variable x_i will be a path of length $3c+3$, where $c = 2d$ and d is a bound on the number of times each literal appears in the formula, instead of length $3m+3$ as in [42]. Each such gadget is made of d many *positive occurrence pairs* of vertices, followed by d many *negative occurrence pairs*, each pair separated from one another and from the two endpoints by $c+1$ many *separator* nodes. See the top part of Figure 1 and ignore, for now, the bottom clause-vertices. If nothing else is modified, this more economical way of setting up the variable-gadgets leaves the correctness of the reduction intact. The effect it has is that if the input 3-CNF formula has all its degrees bounded by a constant c, then the new directed graph has a total of $O(n)$ instead of $O(nm)$ many vertices.

This is not yet our reduction because, in order to define the edges of this graph, one needs a linear order on the variables of the input formula to create the path of variable-gadgets as in Figure 7.14 of [42]. Similarly, one also needs a linear order on the clauses to encode the literals as in Figures 7.16–17 of [42]. Such linear orders are not available in our encoding (and cannot be made available in Theorem 3 as otherwise its proof breaks down). Here is where we modify the construction.

First, instead of aligning the variable-gadgets in the form of a path, we arrange them in the form of a big clique as in Figure 1. Second, for each variable x_j, we connect the first d occurrence pairs of x_j with all the clause-vertices of the clauses in which x_j occurs positively as in Figure 7.16 of [42], and the next d occurrence pairs of x_j with all the clause-vertices of the clauses in which x_j occurs negatively as in Figure 7.17 of [42]. In order to tell whether an occurrence pair is on the positive or negative side of the gadget, we let all the left endpoints of the gadgets be distinguished as *left-endpoint*, and the right endpoints of the gadgets be distinguished as *right-endpoint*. The edges that go from a variable-gadget vertex into a clause-vertex are called *down edges*, and those that go from clause-vertex into a variable-gadget vertex are called *up edges*. See again Figure 1.

Besides shortening the gadgets and arranging them in the form of a clique, the only other difference between the construction we described in the previous paragraph and the one in [42] is that we connect the occurrence pairs of a literal with the clause-vertices where it occurs in the shape of a complete bipartite graph instead of in the shape of a matching. This modification does not introduce any new vertices and its main feature is that it does not depend on any given ordering of the variables or clauses of the input formula. In particular, as in the proof of Lemma 15 but crucially relying on the assumption that c is a constant, it is not hard to see that this reduction is definable by a uniform quantifier-free interpretation without parameters or linear orders. Thus, once we show that the reduction is correct, the lemma will follow from Theorem 3 and the closure of the logic under quantifier-free

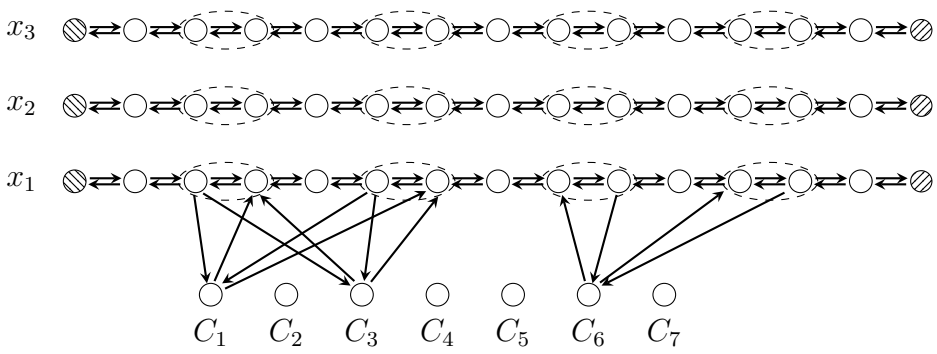

Figure 1: The (end-points of the) variable-gadgets are arranged in the form of a big clique: all striped vertices are connected by all possible directed edges between them. All variables in this instance appear at most twice positively and at most twice negatively. Therefore, $d = 2$ and $c = 4$, and each variable-gadget has 4 occurrence pairs, 5 separator vertices, and 2 endpoints, for a total of 15 vertices. Variable x_1 appears twice positively, in clauses C_1 and C_3, and once negatively in clause C_6. To reduce clutter, the arrows of the variable-gadgets for x_2 and x_3 are not displayed. The left-striped vertices are distinguished from the right-striped vertices so that we can tell which side of the gadget corresponds to the positive occurrence pairs and which side corresponds to the negative occurrence pairs.

interpretations.

To complete the proof of the lemma we show, by an analysis that refines the one in [42], that the reduction is correct.

The argument to show that if the 3-CNF formula is satisfiable then the graph has a Hamilton cycle is the same as in [42]. Any satisfying assignment gives a way to traverse each variable-gadget: traverse it from left-to-right if the variable is set to true, traverse it from right-to-left if the variable is set to false, and visit each clause-vertex through a down-up pair of edges from a uniquely selected occurrence pair whose corresponding literal satisfies the clause. The traversals of the variable-gadgets are connected one to the next, in arbitrary order, through the edges of the big clique that connects their endpoints. Clearly, this is a Hamilton cycle.

What requires modification is the argument to show that if the graph has a Hamilton cycle then the 3-CNF formula is satisfiable. Similarly as in [42], let us say that a Hamilton cycle is *normal* if, for each variable-gadget, either all occurrence pairs in the gadget are traversed from left-to-right, or they are all traversed from right-to-left. Slightly deviating from [42], and due to the redundant edges that ensure the symmetry of our reduction, we need to be a little more precise on the types of left-to-right or right-to-left traversals that are allowed.

We say that an occurrence pair (u, v) is *traversed from left-to-right* by the Hamilton cycle if the cycle uses the left-to-right edge between them, or if (u, v) is a positive occurrence pair and the cycle uses one of the down arrows that leaves u and one of the up arrows that enters v. Dually, we say that an occurrence pair (u, v) is *traversed from right-to-left* if the cycle uses the right-to-left edge between them, or if (u, v) is a negative occurrence pair and

the cycle uses one of the down arrows that leave v and one of the up arrows that enters u. Note that, in either case, we do not require the up and down arrows to be consecutive edges in the cycle; they need not even belong to the same clause-vertex. This is the only difference between our more relaxed notion of normal cycle and the notion of normal cycle in [42]. The point is, however, that each normal cycle in our relaxed sense still gives a satisfying assignment: if the variable-gadget of x_i is traversed from left-to-right, we set x_i to true; if the variable-gadget of x_i is traversed from right-to-left, we set x_i to false. Since each clause-vertex must be visited exactly once and the only two ways it can be accessed is through the down arrow of a left-vertex of an occurrence pair of a variable whose gadget is traversed from left-to-right, or through the down arrow of a right-vertex of an occurrence pair of a variable whose gadget is traversed from right-to-left, it follows that all clauses are satisfied by the assignment.

We are left to show that all Hamilton cycles must be normal in our sense. In order to see this, assume for contradiction that some Hamilton cycle is not normal and let (u_1, v_1) and (u_2, v_2) be two occurrence pairs that are traversed in different directions in the same variable-gadget and that are as close as possible within that gadget. Without loss of generality, let us say that u_1, v_1, u_2, v_2 appear in this left-to-right order in the variable-gadget. Since we chose the occurrence pairs to be as close as possible, they must be consecutive in the gadget. This means that v_1 is the right-vertex of an occurrence pair, that u_2 is the left-vertex of an occurrence pair, and that both are connected to a separator vertex w inbetween. Moreover, since the pairs are traversed in different directions, the only two possibilities are (1) that (u_1, v_1) is traversed left-to-right and (u_2, v_2) is traversed right-to-left, or (2) that (u_1, v_1) is traversed right-to-left and (u_2, v_2) is traversed left-to-right. Note, however, that case (2) is not possible since in that case the cycle would leave v_1 towards a vertex other than w and, likewise, the cycle would leave u_2 towards a vertex other than w, which means that w would not be visited.

Assume then that (u_1, v_1) is traversed left-to-right and that (u_2, v_2) is traversed right-to-left. If (u_1, v_1) is a positive occurrence pair, then the cycle leaves v_1 through w and then through u_2, which contradicts the assumption that (u_2, v_2) is traversed right-to-left. Similarly, if (u_2, v_2) is a negative occurrence pair, then the cycle leaves u_2 through w and then through v_1, which contradicts the assumption that (u_1, v_1) is traversed left-to-right. Finally, since all positive occurrence pairs appear to the left of all negative occurrence pairs within the gadgets, these two cases cover all possibilities and the proof is complete. □

Proof of Theorem 2. Apply Theorem 3 and Lemmas 15 and 16 to Lemma 6, for LP lifts, and then to Lemma 1, for circuits, and in both cases use equation (12). □

Lower Bound on the TSP Polytope As stated in the introduction, Yannakakis proved that the travelling salesman polytope does not have symmetric LP lifts with subexponentially many variables and facets. In particular, it does not have symmetric LP lifts of subexponential size. Here we show that this lower bound on the size of symmetric LP lifts follows from Theorem 2. In the next section we will see that the same type of argument cannot work for the matching polytope.

If $G = (V, E)$ is a graph with $V = [n]$, the incidence vector of G is $\mathbf{x}^G = (x^G_{ij} : i, j \in [n])$,

the vector in $\mathbb{R}^{[n]^2}$ defined by $x^G_{ij} = 1$ if (i,j) is an edge of G, and $x^G_{ij} = 0$ if (i,j) is not an edge of G. Let TSP_n denote the convex hull of all the vectors of the form $\mathbf{x}^C$, where C is a Hamilton cycle of the complete graph K_n on n vertices. We want to show that every graph-symmetric LP lift that has TSP_n as shadow must be of exponential size as a consequence of Theorem 2.

Theorem 4 (follows from Theorem 2 in [47])**.** *Every graph-symmetric LP lift that has* TSP_n *as shadow has size* $2^{\Omega(n)}$*.*

Proof. By Theorem 2, it suffices to show that if TSP_n were the shadow of a subexponential-size graph-symmetric LP lift, then there would be a subexponential-size graph-symmetric LP lift that separates the $\mathbf{x}^G$ for which G is Hamiltonian from the $\mathbf{x}^G$ for which G is not Hamiltonian. Assume then that P were symmetric LP lift of size $2^{o(n)}$ whose shadow is TSP_n; let us say that its principal variables are $\mathbf{y} = (y_{ij} : i,j \in [n])$ and that its auxiliary variables are $\mathbf{z} = (z_k : 1 \leq k \leq p(n))$. Let Q be the following LP on the x_{ij}, y_{ij} and z_k variables:

$$\begin{array}{ll} 0 \leq y_{ij} \leq x_{ij} & \text{for each } i,j \in [n] \\ (\mathbf{y},\mathbf{z}) \in P. & \end{array} \tag{13}$$

Since by assumption P is graph-symmetric, Q is also graph-symmetric. Hence, it suffices to show that the projection of Q on the x-variables separates Hamiltonian graphs from non-Hamiltonian graphs.

It is clear that if G is a graph that contains a Hamilton cycle C, then $\mathbf{x}^G$ is in the projection of Q on the x-variables: choose $\mathbf{y} = \mathbf{x}^C$, and let $\mathbf{z}$ witness its membership in P. Conversely, assume that G is a graph and that $\mathbf{x}^G$ is in the projection of Q on the x-variables. Then there exists $\mathbf{y}^* = (y^*_{ij})_{ij}$ that is in the TSP polytope and satisfies the inequalities $0 \leq y^*_{ij} \leq x^G_{ij}$ for each $i,j \in [n]$. This means that the support of $\mathbf{y}^*$ defines a subgraph of G, and at the same time that $\mathbf{y}^*$ is a convex combination of Hamilton cycles of K_n. Let $\mathbf{y}^C$ be one of the vectors in this convex combination, where C is a Hamilton cycle of K_n. The support of $\mathbf{y}^C$ is of course included in the support of $\mathbf{y}^*$, which means that C is also a subgraph of G. So G contains a Hamilton cycle and is thus Hamiltonian. □

It should be noted that the lower bound stated in Theorem 4 is weaker than Yannakakis' Theorem 2 in two respects: 1) our lower bound applies to LP lifts that are symmetric in the syntactic sense of our definition, and 2) it applies to the size of the lift measured by number of bits. In contrast, Yannakakis' proof applies to arbitrary LP lifts that define symmetric polytope lifts, and, as remarked in page 448 in the beginning of the proof of Theorem 1 in [47], to the size of the lift measured by the number of variables and constraints.

Upper bounds Let us start with the simple observation that the lower bounds of type $2^{\Omega(n)}$ on 3-colouring, 3-SAT and 3-XOR are optimal: all three cases can be solved by symmetric Boolean threshold circuits of size $2^{O(n)}$, and hence by symmetric LP lifts of size $2^{O(n)}$ by Lemma 1. Here n is the number of vertices or variables, respectively. This follows from the fact that all three problems are definable in the existential fragment of monadic second-order logic (i.e., monadic NP), which on structures of size n, straightforwardly translate into symmetric Boolean circuits of size $2^{O(n)}$. For Hamiltonicity, the straightforward symmetric upper

bound is only $2^{O(n \log n)}$ on n-vertex graphs. Although it looks plausible that the dynamic programming algorithms of Bellman [11] and Held-Karp [24] that run in time $O(n^2 2^n)$ could be implemented in a symmetric Boolean circuit, we are not aware of a reference where this has been worked out. What is known is that Hamiltonicity is not definable in (full) monadic second-order logic (see [19]).

The list of problems for which we proved a lower bound in Theorem 2 includes one that is decidable in polynomial time, i.e., 3-XOR. This means that any polynomial-size family of LP lifts or threshold circuits that recognizes 3-XOR must *a fortiori* be asymmetric. On the other hand, Theorem 1 says that any problem that is definable in FPC has polynomial-size symmetric LP lifts. This includes graph planarity [22], any polynomial-time decidable property of graphs that exclude some minor [23], matrix singularity over rationals [12], solving systems of linear equations over rationals [27], and many others. By the results in [4], it also includes the problem of deciding whether a (general, not necessarily bipartite) graph contains a perfect matching. The family can even be taken to be polynomial-time uniform by applying the construction of Lemma 1 to the "easy half" of the equivalence between FPC and polynomial-size symmetric threshold circuits in [3].

Corollary 2. *There is a (polynomial-time uniform) family of graph-symmetric LP lifts of polynomial size that recognizes the class of graphs that have a perfect matching.*

This should be contrasted with the fact, proved by Yannakakis, that any symmetric LP lift of the perfect matching polytope PM_n has size $2^{\Omega(n)}$. Here, PM_n is defined as the convex hull of all vectors of the form $\mathbf{x}^M$, where M is the edge set of a perfect matching of the complete graph K_{2n} on $2n$ vertices. Capturing PM_n by an LP lift or recognizing the class of graphs that have a perfect matching by an LP lift are different tasks. Both objects could be used for deciding whether a given graph has a perfect matching, but capturing PM_n has a demanding structural requirement that has no analogue in the other task. The upper bound of Corollary 2 also means that the argument that was used for deriving lower bounds for TSP_n in the proof of Theorem 4 cannot be adapted to PM_n. Indeed, we do not know whether there is any route at all for deriving lower bounds for PM_n via our results.

In view of Corollary 2 one may wonder whether the convex hull of the incidence vectors of graphs that have a perfect matching has a small symmetric LP lift. This, however, is easily seen to not be the case: if it had, then its intersection with the halfplane $\sum_{i,j \in [2n]: i \neq j} x_{ij} = 2n$ would be a small symmetric LP lift for PM_n.

5.3 Problems on Erdős-Rényi Random Graphs

Let $\mathscr{G}(n,p)$ denote the Erdős-Rényi distribution on n-vertex labelled graphs with edge probability p. We write $G \sim \mathscr{G}(n,p)$ to mean that G is a random graph distributed as in $\mathscr{G}(n,p)$. In this section we argue that, for average-case problems with respect to the uniform distribution $\mathscr{G}(n,1/2)$, as well as for the type of problems that ask to distinguish $\mathscr{G}(n,1/2)$ from some other distribution, polynomial-size symmetric LPs are as powerful as arbitrary not necessarily symmetric Boolean circuits. For average-case problems, this is indeed a direct consequence of our main result and the following well-known fact in descriptive complexity theory:

Theorem 5 (Corollary 4.8 in [25]). *For every polynomial-time decidable class of graphs $\mathscr{C}$ there is an* FPC*-definable class of graphs $\mathscr{C}'$ for which the probability that a random graph $G \sim \mathscr{G}(n, 1/2)$ falls in the symmetric difference $\mathscr{C} \Delta \mathscr{C}'$ is $o(1)$.*

The point of Theorem 5 is that the FPC formula that defines $\mathscr{C}'$ does not require any order on the input graph, hence our Theorem 1 applies. Theorem 5 is indeed a consequence of the Immerman-Vardi Theorem [29, 44] and the fact that a linear order is, asymptotically almost surely on $\mathscr{G}(n, 1/2)$, definable in FPC. We return to this later. For the rest of this section let us focus our discussion on the problem of distinguishing $\mathscr{G}(n, 1/2)$ from some other distribution of random graphs, to which a direct application of Theorem 5 does not look possible.

We focus the discussion on the planted clique problem since it is one of the best studied such problems, although it will be clear from the discussion that the phenomenon is more general. Let $\mathscr{G}(n, p, k)$ denote the distribution that results from drawing a random graph from $\mathscr{G}(n, p)$ and then *planting* a random k-clique in it, i.e., adding the edges of a k-clique on a uniformly chosen subset of k vertices. Following [10], the planted clique problem, also known as the *hidden* clique problem, comes in three flavours. Informally stated, these are:

- search: given $G \sim \mathscr{G}(n, p, k)$, find the planted clique,
- refutation: given $G \sim \mathscr{G}(n, p)$, certify that the clique number is less than k,
- decision: given $G \sim \mathscr{G}(n, p)$ or $G \sim \mathscr{G}(n, p, k)$, determine which is the case.

We focus on the decision version which, formally, can be stated as follows. We say that a class of graphs $\mathscr{C}$ solves the planted clique problem with parameters $p = p(n)$ and $k = k(n)$ and advantage $\epsilon = \epsilon(n) > 0$ if for every large enough n the following hold:

1. if $G \sim \mathscr{G}(n, p)$, then G is in $\mathscr{C}$ with probability at least $1/2 + \epsilon$,
2. if $G \sim \mathscr{G}(n, p, k)$, then G is in $\mathscr{C}$ with probability at most $1/2 - \epsilon$.

Solvable in polynomial time (or in FPC, or by a family of LP lifts, etc.) means that it is solvable by a class of graphs $\mathscr{C}$ that can be recognized in polynomial time (or in FPC, or by a family of LP lifts, etc.). It should be clear that if the decision version is hard, then the other two versions of the problem can only be harder.

The planted clique problem is an easy-to-state average case problem that has a long history. Its formulation is attributed to Jerrum [31] and Kučera [33], independently. In the range $k(n) = \omega(\sqrt{n \log n})$, a simple algorithm based on degree sequences solves the search problem in polynomial time [33]. In the range $k(n) = \Omega(\sqrt{n})$, a spectral based algorithm is known to solve the problem in polynomial time [2], but spectral-free algorithms are also known that run even in linear time [20]. For $k(n) = o(\sqrt{n})$ the status of the problem is famously open, even for its decision variant. Some lower bounds are known in restricted models, such as the original lower bound for Markov chain methods in [31]. Lower bounds are also known for (symmetric) linear and semidefinite program formulations of the problem. We discuss these next.

We formulate the clique problem on a graph $G = (V, E)$ as a non-linear polynomial optimization problem: For each vertex $v \in V$ introduce one variable y_v that stands for the

indicator whether v belongs to the clique. The clique number is the maximum of $\sum_{v\in V} y_v$ subject to the constraints that $y_u y_v = 0$ for each non-edge $(u,v) \notin E$, and $y_v^2 - y_v = 0$ for each $u \in V$. For the refutation and the decision versions of the problem, it is more natural to turn the objective function into a constraint $\sum_{v\in V} y_v \geq k$. This gives a different quadratic program feasibility problem for each graph G. Thinking of G as given by the $\{0,1\}$-vector $(x_{u,v})_{u,v\in[n]}$ in the usual way, the programs can be made uniform, i.e., a single quadratic program feasibility problem serves all graphs with $V = [n]$:

$$\begin{array}{ll} \sum_{v\in[n]} y_v \geq k & \\ y_u y_v \leq x_{u,v} & \text{for } u,v \in [n] \\ y_v^2 - y_v = 0 & \text{for } v \in [n] \end{array}$$

While this a hard-to-solve quadratic program, there are several tractable relaxations that one can study. Two systematic methods for generating such relaxations were introduced by Lovász and Schrijver in [35], and Sherali and Adams in [41]. Both cases start by (naively) *linearizing* the quadratic program, i.e., replacing each quadratic term $y_u y_v$ in the constraints by a new variable $y_{\{u,v\}}$. The result is a very weak symmetric LP relaxations with auxiliary variables, but this is only the *first level* of the LS and SA hierarchies. The successive levels of the hierarchies are obtained by adding linear constraints that can be proved valid for the convex hull of solutions of the quadratic program by iterated applications of a simple rule of inference.

It can be seen that the levels of the hierarchies are always graph-symmetric LPs, with the d-th level having $n^{O(d)}$ auxiliary variables and constraints, where $n = |V|$. The strengths of the relaxations converge to the quadratic program in the sense that the polytopes they project to are tighter and tighter approximations of the convex hull of solutions of the quadratic program. Moreover, the exact convex hull is eventually reached no later than the n-th level.

In order to appreciate the symmetry of the LPs that define the sucessive levels of the hierarchy, it useful to enter the details for the definition of the SA hierarchy. The first step in producing the d-th level of the SA hierarchy is to formally multiply each constraint of the quadratic program by an extended monomial of the form $\prod_{w\in I} y_w \prod_{w\in J}(1-y_w)$ for $I, J \subseteq V$ with $I \cap J = \emptyset$ and $|I \cup J| \leq d-1$. The second step is to expand out the formal expressions into sums of degree $d+1$ monomials, and multilinearize every monomial on the y-variables; note that this is a valid step only under the constraint $y_v^2 - y_v = 0$, hence for $\{0,1\}$-assignments. In the third step we introduce a new variable y_I for each $I \subseteq V$ with $|I| \leq d+1$, and a new variable $y_{u,v,I}$ for each $u, v \in V$ and each $I \subseteq V$ with $|I| \leq d-1$. The fourth and final step is to linearize: replace each monomial $\prod_{w\in I} y_w$ by the new variable y_I, and each monomial $x_{u,v}\prod_{w\in I} y_w$ by the new variable $y_{u,v,I}$. The link between the two types of variables is established by setting $x_{u,v} = y_{u,v,\emptyset}$ and $y_\emptyset = 1$. The symmetry of the resulting LP is obvious since the image $I^{(\pi)}$ of a set $I \subseteq V$ by a permutation $\pi \in \mathrm{Sym}_V$ has the same cardinality as I. The result is a graph-symmetric LP with $n^{O(d)}$ auxiliary variables and constraints, whose shadow eventually captures the convex hull of solutions of the quadratic program. Any level that approximates this hull well enough, even on average on $\mathscr{G}(n,1/2)$, can be used for solving the planted clique problem.

The strengths and limitations of the LS and SA hierarchies (and beyond) for the planted clique problem have been the object of intensive study. Starting with the work of Feige and Krauthgamer [21], it is known that for each constant d, the d-th level of the LS hierarchy cannot distinguish between $\mathscr{G}(n, 1/2)$ and $\mathscr{G}(n, 1/2, k(n))$ when $k(n) = o(\sqrt{n})$ with any significant advantage. Their lower bound is much stronger than we stated it since it applies even to LS$^+$, the semi-definite programming variant of the LS-hierarchy. For the SA hierarchy, the same result is attributed to the folklore, and more recent works [9, 28] have obtained analogous results for the even stronger Lasserre/Sums-of-Squares (SOS) hierarchies, also proving that the problem stays hard for their constant d level when $k(n) = o(\sqrt{n})$. Further, in certain contexts, it is possible to prove that the Sherali-Adams hierarchy is *optimal* among symmetric LP lifts of comparable size. This includes a model of LP lifts for solving Boolean constraint satisfaction problems (see Theorem 4.1 in [13]). This means that, in such restricted contexts, proving lower bounds on the levels of Sherali-Adams is enough for getting size lower bounds for *any* symmetric LP lift.

In view of such success in proving lower bounds on the size of symmetric LP lifts, starting with Yannakakis, and including the discussion above on hierarchies for the planted clique problem, and also given our own lower bounds from Section 5.2, the following consequence of Theorem 1 may come as a surprise:

Corollary 3. *If the planted clique problem with parameters $p = 1/2$ and $k = k(n)$ is solvable in polynomial time with advantage $\epsilon > 0$, then it is also solvable by a (polynomial-time uniform) family of polynomial-size graph-symmetric LP lifts with advantage $\epsilon - o(1)$.*

In the rest of this section we show how to derive the descriptive complexity version of Corollary 3, from which Corollary 3 follows at once from Theorem 1. The descriptive complexity variant relies on the following well-known fact, which builds on the almost sure graph canonization methods of [8]:

Theorem 6 (Theorem 4.6 in [25])**.** *There is an* FOC*-formula $\phi(x, y)$ such that, for all $\epsilon > 0$ and all sufficiently large n, if $G \sim \mathscr{G}(n, 1/2)$, then $\phi(x, y)$ defines a strict linear order on the vertices of G with probability at least $1 - \epsilon$.*

We note that Theorem 6 is one of the two ingredients in the proof of Theorem 5; the second one is the Immerman-Vardi Theorem. While our proof uses only these two ingredients too, we do not see a direct way of getting it from Theorem 5.

Theorem 7. *If the planted clique problem with parameters $p = 1/2$ and $k = k(n)$ is solvable in polynomial time with advantage $\epsilon > 0$, then it is solvable in* FPC *with advantage $\epsilon - o(1)$.*

Proof. Suppose that $\mathscr{C}$ is a polynomial time decidable class of graphs that solves the problem with advantage $\epsilon > 0$. Let $\mathscr{C}'$ denote the class of all ordered expansions of graphs in $\mathscr{C}$, i.e., the structures in $\mathscr{C}'$ are finite structures over the vocabulary $L = \{E, R\}$, where E and R are binary relations symbols, and E is interpreted as the edge relation of some graph in $\mathscr{C}$, and R is interpreted as a strict linear order on its set of vertices. By the Immerman-Vardi Theorem, there is an FP formula ψ that defines $\mathscr{C}'$ on the class of all ordered graphs. Let $\phi(x, y)$ be

the FOC formula from Theorem 6, and let θ be the conjunction of the following sentences:

$$\forall x(\neg\phi(x,x))$$
$$\forall x\forall y\forall z(\neg\phi(x,y)\vee\neg\phi(y,z)\vee\phi(x,z))$$
$$\forall x\forall y(x=y\vee\phi(x,y)\vee\phi(y,x))$$
$$\psi[R/\phi].$$

This sentence says that ϕ defines a strict linear order and ψ holds when each occurrence of R is replaced by ϕ. This is an FPC formula over the vocabulary of unordered graphs that defines a class $\mathscr{D}$ of graphs. We claim that $\mathscr{D}$ solves the problem with advantage $\epsilon - o(1)$.

By assumption and the fact that $\mathscr{C}'$ contains all ordered expansions of graphs in $\mathscr{C}$ we have that the probability that some and hence every ordered expansion of G satisfies ψ is at least $1/2+\epsilon$ when $G \sim \mathscr{G}(n,1/2)$, and at most $1/2-\epsilon$ when $G \sim \mathscr{G}(n,1/2,k)$. Now, if $G \sim \mathscr{G}(n,1/2)$, then the probability that ϕ does not define a linear order is $o(1)$, and the probability that some and hence every ordered expansion of G satisfies ψ is at least $1/2+\epsilon$, so the probability that G satisfies θ is at least $1/2+\epsilon-o(1)$. On the other hand, if $G \sim \mathscr{G}(n,1/2,k)$, then the probability that some and hence every ordered expansion of G satisfies ψ is at most $1/2-\epsilon$, so the probability that G satisfies θ is even smaller, and $1/2-\epsilon \leq 1/2-\epsilon+o(1)$. It follows that $\mathscr{D}$ solves the problem with advantange $\epsilon-o(1)$. □

6 Concluding Remarks

Our main result Theorem 1 establishes a tight three-way correspondence between symmetric Boolean threshold circuits, symmetric LP lifts, and bounded-variable formulas of counting logic. We used this to derive upper and lower bounds on the size of symmetric LPs and symmetric Boolean threshold circuits. We also used it to bound the asymmetric circuit complexity of the planted-clique problem by its symmetric LP lift complexity, up to a polynomial factor. There are several directions for further investigation that are suggested by this work.

The first one concerns the problem of *circuits* vs *formulas*. Composing the first inequality in the first item of Theorem 1 with the second inequality in the second item of Theorem 1, we get the result that, for every constant k, every symmetric Boolean threshold circuit of size at most n^k translates into an equivalent $\mathrm{C}^{O(k)}$-formula of size $n^{O(k)}$. This is a size-efficient translation from circuits into formulas, albeit of different types: the source is a Boolean threshold circuit and the target is a counting logic formula. An explicit and uniform such translation is given in [3], and similarly, AND-OR-NOT circuits can translate into families of L^k-formulas, where L^k stands for the k-variable fragment of first-order logic, without counting.

At first sight, the translation from circuits to formulas is unexpected as the circuit value problem is P-complete, while the formula value problem is in NC^1. However, as we noted, these are not Boolean formulas and the natural translation of formulas of C^k or L^k into circuits necessarily yields circuits with high fan-out. This also accords with the well-known fact that the combined complexity of the logic L^k is P-complete for each $k \geq 3$ [45]. An intriguing question at this point is: What happens when we start the translation not from a circuit but from a symmetric Boolean (threshold) formula? Does this correspond to a

natural syntactic fragment of the logic L^k (or C^k)? Could such a translation shed light on the descriptive complexity of NC^1? On the NC^1 vs P question? Similar questions can be asked for symmetric bounded-depth threshold circuits, and other classes of circuits. It is worth noting that both Theorem 5 and Theorem 7 on the planted-clique problem scale all the way down to uniform TC^0 and FOC. This is so because Theorem 6 gives a FOC-formula and, in the presence of a linear order on the input, FOC captures uniform TC^0 (see Proposition 12.6 in [30]). We view all this as motivation for finding a symmetric LP model of symmetric bounded-depth threshold circuits.

Our results can also be understood as giving a logical interpretation of the decision problems which are *recognized* by polynomial-size symmetric polytope lifts. This, as we have shown, is a larger class of problems than those where the convex hull is obtained as the shadow of a polynomial-size symmetric polytope lifts, i.e. those with polynomial-size symmetric extended formulations. A natural question to ask is if the latter class has a natural logical formulation. Does it admit a descriptive complexity characterization? One line of work that does establish a link between logical definability and extended formulations is [32] which shows that a certain class of problems definable in monadic second-order logic on graphs of bounded treewidth admits linear-size extended formulations.

A different line of research that is suggested by our work relates to the computational complexity of the linear programming feasibility problem. It is well-known that this problem is P-complete under logspace reductions and a question posed in [4], where it was shown that this problem is in FPC, is whether it is complete for FPC under (say) FOC reductions. Our proof of Lemma 1 gives one route towards answering this question. The construction in the proof of the lemma should yield a first-order interpretation from the threshold circuit value problem to the LP feasibility problem. Since the known translations from FPC into symmetric threshold circuits are first-order interpretations themselves, the result should follow by composition. The FPC-completeness of the LP feasibility problem was also proved by Pakusa (unpublished manuscript), independently of our work. Pakusa's construction is very different from ours and raises the question whether the construction in his proof could yield a different proof of Lemma 1. His construction is inspired by the Sherali-Adams hierarchy of LP relaxations [41] (discussed in Section 5) and could well provide a more principled method than ours for constructing LP lifts with useful properties.

Acknowledgments First author partially funded by European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, grant agreement ERC-2014-CoG 648276 (AUTAR) and MICCIN grant TIN2016-76573-C2-1P (TASSAT3). The second author was partially supported by a Fellowship of the Alan Turing Institute under the EPSRC grant EP/N510129/1. The third author funded by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 795936. Some of the work reported here was initiated at the Simons Institute for the Theory of Computing during the programme on Logical Structures in Computation in autumn 2016. We are grateful to Matthew Anderson for some very stimulating discussions on this topic. In particular, he suggested using Lemma 10 to prove the existence of supports. We also thank Wied Pakusa for sharing his manuscript on the FPC-completeness of linear programming with us. Finally, we would like to thank the reviewers for their insightful

comments and constructive suggestions.

References

[1] Noga Alon, Michael Krivelevich, and Benny Sudakov. Finding a Large Hidden Clique in a Random Graph. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '98, pages 594–598, Philadelphia, PA, USA, 1998. Society for Industrial and Applied Mathematics.

[2] Noga Alon, Michael Krivelevich, and Benny Sudakov. Finding a large hidden clique in a random graph. *Random Structures & Algorithms*, 13(34):457–466, 1998.

[3] Matthew Anderson and Anuj Dawar. On Symmetric Circuits and Fixed-Point Logics. *Theory of Computing Systems*, 60(3):521–551, Apr 2017.

[4] Matthew Anderson, Anuj Dawar, and Bjarki Holm. Solving Linear Programs Without Breaking Abstractions. *J. ACM*, 62(6):48:1–48:26, December 2015.

[5] Albert Atserias, Andrei A. Bulatov, and Anuj Dawar. Affine systems of equations and counting infinitary logic. *Theor. Comput. Sci.*, 410(18):1666–1683, 2009.

[6] Albert Atserias and Anuj Dawar. Definable Inapproximability: New Challenges for Duplicator. In *27th EACSL Annual Conference on Computer Science Logic, CSL 2018, September 4-7, 2018, Birmingham, UK*, pages 7:1–7:21, 2018.

[7] Albert Atserias, Anuj Dawar, and Joanna Ochremiak. On the power of symmetric linear programs. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*, pages 1–13. IEEE, 2019.

[8] László Babai, Paul Erdös, and Stanley M. Selkow. Random Graph Isomorphism. *SIAM J. Comput.*, 9(3):628–635, 1980.

[9] Boaz Barak, Samuel B. Hopkins, Jonathan A. Kelner, Pravesh Kothari, Ankur Moitra, and Aaron Potechin. A Nearly Tight Sum-of-Squares Lower Bound for the Planted Clique Problem. In *FOCS*, pages 428–437. IEEE Computer Society, 2016.

[10] Boaz Barak and David Steurer. Proofs, beliefs, and algorithms through the lens of sum-of-squares. Last accessed Jan 8, 2019, 2016.

[11] Richard Bellman. Dynamic Programming Treatment of the Travelling Salesman Problem. *J. ACM*, 9(1):61–63, January 1962.

[12] Andreas Blass, Yuri Gurevich, and Saharon Shelah. Choiceless Polynomial Time. *Ann. Pure Appl. Logic*, 100(1-3):141–187, 1999.

[13] Siu On Chan, James R. Lee, Prasad Raghavendra, and David Steurer. Approximate Constraint Satisfaction Requires Large LP Relaxations. *J. ACM*, 63(4):34:1–34:22, 2016.

[14] Elias Dahlhaus. Reduction to NP-complete Problems by Interpretations. In *Proceedings of the Symposium "Rekursive Kombinatorik" on Logic and Machines: Decision Problems and Complexity*, pages 357–365, Berlin, Heidelberg, 1984. Springer-Verlag.

[15] Anuj Dawar. A Restricted Second Order Logic for Finite Structures. *Inf. Comput.*, 143(2):154–174, 1998.

[16] Anuj Dawar. The nature and power of fixed-point logic with counting. *ACM SIGLOG News*, 2(1):8–21, 2015.

[17] Anuj Dawar and Pengming Wang. Definability of Semidefinite Programming and Lasserre Lower Bounds for CSPs. In *LICS*, pages 1–12. IEEE Computer Society, 2017.

[18] John D. Dixon and Brian Mortimer. *Permutation Groups*, volume 163 of *Graduate Texts in Mathematics*. Springer-Verlag New York, 1996.

[19] Heinz-Dieter Ebbinghaus and Jörg Flum. *Finite model theory.* Perspectives in Mathematical Logic. Springer, 1995.

[20] Uriel Feige and Robert Krauthgamer. Finding and certifying a large hidden clique in a semirandom graph. *Random Structures & Algorithms*, 16(2):195–208, 2000.

[21] Uriel Feige and Robert Krauthgamer. The Probable Value of the Lovász–Schrijver Relaxations for Maximum Independent Set. *SIAM J. Comput.*, 32(2):345–370, 2003.

[22] Martin Grohe. Fixed-Point Logics on Planar Graphs. In *Thirteenth Annual IEEE Symposium on Logic in Computer Science, Indianapolis, Indiana, USA, June 21-24, 1998*, pages 6–15, 1998.

[23] Martin Grohe. From polynomial time queries to graph structure theory. *Commun. ACM*, 54(6):104–112, 2011.

[24] Michael Held and Richard M. Karp. A Dynamic Programming Approach to Sequencing Problems. In *Proceedings of the 1961 16th ACM National Meeting*, ACM '61, pages 71.201–71.204, New York, NY, USA, 1961. ACM.

[25] Lauri Hella, Phokion G. Kolaitis, and Kerkko Luosto. Almost everywhere equivalence of logics in finite model theory. *Bulletin of Symbolic Logic*, 2(4):422–443, 1996.

[26] Wilfrid Hodges. *Model Theory.* Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1993.

[27] Bjarki Holm. *Descriptive complexity of linear algebra.* PhD thesis, University of Cambridge, UK, 2011.

[28] Samuel B. Hopkins, Pravesh Kothari, Aaron Henry Potechin, Prasad Raghavendra, and Tselil Schramm. On the Integrality Gap of Degree-4 Sum of Squares for Planted Clique. *ACM Trans. Algorithms*, 14(3):28:1–28:31, 2018.

[29] Neil Immerman. Languages that Capture Complexity Classes. *SIAM J. Comput.*, 16(4):760–778, 1987.

[30] Neil Immerman. *Descriptive complexity*. Graduate texts in computer science. Springer, 1999.

[31] Mark Jerrum. Large Cliques Elude the Metropolis Process. *Random Struct. Algorithms*, 3(4):347–360, 1992.

[32] Petr Kolman, Martin Koutecký, and Hans Raj Tiwary. Extension complexity, MSO logic, and treewidth. In *15th Scandinavian Symposium and Workshops on Algorithm Theory, SWAT*, pages 18:1–18:14, 2016.

[33] Luděk Kučera. Expected complexity of graph partitioning problems. *Discrete Applied Mathematics*, 57(2):193 – 212, 1995. Combinatorial optimization 1992.

[34] Martin W. Liebeck. On Graphs Whose Full Automorphism Group is an Alternative Group or a Finite Classical Group. *Proceedings of the London Mathematical Society*, s3-47(2):337–362, 1983.

[35] László Lovász and Alexander Schrijver. Cones of Matrices and Set-Functions and 0-1 Optimization. *SIAM Journal on Optimization*, 1(2):166–190, 1991.

[36] Martin Otto. *Bounded Variable Logics and Counting: A Study in Finite Models*, volume 9 of *Lecture Notes in Logic*. Cambridge University Press, 2017.

[37] Christos H. Papadimitriou. *Computational complexity*. Academic Internet Publ., 2007.

[38] Kanstantsin Pashkovich. Tight lower bounds on the sizes of symmetric extensions of permutahedra and similar results. *Mathematics of Operations Research*, 39(4):1330–1339, 2014.

[39] Thomas Rothvoss. The Matching Polytope has Exponential Extension Complexity. *J. ACM*, 64(6):41:1–41:19, 2017.

[40] Alexander Schrijver. *Theory of linear and integer programming*. Wiley-Interscience series in discrete mathematics and optimization. Wiley, 1999.

[41] Hanif D. Sherali and Warren P. Adams. A Hierarchy of Relaxations Between the Continuous and Convex Hull Representations for Zero-One Programming Problems. *SIAM J. Discrete Math.*, 3(3):411–430, 1990.

[42] Michael Sipser. *Introduction to the Theory of Computation*. International Thomson Publishing, 1st edition, 1996.

[43] Salil P. Vadhan. Pseudorandomness. *Found. Trends Theor. Comput. Sci.*, 7(1-3):1–336, 2012.

[44] Moshe Y. Vardi. The Complexity of Relational Query Languages (Extended Abstract). In *STOC*, pages 137–146. ACM, 1982.

[45] Moshe Y. Vardi. On the Complexity of Bounded-Variable Queries. In *PODS*, pages 266–276. ACM Press, 1995.

[46] Heribert Vollmer. *Introduction to Circuit Complexity: A Uniform Approach*. Springer-Verlag, Berlin, Heidelberg, 1999.

[47] Mihalis Yannakakis. Expressing Combinatorial Optimization Problems by Linear Programs. *J. Comput. Syst. Sci.*, 43(3):441–466, 1991.