Theory Comput Syst (2017) 60:521–551
DOI 10.1007/s00224-016-9692-2

On Symmetric Circuits and Fixed-Point Logics

Matthew Anderson[1,2] · **Anuj Dawar**[1]

Published online: 13 July 2016

Abstract We study properties of relational structures, such as graphs, that are decided by families of Boolean circuits. Circuits that decide such properties are necessarily invariant to permutations of the elements of the input structures. We focus on families of circuits that are symmetric, i.e., circuits whose invariance is witnessed by automorphisms of the circuit induced by the permutation of the input structure. We show that the expressive power of such families is closely tied to definability in logic. In particular, we show that the queries defined on structures by uniform families of symmetric Boolean circuits with majority gates are exactly those definable in fixed-point logic with counting. This shows that inexpressibility results in the latter logic lead to lower bounds against polynomial-size families of symmetric circuits.

Keywords Symmetric circuit · Fixed-point logic · Majority · Counting · Uniformity

1 Introduction

A property of graphs on n vertices can be seen as a Boolean function which takes as inputs the $\binom{n}{2}$ potential edges (each of which can be 0 or 1) and outputs either 0 or 1.

This research was supported by EPSRC grant EP/H026835. An extended abstract of this paper appeared in STACS 2014 [1].

✉ Matthew Anderson
andersm2@union.edu

Anuj Dawar
anuj.dawar@cl.cam.ac.uk

[1] University of Cambridge Computer Laboratory, 15 JJ Thomson Ave, Cambridge, CB3 0FD, UK

[2] Present address: Department of Computer Science, Union College, 807 Union St, Schenectady, NY, 12308, USA

For the function to really determine a property of the graph, as opposed to a function of a particular presentation of it, the function must be invariant under re-ordering the vertices of the graph. That is, permuting the $\binom{n}{2}$ inputs according to some permutation of $[n]$ leaves the value of the function unchanged. We call such Boolean functions *invariant*. Note that this does not require the Boolean function to be invariant under *all* permutations of its inputs, which would mean that it was entirely determined by the number of inputs that are set to 1.

The interest in invariant functions arises in the context of characterising the properties of finite relational structures (such as finite graphs) that are decidable in polynomial time. It is a long-standing open problem in descriptive complexity to give a characterisation of the polynomial-time properties of finite relational structures (or, indeed, just graphs) as the classes of structures definable in some suitable logic (see, for instance, [11, Chapter 11]). It is known that fixed-point logic FP and its extension with counting FPC are strictly less expressive than deterministic polynomial time P [4].

It is easy to see that every polynomial-time property of graphs is decided by a polynomial-time uniform family of polynomial-size circuits that are *invariant* in the sense above. On the other hand, when a property of graphs is expressed in a formal logic, it gives rise to a family of circuits that is *explicitly invariant* or *symmetric*. By this we mean that its invariance is witnessed by the automorphisms of the circuit itself. For instance, any sentence of FP translates into a polynomial-size family of symmetric Boolean circuits, while any sentence of FPC translates into a polynomial-size family of symmetric Boolean circuits with majority gates.

Concretely, a circuit C_n consists of a directed acyclic graph whose internal gates are marked by operations from a basis (e.g., the standard Boolean basis $\mathbb{B}_{\text{std}} := \{\text{AND}, \text{OR}, \text{NAND}\}$ or the majority basis $\mathbb{B}_{\text{maj}} = \mathbb{B}_{\text{std}} \cup \{\text{MAJ}\}$) and input gates which are marked with pairs of vertices representing potential edges of an n-vertex input graph. Such a circuit is *symmetric* if C_n has an automorphism π induced by each permutation σ of the n vertices, i.e., π moves the input gates of C_n according to σ and preserves operations and wiring of the internal gates of C_n. Clearly, any symmetric circuit is invariant.

Are symmetric circuits a weaker model of computation than invariant circuits? We aim at characterising the properties that can be decided by uniform families of polynomial-size symmetric circuits. Our main result shows that, indeed, any property that is decided by a uniform family of polynomial-size symmetric majority circuits can be expressed in FPC.

Theorem 1 *A graph property is decided by a polynomial-time uniform family of polynomial-size symmetric majority circuits if, and only if, it is defined by a fixed-point with counting sentence.*

A consequence of this result is that inexpressibility results that have been proved for FPC can be translated into lower bound results for symmetric circuits. For instance, it follows (using [6]) that there is no polynomial-size family of symmetric majority circuits deciding 3-colourability or Hamiltonicity of graphs.

We also achieve a characterisation similar to Theorem 1 of symmetric Boolean circuits.

Theorem 2 *A graph property is decided by a polynomial-time uniform family of polynomial-size symmetric Boolean circuits if, and only if, it is defined by a fixed-point sentence interpreted in $\mathcal{G} \oplus \langle [n], \leq \rangle$, i.e., the structure that is the disjoint union of an n-vertex graph $\mathcal{G}$ with a linear order of length n.*

Note that symmetric majority circuits can be transformed into symmetric Boolean circuits. But, since FP, even interpreted over $\mathcal{G} \oplus \langle [n], \leq \rangle$, is strictly less expressive than FPC, our results imply that any such translation must involve a super-polynomial blow-up in size. Similarly, our results imply with [4] that *invariant* Boolean circuits cannot be transformed into symmetric circuits (even with majority gates) without a super-polynomial blow-up in size. On the other hand, it is clear that symmetric majority circuits can still be translated into *invariant* Boolean circuits with only a polynomial blow-up.

Support The main technical tool in establishing the translation from uniform families of symmetric circuits to sentences in fixed-point logics is a *support theorem* (stated informally below) that establishes properties of the stabiliser groups of gates in symmetric circuits.

We say that a set $X \subseteq [n]$ *supports* a gate g in a symmetric circuit C on an n-element input structure if every automorphism of C that is generated by a permutation of $[n]$ fixing X also fixes g. It is not difficult to see that for a family of symmetric circuits obtained from a given first-order formula ϕ there is a constant k such that all gates in all circuits of the family have a support of size at most k. To be precise, the gates in such a circuit correspond to sub-formulas ψ of ϕ along with an assignment of values from $[n]$ to the free variables of ψ. The set of elements of $[n]$ appearing in such an assignment forms a support of the gate and its size is bounded by the number of free variables of ψ. Using the fact that any formula of FP is equivalent, on structures of size n, to a first-order formula with a constant bound k on the number of variables and, similarly, that any formula of FPC is equivalent to a first-order formula *with majority quantifiers* (see [16]) and a constant bound on the number of variables, we see that the resulting circuits have supports of constant bounded size. Our main technical result is that the existence of supports of bounded size holds, in fact, for all polynomial-size families of symmetric circuits. In its general form, we show the following theorem in Section 3 via an involved combinatorial argument.

Theorem 3 (Informal Support Thm) *Let C be a symmetric circuit with s gates over a graph of size n. If n is sufficiently large and s is sub-exponential in n, then every gate in C has a support of size $O\left(\frac{\log s}{\log n}\right)$.*

In the typical instantiation of the Support Theorem the circuit C contains a polynomial number of gates $s = \text{poly}(n)$ and hence the theorem implies that every gate

has a support that is bounded in size by a constant. The proof of the Support Theorem mainly relies on the structural properties of symmetric circuits and is largely independent of the semantics of such circuits; this means it may be of independent interest for other circuit bases and in other settings.

Symmetric Circuits and FP In Section 4 we show that each polynomial-size family $\mathcal{C}$ of symmetric circuits can be translated into a formula of fixed-point logic. If the family $\mathcal{C}$ is polynomial-time uniform, by the Immerman-Vardi Theorem [14, 19] there is an FP-definable interpretation of the circuit C_n in the ordered structure $\langle [n], \leq \rangle$. We show that the support of a gate is computable in polynomial time, and hence we can also interpret the support of each gate in $\langle [n], \leq \rangle$. The circuit C_n can be evaluated on an input graph (say) $\mathcal{G}$ by fixing a bijection between $[n]$ and the set U of vertices of $\mathcal{G}$. We associate with each gate g of C_n the set of those bijections that cause g to evaluate to 1 on $\mathcal{G}$. This set of bijections admits a compact (i.e., polynomial-size) representation as the set of injective maps from the support of g to U. We show that these compact representations can be inductively defined by formulas of FP, or FPC if the circuit also admits majority gates.

Thus, we obtain that uniform families of polynomial-size symmetric Boolean circuits can be translated into formulas of FP interpreted in $\mathcal{G}$ combined with a disjoint linear order $\langle [|\mathcal{G}|], \leq \rangle$, while families containing majority gates can be simulated by sentences of FPC. The reverse containment follows using classical techniques. As a consequence we obtain the equivalences of Theorems 1 & 2, and a number of more general results as this sequence of arguments naturally extends to: (i) inputs given as an arbitrary relational structure, (ii) outputs defining arbitrary relational queries, and (iii) (non-uniform) families of polynomial-size circuits, provided the logic is allowed additional advice on the disjoint linear order.

Related Work We note that the term "symmetric circuit" is used by Denenberg et al. in [10] to mean what we call invariant circuits. They give a characterisation of first-order definability in terms of a restricted invariance condition, namely circuits that are invariant and whose relativisation to subsets of the universe remains invariant. Our definition of symmetric circuits follows that in [17] where Otto describes it as the "natural and straightforward combinatorial condition to guarantee generic or isomorphism-invariant performance". He then combines it with a size restriction on the orbits of gates along with a strong uniformity condition, which he calls "coherence", to give an exact characterisation of definability in infinitary logic. A key element in his construction is the proof that if the orbits of gates in such a circuit are polynomially bounded in size then they have supports of bounded size. We remove the assumption of coherence from this argument and show that constant size supports exist in any polynomial-size symmetric circuit. This requires a generalisation of what Otto calls a "base" to supporting partitions. See Section 6 for more discussion of connections with Otto's work. Clote and Kranakis [5] show that Boolean functions on binary strings that are close to being invariant under all permutations can be computed by constant-depth threshold circuits (here "close" means that the index of the invariance group in the symmetric group is bounded by a polynomial in the input length of the Boolean function).

2 Preliminaries

Let $[n]$ denote the set of positive integers $\{1, \ldots, n\}$. Let Sym_S denote the group of all permutations of the set S. When $S = [n]$, we write Sym_n for $\mathrm{Sym}_{[n]}$.

2.1 Vocabularies, Structures, and Logics

A *relational vocabulary* (always denoted by τ) is a finite sequence of relation symbols $(R_1^{r_1}, \ldots, R_k^{r_k})$ where for each $i \in [k]$ the relation symbol R_i has an associated arity $r_i \in \mathbb{N}$. A τ-structure $\mathcal{A}$ is a tuple $\langle A, R_1^{\mathcal{A}}, \ldots, R_k^{\mathcal{A}} \rangle$ consisting of (i) a non-empty set U called the *universe* of $\mathcal{A}$, and (ii) relations $R_i^{\mathcal{A}} \subseteq U^{r_i}$ for $i \in [k]$. Members of the universe U are called *elements* of $\mathcal{A}$. A *multi-sorted* structure is one whose universe is given as a disjoint union of several distinct sets called *sorts*. Define the *size* of a structure $|\mathcal{A}|$ to be the cardinality of its universe. All structures considered in this paper are *finite*, i.e., their universes have finite cardinality. Let $\mathrm{fin}[\tau]$ denote the set of all finite τ-structures.

First-Order and Fixed-Point Logics Let $\mathrm{FO}(\tau)$ denote *first-order logic* with respect to the vocabulary τ. The logic $\mathrm{FO}(\tau)$ is the set of formulas whose atoms are formed using the relation symbols in τ, an equality symbol $=$, an infinite sequence of variables $(x, y, z \ldots)$, and that are closed under the Boolean connectives ($\wedge$ and $\vee$), negation ($\neg$), and universal and existential quantification ($\forall$ and $\exists$). Let *fixed-point logic* $\mathrm{FP}(\tau)$ denote the extension of $\mathrm{FO}(\tau)$ to include an inflationary fixed-point operator ifp (see [7] for a definition). Assume standard syntax and semantics for FO and FP (see the textbook [11] for more background). For a formula ϕ write $\phi(x)$ to indicate that x is the tuple of the free variables of ϕ. For a logic $\mathcal{L}$, a formula $\phi(x) \in \mathcal{L}(\tau)$ with k free variables, $\mathcal{A} \in \mathrm{fin}[\tau]$, and tuple $a \in A^k$ write $\mathcal{A} \models_{\mathcal{L}} \phi[a]$ to express that the tuple a makes the formula ϕ true in the structure $\mathcal{A}$ with respect to the logic $\mathcal{L}$. We usually drop the subscript $\mathcal{L}$ and write $\mathcal{A} \models \phi[a]$ when no confusion would arise.

Logics with Disjoint Advice Let τ_{arb} be a relational vocabulary without a binary relation symbol $\leq$. Let $\Upsilon : \mathbb{N} \to \mathrm{fin}[\tau_{\mathrm{arb}} \uplus \{\leq^2\}]$ be an *advice function*, where for $n \in \mathbb{N}$, $\Upsilon(n)$ has universe $\{0, 1, \ldots, n\}$ naturally ordered by $\leq$. For a logic $\mathcal{L}$, typically FO or FP, let $(\mathcal{L} + \Upsilon)(\tau)$ denote the set of formulas of $\mathcal{L}(\tau')$ where $\tau' := \tau \uplus \tau_{\mathrm{arb}} \uplus \{\leq^2\}$ and τ is a vocabulary disjoint from $\tau_{\mathrm{arb}} \uplus \{\leq^2\}$. For a structure $\mathcal{A} \in \mathrm{fin}[\tau]$ define the semantics of $\phi \in (\mathcal{L} + \Upsilon)(\tau)$ to be $\mathcal{A} \models_{(\mathcal{L}+\Upsilon)} \phi$ iff $\mathcal{A}^{\Upsilon} \models_{\mathcal{L}} \phi$, where $\mathcal{A}^{\Upsilon} := \mathcal{A} \oplus \Upsilon(|\mathcal{A}|)$ is the multi-sorted τ'-structure formed by taking the disjoint union of $\mathcal{A}$ with a structure coding a linear order of corresponding cardinality endowed with interpretations of the relations in τ_{arb}. The universe of the multi-sorted structure $\mathcal{A}^{\Upsilon}$ is written as $U \uplus \{0, 1, \ldots, |U|\}$; refer to U as the *point sort* of $\mathcal{A}^{\Upsilon}$ and to $\{0, 1, \ldots, |U|\}$ as the *number sort* of $\mathcal{A}^{\Upsilon}$. We are primarily interested in the special case when τ_{arb} is empty and hence $\Upsilon(|\mathcal{A}|) = \langle \{0, 1, \ldots, |U|\}, \leq \rangle$ is simply a linear order. Denote formulas of this logic by $(\mathcal{L} + \leq)(\tau)$ and extended structures by $\mathcal{A}^{\leq}$ to emphasise the disjoint linear order.

Let $\mathrm{FPC}(\tau)$ denote the extension of $(\mathrm{FP}+\leq)(\tau)$ with a counting operator $\#_x$ where x is a point or number variable. For a structure $\mathcal{A} \in \mathrm{fin}[\tau]$ and a formula $\phi(x) \in \mathrm{FPC}(\tau)$, $\#_x\phi(x)$ is a term denoting the element in the number sort corresponding to $|\{a \in \mathcal{A} \mid \mathcal{A} \models \phi[a]\}|$. See [11, Section 8.4.2] for more details. Finally, we consider the extension of fixed-point logic with both advice functions and counting quantifiers $(\mathrm{FPC}+\Upsilon)(\tau)$.

Using k-tuples of number variables, it is possible in $\mathrm{FP}+\leq$ and FPC to represent numbers up to n^k and perform arithmetic operations on them (see [15, 1.32–1.33]). We omit details but use such constructions freely.

2.2 Symmetric and Uniform Circuits

A *Boolean basis* (always denoted by $\mathbb{B}$) is a finite set of Boolean functions from $\{0,1\}^*$ to $\{0,1\}$. We consider only bases containing symmetric functions, i.e., for all $f \in \mathbb{B}$, $f(x) = f(y)$ for all $n \in \mathbb{N}$ and $x, y \in \{0,1\}^n$ with the same number of ones. The *standard Boolean basis* $\mathbb{B}_{\mathrm{std}}$ consists of unbounded fan-in AND, OR, and NAND operators.[1] The *majority basis* $\mathbb{B}_{\mathrm{maj}}$ extends the standard basis with an operator MAJ which is one iff the number of ones in the input is at least the number of zeroes.

Definition 1 (Circuits on Structures) A Boolean $(\mathbb{B}, \tau)$-circuit C_n computing a q-ary query Q is a structure $\langle G, W, \Omega, \Sigma, \Lambda\rangle$.

- G is a set called the *gates* of C_n. The *size* of C_n is $|C_n| := |G|$.
- $W \subseteq G \times G$ is a binary relation called the *wires* of the circuit. We require that (G, W) forms a *directed acyclic graph*. Call the gates with no incoming wires *input gates*, and all other gates *internal gates*. Gates h with $(h, g) \in W$ are called the *children* of g.
- Ω is an injective function from $[n]^q$ to G. The gates in the image of Ω are called the *output gates*. When $q = 0$, Ω is a constant function mapping to a single output gate.
- Σ is a function from G to $\mathbb{B} \uplus \tau \uplus \{0,1\}$ which maps input gates into $\tau \uplus \{0,1\}$ with $|\Sigma^{-1}(0)|, |\Sigma^{-1}(1)| \leq 1$ and internal gates into $\mathbb{B}$. Call the input gates marked with a relation from τ *relational gates* and the input gates marked with 0 or 1 *constant gates*.
- Λ is a sequence of injective functions $(\Lambda_R)_{R\in\tau}$ where for each $R \in \tau$, Λ_R maps each relational gate g with $R = \Sigma(g)$ to the tuple $\Lambda_R(g) \in [n]^r$ where r is the arity of R. Where no ambiguity arises, we write $\Lambda(g)$ for $\Lambda_R(g)$.

We write $C = C_n$ to emphasise that C accepts input structures of size n. The variable n will always be used to represent this quantity, we often drop the subscript for clarity.

[1] This basis is *universal* in the sense that every Boolean function can be computed by some circuit with gates from this basis. Indeed the basis containing only NAND is universal as AND and OR operations can be computed using several NAND operations. We permit the use of AND and OR gates to make several constructions easier.

Let $C = C_n$ be a Boolean $(\mathbb{B}, \tau)$-circuit, $\mathcal{A} \in \text{fin}[\tau]$ with $|\mathcal{A}| = n$, and $\gamma : U \to [n]$ be a bijection. Let $\gamma\mathcal{A}$ denote the τ-structure over the universe $[n]$ obtained by relabeling the universe of $\mathcal{A}$ according to γ. Recursively evaluate C on $\gamma\mathcal{A}$ by determining a value $C[\gamma\mathcal{A}](g)$ for each gate g: (i) a constant gate evaluates to the bit given by $\Sigma(g)$, (ii) a relational gate evaluates to 1 iff $\gamma\mathcal{A} \models \Sigma(g)(\Lambda(g))$, and (iii) an internal gate evaluates to the result of applying the Boolean operation $\Sigma(g)$ to the values for g's children. C defines the q-ary query $Q \subseteq \mathcal{A}^q$ where $a \in Q$ iff $C[\gamma\mathcal{A}](\Omega(\gamma a)) = 1$.

Definition 2 (Invariant Circuit) Let C be a $(\mathbb{B}, \tau)$-circuit computing a q-ary query on structures of size n. The circuit C is *invariant* if for every $\mathcal{A} \in \text{fin}[\tau]$ with $|\mathcal{A}| = n$, $a \in \mathcal{A}^q$, and bijections γ_1, γ_2 from A to $[n]$, $C[\gamma_1\mathcal{A}](\Omega(\gamma_1 a)) = C[\gamma_2\mathcal{A}](\Omega(\gamma_2 a))$.

Invariance indicates that C computes a property of τ-structures which is invariant to presentations of the structure. A *family* $\mathcal{C} = (C_n)_{n\in\mathbb{N}}$ of invariant $(\mathbb{B}, \tau)$-circuits naturally computes a q-ary query on τ-structures. When $q = 0$ the family computes a Boolean property of structures (i.e. a decision problem). We now discuss a structural property of circuits called *symmetry* that implies invariance.

Symmetric Circuits Permuting a circuit's universe may induce automorphisms of the circuit.

Definition 3 (Automorphism) Let $C = \langle G, W, \Omega, \Sigma, \Lambda\rangle$ be a $(\mathbb{B}, \tau)$-circuit computing a q-ary query on structures of size n. Let $\sigma \in \text{Sym}n$ and $\pi : G \to G$ be a bijection such that

- for all gates $g, h \in G$, $W(g, h)$ iff $W(\pi(g), \pi(h))$,
- for all output tuples $x \in [n]^q$, $\pi\Omega(x) = \Omega(\sigma(x))$,
- for all gates $g \in G$, $\Sigma(g) = \Sigma(\pi(g))$, and
- for each relational gate $g \in G$, $\sigma\Lambda(g) = \Lambda(\pi(g))$.

We say that π is an *automorphism* of C, and that σ *induces the automorphism* π of C. We label the group of C's automorphisms $\text{Aut}_n(C)$.

Note that a permutation σ may induce more than one automorphism of the circuit.

The principal goal of this paper is to understand the computational power of circuit classes with the following type of structural symmetry.

Definition 4 (Symmetric) A circuit C on structures of size n is called *symmetric* if for every permutation $\sigma \in \text{Sym}_n$, σ induces an automorphism of C.

It is not difficult to see that, for a symmetric circuit C, there is a homomorphism $h : \text{Sym}_n \to \text{Aut}_n(C)$ such that $h(\sigma)$ is an automorphism induced by σ. As long as some element of $[n]$ appears in the label of some input gate of C, by symmetry every element of $[n]$ appears and it follows that h is an injective homomorphism. Henceforth we assume that this is always the case as otherwise C has no relational

inputs and computes a constant function. Circuits where the homomorphism is not also surjective introduce artifacts into our arguments. To avoid this we require the circuits we consider to be *rigid*.

Definition 5 (Rigid) Let $C = \langle G, W, \Omega, \Sigma, \Lambda \rangle$ be a $(\mathbb{B}, \tau)$-circuit. Call C *rigid* if there do not exist distinct internal gates $g, g' \in G$ with $\Sigma(g) = \Sigma(g')$, $\Omega^{-1}(g) = \Omega^{-1}(g')$, and for every $g'' \in G$, $W(g'', g)$ iff $W(g'', g')$ and $W(g, g'')$ iff $W(g', g'')$.

To show that for rigid symmetric circuits C, any injective homomorphism from Sym_n to $\mathrm{Aut}_n(C)$ is surjective, it suffices to show that each $\sigma \in \mathrm{Sym}_n$ induces a unique automorphism in $\mathrm{Aut}_n(C)$.

Proposition 1 *Let C be a rigid circuit on structures of size n, and $\sigma \in \mathrm{Sym}_n$. If σ induces an automorphism of C, that automorphism is unique.*

We defer the proof of this proposition to Section 4.1 where we also show that symmetric circuits can be transformed into equivalent rigid symmetric circuits in polynomial time, and hence show that rigidity can be assumed of circuits without loss of generality in our setting. For a rigid symmetric circuit C, the group of automorphisms of C is exactly Sym_n acting faithfully. We shall therefore abuse notation and use these interchangeably. In particular, we shall write σg to denote the image of a gate g in C under the action of the automorphism induced by a permutation σ in Sym_n.

An examination of the definitions suffices to show that symmetry implies invariance. In symmetric circuits it is useful to consider those permutations which induce automorphisms that fix gates. Let $\mathcal{P}$ be a partition of a set $[n]$. Let the *pointwise stabiliser* of $\mathcal{P}$ be

$$\mathrm{Stab}_n(\mathcal{P}) := \{\sigma \in \mathrm{Sym}_n \mid \forall P \in \mathcal{P}, \sigma P = P\},$$

and similarly define the *setwise stabiliser*

$$\mathrm{SetStab}_n(\mathcal{P}) := \{\sigma \in \mathrm{Sym}_n \mid \forall P \in \mathcal{P}, \sigma P \in \mathcal{P}\}.$$

For a gate g in a rigid symmetric circuit C, let the *stabiliser* of g be $\mathrm{Stab}_n(g) := \{\sigma \in \mathrm{Sym}_n \mid \sigma(g) = g\}$, and let the *orbit* of g under the automorphism group $\mathrm{Aut}_n(C)$ of C be $\mathrm{Orb}(g) := \{\pi g \mid \pi \in \mathrm{Aut}_n(C)\}$.

Uniform Circuits One natural class of circuits are those with polynomial-size descriptions that can be generated by a deterministic polynomial-time Turing machine.

Definition 6 (P-Uniform Polynomial-Size Circuits) A polynomial-size $(\mathbb{B}, \tau)$-circuit family $\mathcal{C} = (C_n)_{n \in \mathbb{N}}$ computing a q-ary query is P-*uniform* if there exists an integer $t \geq q$ and function $\Upsilon : \mathbb{N} \to \{0, 1\}^*$, computable by a determininistic polynomial-time Turing machine M, which takes an integer n to a binary string

$\Upsilon(n)$ such that $|\Upsilon(n)| = \text{poly}(n)$, and $\Upsilon(n)$ describes[2] the circuit C_n whose gates are indexed by t-tuples of $\{0, 1, \ldots, n\}$, inputs are labeled by tuples from $[n]$, and outputs are labeled by q-tuples of $[n]$.

Note that such uniform families explicitly have polynomial size. For more background on circuit uniformity, c.f., e.g., the text [20]. It follows from the Immerman-Vardi Theorem [14, 19] that any P-uniform family $\mathcal{C} = (C_n)_{n\in\mathbb{N}}$ of polynomial-size circuits is definable by an FP interpretation in the sense that there is a sequence of formulas $(\phi_G, \phi_W, \phi_\Omega, (\phi_s)_{s\in\mathbb{B}\uplus\tau\uplus\{0,1\}}, (\phi_{\Lambda_R})_{R\in\tau})$ which, interpreted in the structure $\langle[n], \leq\rangle$, describes the circuit $C_n = \langle G, W, \Omega, \Sigma, \Lambda\rangle$, with

- $G \subseteq [n]^t$ such that $g \in G$ iff $\langle[n], \leq\rangle \models \phi_G[g]$.
- For all $g, g' \in G$ and $W(g, g')$ iff $\langle[n], \leq\rangle \models \phi_W[g, g']$.
- For all $g \in G$ and $a \in [n]^q$, $\Omega(a) = g$ iff $\langle[n], \leq\rangle \models \phi_\Omega[a, g]$.
- For all $g \in G$ and $s \in \mathbb{B} \uplus \tau \uplus \{0, 1\}$, $\Sigma(g) = s$ iff $\langle[n], \leq\rangle \models \phi_s[g]$.
- For all relational gates $g \in G$ and $a \in [n]^r$, $\Lambda_R(g) = a$ iff $\langle[n], \leq\rangle \models \phi_\Lambda[g, a]$, where r is the arity of $R = \Sigma(g)$.

More generally, if we do not know how to efficiently compute Υ for a polynomial-size circuit family $\mathcal{C} = (C_n)_{n\in\mathbb{N}}$, there is still an $(\text{FP} + \Upsilon)$-definable interpretation of C_n in $\mathcal{A}^\Upsilon$.

Circuits need not compute invariant queries as their computation may implicitly depend on the order associated with $[n]$. To obtain invariance for such circuits we assert symmetry. The next section proves a natural property of symmetric circuits that ultimately implies that *symmetric* P-uniform circuits of polynomial size coincide with FP definitions on the standard and majority bases.

3 Symmetry and Support

In this section we analyse the structural properties of symmetric circuits. We begin with a formal definition of support.

Definition 7 (Support) Let C be a rigid symmetric circuit on structures of size n and let g be a gate in C. A set $X \subseteq [n]$ *supports* g if, for any permutation $\sigma \in \text{Sym}_n$ such that $\sigma x = x$ for all $x \in X$, we have $\sigma g = g$ (i.e., $\sigma \in \text{Stab}_n(g)$).[3]

We show how to associate supports of constant size in a canonical way to all gates in *any* rigid symmetric circuit of polynomial size. Indeed, our result is more general as it associates moderately growing supports to gates in circuits of sub-exponential size. As a preliminary to the proof, we introduce, in Section 3.1, the more general

[2]Formally one must define a particular way of encoding circuits via binary strings. However, since the details of the representation are largely irrelevant for our purposes we omit them.

[3]Note that this definition of support is different than the group-theoretic support of a permutation, i.e., for a permutation $\pi \in \text{Sym}_n$, $\text{support}(\pi) := \{x \in [n] \mid \pi x \neq x\}$.

notion of a *supporting partition* for a permutation group. We show how to associate a canonical such supporting partition with any permutation group G and obtain bounds on the size of such a partition based on the index of G in the symmetric group. These results are then used, in Section 3.2, to bound the size of partitions supporting stabiliser groups of gates based on the size of the circuit, proving our main technical result—the Support Theorem.

3.1 Supporting Partitions

The notion of a supporting partition generalises the notion of a support of a gate by replacing the set with a partition and the stabiliser group of the gate with an arbitrary permutation group.

Definition 8 (Supporting Partition) Let $G \subseteq \mathrm{Sym}_n$ be a group and $\mathcal{P}$ a partition of $[n]$. We say that $\mathcal{P}$ is a *supporting partition* of G if $\mathrm{Stab}_n\mathcal{P} \subseteq G$.

For intuition consider two extremes. When G has supporting partition $\mathcal{P} = \{[n]\}$, it indicates $G = \mathrm{Sym}_n$. Saying that G has supporting partition $\mathcal{P} = \{\{1\}, \{2\}, \ldots, \{n\}\}$ indicates only that G contains the identity permutation, which is always true.

A natural partial order on partitions is the coarseness relation, i.e., $\mathcal{P}'$ is as coarse as $\mathcal{P}$, denoted $\mathcal{P}' \supseteq \mathcal{P}$, if every part in $\mathcal{P}$ is contained in some part of $\mathcal{P}'$. For two partitions $\mathcal{P}$ and $\mathcal{P}'$, there is a most refined partition that is as coarse as either partition:

Definition 9 Let $\mathcal{P}, \mathcal{P}'$ be partitions of $[n]$. Define a binary relation $\sim$ on $[n]$: $x_1 \sim x_2$ iff there exists $P \in \mathcal{P}$ or $P \in \mathcal{P}'$ such that $x_1, x_2 \in P$. Let $\mathcal{E}(\mathcal{P}, \mathcal{P}')$ denote the partition of $[n]$ corresponding to the equivalence classes of $[n]$ under the transitive closure of $\sim$.[4]

Now it is easy to show that $\mathcal{E}$ preserves supporting partitions (the proof is similar to that of (*) on page 379 of [17]).

Proposition 2 *Let $G \subseteq \mathrm{Sym}_n$ be a group and $\mathcal{P}, \mathcal{P}'$ be supporting partitions of G. Then $\mathcal{E}(\mathcal{P}, \mathcal{P}')$ is also a supporting partition of G.*

Proof Let $\mathcal{E} := \mathcal{E}(\mathcal{P}, \mathcal{P}') = \{E_1, \ldots, E_m\}$. Suppose $\sigma \in \mathrm{Stab}(\mathcal{E})$ and we now show that $\sigma \in G$. Because the parts E_i are disjoint write σ as $\sigma_1 \cdots \sigma_m$ where $\sigma_i \in \mathrm{Sym}_{E_i}$ (i.e., it permutes only the elements of E_i). Indeed each σ_i may be written as a sequence of transpositions of elements in E_i. Thus it suffices to show

[4] By interpreting partitions as equivalence relations one can can alternatively view $\mathcal{E}(\mathcal{P}, \mathcal{P}')$ as the relational join of $\mathcal{P}$ and $\mathcal{P}'$.

that each transposition (xx') with $x, x' \in E_i$ can be written as a sequence of permutations in $\mathrm{Stab}(\mathcal{P}) \cup \mathrm{Stab}(\mathcal{P}') \subseteq G$. Since $x, x' \in E_i$ there is a sequence of elements of $x_1, \ldots, x_\ell$ with $x_1 = x$, $x_\ell = x'$ and $x_j \sim x_{j+1}$ for $j \in [\ell - 1]$ witnessing the path from x to x'. By the definition of $\sim$, for each $j \in [\ell - 1]$ there is $P \in \mathcal{P}$ or $P \in \mathcal{P}'$ such that $x_j, x_{j+1} \in P$ and therefore $(x_j x_{j+1})$ is a transposition in $\mathrm{Stab}(\mathcal{P}) \cup \mathrm{Stab}(\mathcal{P}')$. Conclude that the transposition $(xx') = (x_1 x_\ell) = (x_1 x_2)(x_2 x_3) \cdots (x_{\ell-2} x_{\ell-1})(x_{\ell-1} x_\ell)(x_{\ell-2} x_{\ell-1}) \cdots (x_1 x_2)$ is a sequence of transpositions from $\mathrm{Stab}(\mathcal{P}) \cup \mathrm{Stab}(\mathcal{P}')$ and the proof is complete. □

This implies that each permutation group has a unique coarsest partition that supports it.

Lemma 1 *Each permutation group $G \subseteq \mathrm{Sym}_n$ has a unique coarsest supporting partition.*

Proof Suppose G has two distinct coarsest partitions $\mathcal{P}, \mathcal{P}'$ of $[n]$ that support it, then Proposition 2 implies that the coarser partition $\mathcal{E}(\mathcal{P}, \mathcal{P}')$ also supports G. This is a contradiction assuming $\mathcal{E}(\mathcal{P}, \mathcal{P}')$ is not either $\mathcal{P}$ or $\mathcal{P}'$. □

We write $\mathsf{SP}(G)$ for *the unique coarsest partition supporting G*. For a partition $\mathcal{P}$ of $[n]$ we write $|\mathcal{P}|$ to denote its *size*, i.e., the number of parts $\mathcal{P}$ contains, and for a permutation $\sigma \in \mathrm{Sym}_n$, we write $\sigma\mathcal{P}$ for the partition $\{\sigma P \mid P \in \mathcal{P}\}$. Note that this commutes with the operation $\mathcal{E}$, so $\sigma\mathcal{E}(\mathcal{P}, \mathcal{P}') = \mathcal{E}(\sigma\mathcal{P}, \sigma\mathcal{P}')$. The next lemma shows how supporting partitions are affected by the conjugacy action of Sym_n.

Lemma 2 *If $\mathcal{P}$ is a partition supporting a group G, then for any $\sigma \in \mathrm{Sym}_n$, $\sigma\mathcal{P}$ supports the group $\sigma G \sigma^{-1}$.*

Proof Let $\pi \in \mathrm{Stab}_n(\sigma\mathcal{P})$ and let P be a part in $\mathcal{P}$, then:

$$(\sigma^{-1}\pi\sigma)P = (\sigma^{-1}\pi)(\sigma P) = \sigma^{-1}\sigma P = P,$$

where the second equality follows from the fact that π fixes σP. Thus, $\sigma^{-1}\pi\sigma$ fixes $\mathcal{P}$ pointwise, therefore $\sigma^{-1}\pi\sigma \in G$ and hence $\pi \in \sigma G \sigma^{-1}$. □

This indicates how the unique coarsest supporting partition of a group translates under conjugation.

Lemma 3 *For any $G \subseteq \mathrm{Sym}_n$ and any $\sigma \in \mathrm{Sym}_n$, $\sigma\mathsf{SP}(G) = \mathsf{SP}(\sigma G \sigma^{-1})$.*

Proof Immediate from Lemma 2 and the fact that the action of $\mathcal{E}$ commutes with σ. □

We conclude that any group G is sandwiched between the pointwise and setwise stabilisers of $\mathsf{SP}(G)$.

Lemma 4 *For any group $G \subseteq \mathrm{Sym}_n$, we have*

$$\mathrm{Stab}_n(\mathsf{SP}(G)) \subseteq G \subseteq \mathrm{SetStab}_n(\mathsf{SP}(G)).$$

Proof The first inclusion is by definition of supporting partitions. For the second, note that if $\sigma \in G$, then $\sigma G \sigma^{-1} = G$. Then, by Lemma 3, $\sigma \mathsf{SP}(G) = \mathsf{SP}(G)$. □

Note that these bounds need not be tight. For example, if G is the alternating group on n (or, indeed, any transitive, primitive proper subgroup of Sym_n), then $\mathsf{SP}(G)$ is the partition of $[n]$ into singletons, i.e., $\mathsf{SP}(G) = \{\{x_1\}, \{x_2\}, \ldots, \{x_n\}\}$. In this case, $\mathrm{Stab}_n(\mathsf{SP}(G))$ is the trivial group while $\mathrm{SetStab}_n(\mathsf{SP}(G))$ is all of Sym_n.

We now use the bounds given by Lemma 4, in conjunction with bounds on G to obtain size bounds on $\mathsf{SP}(G)$. Recall that the index of G in Sym_n, denoted $[\mathrm{Sym}_n : G]$ is the number of cosets of G in Sym_n or, equivalently, $\frac{|\mathrm{Sym}_n|}{|G|}$. The next lemma says that if $\mathcal{P}$ is a partition of $[n]$ where the index of $\mathrm{SetStab}_n(\mathcal{P})$ in Sym_n is sufficiently small then the number of parts in $\mathcal{P}$ is either very small or very big.

Lemma 5 *For any ϵ and n such that $0 < \epsilon < 1$ and $\log n \geq \frac{4}{\epsilon}$, if $\mathcal{P}$ is a partition of $[n]$ with k parts, $s := [\mathrm{Sym}_n : \mathit{SetStab}_n(\mathcal{P})]$ and $n \leq s \leq 2^{n^{1-\epsilon}}$, then $\min\{k, n-k\} \leq \frac{8}{\epsilon} \frac{\log s}{\log n}$.*

Proof Let $p_1 \leq p_2 \leq \ldots \leq p_k$ be the respective sizes of the parts in $\mathcal{P}$. Thus,

$$s = \frac{n!}{|\mathrm{SetStab}_n(\mathcal{P})|} \geq \frac{1}{k!} \frac{n!}{p_1! p_2! \cdots p_k!}. \tag{1}$$

Observe that, if $p_i > 1$, then $p_1! p_2! \cdots p_k! \leq p_1! p_2! \cdots (p_i - 1)! \cdots (p_k + 1)!$. By repeatedly applying this, we see that in the lower bound on s given by (1), we can replace $p_1! p_2! \cdots p_k!$ by $(n - (k-1))!$. Let $k' := \min\{k, n-k\}$ and we have

$$\begin{aligned} s &\geq \frac{n!}{k!(n-(k-1))!} = \frac{1}{n+1}\binom{n+1}{k} \geq \frac{1}{n+1}\binom{n}{k} \\ &= \frac{1}{n+1}\binom{n}{k'} \geq \frac{1}{n+1}\left(\frac{n}{k'}\right)^{k'} \end{aligned}$$

where the second equality follows because $\binom{n}{k} = \binom{n}{n-k} = \binom{n}{k'}$, and the final inequality follows from a simple combinatorial bound. Take the logarithm of both sides of the above equation to get $\log s \geq k'(\log n - \log k') - \log(n+1)$. Using the fact that $s \geq n \geq 2$ ($\log n \geq \frac{4}{\epsilon} \geq 4$) then implies that

$$4 \log s \geq k'(\log n - \log k'), \tag{2}$$

The definition of k' implies that $k' \leq \frac{n}{2}$ and $\log n - \log k' \geq 1$. Plugging this into (2) gives that $4 \log s \geq k'$. Take the logarithm of this inequality and apply the upper bound on s to determine that $(1-\epsilon)\log n + 2 \geq \log k'$. Inserting this inequality back into (2) gives $4 \log s \geq k'(\epsilon \log n - 2)$. Since $\frac{\epsilon}{2} \log n \geq 2$, conclude that $k' \leq \frac{8}{\epsilon} \frac{\log s}{\log n}$. □

We use a similar argument to establish that, under the assumptions of the previous lemma, when the number of parts in $\mathcal{P}$ is less than $\frac{n}{2}$ the largest part is very big. The intuition is that the number of elements in any union of the parts must either be very small or very big, in the same sense as before, because otherwise it violates the bound on the orbit size. Then, because we assume there are few parts, one part must be large and the rest small.

Lemma 6 *For any ϵ and n such that $0 < \epsilon < 1$ and $\log n \geq \frac{8}{\epsilon^2}$, if $\mathcal{P}$ is a partition of $[n]$ with $|\mathcal{P}| \leq \frac{n}{2}$, $s := [\mathrm{Sym}_n : \mathit{SetStab}_n(\mathcal{P})]$ and $n \leq s \leq 2^{n^{1-\epsilon}}$, then $\mathcal{P}$ contains a part P with at least $n - \frac{33}{\epsilon} \cdot \frac{\log s}{\log n}$ elements.*

Proof The initial setup is the same as in the proof of Lemma 5. Let $p_1 \leq p_2 \leq \ldots \leq p_k$ be the respective sizes of the parts in $\mathcal{P}$ and let $S := \sum_{i=1}^{k-1} p_i$. Our aim is to show that $S \leq \frac{33}{\epsilon} \cdot \frac{\log s}{\log n}$. Denote the size of the second largest part by $p := p_{k-1}$. We have

$$s = \frac{n!}{|\mathrm{SetStab}_n(\mathcal{P})|} \geq \frac{1}{k!} \frac{n!}{p_1! p_2! \cdots p_k!}. \tag{3}$$

Let $\ell \in \mathbb{N}$ be such that

$$\ell \leq k-1, \text{ and } k - 1 + \ell(p-1) \leq \sum_{i=1}^{k-1} p_i = S. \tag{4}$$

Provided $\mathcal{P}$ contains more than one part both $\ell \in \{0, 1\}$ satisfy (4). We may assume that $p > 1$ otherwise $S \leq |\mathcal{P}|$ and we are done by Lemma 5. For any $\ell \geq 1$ satisfying (4), redistributing weight from a p_i to p_j with $i < j$ in a way similar to the proof of Lemma 5 gives the following,

$$\begin{aligned} s &\geq \frac{1}{k!} \frac{n!}{\underbrace{1! \cdots 1!}_{k-1-\ell \text{ times}} \underbrace{p! \cdots p!}_{\ell \text{ times}} (n - (k - 1 - \ell + \ell p))!} \geq \frac{n!}{(p!)^{\ell} (n - \ell(p-1) + 1)!} \\ &\geq \frac{(\frac{n}{e})^n}{(e\sqrt{p}(\frac{p}{e})^p)^{\ell} (e\sqrt{n}(\frac{n}{e})^{n-\ell(p-1)+1})} = \frac{n^{\ell(p-1)-3/2}}{p^{\ell(p+1/2)}} \underbrace{\frac{e^{\ell p} e^{n-\ell(p-1)+1}}{e^{\ell+1} e^n}}_{=1} \end{aligned}$$

where the third inequality follows from Stirling's Formula, i.e., that for any $x \geq 2$, $(\frac{x}{e})^x \leq x! \leq \sqrt{2\pi x}(\frac{x}{e})^x e^{\frac{1}{12x}} \leq e\sqrt{x}(\frac{x}{e})^x$. Take the logarithm of the above equation to determine that

$$\begin{aligned}
\log s &\geq \left[\ell(p-1) - \frac{3}{2}\right]\log n - \ell\left(p+\frac{1}{2}\right)\log p \qquad (5)\\
&= \ell\left(p+\frac{1}{2}\right)(\log n - \log p) - \frac{3}{2}(\ell+1)\log n\\
&\geq \ell p(\log n - \log p) - \frac{3}{2}\ell\log n - \frac{3}{2}\log n,\\
\frac{5}{2}\log s &\geq \ell\left[p(\log n - \log p) - \frac{3}{2}\log n\right] \geq p(\log n - \log p) - \frac{3}{2}\log n, \qquad (6)\\
4\log s &\geq p(\log n - \log p) \geq p, \qquad (7)
\end{aligned}$$

where (6) follows from (5) since $s \geq n$ and $\ell \geq 1$, and (7) follows from (6) because p is the size of the second largest part of $\mathcal{P}$ and hence $p \leq \frac{n}{2}$ and $(\log n - \log p) \geq 1$. Take the logarithm of (7) and use the bound on s to determine that $\log p \leq \log\log s + 2 \leq (1-\epsilon)\log n + 2$. Plug this bound into (6) to get that $\frac{5}{2}\log s \geq \ell p(\epsilon \log n - 2) - \frac{3}{2}\ell\log n$. Using $\frac{\epsilon}{2}\log n \geq 2$ and dividing by $\log n$, $\frac{5}{2}\frac{\log s}{\log n} \geq \ell(\frac{p\epsilon}{2} - \frac{3}{2})$.

If $\frac{p\epsilon}{4} \geq \frac{3}{2}$, then $\frac{10}{\epsilon}\frac{\log s}{\log n} \geq \ell p$. For the largest value of ℓ, $k-1+(\ell+1)(p-1) \geq S$, and hence $k - 1 + 2\ell p \geq S$. Thus Lemma 5 implies that $S \leq \frac{8+20}{\epsilon}\frac{\log s}{\log n}$. Otherwise $p < \frac{6}{\epsilon}$ and hence $\log p \leq 3 - \log\epsilon$. Plugging this into (6) and using $\log n \geq \frac{8}{\epsilon^2} \geq 2(3 - \log\epsilon) \geq 2\log p$ gives

$$\frac{5}{2}\log s \geq \ell\frac{p-3}{2}\log n. \qquad (8)$$

If $p \geq 5$, then we can recover

$$\frac{5}{2\epsilon}\frac{\log s}{\log n} \geq \frac{5}{2}\frac{\log s}{\log n} \geq \ell\frac{p-3}{2} \geq \ell\frac{p}{5}$$

from (8) and conclude $S \leq \frac{8+25}{\epsilon}\frac{\log s}{\log n}$ analogously to before. Otherwise $p \leq 4$, and $S \leq p(k-1) \leq 4 \cdot \frac{8}{\epsilon}\frac{\log s}{\log n}$ by Lemma 5. Since in each case we concluded that $S \leq \frac{33}{\epsilon}\frac{\log s}{\log n}$ the proof is complete. □

3.2 Support Theorem

Let g be a gate in a rigid symmetric circuit C on structures of size n. From now on we abuse notation and write $\mathsf{SP}(g)$ for $\mathsf{SP}(\mathrm{Stab}_n(g))$. Note that, if P is any part in $\mathsf{SP}(g)$, then $[n] \setminus P$ is a support of g in the sense of Definition 7. We write $\|\mathsf{SP}(g)\|$ to denote the smallest value of $|[n] \setminus P|$ over all parts P in $\mathsf{SP}(g)$. Also, let $\mathsf{SP}(C)$ denote the maximum of $\|\mathsf{SP}(g)\|$ over all gates g in C.

The orbit-stabiliser theorem implies that $|\mathrm{Orb}(g)| = [\mathrm{Sym}_n : \mathrm{Stab}_n(g)]$. Using Lemma 4, we have that $\mathrm{Stab}_n(g) \subseteq \mathrm{SetStab}_n(\mathsf{SP}(g))$ and thus, if s is an upper bound on $|\mathrm{Orb}(g)|$, $s \geq [\mathrm{Sym}_n : \mathrm{Stab}_n(g)] \geq [\mathrm{Sym}_n : \mathrm{SetStab}_n(\mathsf{SP}(g))]$. Then, by

Lemma 6, g has a support of small size provided that (i) s is sub-exponential, and (ii) $\mathsf{SP}(g)$ has fewer than $n/2$ parts. Thus, to prove our main technical theorem, which formalises Theorem 3 from the introduction, it suffices to show that if s is sufficiently sub-exponential, (ii) holds.

Theorem 4 (Support Theorem) *For any ϵ and n with $\frac{2}{3} \leq \epsilon \leq 1$ and $n > 2^{\frac{96}{\epsilon^2}}$, if C is a rigid symmetric circuit on structures of size n and $s := \max_{g \in C} |\mathrm{Orb}(g)| \leq 2^{n^{1-\epsilon}}$, then, $\mathsf{SP}(C) \leq \frac{33}{\epsilon} \frac{\log s}{\log n}$.*

Proof Suppose $1 \leq s < n$. C cannot have relational inputs, because each relational gate must have an orbit of size at least n, so each gate of C computes a constant Boolean function. In this case the inputs to C must be constant gates, which are fixed under all permutations because they are the only gates with their label, i.e., 0 or 1. We have $|\mathsf{SP}(g)| = |\{[n]\}| = 1$ for each input gate g. This property extends inductively to the rest of C because C is rigid. Therefore the coarsest supporting partition of every gate g in C must be $\{[n]\}$, and hence $0 = \|\mathsf{SP}(g)\| = \mathsf{SP}(C)$. Therefore assume $s \geq n$.

To conclude the theorem from Lemma 6 it suffices to argue that for all gates g, $|\mathsf{SP}(g)| \leq \frac{n}{2}$. Suppose g is a constant gate, then, as we argued before, $|\mathsf{SP}(g)| = |\{[n]\}| = 1 < \frac{n}{2}$. If g is a relational gate, then it is fixed by any permutation that fixes all elements appearing in the tuple $\Lambda(g)$ and moved by all others. Thus, $\mathsf{SP}(g)$ must contain singleton parts for each element of $[n]$ in the tuple $\Lambda(g)$ and a part containing everything else. For the sake of contradiction suppose that $|\mathsf{SP}(g)| > \frac{n}{2}$ (i.e., the arity of the relation is large with respect to n), and therefore that $\mathsf{SP}(g)$ contains more than $\frac{n}{2}$ singleton parts. By the orbit-stabiliser theorem the orbit of g can be bounded as $|\mathrm{Orb}(g)| = \frac{n!}{(n-|\Lambda(g)|)!} \geq \frac{n!}{(\frac{n}{2})!}$. This is contradiction with the stated bound on s, because $s \geq |\mathrm{Orb}(g)| \geq \frac{n!}{(\frac{n}{2})!} \geq 2^{\frac{n}{4}} > 2^{n^{1-\epsilon}}$.

It remains to consider internal gates. For the sake of contradiction let g be a topologically first internal gate such that $\mathsf{SP}(g)$ has more than $\frac{n}{2}$ parts. Define $k' := \left\lceil \frac{8}{\epsilon} \frac{\log s}{\log n} \right\rceil$. Note that the assumptions on s, n,and ϵ imply that $k' \leq \frac{1}{4} n^{1-\epsilon} < \frac{n}{2}$. Lemma 5 implies that $n - |\mathsf{SP}(g)| \leq k'$.

Let H denote the children of g. Because g is a topologically first gate with $|\mathsf{SP}(g)| > \frac{n}{2}$, for all $h \in H$, $\mathsf{SP}(h)$ has at most $\frac{n}{2}$ parts. As before, we argue a contradiction with the upper bound on s. This is done by demonstrating that there is a set of gate-automorphism pairs $S = \{(h, \sigma) \mid h \in H, \sigma \in \mathrm{Sym}_n\}$ that are: (i) *useful* – the automorphism moves the gate out of the set of g's children, i.e., $\sigma h \notin H$, and (ii) *independent* – each child and its image under the automorphism are fixed points of the other automorphisms in the set, i.e., for all $(h, \sigma), (h', \sigma') \in S$, $\sigma' h = h$ and $\sigma' \sigma h = \sigma h$. Note that sets which are useful and independent contain tuples whose gate and automorphism parts are all distinct. The set S describes elements in the orbit of H with respect to Sym_n.

Claim Let S be useful and independent, then $|\mathrm{Orb}(H)| \geq 2^{|S|}$.

Proof Let R be any subset of S. Derive an automorphism from R: $\sigma_R := \prod_{(h,\sigma)\in R} \sigma$ (since automorphisms need not commute fix an arbitrary ordering of S).

Let R and Q be distinct subsets of S where without loss of generality $|R| \geq |Q|$. Pick any $(h, \sigma) \in R \backslash Q \neq \emptyset$. Because S is independent $\sigma_R h = \sigma h$ and $\sigma_Q \sigma h = \sigma h$. Since S is useful, $\sigma h \notin H$. Thus $\sigma h \in \sigma_R H$, but $\sigma h \notin \sigma_Q H$. Hence $\sigma_R H \neq \sigma_Q H$. Therefore each subset of S can be identified with a distinct element in $\mathrm{Orb}(H)$ and hence $|\mathrm{Orb}(H)| \geq 2^{|S|}$. □

Thus to reach a contradiction it suffices to construct a sufficiently large set S of gate-automorphism pairs. To this end, divide $[n]$ into $\lfloor \frac{n}{k'+2} \rfloor$ disjoint sets S_i of size $k'+2$ and ignore the elements left over. It follows that for each i there is a permutation σ_i which fixes $[n]\backslash S_i$ pointwise but moves g, as otherwise it implies $|\mathsf{SP}(g)| \leq n-(k'+2)+1 = n-k'-1$ which directly contradicts the bound of $n-|\mathsf{SP}(g)| \leq k'$. Since g is moved by each σ_i and C is rigid, there must be an associated child $h_i \in H$ with $\sigma_i h_i \notin H$. Thus let (h_i, σ_i) be the gate-automorphism pair for S_i, these pairs are *useful*. Let Q_i be the union of all the parts of $\mathsf{SP}(h_i)$ except the largest part together with all the parts of $\mathsf{SP}(\sigma_i h_i)$ except the largest part. Consider a σ_j which fixes Q_i pointwise, then, by the construction of Q_i, $\sigma_j \in \mathrm{Stab}_n(\mathsf{SP}(h_i)) \cap \mathrm{Stab}_n(\mathsf{SP}(\sigma_i h_i))$. This implies σ_j fixes both h_i and $\sigma_i h_i$.

Define a directed graph K on the sets S_i as follows. Include an edge from S_i to S_j, with $i \neq j$, if $Q_i \cap S_j \neq \emptyset$. An edge in K indicates a potential lack of independence between (h_i, σ_i) and (h_j, σ_j), and on the other hand if there are no edges between S_i and S_j, the associated pairs are independent (as, for example, $Q_i \cap S_j = \emptyset$ implies $Q_i \subseteq [n]\backslash S_j$ so that σ_i fixes Q_i pointwise). Thus it remains to argue that K has a large independent set. This is possible because the out-degree of S_i in K is bounded by

$$|Q_i| \leq \|\mathsf{SP}(h_i)\| + \|\mathsf{SP}(\sigma_i h_i)\| \leq 2 \cdot \frac{33}{\epsilon} \frac{\log s}{\log n}$$

as the sets S_i are disjoint and Lemma 6 can be applied to h_i. Thus the average total degree (in-degree + out-degree) of K is at most $2|Q_i| \leq 18k'$. Greedily select a maximal independent set in K by repeatedly selecting the S_i with the lowest total degree and eliminating it and its neighbours. This action does not affect the bound on the average total degree of K and hence determines an independent set I in K of size at least

$$\frac{\lfloor \frac{n}{k'+2} \rfloor}{18k'+1} \geq \frac{n-(k'+2)}{(18k'+1k'+2)} \geq \frac{\frac{n}{2}-1}{18k'^2+37k'+2} \geq \frac{\frac{7}{16}n}{18k'^2+37k'+2} \geq \frac{n}{(12k')^2}$$

where the first inequality follows by expanding the floored expression, the second follows because $k' < \frac{n}{2}$, the third follows from the lower bound on n, and the last follows because $k' \geq 1$ as it is the ceiling of a positive non-zero quantity by definition.

Take $S := \{(h_i, \sigma_i) \mid S_i \in I\}$. By the argument above S is useful and independent. By the above claim, conclude that $s \geq |\mathrm{Orb}(g)| \geq |\mathrm{Orb}(H)| \geq 2^{|S|} \geq 2^{\frac{n}{(12k')^2}}$. For $\epsilon \geq \frac{2}{3}$, $s \leq 2^{n^{1-\epsilon}}$, and $\frac{\epsilon}{96} \log n > 1$ the following is a contradiction $\log s \geq$

$n \cdot (\frac{96}{\epsilon}\frac{\log s}{\log n})^{-2} > n \cdot (n^{1-\epsilon})^{-2} = n^{2\epsilon-1} \geq n^{1-\epsilon}$. Thus $|\mathsf{SP}(g)| \leq \frac{n}{2}$ for all $g \in C$ and the proof is complete by Lemma 6. □

Observe that when s is polynomial in n the support of a rigid symmetric circuit family is asymptotically constant. This is the case for polynomial-size families.

Corollary 1 *Let $\mathcal{C}$ be a polynomial-size rigid symmetric circuit family, then* $\mathsf{SP}(\mathcal{C}) = O(1)$.

4 Translating Symmetric Circuits to Formulas

In this section, we deploy the support theorem to show that P-uniform families of polynomial-size symmetric circuits can be translated into formulas of fixed-point logic. As a first step, we argue in Section 4.1 that we can restrict our attention to rigid circuits, by showing that every symmetric circuit can be converted, in polynomial time, into an equivalent rigid symmetric circuit. In Section 4.2 we show that there are polynomial-time algorithms that will determine whether a circuit is symmetric and, if so, compute for every gate its coarsest supporting partition and therefore its canonical support. In Section 4.3 we give an inductive construction of a relation that associates to each gate g of C a set of tuples that when assigned to the support of g result in g being evaluated to true. This construction is turned into a definition in fixed-point logic in Section 4.4.

4.1 Rigid Circuits

We first argue that rigid circuits uniquely induce automorphisms.

Proof of Proposition 1 Let $\sigma \in \mathrm{Sym}_n$ induce the automorphisms π, π' of C. We show $\pi g = \pi' g$ for all gates g in C, and hence $\pi = \pi'$.

Observe that if g is an output gate, the image of g under any automorphism induced by σ must be $\Omega(\sigma\Omega^{-1}(g))$, because Ω is a function, and hence $\pi g = \pi' g$ is unique and completely determined by σ. Therefore assume that g is not an output gate. We proceed by induction on the height of g to show that $\pi g = \pi' g$.

In the base case g is an input gate. If g is a constant gate, g is the only constant gate of its type and hence all automorphisms of C must fix it. If g is a relational gate, g is the only relational gate with its type $\Sigma(g)$ and label $\Lambda(g)$ and it must map to the similarly unique gate with type $\Sigma(g)$ and tuple $\sigma\Lambda(g)$ and hence $\pi g = \pi' g$.

In the induction step g is an internal gate. By rigidity of C, g is unique for its children and type. Moreover, by induction the children of g map in the same way under π and π', and hence the image of g must be the same in both automorphisms. Thus $\pi g = \pi' g$ for all gates of C. □

To see that any symmetric circuit can be transformed in polynomial time into an equivalent rigid symmetric circuit, observe that we can proceed inductively from the input gates, identifying gates whenever they have the same label and the same set of children. This allows us to establish the following lemma.

Lemma 7 *Let $C = \langle G, W, \Omega, \Sigma, \Lambda \rangle$ be a $(\mathbb{B}, \tau)$-circuit on structures of size n. There is a deterministic algorithm which runs in time* $\mathrm{poly}(|C|)$ *and outputs a rigid $(\mathbb{B}, \tau)$-circuit C' with gates $G' = G$ such that for any $g \in G$, any input τ-structure $\mathcal{A}$ and any bijection γ from A to $[n]$, $C[\gamma\mathcal{A}](g) = C'[\gamma\mathcal{A}](g)$. Moreover, C' is symmetric if C is.*

Proof Partition the internal gates of G into equivalence classes where gates in the same class share the same operation, and have the same output markings and children. If C is rigid every class has size one, otherwise there is at least one class containing two gates.

Let E be a minimum height equivalence class containing at least two gates (here, the *height* of a gate is the length of a longest path from it to an input). Order the gates in E: $g_1, g_2, \ldots, g_{|E|}$. For each gate $f \in G \backslash E$, let c_f denote the number of wires from E to f, and note that $c_f \leq |E|$. For all gates in E remove all outgoing wires. For all gates $E \backslash \{g_1\}$: (i) remove all input wires, and (ii) set their operation to AND. For each i, $1 \leq i \leq |E| - 1$, add a wire from g_i to g_{i+1}. For each $f \in G \backslash E$ and $i \in [|E|]$, add a wire from g_i to f if $c_f \geq i$. This completes the transformation of the gates in E.

We now argue that this does not affect the result computed at any gate g. First observe that no output gates appear in E, because Ω is injective and hence each output gate must be the sole member of its equivalence class. All gates in E originally had identical sets of children and labels and hence they must have evaluated to the same value. The modifications made do not change this property as g_1 computes the value it originally would have, then passes this value to the other gates in E, along a chain of single input AND gates. The modifications to the outgoing wires of E insure that each gate that originally took input from E has the same number of inputs from E (each with the same value) in the modified circuit. Taken together this means that the result computed at any gate in the modified circuit is the same as that computed at that gate in C.

We next argue that the local modification of E makes strict progress towards producing a rigid circuit C'. The local modification of E can only change equivalence classes above E because the changes to the output wires of E are the only thing that can possibly affect other equivalence classes. After the modification all gates in E must be in singleton equivalence classes because each gate in E is designed to have a unique set of children.

Applying the above local modification simultaneously to all topologically minimal non-singleton equivalence classes of C, until none remain, produces a rigid circuit C' that computes the same query as C, because, as we have just argued, equivalence classes cannot grow as a result of this local modification. Moreover, this must happen after at most $|C|$ many local modifications, because the number of equivalence classes is at most $|C|$.

We now show that this transformation preserves symmetry. Suppose C is symmetric. Fix any permutation $\sigma \in \mathrm{Sym}_n$. Let π be an automorphism induced by σ on C. Observe that any induced automorphism on C must map equivalence classes to equivalence classes because labels and children are preserved. It is easy to translate

π into an induced automorphism of C'. Let E and E' be two topologically-minimal equivalence classes such that $\pi E = E'$ where $g_1, \ldots, g_{|E|}$ and $g'_1, \ldots, g'_{|E'|}$ are the ordering of the gates in E and E' in C', respectively. It can be argued by induction that mapping g_i to g'_i for all $1 \leq i \leq |E| = |E'|$ preserves all labels and wires and hence is an induced automorphism of σ in C'. Since σ is arbitrary, we conclude that the resulting circuit is symmetric.

The construction of equivalence classes and, indeed, the overall construction of C' can be easily implemented in time polynomial in $|C|$ when given the circuit in a reasonable binary encoding. Finally, as gates are only being rewired and relabeled, $G = G'$. □

4.2 Computing Supports

By Lemma 7, we know that there is a polynomial-time algorithm that converts a circuit into an equivalent rigid circuit while preserving symmetry. In this subsection we show how to, in polynomial time, check whether the resulting circuit is symmetric, and if it is, compute the support of each gate. To this end we first describe an algorithm for determining induced automorphisms of a rigid circuit.

Lemma 8 *Let C be a rigid $(\mathbb{B}, \tau)$-circuit on structures of size n and $\sigma \in \mathrm{Sym}_n$. There is a deterministic algorithm which runs in time* $\mathrm{poly}(|C|)$ *and outputs for each gate $g \in G$ its image under the automorphism π induced by σ, if it exists.*

Proof Process the gates of C recursively building up a mapping π. Compute the mapping for the children of a gate g before determining the mapping for g. If at any point an image for g cannot be located, halt and output that there is no induced automorphism.

Let g be a constant gate, then g is fixed under every automorphism. Let g be a relational gate, then there is at most one gate g' in C with $\Sigma(g) = \Sigma(g')$, $\sigma\Lambda(g) = \Lambda(g')$, and $\sigma\Omega^{-1}(g) = \Omega^{-1}(g')$. If g' exists, set πg to g', otherwise halt with failure. Similarly, when g is an internal gate use Σ, Ω, and the action of π on the children of G (via W) to determine a unique image of g, if it exists.

By Proposition 1 if σ induces an automorphism of C, it is unique and will be discovered by the above algorithm. This algorithm clearly runs in time polynomial in $|C|$. □

Using the preceding lemma we can determine whether a given rigid circuit is symmetric by computing the set of automorphisms induced by transpositions of the universe. If an induced automorphism fails to exist the circuit cannot be symmetric. Otherwise, it must be symmetric because such transpositions generate the symmetric group. If the circuit is symmetric, the coarsest supporting partitions and orbits of each gate can be determined by examining the transitive closure of the action of the automorphisms induced by transpositions on the universe and the gates, respectively.

Lemma 9 *Let C be a rigid $(\mathbb{B}, \tau)$-circuit for structures of size n. There is a deterministic algorithm which runs in time* poly($|C|$) *and decides whether C is symmetric. If C is symmetric the algorithm also outputs the orbits and coarsest supporting partitions of every gate.*

Proof For all transpositions $(uv) \in \mathrm{Sym}_n$ run the algorithm of Lemma 8 to determine the unique automorphism $\pi_{(uv)}$ of C induced by (uv), if it exists. Output that C is symmetric iff every induced automorphism $\pi_{(uv)}$ exists. This is correct because the set of transpositions generates all of Sym_n, and therefore the automorphisms $\pi_{(uv)}$ generate all induced automorphisms of C.

If C is symmetric, these induced automorphisms also indicate the supporting partitions and orbits of each gate g. Let $\mathcal{P}_{(uv)} := \{\{u, v\}\} \cup_{w \in [n] \setminus \{u,v\}} \{\{w\}\}$ be a partition of $[n]$. Note that $\pi_{(uv)}$ fixes g iff $\mathcal{P}_{(uv)}$ supports g. Let $\mathcal{P}$ be the partition determined by combining the partitions $\mathcal{P}_{(uv)}$ which support g using $\mathcal{E}$. Proposition 2 implies that $\mathcal{P}$ supports g. Suppose $\mathcal{P}$ is not the coarsest partition supporting g. Then, there exists $u, v \in [n]$ which are not in the same part of $\mathcal{P}$ but in the same part of some partition supporting g. But by the definition of $\mathcal{P}$, $\pi_{(uv)}$ cannot fix g—a contradiction. Therefore $\mathcal{P}$ is the coarsest partition supporting g.

To compute the orbit of a gate g: Start with $S_0 := \{g\}$, and for $i \geq 0$, compute $S_{i+1} := S_i \cup_{(uv) \in \mathrm{Sym}_{[n]}} \pi_{(uv)} S_i$. Let S be the least fixed point of this process. We argue that $S = \mathrm{Orb}(g)$. $S \subseteq \mathrm{Orb}(g)$, because it consists of gates reachable from g via a sequence of induced automorphisms of C. $S \supseteq \mathrm{Orb}(g)$, because the set of automorphisms induced by transpositions generate the group of all induced automorphisms. (This algorithm for orbit finding is well-known, c.f., e.g., [12]).

Since there are only $\binom{n}{2}$ transpositions, and we can determine whether there is an induced automorphism for each transposition in time poly($|C|$), we can determine whether C is symmetric in time poly($|C|$). If C is symmetric the computation of the supports and orbits of all gates also is computed in time poly($|C|$) because each output is completely determined by the equivalence classes induced by the relations defined by the induced automorphisms $\pi_{(uv)}$. Therefore the overall algorithm runs in time poly($|C|$). □

4.3 Succinctly Evaluating Symmetric Circuits

Let $\mathcal{C} = (C_n)_{n \in \mathbb{N}}$ be a family of polynomial-size rigid symmetric circuits computing a q-ary query. Let n_0 be a constant sufficient to apply the Support Theorem to C_n for $n \geq n_0$ and fix such an n. By Theorem 4, there is a constant bound k so that for each gate g in C_n the union of all but the largest part of the coarsest partition supporting g, $\mathsf{SP}(g)$, has at most k elements. Moreover, this union is a *support* of g in the sense of Definition 7. We call it the *canonical support* of g and denote it by $\mathrm{sp}(g)$.

Consider a structure $\mathcal{A}$ with universe U of size n. In this subsection we show that how a gate g evaluates in C_n with respect to $\mathcal{A}$ depends only on the injective partial

mapping of U to the elements in the canonical support of g (and not a complete bijection $\gamma : U \to [n]$). This allows us to succinctly encode the bijections which make a gate true (first as injective partial functions and then as tuples). This ultimately lets us build a fixed-point formula for evaluating C_n—indeed, all symmetric circuits—in the next subsection.

For any set $X \subseteq [n]$, let $U^{\underline{X}}$ denote the set of *injective* functions from X to U. For $X, Y \subseteq [n]$ and $\alpha \in U^{\underline{X}}$, $\beta \in U^{\underline{Y}}$, we say α and β are *consistent*, denoted $\alpha \sim \beta$, if for all $z \in X \cap Y, \alpha(z) = \beta(z)$, and for all $x \in X \backslash Y$ and $y \in Y \backslash X$, $\alpha(x) \neq \beta(y)$. Recall that any bijection $\gamma : U \to [n]$ determines an evaluation of the circuit C_n on the input structure $\mathcal{A}$ which assigns to each gate g the Boolean value $C_n[\gamma\mathcal{A}](g)$. (Note that $\gamma^{-1} \in U^{\underline{[n]}}$). Let g be a gate and let $\Gamma(g) := \{\gamma \mid C_n[\gamma\mathcal{A}](g) = 1\}$ denote the set of those bijections which make g evaluate to 1. The following lemma proves that the membership of γ in $\Gamma(g)$ (moreover, the number of 1s input to g) depends only on what γ maps $\mathrm{sp}(g)$ to.

Lemma 10 *Let g be a gate in C_n with children H. Let $\alpha \in U^{\underline{\mathrm{sp}(g)}}$, then for all $\gamma_1, \gamma_2 : U \to [n]$ with $\gamma_1^{-1} \sim \alpha$ and $\gamma_2^{-1} \sim \alpha$,*

1. *$\gamma_1 \in \Gamma(g)$ iff $\gamma_2 \in \Gamma(g)$.*
2. *$|\{h \in H \mid \gamma_1 \in \Gamma(h)\}| = |\{h \in H | \gamma_2 \in \Gamma(h)\}|$.*

Proof There is a unique permutation $\pi \in \mathrm{Sym}_n$ such that $\pi\gamma_1 = \gamma_2$. Moreover, π fixes sp$g$ pointwise, since γ_1^{-1} and γ_2^{-1} are consistent with α. Since C_n is rigid and symmetric, π is an automorphism of C_n, and we have that $C_n[\gamma_1\mathcal{A}](g) = C_n[(\pi\gamma_1)\mathcal{A}](\pi g)$. Since π fixes $\mathrm{sp}(g)$ pointwise, we have $\pi g = g$ and therefore $C_n[\gamma_1\mathcal{A}](g) = C_n[(\pi\gamma_1)\mathcal{A}](g) = C_n[\gamma_2\mathcal{A}](g)$, proving part 1. Similarly, for any child $h \in H$ we have that $C_n[\gamma_1\mathcal{A}](h) = C_n[(\pi\gamma_1)\mathcal{A}](\pi h) = C_n[\gamma_2\mathcal{A}](\pi h)$. Since π fixes g, π fixes H setwise. As this establishes a bijection between the children H that evaluate to 1 for γ_1 and γ_2, we conclude part 2. □

We associate with each gate g a set of injective functions $\mathrm{EV}_g \subseteq U^{\underline{\mathrm{sp}(g)}}$ defined by $\mathrm{EV}_g := \{\alpha \in U^{\underline{\mathrm{sp}(g)}} \mid \exists\gamma \in \Gamma(g) \wedge \alpha \sim \gamma^{-1}\}$ and note that, using Lemma 10, this completely determines $\Gamma(g)$. We can use the following lemma to recursively construct EV_g for all gates in C.

Lemma 11 *Let g be a gate in C with children H. Let $\alpha \in U^{\underline{\mathrm{sp}(g)}}$, then for all $\gamma : U \to [n]$ with $\gamma^{-1} \sim \alpha$,*

$$|\{h \in H \mid \gamma \in \Gamma(h)\}| = \sum_{h \in H} \frac{|A_h \cap \mathrm{EV}_h|}{|A_h|}, \tag{9}$$

where for $h \in H$, $A_h := \{\beta \in U^{\underline{\mathrm{sp}(h)}} \mid \alpha \sim \beta\}$.

Proof Write $\chi_{\Gamma(h)}$ for the characteristic (or indicator) function of $\Gamma(h)$. We have,

$$\begin{aligned}
|\{h \in H \mid \gamma \in \Gamma(h)\}| \;\cdot\; & |\{\delta \in U^{\underline{[n]}} \mid \delta \sim \alpha\}| \\
&= \sum_{\{\delta \in U^{\underline{[n]}} \mid \delta \sim \alpha\}} |\{h \in H \mid \delta^{-1} \in \Gamma(h)\}| \\
&= \sum_{h \in H} \sum_{\{\delta \in U^{\underline{[n]}} \mid \delta \sim \alpha\}} \chi_{\Gamma(h)}(\delta^{-1}) \\
&= \sum_{h \in H} \sum_{\beta \in A_h} \sum_{\{\delta \in U^{\underline{[n]}} \mid \delta \sim \alpha \wedge \delta \sim \beta\}} \chi_{\Gamma(h)}(\delta^{-1}) \\
&= \sum_{h \in H} \sum_{\beta \in A_h} |\{\beta \in \mathrm{EV}_h\}| \cdot |\{\delta \in U^{\underline{[n]}} \mid \delta \sim \alpha \wedge \delta \sim \beta\}|
\end{aligned} \tag{10}$$

where the first equality follows from Lemma 10 Part 2, the second by linearity of addition (note that $|\{\delta^{-1} \in \Gamma(h)\}| \in \{0, 1\}$), the third by the definitions of $\sim$ and A_h, and the fourth by the definition of EV_h. Observe that as β ranges over A_h, the sets $\{\delta \in U^{\underline{[n]}} \mid \delta \sim \alpha \wedge \delta \sim \beta\}$ are pairwise disjoint and all have the same size. Thus, $|A_h| \cdot |\{\delta \in U^{\underline{[n]}} \mid \delta \sim \alpha \wedge \delta \sim \beta\}| = |\{\delta \in U^{\underline{[n]}} \mid \delta \sim \alpha\}|$, and we conclude that

$$|\{h \in H \mid \gamma \in \Gamma(h)\}| = \sum_{h \in H} \sum_{\beta \in A_h} \frac{|\{\beta \in \mathrm{EV}_h\}|}{|A_h|} = \sum_{h \in H} \frac{|A_h \cap \mathrm{EV}_h|}{|A_h|}.$$

□

Note that implicit in the lemma is that the r.h.s. side of (9) is integral.

Since $[n]$ is linearly ordered, $X \subseteq [n]$ inherits this order and we write $\mathbf{X}$ for the ordered $|X|$-tuple consisting of the elements of X in the inherited order. For $\alpha \in U^{\underline{\mathbf{X}}}$ we write $\alpha \in U^{\underline{\mathbf{X}}}$ to indicate the $|X|$-tuple resulting from applying α to each component of $\mathbf{X}$ in order. Observe that this transformation is invertible. This allows us to succinctly encode such injective functions as tuples over U and, further, to write relational analogs of the sets of injective functions we considered before, e.g., $\mathbf{EV}_g := \{\alpha \mid \alpha \in \mathrm{EV}_g\}$, and $\mathbf{A}_h = \{\alpha \mid \alpha \in A_h\}$. Using Lemma 11 it is easy to recursively define $\mathbf{EV}_g$ over C_n.

- Let g be a constant input gate, then $\mathrm{sp}(g)$ is empty. If $\Sigma(g) = 0$, then $\Gamma(g) = \emptyset$ and $\mathbf{EV}_g = \emptyset$. Otherwise $\Sigma(g) = 1$, then $\Gamma(g)$ is all bijections and $\mathbf{EV}_g = \{\langle\rangle\}$, i.e., the set containing the empty tuple.
- Let g be a relational gate with $\Sigma(g) = R \in \tau$, then $\mathrm{sp}(g)$ is the set of elements in the tuple $\Lambda_R(g)$. By definition we have

$$\mathbf{EV}_g = \{\alpha \in U^{\underline{\mathbf{sp}(g)}} \mid \alpha(\Lambda_R(g)) \in R^{\mathcal{A}}\}.$$

- Let $\Sigma(g) = \mathrm{AND}$ and consider $\alpha \in U^{\underline{\mathbf{sp}(g)}}$. By Lemma 11, $\alpha \in \mathbf{EV}_g$ iff $\mathbf{A}_h \subseteq \mathbf{EV}_h$ for *every* child h of g, i.e., for *every* child h and *every* $\beta \in U^{\underline{\mathrm{sp}(h)}}$ with $\alpha \sim \beta$, we have $\beta \in \mathbf{EV}_h$.

- Let $\Sigma(g) = \text{OR}$ and consider $\alpha \in U^{\underline{\mathbf{sp}(g)}}$. By Lemma 11, $\alpha \in \mathbf{EV}_g$ iff there is a child h of g where $\mathbf{A}_h \cap \mathbf{EV}_h$ is non-empty, i.e., for *some* child h of g and *some* $\beta \in U^{\underline{\mathbf{sp}(h)}}$ with $\alpha \sim \beta$, we have $\beta \in \mathbf{EV}_h$.
- Let $\Sigma(g) = \text{NAND}$ and consider $\alpha \in U^{\underline{\mathbf{sp}(g)}}$. By Lemma 11, $\alpha \in \mathbf{EV}_g$ iff there is a child h of g where $\mathbf{A}_h \not\subseteq \mathbf{EV}_h$, i.e., for *some* child h and *some* $\beta \in U^{\underline{\mathbf{sp}(h)}}$ with $\alpha \sim \beta$, we have $\beta \notin \mathbf{EV}_h$.
- Let $\Sigma(g) = \text{MAJ}$ and consider $\alpha \in U^{\underline{\mathbf{sp}(g)}}$. Let H be the set of children of g. Then Lemma 11 implies that $\alpha \in \mathbf{EV}_g$ if, and only if,

$$\sum_{h \in H} \frac{|\mathbf{A}_h \cap \mathbf{EV}_h|}{|\mathbf{A}_h|} \geq \frac{|H|}{2}. \tag{11}$$

From EV we can recover the q-ary query Q computed by $C_n = \langle G, W, \Omega, \Sigma, \Lambda \rangle$ on the input structure $\mathcal{A}$ because the support of an output gate $g \in G$ is exactly the set of elements in the marking of g by Λ_Ω. In particular:

$$Q = \{\bar{a} \in U^q \mid \exists g \in G, \alpha \in \mathbf{EV}_g \text{ such that} \Lambda_\Omega(\alpha^{-1}(\bar{a})) = g\}.$$

For Boolean properties $q = 0$, and $Q = \{\langle\rangle\}$ indicates that $\mathcal{A}$ has the property and $Q = \emptyset$ indicates that it does not.

4.4 Translating to Formulas of FP

Let $\mathcal{C} = (C_n)_{n \in \mathbb{N}}$ be a P-uniform family of polynomial-size symmetric $(\mathbb{B}, \tau)$ circuits, where $\mathbb{B}$ is either $\mathbb{B}_{\text{std}}$ or $\mathbb{B}_{\text{maj}}$. Our aim is to show that there is a formula Q of FP, or FPC in the case of $\mathbb{B}_{\text{maj}}$, in the vocabulary $\tau \uplus \{\leq\}$ such that for any n and τ-structure $\mathcal{A}$ over a universe U with $|U| = n$, the q-ary query defined by C_n on input $\mathcal{A}$ is defined by the formula Q when interpreted in the structure $\mathcal{A}^{\leq} := \mathcal{A} \uplus \langle [n], \leq \rangle$.

Since $\mathcal{C}$ is P-uniform, by the Immerman-Vardi theorem and Lemma 7, we have an FP interpretation defining a rigid symmetric circuit equivalent to C_n—that we also call C_n—over the number sort of $\mathcal{A}^{\leq}$, i.e., a sequence $\Phi := (\phi_G, \phi_W, \phi_\Omega, (\phi_s)_{s \in \mathbb{B} \uplus \tau \uplus \{0,1\}}, (\phi_{\Lambda_R})_{R \in \tau})$ of formulas of FP$(\leq)$ that define the circuit when interpreted in $\langle [n], \leq \rangle$. Note that C_n is defined over the universe $[n]$. Let t be the arity of the interpretation, i.e., ϕ_G defines a t-ary relation $G \subseteq [n]^t$. If n is less than n_0, the length threshold for applying the support theorem, C_n can trivially be evaluated by a FP formula which quantifies over all (constantly-many) bijections from the point sort of $\mathcal{A}^{\leq}$ to the number sort of $\mathcal{A}^{\leq}$ and then directly evaluates the circuit with respect to the bijection. Thus we only need to consider the case when $n \geq n_0$, and are able to use the recursive construction of $\vec{\text{EV}}$ from the Section 4.3 along with a constant bound k on the size of the gate supports in C_n.

A small technical difficulty arises from the fact that we want to define the relation $\vec{\text{EV}}_g$ inductively, but these are actually relations of varying arities, depending on the size of $\text{sp}(g)$. For the sake of a uniform definition, we extend $\vec{\text{EV}}_g$ to a k-ary relation for all g by padding it with all possible values to obtain tuples of length k. If $|\text{sp}(g)| = \ell \leq k$, define

$$\overline{\text{EV}}_g = \{(a_1 \cdots a_k) \mid (a_1 \cdots a_\ell) \in \mathbf{EV}_g \text{ and} a_i \neq a_j \text{ for} i \neq j\}.$$

Define the relation $V \subseteq [n]^t \times U^k$ by $V(g, \bar{a})$ if, and only if, $\bar{a} \in \overline{\mathrm{EV}}_g$. Our aim is to show that the relation V is definable by a formula of FP. Throughout this subsection we use μ and ν to indicate t-tuples of number variables which denote gate indexes in $[n]^t$, and use the k-tuples of point variables $\bar{x} = (x_1, \ldots, x_k)$ and $\bar{y} = (y_1, \ldots, y_k)$ to denote injective functions that have been turned into tuples and then padded.

By Lemma 9 and invoking the Immerman-Vardi theorem again, we have a formula SUPP such that $\langle [n], \leq \rangle \models \mathrm{SUPP}[g, u]$ if, and only if, $\langle [n], \leq \rangle \models \phi_G[g]$ (i.e., g is a gate of C_n as defined by the interpretation Φ) and u is in $\mathrm{sp}(g)$. We use SUPP to define some additional auxiliary formulas. First we define, for each i with $1 \leq i \leq k$ a formula SUPP_i such that $\langle [n], \leq \rangle \models \mathrm{SUPP}_i[g, u]$ if, and only if, u is the i^{th} element of $\mathrm{sp}(g)$. These formulas can be defined inductively as follows, where η is a number variable

$$\begin{aligned}
\mathrm{SUPP}_1(\mu, \eta) &:= \mathrm{SUPP}(\mu, \eta) \wedge \forall \chi (\chi < \eta) \implies \neg \mathrm{SUPP}(\mu, \chi) \\
\mathrm{SUPP}_{i+1}(\mu, \eta) &:= \mathrm{SUPP}(\mu, \eta) \wedge \exists \chi_1 (\chi_1 < \eta \wedge \mathrm{SUPP}_i(\mu, \chi_1) \\
&\quad \wedge \forall \chi_2 [(\chi_1 < \chi_2 < \eta) \implies \neg \mathrm{SUPP}(\mu, \chi_2)]).
\end{aligned}$$

We now define a formula $\mathrm{AGREE}(\mu, \nu, \bar{x}, \bar{y})$ so that for a structure $\mathcal{A}$, $\mathcal{A}^{\leq} \models \mathrm{AGREE}[g, h, \bar{a}, \bar{b}]$ if, and only if, $\alpha \sim \beta$ for $\vec{\alpha} \in U^{\underline{\mathrm{sp}(g)}}$, $\vec{\beta} \in U^{\underline{\mathrm{sp}(h)}}$ that are the restrictions of the k-tuples $\bar{a}$ and $\bar{b}$ to the length of $\mathrm{sp}(g)$ and $\mathrm{sp}(h)$ respectively.

$$\mathrm{AGREE}(\mu, \nu, \bar{x}, \bar{y}) :=$$

$$\begin{aligned}
\bigwedge_{1 \leq i, j, \leq k} & (\forall \eta (\mathrm{SUPP}_i(\mu, \eta) \wedge \mathrm{SUPP}_j(\nu, \eta)) \implies x_i = y_j) \wedge \\
& (\forall \eta_1 \eta_2 (\mathrm{SUPP}_i(\mu, \eta_1) \wedge \mathrm{SUPP}_j(\nu, \eta_2) \wedge x_i = y_j) \implies \eta_1 = \eta_2)
\end{aligned}$$

With these, we now define a series of formulas $(\theta_s)_{s \in \mathbb{B} \uplus \tau \uplus \{0,1\}}(\mu, x)$ corresponding to the various cases of the construction of the relation $\mathbf{EV}_g$ from Section 4.3. In these, V is a relational variable for the relation being inductively defined.

$$\begin{aligned}
\theta_0(\mu, \bar{x}) &:= \text{false} \\
\theta_1(\mu, \bar{x}) &:= \bigwedge_{1 \leq i < j \leq k} x_i \neq x_j \\
\theta_R(\mu, \bar{x}) &:= \bigwedge_{1 \leq i < j \leq k} x_i \neq x_j \wedge \exists z_1 \cdots z_r \exists \eta_1 \cdots \eta_r R(z_1, \ldots, z_r) \wedge \phi_{\Lambda_R}(\mu, \eta) \wedge \\
&\qquad \bigwedge_{1 \leq i \leq r} \bigwedge_{1 \leq j \leq k} (\mathrm{SUPP}_j(\mu, \eta_i) \implies z_i = x_j) \\
\theta_{\mathrm{OR}}(\mu, \bar{x}) &:= \bigwedge_{1 \leq i < j \leq k} x_i \neq x_j \wedge \exists \nu \exists \bar{y} (W(\nu, \mu) \wedge \mathrm{AGREE}(\mu, \nu, \bar{x}, \bar{y}) \wedge V(\nu, \bar{y})) \\
\theta_{\mathrm{AND}}(\mu, \bar{x}) &:= \bigwedge_{1 \leq i < j \leq k} x_i \neq x_j \wedge \forall \nu \forall \bar{y} ((W(\nu, \mu) \wedge \mathrm{AGREE}(\mu, \nu, \bar{x}, \bar{y})) \implies V(\nu, \bar{y})) \\
\theta_{\mathrm{NAND}}(\mu, \bar{x}) &:= \bigwedge_{1 \leq i < j \leq k} x_i \neq x_j \wedge \exists \nu \exists \bar{y} (W(\nu, \mu) \wedge \mathrm{AGREE}(\mu, \nu, \bar{x}, \bar{y}) \wedge \neg V(\nu, \bar{y}))
\end{aligned}$$

To define θ_{MAJ} we start with some observations. We wish to formalise (11), but there are a few complications. The k-*ary* relation $\overline{\mathrm{EV}}_h$ we are defining inductively is the result of padding $\mathbf{EV}_h$ with all tuples of $k - |\mathrm{sp}(h)|$ distinct elements. Thus the

number of elements in $\overline{\mathrm{EV}}_h$ is $|\mathbf{EV}_h| \cdot \frac{(n-|\mathrm{sp}(h)|)!}{(n-k)!}$. Similarly, for any fixed g, h and $\bar{a}$, if we write $\bar{A}_h$ for the set of tuples $\bar{b}$ satisfying $\mathrm{AGREE}(g, h, \bar{a}, \bar{b})$, then $|\bar{A}_h| = |\mathbf{A}_h| \cdot \frac{(n-|\mathrm{sp}(h)|)!}{(n-k)!}$. Finally, the tuples in $\bar{A}_h \cap \overline{\mathrm{EV}}_h$ are exactly those obtained by padding tuples in $\mathbf{A}_h \cap \mathbf{EV}_h$ to length k and there are therefore $|\mathbf{A}_h \cap \mathbf{EV}_h| \cdot \frac{(n-|\mathrm{sp}(h)|)!}{(n-k)!}$ many of these. Thus, $\frac{|\vec{A}_h \cap \vec{\mathrm{EV}}_h|}{|\vec{A}_h|} = \frac{|\bar{A}_h \cap \overline{\mathrm{EV}}_h|}{|\bar{A}_h|}$ and it suffices to compute the latter. Observe that $|\vec{A}_h|$ and $|\bar{A}_h|$ are completely determined by $|\mathrm{sp}(g)|$, $|\mathrm{sp}(h)|$ and $|\mathrm{sp}(g) \cap \mathrm{sp}(h)|$. We avoid dealing explicitly with fractions by noting that for any gate h, the sum $\sum_{h' \in \mathrm{Orb}(h)} \frac{|\vec{A}_{h'} \cap \vec{\mathrm{EV}}_{h'}|}{|\vec{A}_{h'}|}$ is an integer (by an argument analogous to Lemma 11). Since $|\bar{A}_{h'}|$ is the same for all $h' \in \mathrm{Orb}(h)$, it suffices to compute the sum of $|\bar{A}_{h'} \cap \overline{\mathrm{EV}}_{h'}|$ for all h' with a fixed size of $|\bar{A}_h|$ and then divide the sum by $|\bar{A}_h|$. This is what we use to compute the sum on the l.h.s. of (11).

For any fixed i and j with $0 \leq i \leq j \leq k$, define the formula $\mathrm{OVERLAP}_{ij}(\mu, \nu)$ so that $\mathcal{A}^{\leq} \models \mathrm{OVERLAP}_{ij}[g, h]$ iff $|\mathrm{sp}(h)| = j$ and $|\mathrm{sp}(g) \cap \mathrm{sp}(h)| = i$. This formula can be defined in FO.

Using k-tuples of number variables in FPC we can represent natural numbers less than n^k. We assume, without giving a detailed construction of the formulas involved, that we can define arithmetic operations on these numbers. In particular, we assume we have for each i, j as above a formula $\mathrm{ASIZE}_{ij}(\mu, \xi)$, with ξ a k-tuple of number variables, such that $\mathcal{A}^{\leq} \models \mathrm{ASIZE}_{ij}[g, e]$ iff $e = |\bar{A}_h|$ for any gate h with $|\mathrm{sp}(h)| = j$ and $|\mathrm{sp}(g) \cap \mathrm{sp}(h)| = i$.

Using this, we define the formula $\mathrm{NUM}_{ij}(\mu, \bar{x}, \xi)$, with ξ a k-tuple of number variables, so that $\mathcal{A}^{\leq} \models \mathrm{NUM}_{ij}[g, \bar{a}, e]$ iff e is the number of gates h with $\mathcal{A}^{\leq} \models \mathrm{OVERLAP}_{ij}[g, h]$ which are made true by some bijection that assigns the tuple $\bar{a}$ to $\mathrm{sp}(g)$. This formula is given by

$$\mathrm{NUM}_{ij}(\mu, \bar{x}, \xi) :=$$

$$\begin{aligned} \exists \xi_1 \xi_2 \ & \mathrm{ASIZE}_{ij}(\mu, \xi_1) \wedge \\ & \xi_2 = \#_{\nu \bar{y}}(W(\nu, \mu) \wedge \mathrm{OVERLAP}_{ij}(\mu, \nu) \wedge V(\nu, \bar{y}) \wedge \mathrm{AGREE}(\mu, \nu, \bar{x}, \bar{y})) \wedge \\ & \xi \cdot \xi_1 = \xi_2. \end{aligned}$$

Now we can define the required formula θ_{MAJ} by

$$\theta_{\mathrm{MAJ}}(\mu, \bar{x}) :=$$

$$\bigwedge_{1 \leq i < j \leq k} x_i \neq x_j \wedge \exists \xi (2 \cdot \xi \geq \#_{\nu} W(\nu, \mu)) \wedge \xi = \sum_{0 \leq i \leq j \leq k} \{\xi' \mid \mathrm{NUM}_{ij}(\mu, \bar{x}, \xi')\},$$

where the sum inside the formula is to be understood as shorthand for taking the sum over the bounded number of possible values of i and j.

Now, we can define the relation $V \subseteq [n]^t \times U^k$ given by $V(g, \bar{a})$ if, and only if, $\bar{a} \in \overline{\mathrm{EV}}_g$ by the following formula

$$\theta(\mu, \bar{x}) := [\mathrm{ifp}_{V, \nu \bar{y}} \bigvee_{s \in \mathbb{B} \uplus \tau \uplus \{0,1\}} (\phi_s(\mu) \wedge \theta_s(\nu, \bar{y}))](\mu, \bar{x}).$$

The overall q-ary output query computed by the circuit is given by the following formula derived from the final construction in the last subsection

$$Q(z_1, \ldots, z_q) :=$$

$$\begin{aligned}\exists \mu \bar{x} \nu_1 \cdots \nu_q \eta_1 \cdots \eta_k [\theta(\mu, \bar{x}) \wedge \phi_\Omega(\nu_1, \ldots, \nu_q, \mu) \wedge \\ \textstyle\bigwedge_{1 \le i \le k} (\mathrm{SUPP}_i(\mu, \eta_i) \vee \forall \eta [\neg \mathrm{SUPP}_i(\mu, \eta)]) \wedge \\ \textstyle\bigwedge_{1 \le i \le k} \bigwedge_{1 \le j \le q} ([\mathrm{SUPP}_i(\mu, \eta_i) \wedge x_i = z_j] \implies \nu_j = \eta_i) \wedge \\ \textstyle\bigwedge_{1 \le j \le q} \bigvee_{1 \le i \le k} (x_i = z_j \wedge \mathrm{SUPP}_i(\mu, \eta_i))]\end{aligned}$$

where the purpose of the last three lines is to invert the injective function encoded in $\bar{x}$ and then apply it to z_i to produce ν_i; in particular: the second line puts the ordered support of μ into $\eta_1, \ldots, \eta_k$, the third line defines the map from z_i to ν_i, and the fourth line ensures that this map covered all coordinates of z_i.

Note that this is a formula of FP $+ \le$ if $\mathbb{B}$ is the standard basis and a formula of FPC if $\mathbb{B}$ is the majority basis. Moreover, if the family $\mathcal{C} = (C_n)_{n \in \mathbb{N}}$ of polynomial-size symmetric circuits is not uniform, but given by an advice function Υ, the construction gives us an equivalent formula of FP $+ \Upsilon$ (for the standard basis) or FPC $+ \Upsilon$ (for the majority basis). This may be formalised as follows.

Lemma 12 1. *Any relational query defined by a* P*-uniform family of polynomial-size symmetric circuits over the standard basis is definable in* FP$+ \le$.
2. *Any relational query defined by a* P*-uniform family of polynomial-size symmetric circuits over the majority basis is definable in* FPC.
3. *Any relational query defined by a* P*-uniform family of polynomial-size symmetric circuits over the standard basis is definable in* FP $+ \Upsilon$*, for some advice function* Υ.

5 Consequences

Formulas of FP $+ \le$ can be translated into P-uniform families of polynomial-size symmetric Boolean circuits by standard methods (see [16]) and similar translations hold for FPC and FP $+ \Upsilon$. This combined with Lemma 12 proves our main theorem.

Theorem 5 (Main) *The following pairs of classes define the same queries on structures:*

1. *Symmetric* P*-uniform polynomial-size Boolean circuits and* FP $+ \le$.
2. *Symmetric* P*-uniform polynomial-size majority circuits and* FPC.
3. *Symmetric polynomial-size majority circuits and* FPC $+ \Upsilon$.

One consequence is that properties of graphs which we know not to be definable in FPC are also not decidable by P-uniform families of polynomial-size symmetric circuits. The results of Cai-Fürer-Immerman [4] give graph properties that are polynomial-time decidable, but not definable in FPC. Furthermore, there are a number of NP-complete graph problems known not to be definable in FPC, including Hamiltonicity and 3-colourability (see [6]). This contrasts with a number of graph

properties that have been shown to be definable in FPC, including graphs that have perfect matchings [2] and any proper minor-closed class of graphs (see [13]).

The proofs establishing that the above-mentioned properties are not definable in FPC actually show that these properties are not even definable in the infinitary logic with a bounded number of variables and counting ($C^{\omega}_{\infty\omega}$—see [16]). For our purposes, we can think of formulas of this infinitary logic as families $(\phi_n)_{n\in\mathbb{N}}$ of formulas of first-order logic with counting quantifiers (FOC) such that there is a $k \in \mathbb{N}$ so that no formula ϕ_n uses more than k variables. Such a family defines the class $\mathcal{C}$ of structures such that $\mathcal{A} \in \mathcal{C}$ iff $|\mathcal{A}| = n$ and $\mathcal{A} \models \phi_n$. It is not difficult to show that formulas of FPC + Υ can be translated into $C^{\omega}_{\infty\omega}$, using the fact that for any vocabulary τ there is a k such that any property of τ-structures with at most n elements can be expressed by a first-order formula with at most k variables (see [8]). This gives us the following.

Corollary 2 *Hamiltonicity and 3-colourability of graphs are not decidable by polynomial-size families of symmetric majority circuits.*

We can say more about the correspondence between the infinitary logic $C^{\omega}_{\infty\omega}$ and symmetric circuits. In the proof of the Support Theorem (Theorem 4), we only use the upper bound on the size of the circuits to bound the size of *orbits* of individual gates. This motivates the following definition (cp. Definition 12 below).

Definition 10 A family of symmetric $(\mathbb{B}, \tau)$-circuits $\mathcal{C} = (C_n)_{n\in\mathbb{N}}$ is *orbit-polynomial* if there is polynomial p such that for any gate g in C_n, the orbit of g in C_n under the action of Sym_n has size at most $p(n)$.

Then the following strengthening of Corollary 1 is a direct consequence of the proof of Theorem 4.

Corollary 3 *Let $\mathcal{C}$ be an orbit-polynomial rigid symmetric circuit family, then* $\mathsf{SP}(\mathcal{C}) = O(1)$.

It is not hard to see that a formula of $C^{\omega}_{\infty\omega}$ can be translated into an orbit-polynomial family of symmetric majority circuits. This is by the standard translation of a formula ϕ_n of FOC into a circuit C_n with majority gates. This creates a gate for each pair (ψ, a) where ψ is a sub-formula of ϕ_n and a is a tuple from $[n]$ that acts as an assignment to the free variables of ψ. The orbit of the gate (ψ, a) is the set of all gates $(\psi, \pi a)$ for $\pi \in \mathrm{Sym}_n$. Since the number of free variables of ψ and hence the length of a is bounded by a constant k, we see that all orbits have size at most n^k. Moreover, a simple adaptation of the translation of Section 4.4 shows that any family of orbit-polynomial symmetric circuits can be translated into a formula of $C^{\omega}_{\infty\omega}$. This is obtained by translating each circuit C_n into a formula of FOC. Once n is fixed, we do not need fixed-points to define the relations $\overline{\mathrm{EV}}_g$ as the recursion can be unfolded to give a first-order formula. Moreover, the maximum number of free variables occurring in any sub-formula is given by a function of the size of the supports $\mathrm{sp}(g)$. We avoid the fixed-points introduced by the use of the Immerman-Vardi

theorem by invoking the fact that on ordered structures of a fixed signature τ and a fixed size n, we can express any property using a first-order formula with a constant number of variables (see [8]). Together these establish that the total number of variables required in our formulas is bounded by a constant. Taken all together, this establishes the following.

Theorem 6 *A query is decidable by an orbit-polynomial family of symmetric majority circuits if and only if it is definable in* $C^{\omega}_{\infty\omega}$.

6 Coherent and Locally Polynomial Circuits

In this section we discuss connections with the prior work of Otto [17]. Otto studies rigid symmetric Boolean circuits deciding Boolean properties of structures and provides uniformity conditions on such families that characterise bounded-variable fragments of finite and infinitary first-order logic. Otto defines two properties to establish his notion of uniformity. The first property is called coherence; informally, a circuit family $(C_n)_{n\in\mathbb{N}}$ is coherent if C_n appears as a subcircuit of all but finitely many of the circuits at larger input lengths.

Definition 11 (Coherence) Let $\mathcal{C} := (C_n)_{n\in\mathbb{N}}$ be a family of rigid symmetric $(\mathbb{B}_{\text{std}}, \tau)$-circuits computing a Boolean function. The circuit C_n *embeds* into the circuit C_m with $m > n$ if there is a subcircuit of C_m which is isomorphic to C_n. An embedding is *complete* if its images are exactly those gates of C_m which are fixed by $\text{Sym}[m]\backslash[n]$. The circuit family $\mathcal{C}$ is *coherent* if for each $n \in \mathbb{N}$, C_n completely embeds into C_m for all large enough $m > n$.

The second property is locally polynomial; informally, a circuit family is locally polynomial if the size of the orbit of every wire is polynomially bounded.

Definition 12 (Locally Polynomial) A rigid circuit family $(C_n)_{n\in\mathbb{N}}$ is *locally polynomial of degree* k if there is a $k \in \mathbb{N}$ such that each C_n and every subset $S \subseteq [n]$, the size of the orbit of every wire with respect to the automorphisms of the circuit induced by Sym_S is at most $|S|^k$.

The main result of [17, Theorem 6] establishes an equivalence between coherent locally-polynomial (of degree k) families of rigid symmetric $(\mathbb{B}_{\text{std}}, \tau)$-circuits computing Boolean functions on $\text{fin}[\tau]$ and infinitary first-order logic with k variables. It should be noted that in Otto's definition of circuit families the individual circuits in the family may themselves be infinite, as the only size restriction is on the orbits of gates. The theorem also shows that if the circuit families are also constant depth they correspond to the fragment of first-order logic with k variables.

The common restriction of notions of uniformity we consider in this paper is that the circuits have size polynomial in their input length. If we restrict ourselves to locally-polynomial coherent symmetric families where the individual circuits are *finite*, we can use the Support Theorem (Corollary 1) to establish a direct connection

with polynomial-size symmetric circuit families, formally stated in the following proposition.

Proposition 3 *Let $\mathcal{C} := (C_n)_{n\in\mathbb{N}}$ be a family of finite rigid symmetric Boolean circuits.*

1. *If $\mathcal{C}$ is a locally-polynomial coherent family, then $\mathcal{C}$ is polynomial size.*
2. *If $\mathcal{C}$ is polynomial size, then $\mathcal{C}$ is locally polynomial.*

Proof We prove the two parts separately.

Part 1. Suppose to the contrary that C_n has $s(n) = \omega(\text{poly}(n))$ gates. Because $\mathcal{C}$ is locally polynomial the Support Theorem gives a bound $k \in \mathbb{N}$ on the size of the support of gates in $\mathcal{C}$. Take $m \in \mathbb{N}$ such that C_m is a circuit such that C_k completely embeds into C_m and $s(k) \cdot m^k < s(m)$, such m exists because $\mathcal{C}$ is coherent and s is super polynomial. By symmetry and averaging there are at least $\frac{s(m)}{m^k}$ gates of C_m whose supports are drawn from $[k]$. These gates are necessarily fixed by $\text{Sym}_{[m]\setminus[k]}$. Since the embedding is complete, C_k maps onto at least these gates. But this is a contradiction because $s(k) < \frac{s(m)}{m^k}$. Thus $\mathcal{C}$ has polynomially many gates.

Part 2. If $\mathcal{C}$ has polynomially many gates then the Support Theorem immediately implies that the supports of all gates in $\mathcal{C}$ is bounded by some $k \in \mathbb{N}$. Therefore for every $S \subseteq [n]$ every wire in $C_n \in \mathcal{C}$ has its orbit size bounded by $|S|^{2k}$. This is exactly the definition of locally polynomial.

□

Since there are properties definable in an infinitary logic with finitely many variables that are not decidable by polynomial-size circuits, it follows from the above proposition that the use of infinite circuits is essential in Otto's result.

Proposition 3 implies that all uniform circuit families we consider are locally polynomial. However, it does not establish an equivalence between a circuit family having polynomially many gates and being locally polynomial and coherent. Indeed there are Boolean circuit families uniformly definable in FO + ≤ that are not coherent. To see this observe that such circuit families may include gates that are completely indexed by the number sort and hence are fixed under all automorphisms induced by permutations of the point sort. Moreover the number of such gates may increase as a function of input length. However, because coherence requires that *complete* embedding exist, the number of gates in each circuit of a coherent family that are not moved by any automorphism must be identical. Thus there are uniform circuits that are not coherent.

Consider weakening the definition of coherence to require only that an embedding exists but not that the embedding is complete, and call this *partial coherence*. One can show that any relation which can be computed by a Boolean circuit family uniformly definable in FO + ≤ can also be computed by a partially coherent Boolean circuit family with the same uniformity by appropriately creating copies of circuits relativised for all shorter lengths. We omit any formal discussion of this construction.

7 Future Directions

One of our original motivations for studying symmetric majority circuits was the hope that they had the power of choiceless polynomial time with counting (CPTC) [3], and that, perhaps, techniques from circuit complexity could improve our understanding of the relationship between CPTC and the invariant queries definable in polynomial-time. However, because FPC $\subsetneq$ CPTC [9], our results indicate that symmetry is too much of restriction on P-uniform circuit families to recover CPTC.

A natural way to weaken the concept of symmetry is to require that induced automorphisms exist only for a certain subgroup of the symmetric group. This interpolates between our notion of symmetric circuits and circuits on linearly-ordered structures, with the latter case occurring when the subgroup is trivial. An easier first step may be to consider the action on structures with a finite number of disjoint sorts and require only that automorphisms be induced by permutations which preserve the sorts, e.g., structures interpreting Boolean matrices whose rows and columns are indexed by disjoint sets.

The Support Theorem is a fairly general statement about the structure of symmetric circuits and is largely agnostic to the particular semantics of the basis. To that end the Support Theorem may find application to circuits over bases not consider here. The Support Theorem can be applied to arithmetic circuits computing invariant properties of matrices over a field; e.g., the Permanent polynomial is invariant and one standard way to compute it is as a symmetric arithmetic circuit, i.e., Ryser's formula [18].

Known inexpressibility results for FPC and $C^{\omega}_{\infty\omega}$ often give explicit lower bounds on the number of variables required to express a problem. It may be instructive to translate these into size lower bounds for symmetric circuits. One barrier to obtaining exponential lower bounds is that the Support Theorem has a subexponential bound built into it. Perhaps the form of the Support Theorem can be improved as the particular bound required on the orbit size does not appear to be fundamental to the conclusion of the Support Theorem. Inspired by the proof of Theorem 29 in [5], one approach might be to apply the O'Nan-Scott Theorem to more tightly characterise the stabilizer group of a gate.

Acknowledgments The authors thank Dieter van Melkebeek for looking at an early draft of this paper, and Joanna Fawcett for useful comments.

References

1. Anderson, M., Dawar, A.: On symmetric circuits and fixed-point logics. In: 31st International Symposium on Theoretical Aspects of Computer Science, volume 25, pages 41–52 (2014)

2. Anderson, M., Dawar, A., Holm, B.: Maximum matching and linear programming in fixed-point logic with counting (2013)
3. Blass, A., Gurevich, Y., Shelah, S.: Choiceless polynomial time. Annals of Pure and Applied Logic **100**, 141–187 (1999)
4. Cai, J.-Y., Fürer, M., Immerman, N.: An optimal lower bound on the number of variables for graph identification. Combinatorica **12**(4), 389–410 (1992)
5. Clote, P., Kranakis, E.: Boolean functions, invariance groups, and parallel complexity. SIAM J. Comput. **20**(3), 553–590 (1991)
6. Dawar, A.: A restricted second order logic for finite structures. Inf. Comput. **143**, 154–174 (1998)
7. Dawar, A., Gurevich, Y.: Fixed point logics. Bull. Symb. Log. **8**, 65–88 (2002)
8. Dawar, A., Lindell, S., Weinstein, S.: First order logic, fixed point logic and linear order. In: Kleine-Büning, H. (ed.) Computer Science Logic '95, volume 1092 of LNCS, pages 161–177. Springer-Verlag (1996)
9. Dawar, A., Richerby, D., Rossman, B.: Choiceless polynomial time, counting and the Cai–Fürer–Immerman graphs. Annals of Pure and Applied Logic **152**(1), 31–50 (2008)
10. Denenberg, L., Gurevich, Y., Shelah, S.: Definability by constant-depth polynomial-size circuits. Inf. Control. **70**(2), 216–240 (1986)
11. Ebbinghaus, H.D., Flum, J.: Finite Model Theory. Springer (1999)
12. Furst, M., Hopcroft, J., Luks, E.: Polynomial-time algorithms for permutation groups. In: Proceedings of the Twenty-First Annual ACM Symposium on Foundations of Computer Science, pages 36–41. IEEE (1980)
13. Grohe, M.: Fixed-point definability and polynomial time on graphs with excluded minors. J. ACM **59**(5), 27:1–27:64 (2012)
14. Immerman, N.: Relational queries computable in polynomial time. Inf. Control. **68**(1-3), 86–104 (1986)
15. Immerman, N.: Descriptive Complexity Theory. Springer (1999)
16. Otto, M.: Bounded Variable Logics and Counting: A Study in Finite Models, volume 9 of Lecture Notes in Logic. Springer-Verlag (1997)
17. Otto, M.: The logic of explicitly presentation-invariant circuits. In: Dalen, D., Bezem, M. (eds.) Computer Science Logic, volume 1258 of Lecture Notes in Computer Science, pages 369–384. Springer Berlin Heidelberg (1997)
18. Ryser, H.J.: Combinatorial Mathematics. Mathematical Association of America (1963)
19. Vardi, M.: The complexity of relational query languages (1982)
20. Vollmer, H.: Introduction to Circuit Complexity: A Uniform Approach. Springer-Verlag Berlin Heidelberg (1999)