# Symmetric Circuits for Rank Logic

ANUJ DAWAR*, University of Cambridge, United Kingdom
GREGORY WILSENACH*, University of Cambridge, United Kingdom

Fixed-point logic with rank (FPR) is an extension of fixed-point logic with counting (FPC) with operators for computing the rank of a matrix over a finite field. The expressive power of FPR properly extends that of FPC and is contained in P, but it is not known if that containment is proper. We give a circuit characterization for FPR in terms of families of symmetric circuits with rank gates, along the lines of that for FPC given by [Anderson and Dawar 2017]. This requires the development of a broad framework of circuits in which the individual gates compute functions that are not symmetric (i.e., invariant under all permutations of their inputs). This framework also necessitates the development of novel techniques to prove the equivalence of circuits and logic. Both the framework and the techniques are of greater generality than the main result.

CCS Concepts: • **Theory of computation → Finite Model Theory**; **Circuit complexity**; *Complexity theory and logic.*

Additional Key Words and Phrases: finite model theory, descriptive complexity, fixed-point logics, fixed-point logic with rank, circuit complexity, symmetric circuits, uniform families of circuits, circuit characterization, circuit frameworks

**ACM Reference Format:**
Anuj Dawar and Gregory Wilsenach. 2020. Symmetric Circuits for Rank Logic. 1, 1 (May 2020), 35 pages. https://doi.org/10.1145/1122445.1122456

## 1 INTRODUCTION

The study of extensions of fixed-point logics plays an important role in the field of descriptive complexity theory. In particular, fixed-point logic with counting (FPC) has become a reference logic in the search for a logic for polynomial-time (see [3]). In this context, Anderson and Dawar [1] provide an interesting characterization of the expressive power of FPC in terms of circuit complexity. They show that the properties expressible in this logic are exactly those that can be decided by polynomially-uniform families of circuits (with threshold gates) satisfying a natural *symmetry* condition. Not only does this illustrate the robustness of FPC as a complexity class within P by giving a distinct and natural characterization of it, it also demonstrates that the techniques for proving inexpressibility in the field of finite model theory can be understood as lower-bound methods against a natural circuit complexity class. An interesting example of the latter is the lower bounds for arithmetic circuits computing the permanent obtained by these methods [8]. This raises an obvious question (explicitly posed in the concluding section of [1]) of how to obtain circuit characterizations of logics more expressive than FPC, such as choiceless polynomial time (CPT) and fixed-point logic with rank (FPR). It is this last question that we address in this paper.

*Research funded in part by EPSRC grant EP/S03238X/1. An extended abstract of this paper appeared in [7].

Authors' addresses: Anuj Dawar, anuj.dawar@cl.cam.ac.uk, University of Cambridge, The Old Schools, Trinity Ln, Cambridge, CB2 1TN, United Kingdom; Gregory Wilsenach, gregory.wilsenach@cl.cam.ac.uk, University of Cambridge, The Old Schools, Trinity Ln, Cambridge, CB2 1TN, United Kingdom.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2020 Association for Computing Machinery.
Manuscript submitted to ACM

Fixed-point logic with rank extends the expressive power of FPC by means of operators that allow us to define the rank of a matrix over a finite field. Such operators are natural extensions of counting—counting the dimension of a definable vector space rather than just the size of a definable set. At the same time they make the logic rich enough to express many of the known examples that separate FPC from P. Rank logics were first introduced in [5]. The version of FPR we consider here is that defined by Grädel and Pakusa [11] where the prime characteristic is a parameter to the rank operator, and we do not have a distinct operator for each prime number. Formal definitions of these logics are given in Section 2. We give a circuit characterization, in terms of symmetric circuits, of FPR. One might think, at first sight, that this is a simple matter of extending the circuit model with gates for computing the rank of a matrix. It turns out, however, that the matter is not so simple as the symmetry requirement interacts in surprising ways with such rank gates. It requires a new framework for defining classes of such circuits, which yields remarkable new insights.

The word *symmetry* is used in more than one sense in the context of circuits (and also in this paper). We say that a Boolean function $f : \{0,1\}^n \to \{0,1\}$ is symmetric if the value of the function on a string $s$ is determined by the number of 1s in $s$. In other words, $f$ is invariant under *all* permutations of its input. In contrast, when we consider the input to a Boolean function to be the adjacency matrix of an $n$-vertex graph, for example, and $f : \{0,1\}^{\binom{n}{2}} \to \{0,1\}$ decides a graph property, then $f$ is invariant under all permutations of its input induced by permutations of the $n$ vertices of the graph. We call such a function *graph-invariant*. More generally, for a relational vocabulary $\tau$ and a standard encoding of $n$-element $\tau$-structures as strings over $\{0,1\}$, we can say that a function taking such strings as input is $\tau$-invariant if it is invariant under permutations induced by the $n$ elements. A circuit $C$ computing such an invariant function is said to be *symmetric* if every permutation of the $n$ elements extends to an automorphism of $C$. It is families of symmetric circuits in this sense that characterize FPC in [1]. The restriction to symmetric circuits arises naturally in the study of logics and has appeared previously under the names of generic circuits in the work of [9] and explicitly order-invariant circuits in the work of Otto [17]. In this paper, we use the word "symmetric", and context is used to distinguish the meaning of the word as applied to circuits from its meaning as applied to Boolean functions.

The main result of [1] says that the properties of $\tau$-structures definable in FPC are exactly those that can be decided by P-uniform families of symmetric circuits using AND, OR, NOT and majority gates. Note that each of these gates itself computes a Boolean function that is symmetric in the strong sense identified above. On the other hand, a rank threshold function, e.g. one that takes as input an $n \times n$ matrix and outputs 1 if the rank of the matrix is greater than a threshold $t$, is *not* symmetric. In our circuit characterization of FPR we necessarily have to consider such non-symmetric gates. Indeed, we can show that P-uniform families of symmetric circuits with gates labelled by *any* symmetric functions do not take us beyond the power of FPC. This is a further illustration of the robustness of FPC. In order to go beyond it, we need to introduce gates for Boolean functions that are not symmetric. We construct a systematic framework for including isomorphism-invariant functions that take $\tau$-structures as inputs, for $\tau$ an arbitrary multi-sorted relational vocabulary, in Section 3. We also explore what it means for such circuits to be symmetric.

The proof of the circuit characterization of FPC relies on the *support theorem* proved in [1]. This establishes that for any P-uniform family of circuits using AND, OR, NOT and majority gates there is a constant $k$ such that every gate has a support of size at most $k$. That is to say that we can associate with every gate $g$ in the circuit $C_n$ (the circuit in the family that works on $n$-element structures) a subset $X$ of $[n]$ of size at most $k$ such that any permutation of $[n]$ fixing $X$ pointwise extends to an automorphism of $C_n$ that fixes $g$. This theorem is crucial to the translation of the family of circuits into a formula of FPC, which is the difficult (and novel) direction of the equivalence. In attempting to do the same with circuits that now use rank-threshold gates we are faced with the difficulty that the proof of the support theorem in [1] relies in an essential way on the fact that the Boolean function computed at each gate is symmetric. We

are able to overcome this difficulty and prove a support theorem for circuits with rank gates but this requires substantial, novel technical machinery. This result is not only more general in that its applicability to circuits with rank gates, but is stronger even if we restrict our attention to circuits with gates that only compute symmetric functions.

Another crucial ingredient in the proof of Anderson and Dawar is that we can eliminate redundancy in the circuit $C_n$ by making it *rigid*. That is, we can ensure that the *only* automorphisms of $C_n$ are those that are induced by permutations of $[n]$. Here we face the difficulty that identifying the symmetries and eliminating redundancy in a circuit that involves gates computing $\tau$-invariant functions requires us to solve the isomorphism problem for $\tau$-structures. This is a hard problem (or, at least, one that we do not know how to solve efficiently) even when the $\tau$-structures are 0-1-matrices. We overcome this difficulty by placing a further restriction on circuits that we call *transparency*. Circuits satisfying this condition have the property that their lack of redundancy is transparent.

In the characterization of FPC, the translation from formulas into families of circuits is easy and, indeed, standard. In our case, we have to show that formulas of FPR translate into uniform families of circuits using rank-threshold gates that are both symmetric and transparent. This is somewhat more involved technically and presented in Section 6. Finally, with all these tools in place, the translation of such P-uniform families of circuits into formulas of FPR given in Section 7 completes the characterization. This still requires substantial new techniques. The translation of circuits to formulas in [1] relies on the fact that in order to evaluate a gate computing a symmetric Boolean function, it suffices to count the number of inputs that evaluate to true and establish a bijection between the elements of the orbit of a gate and tuple assignments to its support. When counting is no longer sufficient, this bijection has to preserve more structure and demonstrating this in the case of matrices requires new insight.

## 2 BACKGROUND

We write $\mathbb{N}$ to denote the positive integers and $\mathbb{N}_0$ to denote the non-negative integers. For $n \in \mathbb{N}$ we write $[n]$ to denote the set $\{1, \dots, n\}$. We identify a $k$-length tuple $\vec{a}$ from the set $X$ with a function of the form $\vec{a} : [k] \to X$. For sets $X$ and $Y$ we write $X^{\underline{Y}}$ to denote the set of injections from $X$ to $Y$. For $S \subseteq Y$ we let $f^{-1}[S] := \{x \in X : f(x) \in S\}$ denote the inverse image of $S$. We say a relation $R \subseteq \Pi_{j \in J} X_j$ is *full* if $R = \Pi_{j \in J} X_j$.

### 2.1 Group Theory

For a set $S$ we write $\mathbf{Sym}_S$ or $\mathbf{Sym}(S)$ to denote the symmetric group on $S$. For $n \in \mathbb{N}$ we write $\mathbf{Sym}_n$ to abbreviate $\mathbf{Sym}_{[n]}$. Let $G$ be a group acting on a set $X$ and let $S \subseteq X$. Let $\mathbf{Stab}_G(S) := \{\pi \in G : \forall s \in S,\ \pi(s) = s\}$ be the *pointwise stabiliser* of $S$. Let $\mathbf{Orb}_G(x) := \{y \in X : \exists \pi \in G,\ \pi(x) = y\}$ be the *orbit* of $x$. In both cases we omit subscripts when the group $G$ is obvious from context. For $n \in \mathbb{N}$ we write $\mathbf{Stab}_n(S)$ to abbreviate $\mathbf{Stab}_{\mathbf{Sym}_n}(S)$ and $\mathbf{Orb}_n(x)$ to abbreviate $\mathbf{Orb}_{\mathbf{Sym}_n}(x)$.

Let $G$ be a group acting on a set $X$. We denote this as a left action, i.e. $\sigma x$ for $\sigma \in G$, $x \in X$. The action extends in a natural way to powers of $X$. So, for $(x, y) \in X \times X$, $\sigma(x, y) = (\sigma x, \sigma y)$. It also extends to the powerset of $X$ and functions on $X$ as follows. The action of $G$ on $\mathcal{P}(X)$ is defined for $\sigma \in G$ and $S \in \mathcal{P}(X)$ by $\sigma S = \{\sigma x : x \in S\}$. For any set $Y$, the action of $G$ on $Y^X$ is defined for $\sigma \in G$ and $f \in Y^X$ by $(\sigma f)(x) = f(\sigma x)$ for all $x \in X$. We refer to all of these as the *natural action* of $G$ on the relevant set.

### 2.2 Linear Algebra

Let $A$, $B$, and $X$ be non-empty sets such that $A$ and $B$ are finite. An $X$-valued $A \times B$ *matrix* is a function $M : A \times B \to X$. If $X \subseteq \mathbb{N}_0$ we say $M$ is number valued.

We use $\mathbb{F}$ to denote a field. For any prime $p$ we write $\mathbb{F}_p$ for the finite field of order $p$. For an $\mathbb{F}_p$-valued matrix $M$ we write $\mathbf{rk}_p(M)$ to denote the rank. For a number valued matrix $M$ and a prime $p$ we write $(M \bmod p)$ to denote the matrix formed by taking the residue of each element modulo $p$. We write $\mathbf{rk}_p(M)$ to abbreviate $\mathbf{rk}_p(M \bmod p)$.

### 2.3 Logic

A *vocabulary* is a finite set of relation symbols $(R_1, \ldots, R_k)$, each of which has a fixed *arity*. For each relation symbol $R$ we write $r_R$ to denote its arity. We omit the subscript when it is obvious from context.

A *many-sorted vocabulary* is a tuple of the form $(\boldsymbol{R}, \boldsymbol{S}, \zeta)$, where $\boldsymbol{R}$ is a relational vocabulary, $\boldsymbol{S}$ is a finite sequence of *sort* symbols, and $\zeta$ is a function that assigns to each $R \in \boldsymbol{R}$ a tuple $\zeta(R) := (s_1, \ldots, s_r)$ of sort symbols. We call $\zeta(R)$ the *type* of $R$. A *$\tau$-structure* $\mathcal{A}$ is a tuple $(A, (R^{\mathcal{A}})_{R \in \boldsymbol{R}})$ where $A = \uplus_{s \in S} A_s$ a disjoint union of non-empty sets, called the *universe* of $\mathcal{A}$, and for all $R \in \boldsymbol{R}$, $R^{\mathcal{A}} \subseteq A_{s_1} \times \ldots \times A_{s_r}$, where $(s_1, \ldots, s_{r_i}) = \zeta(R_i)$. The *size* of $\mathcal{A}$, denoted by $|\mathcal{A}|$, is the cardinality of $A$. All structures in this paper are finite. Let $\mathrm{fin}[\tau]$ denote the set of all finite $\tau$-structures and for $n \in \mathbb{N}$ let $\mathrm{fin}[\tau, n]$ denote the set of all $\tau$-structures of size $n$. We say that a $\tau$-structure is *complete* if every relation it contains is full.

Let $\mathrm{FO}[\tau]$ denote *first-order logic* over the vocabulary $\tau$. A formula in $\mathrm{FO}[\tau]$ is formed from atomic formulas, each formed using variables from some countable sequence of (first-order) variable symbols $x, y, \ldots$, the relation symbols in $\tau$, and the equality symbol $=$, and then closing the set of atomic formulas under the Boolean connectives, and universal and existential quantification (i.e. $\wedge$, $\vee$, $\neg$, $\forall$, and $\exists$). Let FP denote *fixed-point logic*, the extension of FO with inflationary fixed-point operators. We assume standard syntax and semantics for FO and FP, and direct the reader to [12] or [15] for more detail.

Let $L[\tau]$ be a logic over a vocabulary $\tau$ and $\phi \in L[\tau]$. For a sequence of variables $\vec{x}$ we write $\phi(\vec{x})$ to denote that the free variables in $\phi$ are among $\vec{x}$. For $\mathcal{A} \in \mathrm{fin}[\tau]$ and $\vec{a} \in A^{|\vec{x}|}$ we write $\mathcal{A} \models_L \phi[\vec{a}]$ to denote that $\phi$ holds when interpreted in $\mathcal{A}$ with respect to the logic $L$ when each variable $x_i$ is assigned to $a_i$. We omit the subscript $L$ when it is obvious from context.

#### 2.3.1 *Fixed-Point with Counting.*

Let $\mathrm{FPC}[\tau]$ denote *fixed-point logic with counting* over the vocabulary $\tau$. FPC extends FP with a *counting operator* that allows us to define the cardinality of a definable set, as well as a number sort for doing calculations. Each variable in FPC is either a *number* or *element* variable. Element variables correspond to the variables in FO or FP and range over the universe of the structure. Number variables instead range over the number domain $\mathbb{N}_0$.

The number terms and formulas of $\mathrm{FPC}[\tau]$ are defined by mutual recursion. A *number term* is either a number variable, of the form $t_1 + t_2$ or $t_1 \cdot t_2$ for number terms $t_1, t_2$, or of the form $\#x\phi$, where $\phi \in \mathrm{FPC}[\tau]$ and $x$ is an element variable. The number term $\#x\phi$ denotes the number of distinct elements $a$ for which $\phi[a]$ holds. Let $t(\vec{z})$ be a number term. Let $\mathcal{A} \in \mathrm{fin}[\tau]$ and $\vec{a}$ be a $|\vec{z}|$-sequence such that if $z_i$ is a element variable then $a_i \in A$ and if $z_i$ is a number variable then $a_i \in \mathbb{N}_0$. Let $t^{\mathcal{A}}[\vec{a}]$ denote the number defined by evaluating $t$ in $\mathcal{A}$ under the assignment that maps each $z_i$ to $a_i$.

The atomic formulas of $\mathrm{FPC}[\tau]$ include all atomic formulas in $\mathrm{FP}[\tau]$ as well as formulas of the form $t_1 = t_2$ and $t_1 \leq t_2$ for number terms $t_1$ and $t_2$. The formulas of $\mathrm{FPC}[\tau]$ are formed by closing the set of atomic formulas under the usual Boolean connectives, the first-order quantifiers, and the fixed-point operator. To avoid undecidability, quantification over the number sort must be bounded. If $\nu$ is a number variable, it must be quantified in the form $\exists \nu \leq t\ \phi$ or $\forall \nu \leq t\ \phi$ for $\phi \in \mathrm{FPC}[\tau]$, where $t$ is a number term. We omit these bounds to abbreviate that the number variable is quantified over $\{0, \ldots, |\mathcal{A}|\}$ when evaluated over some $\mathcal{A} \in \mathrm{fin}[\tau]$. In other words, we write $\forall \nu \phi$ to abbreviate $\forall \nu \leq t_m \phi$, where

$t_m$ is a number term that denotes the size of the structure over which it is evaluated. The fixed-point operator may bind sequences of variables consisting of both element and number variables. We similarly require an explicit bound for each number variable quantified.

We assume a standard semantics of FPC[$\tau$] and direct the reader to [12] for more detail.

Let FP$^{\mathbb{N}}$[$\tau$] denote *fixed-point logic with a number sort* over the vocabulary $\tau$. The formulas of FP$^{\mathbb{N}}$[$\tau$] include all those in FPC[$\tau$] that do not include an application of the counting operator.

*2.3.2 Fixed-Point with Rank.* Let FPR[$\tau$] denote *fixed-point logic with rank* over the vocabulary $\tau$. FPR extends FP$^{\mathbb{N}}$ with an operator that denotes the rank of a definable matrix over a finite field. To define FPR we extend the definition of FP$^{\mathbb{N}}$ to include number terms of the form $[\mathbf{rk}(\vec{x}\vec{\mu} \leq \vec{t}, \vec{y}\vec{\nu} \leq \vec{s}, \pi).\eta]$, where $\eta$ is a formula or number term and $\vec{t}$ and $\vec{s}$ are tuples of number terms bounding the sequences of number variables $\vec{\mu}$ and $\vec{\nu}$, respectively.

Informally, this number term evaluates to the rank of the matrix $M_\eta$ over a field with order determined by $\pi$, where $M_\eta$ is a matrix with entries determined by $\eta$ and with rows indexed by the assignments to $\vec{x}\vec{\mu}$ and columns indexed by the assignments to $\vec{y}\vec{\nu}$. More formally, we define the evaluation of this number term for $\mathcal{A} \in \text{fin}[\tau]$ as follows. Let $\mathbb{N}_0^{\leq \vec{t}} := \{\vec{a} \in \mathbb{N}_0^{|\vec{t}|} : \forall i \in [|\vec{t}|] \;\; a_i \leq t_i^{\mathcal{A}}\}$ and let $\mathbb{N}_0^{\leq \vec{s}}$ be defined similarly. Let $R := A^{|\vec{x}|} \times \mathbb{N}_0^{\leq \vec{t}}$ and $C := A^{|\vec{y}|} \times \mathbb{N}_0^{\leq \vec{s}}$. Let $M_\eta : R \times C \to \mathbb{N}_0$ be the matrix defined such that $M_\eta(\vec{a}\vec{m}, \vec{b}\vec{n}) = \eta^{\mathcal{A}}[\vec{a}\vec{m}, \vec{b}\vec{n}]$ for all $(\vec{a}\vec{m}, \vec{b}\vec{n}) \in R \times C$ (so if $\eta$ is a formula then $M_\eta$ is a 0-1 matrix). Let $p = \pi^{\mathcal{A}}$. The number term $[\mathbf{rk}(\vec{x}\vec{\mu} \leq \vec{t}, \vec{y}\vec{\nu} \leq \vec{s}, \pi).\eta]$ evaluates to 0 if $p$ is not prime and otherwise evaluates to $\mathbf{rk}_p(M_\eta)$.

For any formula $\phi$ the number term $[\mathbf{rk}(x, y, 2)(x = y \wedge \phi(x))]$ denotes the number of $a \in A$ such that $\mathcal{A} \models \phi[a]$. It follows that FPR is at least as expressive as FPC.

We should note that there are two different rank logics considered in the literature. The first, introduced in [5], is defined in a similar manner as above but includes a separate rank operator for each prime, rather then a single operator that takes in a prime as a parameter. This logic was shown to be strictly contained in P [11]. The rank logic we have defined here is strictly more expressive and is the focus of current research. For more details on the syntax and semantics of FPR see [11].

## 3 SYMMETRIC CIRCUITS

We aim to derive a characterisation for FPR analogous to the characterisation of FPC in terms of symmetric majority circuits given in [1]. A natural starting point would be to consider circuits defined over a basis more expressive than the majority basis. However, in Section 3.3 we show that any basis of symmetric functions leads to a circuit model that is at most as expressive as FPC. Since FPR is strictly more expressive than FPC we need to consider circuits with gates labelled by non-symmetric functions.

This poses a potential problem as the usual notion of a circuit as a directed acyclic graph implicitly requires that the gates be labelled only by symmetric functions. This follows from the fact that a directed acyclic graph imposes no structure on the inputs of any gate, and so each gate must compute a function invariant under any ordering of its inputs, i.e. a symmetric function. Importantly, the standard basis, majority basis, and most other bases considered in the literature satisfy this assumption.

In this section we introduce the notion of a *structured function*, a Boolean function whose input strings naturally encode relational structures. We generalise the notion of a symmetric function for this framework, and so introduce *isomorphism-invariant structured functions*. We then define a more general circuit model so as to allow for gates to be labelled by isomorphism-invariant structured functions. We generalise the notions of a circuit automorphism and a

symmetric circuit for this new model. We also address many problems that emerge in the analysis of these more general circuits, and discuss a connection with the graph isomorphism problem.

### 3.1 Structured Functions

We now define structured functions formally.

*Definition 3.1.* Let $\tau := (\boldsymbol{R}, \boldsymbol{S}, \zeta)$ be a vocabulary, $Y$ and $Z$ be sets, and $X = \uplus_{s \in \boldsymbol{S}} X_s$ be a disjoint union of finite non-empty sets. Let $K$ be the complete $\tau$-structure with universe $X$. Let $\tau[X] := \uplus_{R \in \boldsymbol{R}} R^K$. We call a function $F : Y^{\tau[X]} \to Z$ a *structured function.*

Let $F : Y^{\tau[X]} \to Z$ be a structured function. We call $\tau$ the *vocabulary* of $F$, $X$ the *universe* of $F$, $K$ the *structure* associated with $F$, and $\tau[X]$ the *index* of $F$. We denote the index of $F$ by $\mathrm{ind}(F)$, the universe of $F$ by $\mathrm{unv}(F)$, and the structure associated with $F$ by $\mathrm{str}(F)$. We abuse notation and write $(x, R) \in \tau[X]$ to denote that $x \in \tau[X]$ and $x \in R^K$. In other words, $(x, R) \in \tau[X]$ denotes that $x$ is a tuple from the relation $R$ in the complete $\tau$-structure with universe $X$. Throughout this paper we assume that each sort symbol appears in the type of some relation, i.e. for each sort $s \in \tau$ there exists a relation symbol $R \in \tau$ such that $s$ appears in $\zeta(R)$.

We identify elements in $\{0,1\}^{\tau[X]}$ with $\tau$-structures over $X$. More precisely, we identify each $f \in \{0,1\}^{\tau[X]}$ with the $\tau$-structure $\mathcal{A}_f$ with universe $X$ such that for each relation symbol $R \in \tau$, $R^{\mathcal{A}_f} = \{x : (x, R) \in \tau[X], f(x) = 1\}$. We can similarly identify elements of $Y^{\tau[X]}$ for some set $Y$ with the $\tau$-structure with universe $X$ and elements of each relation labelled by an element of $Y$.

The action of $\mathbf{Sym}(\tau[X])$ on $\tau[X]$ extends naturally to $Y^{\tau[X]}$ and further to structured functions of the form $Y^{\tau[X]} \to Z$. For a structured function $F : Y^{\tau[X]} \to Z$ the stabiliser group $\mathbf{Stab}(F)$ is the set of all permutations $\sigma \in \mathbf{Sym}(\tau[X])$ such that for all $x \in Y^{\tau[X]}$, $F(\sigma x) = F(x)$. Let $H := \bigoplus_{s \in \boldsymbol{S}} \mathbf{Sym}_{X_s}$ and consider $H$ as a subgroup of $\mathbf{Sym}(\tau[X])$. Then $H \leq \mathbf{Stab}(F)$ if, and only if, $F$ is invariant under isomorphism, where the inputs are interpreted as (edge) labelled structures. In particular, if $Y = Z = \{0,1\}$ then $H \leq \mathbf{Stab}(F)$ if, and only if, $F$ decides an isomorphism-closed class of structures. This motivates the following definition.

*Definition 3.2.* Let $\tau := (\boldsymbol{R}, \boldsymbol{S}, \zeta)$ be a vocabulary, $Y$ and $Z$ be sets, and $X = \uplus_{s \in \boldsymbol{S}} X_s$ be a disjoint union of finite non-empty sets. Let $F : Y^{\tau[X]} \to Z$ be a structured function. We call $H := \bigoplus_{s \in \boldsymbol{S}} \mathbf{Sym}_{X_s}$ the *natural invariance group* of $F$. We say that $F$ is *isomorphism invariant* if $H \leq \mathbf{Stab}(F)$.

We write $\mathrm{NI}(F)$ to denote the natural invariance group of $F$. We identify the usual symmetric Boolean functions with isomorphism-invariant structured functions defined over a vocabulary consisting of a single unary relation. We call these functions *fully symmetric* to distinguish them from isomorphism-invariant structured functions, which are symmetric in a weaker sense.

We now introduce some notation and a few useful conventions. Throughout this paper we can often assume, without loss of generality, that a structured function has as its universe a direct sum of initial segments of the natural numbers. We introduce some notation to simplify the definition of such a function. Let $\tau = (\boldsymbol{R}, \boldsymbol{S}, \zeta)$ be a many-sorted relational vocabulary. Let $e : \boldsymbol{S} \to \mathbb{N}$. We write $\tau[e]$ to denote $\tau[X]$ where $X = \uplus_{s \in \boldsymbol{S}} X_s$, for each $s \in \boldsymbol{S}$, $X_s = [e(s)]$. If $\boldsymbol{S} = \{s_1, \ldots, s_m\}$ then for $p_1, \ldots, p_m \in \mathbb{N}$ we write $\tau[p_1, \ldots, p_m]$ to denote $\tau[e]$ where $e : \boldsymbol{S} \to \mathbb{N}$ is defined by $e(s_i) = p_i$ for each $i \in [m]$.

We now generalise the notion of a basis to this framework.

*Definition 3.3.* A *Boolean basis* (or just a *basis*) is a set of isomorphism-invariant structured functions.

We now define the *rank threshold functions*. Let $\tau_{\text{mat}}$ be a three sorted vocabulary with a single three-sorted ternary relation symbol $R$. We first define a translation between number-valued matrices and $\tau_{\text{mat}}$-structures. Let $S$ be a $\tau_{\text{mat}}$-structure with universe $X = \uplus_{s\in[3]} X_s$. Let $M_S : X_1 \times X_2 \to \mathbb{N}$ be defined for $(x_1, x_2) \in X_1 \times X_2$ by

$$M_S(x_1, x_2) = |\{x_3 \in X_3 : R^S(x_1, x_2, x_3)\}|.$$

We call $S$ a *three sorted matrix* and $M_S$ the *matrix associated with* $S$. For a prime $p$ we write $\mathbf{rk}_p(S)$ to abbreviate $\mathbf{rk}_p(M_S)$.

Let $t, p \in \mathbb{N}_0$ and $X = \uplus_{s\in[3]} X_s$ be a disjoint union of finite non-empty sets. Let $\text{RANK}_p^t[X] : \{0,1\}^{\tau_{\text{mat}}[X]} \to \{0,1\}$ be defined for each $S \in \{0,1\}^{\tau_{\text{mat}}[X]}$ by

$$\text{RANK}_p^t[X](S) = \begin{cases} 1 & \text{if } p \text{ prime and } \mathbf{rk}_p(S) \geq t \\ 0 & \text{otherwise} \end{cases}.$$

These functions are isomorphism-invariant. We write $\text{RANK}_p^t[a, b, c]$ for $a, b, c \in \mathbb{N}$ to denote $\text{RANK}_p^t[X]$ where $X$ is the disjoint union of $[a]$, $[b]$, and $[c]$. We omit $X$ and just write $\text{RANK}_p^t$ when $X$ is clear from context.

For $n \in \mathbb{N}$ we similarly define $\text{AND}[n] : \{0,1\}^n \to \{0,1\}$ to be the usual AND function on $n$ bits, $\text{OR}[n]$ to be the OR function on $n$ bits, NOT to be negation on 1 bit, and $\text{MAJ}[n]$ the majority function on $n$ bits. All of these functions are fully symmetric.

The *standard basis*, denoted by $\mathbb{B}_{\text{std}}$, contains NOT and all $\text{AND}[n]$ and $\text{OR}[n]$ for $n \in \mathbb{N}$. The *majority basis*, denoted by $\mathbb{B}_{\mathbf{maj}}$, extends the standard basis with the majority functions, i.e. $\text{MAJ}[n]$ for all $n \in \mathbb{N}$. The *rank basis*, denoted by $\mathbb{B}_{\mathbf{rk}}$, is the extension of $\mathbb{B}_{\text{std}}$ with the rank threshold functions, i.e. $\text{RANK}_p^t[a, b, c]$ for all $a, b, c, t, p \in \mathbb{N}$

### 3.2 Symmetric Circuits

We now generalise the circuit model of Anderson and Dawar [1] so as to allow for circuits to be defined over bases of isomorphism-invariant functions, rather than just fully symmetric functions. In this model each gate $g$ is not only associated with an element of the basis, as in the conventional case, but also with a labelling function. This labelling function maps the input gates of $g$ to an appropriate set of labels (i.e. the index of the structured function associated with $g$). In concord with this generalisation, we also update the circuit-related notions discussed by Anderson and Dawar [1], e.g. circuit automorphisms, symmetry, etc. Moreover, we briefly discuss some of the important complications introduced by our generalisation, and introduce some of the important tools we use later to address these complications.

*Definition 3.4 (Circuits on Structures).* Let $\mathbb{B}$ be a basis and $\rho$ be a relational vocabulary, we define a $(\mathbb{B}, \rho)$-*circuit* $C$ of order $n$ computing a $q$-ary query $Q$ as a structure $\langle G, \Omega, \Sigma, \Lambda, L\rangle$.

- $G$ is called the set of gates of $C$.
- $\Omega$ is an injective function from $[n]^q$ to $G$. The gates in the image of $\Omega$ are called the output gates. When $q = 0$, $\Omega$ is a constant pointing to a single output gate.
- $\Sigma$ is a function from $G$ to $\mathbb{B} \uplus \rho \uplus \{0,1\}$ such that $|\Sigma^{-1}(0)| \leq 1$ and $|\Sigma^{-1}(1)| \leq 1$. Those gates mapped to $\rho \uplus \{0,1\}$ are called input gates, with those mapped to $\rho$ called relational gates and those mapped to $\{0,1\}$ called constant gates. Those gates mapped to $\mathbb{B}$ are called internal gates.
- $\Lambda$ is a sequence of injective functions $(\Lambda_R)_{R\in\rho}$ such that $\Lambda_R$ maps each relational gate $g$ with $\Sigma(g) = R$ to the tuple $\Lambda_R(g) \in [n]^{r_R}$. When no ambiguity arises we write $\Lambda(g)$ for $\Lambda_R(g)$.
- $L$ associates with each internal gate $g$ a function $L(g) : \text{ind}(\Sigma(g)) \to G$ such that if we define a relation $W \subseteq G^2$ by $W(h_1, h_2)$ iff $h_2$ is an internal gate and $h_1$ is in the image of $L(h_2)$, then $(G, W)$ is a directed acyclic graph.

The definition requires some explanation. Each gate in $G$ computes a function of its inputs and the relation $W$ on $G$ is the set of "wires". That is, $W(h, g)$ indicates that the value computed at $h$ is an input to $g$. However, since the functions are structured, we need more information on the set of inputs to $g$ and this is provided by the labelling $L$. For any $g$, $\Sigma(g)$ tells us what the function computed at $g$ is, and thus the index of $\Sigma(g)$ tells us the structure on the inputs and $L(g)$ maps this to the set of gates that form the inputs to $g$.

Let $C := \langle G, \Omega, \Sigma, \Lambda, L\rangle$ be a $(\mathbb{B}, \rho)$-circuit of order $n$. We define the *size* of $C$, denoted $|C|$ to be the number of elements in $G$. If $(C_n)_{n\in\mathbb{N}}$ is a family of circuits we assume that each $C_n$ is a circuit of order $n$.

For each $g \in G$ we let $W(\cdot, g) := \{h \in G : W(h, g)\}$ and $W(g\cdot) := \{h \in G : W(g, h)\}$. We call the elements of $W(\cdot, g)$ the *children* of $g$ and the elements of $W(g, \cdot)$ the *parents* of $g$. We also abbreviate $W(\cdot, g)$ by $H_g$. We write $W_T$ for the transitive closure of $W$. The *depth* of $g$ is 0 if $g$ is an input gate or otherwise the length of the longest path from an input gate to $g$.

For a gate $g \in G$ with $\Sigma(g) \in \mathbb{B}$, we let the *index* of $g$, denoted by $\mathrm{ind}(g)$, be the index set of $\Sigma(g)$. We abuse notation and let the *vocabulary* and *universe* of $g$ denote the vocabulary and universe of $\Sigma(g)$, respectively. We also use $\mathrm{voc}(g)$ and $\mathrm{unv}(g)$ to denote the vocabulary and universe of $g$, respectively, and write $\mathrm{NI}(g)$ to abbreviate $\mathrm{NI}(\Sigma(g))$. We write $\mathrm{str}(g)$ to denote the structure associated with $\Sigma(g)$. We say that a gate $g$ is *fully symmetric* if $\Sigma(g)$ is fully symmetric. We say $C$ is a *circuit with fully symmetric gates* if every gate in $C$ is fully symmetric.

Let $\rho$ be a relational vocabulary, $\mathcal{A}$ be a $\rho$-structure with universe $A$ of size $n$, and $\gamma \in [n]^{\underline{A}}$. Let $\gamma\mathcal{A}$ be the structure with universe $[n]$ formed by mapping the elements of $A$ in accordance with $\gamma$. The evaluation of a $(\mathbb{B}, \rho)$-circuit $C$ of order $n$ computing a $q$-ary query $Q$ proceeds by recursively evaluating the gates in the circuit. The evaluation of the gate $g$ for the bijection $\gamma$ and input structure $\mathcal{A}$ is denoted by $C[\gamma\mathcal{A}](g)$, and is given as follows:

(1) If $g$ is a constant gate then it evaluates to the value given by $\Sigma(g)$;
(2) if $g$ is a relational gate then $g$ evaluates to 1 iff $\gamma\mathcal{A} \models \Sigma(g)(\Lambda(g))$; and
(3) if $g$ is an internal gate let $L^{\gamma\mathcal{A}}(g) : \mathrm{ind}(g) \to \{0, 1\}$ be defined by $L^{\gamma\mathcal{A}}(g)(x) = C[\gamma\mathcal{A}](L(g)(x))$, for all $x \in \mathrm{ind}(g)$. Then $g$ evaluates to 1 iff $\Sigma(g)(L^{\gamma\mathcal{A}}(g)) = 1$.

We say that $C$ defines the $q$-ary query $Q \subseteq A^q$ under $\gamma$ where $\vec{a} \in Q$ if, and only if, $C[\gamma\mathcal{A}](\Omega(\gamma\vec{a})) = 1$. We write $C[\gamma\mathcal{A}]$ to denote the query $Q$. In general the evaluation $C$ depends on the particular encoding of $\mathcal{A}$ as a structure with universe $[n]$, i.e. on the chosen bijection $\gamma$. We consider circuits whose outputs do not depend on this choice of encoding. Anderson and Dawar [1] call such a circuit *invariant*. We have reproduced their definition below.

*Definition 3.5 (Invariant Circuit).* Let $C$ be a $(\mathbb{B}, \rho)$-circuit of order $n$, computing some $q$-ary query. We say $C$ is *invariant* if for every $\rho$-structure $\mathcal{A}$ of size $n$, $\vec{a} \in A^q$, and $\gamma_1, \gamma_2 \in [n]^{\underline{A}}$ we have that $C[\gamma_1\mathcal{A}](\Omega(\gamma_1\vec{a})) = C[\gamma_2\mathcal{A}](\Omega(\gamma_2\vec{a}))$.

If a family of $(\mathcal{B}, \rho)$-circuits $\mathcal{C}$ is invariant it follows that the query computed is a $q$-ary query on $\rho$-structures. When $q = 0$ then $\mathcal{C}$ computes a property of $\rho$-structures, i.e. a decision problem. Each circuit $C_n \in \mathcal{C}$ then defines a structured function with vocabulary $\rho$ and universe $[n]$.

Lemma 3.6. *Let $C$ be a $(\mathbb{B}, \rho)$-circuit of order $n$ computing a 0-ary query. The structured function computed by $C$ is isomorphism-invariant if, and only if, $C$ is an invariant circuit.*

Proof. Let $F_C$ be the structured function defined by $C$ that maps $\rho$-structures over $[n]$ to $\{0, 1\}$. Suppose $F_C$ is isomorphism-invariant. Let $\mathcal{A}$ be a $\rho$-structure of size $n$ and let $\gamma_1, \gamma_2 \in [n]^{\underline{A}}$. Then $C[\gamma_1\mathcal{A}](\Omega) = F_C(\gamma_1\mathcal{A}) = F_C(\gamma_2\mathcal{A}) = C[\gamma_2\mathcal{A}](\Omega)$. The second equality follows from the fact that $\gamma_1\mathcal{A}$ and $\gamma_2\mathcal{A}$ are isomorphic. So $C$ is invariant.

Suppose $C$ is invariant. Let $\mathcal{B}$ be a $\rho$-structure over $[n]$. Then $[n]^{\underline{B}}$ is the set of permutations of $[n]$. Let $\sigma, e \in \mathbf{Sym}_n$, where $e$ is the identity. Then $F_C(\mathcal{B}) = C[e\mathcal{B}](\Omega) = C[\sigma\mathcal{B}](\Omega) = F_C(\sigma\mathcal{B}) = (\sigma F_C)(\mathcal{B})$. So $F_C$ is isomorphism-invariant. □

We now define an automorphism of a circuit, generalising the definition introduced by Anderson and Dawar. The definition is similar, but adds the requirement that if a gate $g$ is mapped to $g'$, then children of $g$ must be mapped to the children of $g'$ via some appropriate isomorphism of the structure associated with $g$.

*Definition 3.7 (Automorphism).* Let $C = \langle G, \Omega, \Sigma, \Lambda, L\rangle$ be a $(\mathbb{B}, \tau)$-circuit of order $n$ computing a $q$-ary query. Let $\sigma \in \mathbf{Sym}_n$ and $\pi : G \to G$ be a bijection such that

- for all output tuples $x \in [n]^q$, $\pi\Omega(x) = \Omega(\sigma x)$,
- for all gates $g \in G$, $\Sigma(g) = \Sigma(\pi g)$,
- for each relational gate $g \in G$, $\sigma\Lambda(g) = \Lambda(\pi g)$, and
- For each pair of gates $g, h \in G$ we have $W(h, g)$ if, and only if, $W(\pi h, \pi g)$ and for each internal gate $g$ we have that $L(\pi g)$ and $\pi L(g)$ are isomorphic (as labelled structures – see Section 3.1).

We call $\pi$ an *automorphism* of $C$, and we say that $\sigma$ *extends to an automorphism* $\pi$. The group of automorphisms of $C$ is denoted by $\mathbf{Aut}(C)$.

We can equally define an isomorphism between a pair of circuits $C = \langle G, \Omega, \Sigma, \Lambda, L\rangle$ and $C' = \langle G', \Omega', \Sigma', \Lambda', L'\rangle$ as a bijection $\pi : G \to G'$ satisfying conditions as above. We do not usually need to consider distinct, isomorphic circuits and for this reason we only formally define automorphisms.

We are particularly interested in circuits that have the property that *every* permutation in $\mathbf{Sym}_n$ extends to an automorphism of the circuit.

*Definition 3.8 (Symmetry).* A circuit $C$ of order $n$ is called *symmetric* if every $\sigma \in \mathbf{Sym}_n$ extends to an automorphism on $C$.

It follows that for any symmetric circuit $C$ of order $n$ there is a homomorphism $h$ that maps $\mathbf{Sym}_n$ to $\mathbf{Aut}(C)$ such that if $\sigma \in \mathbf{Sym}_n$ then $h(\sigma)$ is an automorphism extending $\sigma$. If $C$ does not contain a relational gate then it computes a constant function. These circuits form a trivial and uninteresting class and so, unless stated otherwise, we assume each circuit in this paper contains at least one relational gate. With this assumption in place it follows that some element of $[n]$ appears in a tuple labelling some relational gate, and so by symmetry every element of $[n]$ appears in a tuple labelling a relational gate. Then no two distinct elements of $\mathbf{Sym}_n$ agree on all input gates, and so the homomorphism $h$ is injective.

Suppose $h$ is not also surjective. Since every automorphism extends some permutation, there exists $\sigma \in \mathbf{Sym}_n$ and $\pi, \pi' \in \mathbf{Aut}(C)$ both of which extend $\sigma$ but disagree on some gate $g$. In the forthcoming Lemmas 3.11 and 3.12 we show that, although $\pi(g)$ and $\pi'(g)$ are not equal, they are in a precise sense "essentially the same". In this way the existence of a non-surjective $h$ witnesses the introduction of "artefacts" into our analysis. We therefore restrict ourselves to circuits with *unique extensions*, a property which ensures that $h$ is an isomorphism.

*Definition 3.9.* We say that a circuit $C$ over order $n$ has *unique extensions* if for every $\sigma \in \mathbf{Sym}_n$ there is at most one $\pi \in \mathbf{Aut}(C)$ such that $\pi$ extends $\sigma$.

Most of the important technical tools needed in this paper are only applicable to circuits with unique extensions. A similar problem arises in the work of Anderson and Dawar [1]. They address this by defining a normal form for circuits,

which they call *rigid circuits*, and show that these circuits have unique extensions. They then show that symmetric circuits can be transformed into rigid symmetric circuits in polynomial time, allowing them to restrict their attention to circuits with unique extensions (and numerous other desirable properties) without loss of generality.

We will define a normal form analogous to rigidity and show that there is a polynomial time algorithm that transforms a circuit into an equivalent circuit of this form. It is at this point that we arrive at the first complication introduced by our generalisation. The polynomial-time translation in [1] makes indispensable use of the polynomial-time decidability of many important circuit properties for circuits defined over bases of fully symmetric functions. However, for the more general circuits discussed here, it is not known if even the most basic circuit properties are polynomial-time decidable. This is essentially due to the requirement built in to Definition 3.7 that an automorphism that takes $g$ to $g'$ must be an isomorphism between $L(g)$ and $L(g')$. This makes checking the condition as hard as isomorphism checking. As such, constructing an argument analogous to [1], as well as establishing the numerous other crucial results whose proofs rely on the polynomial-time decidability of various circuit properties, would be beyond the scope of this paper.

In order to proceed we explicitly restrict our attention to a particular class of circuits characterised by a restriction on the children of gates labelled by functions that are not fully symmetric. We say such circuits are *transparent*. We show in Section 5 that all of the circuit properties of interest are polynomial-time decidable for transparent circuits, and we use these results to define a polynomial-time transformation from transparent circuits to equivalent circuits in our normal form. This normal form not only allows us to ensure the polynomial-time decidability of numerous circuit properties but also ensures the circuits have unique extensions, a necessary requirement to apply the theory of supports we develop in Section 4. Importantly, while the restriction to transparent circuits makes it easier to translate families of circuits into formulas, the usual translation from formulas to circuits does not produce a family of transparent circuits in general. We discuss this difficulty and define a novel translation from formulas to symmetric circuits that ensures transparency in Section 6.

Before we can formally define *transparency* we need to define the *syntactic equivalence* relation on the gates of a circuit. The intuition is that two gates $g$ and $g'$ in a circuit are syntactically equivalent if the circuits underneath them are "hereditarily equivalent", i.e. if the two circuits induced by the restrictions to $W_T(\cdot, g)$ and $W_T(\cdot, g')$ are in some sense just copies of one another.

*Definition 3.10.* Let $C := \langle G, \Omega, \Sigma, \Lambda, L\rangle$ be a $(\mathbb{B}, \rho)$-circuit of order $n$. Let $\equiv'$ be an equivalence relation on $G$ defined such that for $g, h \in G$, $g \equiv' h$ if, and only if, $\Sigma(g) = \Sigma(h)$ and either both $g$ and $h$ are not output gates or $g$ and $h$ are both output gates and $g = h$. We now define a refinement $\equiv$ of $\equiv'$ by induction on depth. Let $g, h \in G$ be such that $g \equiv' h$. If $g$ and $h$ have depth 0 then $g \equiv h$ if, and only if, both $g$ and $h$ are constant gates or both $g$ and $h$ are relational gates and $\Lambda(g) = \Lambda(h)$. Let $r > 0$ and suppose we have defined $\equiv$ for the case that both $g$ and $h$ have depth less than $r$. If either $h$ or $g$ have depth $r$ then $g \equiv h$ if, and only if, there exists $\lambda \in \mathrm{NI}(g)$ such that for all $x \in \mathrm{ind}(g)$, $L(g)(\lambda(x)) \equiv L(h)(x)$. We call the relation $\equiv$ *syntactic equivalence*.

For a circuit $C$ of order $n$ and a gate $g$ we think of $g$ as computing the function that maps an input structure $\mathcal{A}$ and a bijection $\gamma$ from the universe of $\mathcal{A}$ to $[n]$ to the evaluation $C[\gamma\mathcal{A}](g)$. While we would like to be able to identify gates that compute the same function in this sense, it is not hard to show that deciding this equivalence relation for a given circuit is NP-hard. We now show that if two gates are syntactically equivalent then the functions computed at these two gates must be equal. We show later that the syntactic equivalence relation is polynomial-time decidable for the class of circuits of interest to us. In this sense we shall treat syntactic equivalence as a tractable refinement of an NP-complete relation.

Lemma 3.11. *Let $C = \langle G, \Omega, \Sigma, \Lambda, L\rangle$ be a $(\mathbb{B}, \rho)$-circuit of order $n$. Let $\mathcal{A}$ be a $\rho$-structure of size $n$ and let $\gamma$ be a bijection from the universe of $\mathcal{A}$ to $[n]$. For all $g, g' \in G$ if $g \equiv g'$ then $C[\gamma\mathcal{A}](g) = C[\gamma\mathcal{A}](g')$.*

Proof. We prove the result by induction on depth. Suppose $g$ and $g'$ have depth 0 and $g \equiv g'$. Then they are both input gates and so $g = g'$. The result for depth 0 then follows trivially. Suppose $g$ and $g'$ are internal gates, and suppose for all $h, h' \in G$ of depth less than $g$ or $g'$ we have that if $h \equiv h'$ then $C[\gamma\mathcal{A}](h) = C[\gamma\mathcal{A}](h')$. Suppose $g \equiv g'$. There exists $\lambda \in \mathrm{NI}(g)$ such that $L(g)(x) \equiv L(g')(\lambda x)$ for all $x \in \mathrm{ind}(g)$. It follows from the inductive hypothesis that $L^{\gamma\mathcal{A}}(g)(x) = C[\gamma\mathcal{A}](L(g)(x)) = C[\gamma\mathcal{A}](L(g')(\lambda x)) = (L^{\gamma\mathcal{A}}(g)\lambda)(x)$ for all $x \in \mathrm{ind}(g)$. Since $\Sigma(g)$ (and so $\Sigma(g')$) are isomorphism invariant, it follows that $C[\gamma\mathcal{A}](g) = \Sigma(g)(L^{\gamma\mathcal{A}}(g)) = \Sigma(g')(L^{\gamma\mathcal{A}}(g')\lambda) = \Sigma(g')(L^{\gamma\mathcal{A}}(g')) = C[\gamma\mathcal{A}](g')$. The result follows. □

The syntactic equivalence relation identifies gates that have "equivalent" circuits underneath them. A similar intuition is captured by identifying gates that are mapped to one another by automorphisms that extend the trivial permutation. We now show that if two automorphisms extend the same permutation then the two images of each gate must be syntactically equivalent.

Lemma 3.12. *Let $C$ be a circuit of order $n$, $\sigma \in \boldsymbol{Sym}_n$ and $\pi, \pi' \in \boldsymbol{Aut}(C)$ both extend $\sigma$, then for every gate $g$ in the circuit we have that $\pi(g)$ and $\pi'(g)$ are syntactically equivalent.*

Proof. From the definition of an automorphism we have for any gate $g$ in $C$ that $\Sigma(g) = \Sigma(\pi(g)) = \Sigma(\pi'(g))$, and either all of $g$, $\pi(g)$ and $\pi'(g)$ are not output gates or all three are output gates and $\Omega(\Omega^{-1}(g)) = \Omega(\sigma\Omega^{-1}(g)) = \pi'\Omega(\Omega^{-1}(g)) = \pi'(g)$.

We now prove the result by induction on depth. Suppose $g$ is a gate of depth 0. Then $g$ is either a relational or constant gate. In either case $\pi(g) = \pi'(g)$, and so $\pi(g)$ and $\pi'(g)$ are syntactically equivalent.

Suppose $g$ is an internal gate and suppose that for every gate $h$ of depth less than $g$, $\pi(h)$ is syntactically equivalent to $\pi'(h)$. We have that there exists $\lambda, \lambda' \in \mathrm{NI}(g)$ such that $\pi L(g)(x) = L(\pi g)(\lambda x)$ and $\pi' L(g)(x) = L(\pi' g)(\lambda' x)$, for all $x \in \mathrm{ind}(g)$. Then, from the inductive hypothesis, we have that $\pi L(g)(x) \equiv \pi' L(g)(x)$ and so $L(\pi g)(\lambda x) \equiv L(\pi' g)(\lambda' x)$, for all $x \in \mathrm{ind}(g)$. Thus we have that $L(\pi g)(x) = L(\pi g)(\lambda\lambda^{-1}(x)) \equiv L(\pi' g)(\lambda'\lambda^{-1}(x))$, for all $x \in \mathrm{ind}(g)$, and so $L(\pi g)/_{\equiv}$ is isomorphic to $L(\pi' g)/_{\equiv}$. We thus have that $\pi(g)$ and $\pi'(g)$ are syntactically equivalent, and the result follows. □

It follows from Lemma 3.12 that the syntactic equivalence relation of a circuit $C$ constrains the automorphism group of $C$ and the orbits and stabiliser groups of the gates in $C$. It follows then that if all of the internal gates of $C$ have trivial syntactic equivalence classes, i.e. for all $g$ and $h$ in $C$ if $g \equiv h$ then $g = h$, then $C$ has unique extensions. This motivates the importance of the following definition.

*Definition 3.13.* We say that a circuit $C$ is *reduced* if $C$ has trivial syntactic equivalence classes.

We now define the notion of a transparent circuit, as well as numerous other restrictions on the structure of a circuit.

*Definition 3.14.* Let $C$ be a circuit and $g$ be a gate in $C$. We say $g$ has *injective labels* if $L(g)$ is an injection. We say $g$ has *unique children* if no two distinct gates in $H_g$ are syntactically equivalent. We say $g$ has *unique labels* if $g$ has injective labels and unique children.

We say $C$ has *injective labels* (or just $C$ is *injective*) if every gate in $C$ has injective labels. We say $C$ has *unique labels* if every gate in $C$ has unique labels. We say $C$ is *transparent* if every gate $g$ in $C$ that is not fully symmetric has unique labels.

We earlier noted the need for a normal form analogous to the notion of a rigid circuit introduced by Anderson and Dawar [1]. We will show that the notion of a reduced injective circuit suffices as our normal form. We have already noted that these circuits have unique extensions. This allows us to apply the theory of supports developed in Section 4. Moreover, we show in Section 5 that there is a polynomial time translation from transparent circuits to reduced injective circuits. We also show that many natural circuit properties can be computed in polynomial time for such circuits. These results together play a crucial role in our translation from families of circuits to formulas in Section 7.

We say a circuit $C$ is a *rank circuit* if it is defined over the basis $\mathbb{B}_{\mathbf{rk}}$. We are now in a position to state the main theorem of this paper.

Theorem 3.15 (Main Theorem). *Let $\rho$ be a relational vocabulary and let $Q$ be a $\rho$-query. Then $Q$ is definable in* $\mathrm{FPR}[\rho]$ *if, and only if, $Q$ is definable by a* P*-uniform family of transparent symmetric rank circuits with vocabulary $\rho$.*

### 3.3 Limitations of Symmetric Bases

The main result of this subsection, Theorem 3.17, establishes that any symmetric circuit defined over an arbitrary basis of fully symmetric functions can be transformed in polynomial time into a symmetric majority circuit computing the same function. It follows from this result that, so long as we are interested in circuit families of at least polynomial size, it makes no difference to the expressive power of the model if we consider symmetric majority circuits or symmetric circuits defined over the basis containing *all* fully symmetric functions. Moreover, in order to define a symmetric circuit characterisation for FPR analogous to the symmetric circuit characterisation of FPC we must consider bases that include functions that are not fully symmetric.

We first show that any fully symmetric function can be computed by a small symmetric majority circuit. The crucial observation used is that for any fully symmetric function $F : \{0,1\}^n \to \{0,1\}$ and input $\vec{x}$ the value of $F(\vec{x})$ is entirely determined by the number of 1s in $\vec{x}$. In particular, $F$ is characterised by the set $c_F \subseteq [n]$ such that $m \in c_F$ if, and only if, there exists $\vec{x} \in \{0,1\}^n$ with $m$ 1s such that $F(\vec{x}) = 1$. We encode $F$ by the set $c_F$.

Proposition 3.16. *There is a linear time algorithm that takes as input a fully symmetric function $F : \{0,1\}^n \to \{0,1\}$ and outputs a symmetric majority circuit $C$ that computes $F$ and has depth at most 5, width at most $2n+2$ and size at most $5n+3$.*

Proof. We have $c_F$ from the input. We now define $C$. We define the set of gates and wires of $C$ layer by layer as follows. The first layer consists of the input gates $x_1, \ldots, x_n$. The second layer consists of two MAJ gates for each $a \in c_F$, which we denote by $\mathbf{maj}_a$ and $\mathbf{maj}_a^{\neg}$. For each $a \in c_F$ there is one wire from each of $x_1, \ldots, x_n$ to $\mathbf{maj}_a$ and $\mathbf{maj}_a^{\neg}$. For all $a \geq \frac{n}{2}$, there are $2a - n$ wires from 0 to $\mathbf{maj}_a$ and $2a - n + 2$ wires from 0 to $\mathbf{maj}_a^{\neg}$. For all $a < \frac{n}{2}$ there are $n - 2a$ wires from 1 to $\mathbf{maj}_a$ and $n - 2a - 2$ wires from 1 to $\mathbf{maj}_a^{\neg}$. The third layer consists of one NOT gate for each $a \in c_F$, which we denote by $\neg_a$. For each $a \in c_F$ there is a wire from $\mathbf{maj}_a^{\neg}$ to $\neg_a$. The fourth layer consists of one AND gate for each $a \in c_F$, which we denote by $\mathrm{count}_a$. For each $a \in c_F$ there is a wire from each of $\mathbf{maj}_a$ and $\neg_a$ to $\mathrm{count}_a$. The fifth layer consists of just a single OR gate, designated as the output gate, and for each $a \in c_F$ there is a wire from $\mathrm{count}_a$ to the output gate.

We summarise the circuit up to the fourth layer as follows:

$$\mathrm{count}_a = \begin{cases} \wedge(\mathbf{maj}(x_1,\dots,x_n,\underbrace{0,\dots,0}_{2a-n}),\neg(\mathbf{maj}(x_1,\dots,x_n,\underbrace{0,\dots,0}_{2a-n+2}))) & a \geq \frac{n}{2} \\ \wedge(\mathbf{maj}(x_1,\dots,x_n,\underbrace{1,\dots,1}_{n-2a}),\neg(\mathbf{maj}(x_1,\dots,x_n,\underbrace{1,\dots,1}_{n-2a-2}))) & a < \frac{n}{2}. \end{cases}$$

For an input $\vec{x} \in \{0,1\}^n$, $\mathrm{count}_a$ evaluates to 1 if, and only if, the number of 1s in $\vec{x}$ equals $a$. Thus $C$ evaluates to 1 if, and only if, there exists $a \in c_F$ such that the number of 1s in $\vec{x}$ equals $a$ if, and only if, $F(\vec{x}) = 1$.

Let $\sigma \in \mathbf{Sym}_n$. We define the automorphism $\pi$ extending $\sigma$ as follows. If $x_i$ is an input gate let $\pi(x_i) := x_{\sigma(i)}$. For each majority gate $g$ in the second layer there is exactly one wire from each input gate to $g$. We note additionally that every other gate in the circuit is connected to the (non-constant) input gates only through a gate in the second layer. As such, if $g$ is an internal gate we let $\pi(g) := g$. Then $\pi$ is an automorphism extending $\sigma$, and so $C$ is symmetric. It is easy to see that the construction of the circuit $C$ can be implemented by an algorithm running in time polynomial in $n$.

The first layer contains $n$ gates and the second layer contains at most $2|c_F| \leq 2n$ gates. The third and fourth layer each contain at most $n$ gates. The size of $C$ is at most $n + 2n + 2n + 1 + 2 = 5n + 3$ (the additional 2 is for the constant gates), the width of $C$ is at most $2n + 2$, and the depth is at most 5. □

Theorem 3.17. *Let $\mathbb{B}$ be a basis of fully symmetric functions. There is an algorithm that takes as input a symmetric circuit over $\mathbb{B}$ and outputs a symmetric majority circuit computing the same function. This algorithm runs in time polynomial in the size of the input circuit.*

Proof. Let $C$ be a symmetric circuit over $\mathbb{B}$. We construct the corresponding symmetric majority circuit $C'$ by replacing each gate labelled by a fully symmetric function $F$ with the corresponding circuit given in Proposition 3.16. It is easy to see that $C'$ is symmetric and computes the same function as $C$.

□

## 4 SYMMETRY AND SUPPORT

In this section we develop a theory of supports for symmetric circuits. We begin with the general definition of a support.

*Definition 4.1.* Let $G \leq \mathbf{Sym}_n$ and let $S \subseteq [n]$. Then $S$ is a *support* of $G$ if $\mathbf{Stab}_n(S) \leq G$.

We will define what it means for a gate or the element of the universe of a gate to have a support. The main theorem of this section, Theorem 4.10, establishes bounds on the sizes of these supports, which it turn allow us to encode information about the evaluation of a gate succinctly – an important step in translating families of symmetric circuits to formulas of FPR in Section 7. Before we specialise the discussion to circuits, we first present two general lemmas on supports which establish the existence of a unique minimal support.

Lemma 4.2 ([2, Lemma 26]). *Let $G \leq \boldsymbol{Sym}_n$ and let $A, B \subseteq [n]$ be supports of $G$. If $A \cup B \neq [n]$ then $A \cap B$ is a support of $G$.*

As stated [2, Lemma 26] is restricted to the case where $G$ is the stabiliser of a single element. However, the proof given suffices to prove this more general statement as well.

Lemma 4.3. *Let $G \leq \boldsymbol{Sym}_n$ and suppose $G$ has a support of size less than $\frac{n}{2}$. Then there exists a unique minimum-size support of $G$. Moreover, this support is a subset of each support of $G$ of size less than $\frac{n}{2}$.*

Proof. Let $S := \bigcap\{T \subseteq [n] : T \text{ supports } G \text{ and } |T| < \frac{n}{2}\}$. By repeated application of Lemma 4.2 it follows that $S$ is a support of $G$. Let $T \subseteq [n]$ be a support of $G$ such that $|T| \leq |S|$. Then $|T| < \frac{n}{2}$ and so $S \subseteq T$ and thus $S = T$. □

Suppose $G \leq \mathbf{Sym}_n$ has a support of size less than $\frac{n}{2}$. We call the unique minimum-size support of $G$ the *canonical support* of $G$ and denote it by $\mathrm{sp}(G)$.

Let $g$ be a gate in a symmetric circuit $C$ with unique extensions of order $n$. We say $S \subseteq [n]$ is a *support* of $g$ if $S$ is a support of $\mathbf{Stab}_n(g)$. If $g$ has a support of size less than $\frac{n}{2}$ then it follows from Lemma 4.3 that $g$ has a unique minimum-size support. We call this the support the *canonical support* of $g$ and denote it by $\mathrm{sp}(g)$.

Lemma 4.4. *Let $C$ be a symmetric circuit with unique extensions of order $n$. Let $g$ be a gate and $\sigma \in \boldsymbol{Sym}_n$ such that $g$ and $\sigma(g)$ have supports of size less than $\frac{n}{2}$. Then $\sigma sp(g) \subseteq sp(\sigma g)$.*

Proof. It suffices to prove that $\sigma^{-1}\mathrm{sp}(\sigma g)$ is a support of $g$, as then $\mathrm{sp}(g) \subseteq \sigma^{-1}\mathrm{sp}(\sigma g)$ (Lemma 4.3), and the result follows. Let $\pi \in \mathbf{Stab}(\sigma^{-1}\mathrm{sp}(\sigma g))$. Then for all $x \in \mathrm{sp}(\sigma g)$ we have $\pi\sigma^{-1}(x) = \sigma^{-1}(x)$ and so $\sigma\pi\sigma^{-1}(x) = x$. Thus $\sigma\pi\sigma^{-1} \in \mathbf{Stab}(\mathrm{sp}(\sigma g))$ and so $\sigma\pi\sigma^{-1}(\sigma g) = \sigma g$, which implies $\pi(g) = g$. □

Lemma 4.5. *Let $C$ be a symmetric circuit with unique extensions of order $n$. Let $g$ be a gate with supports of size less than $\frac{n}{2}$. If $\sigma_1, \sigma_2 \in \boldsymbol{Stab}_n$ agree on $sp(g)$ then $\sigma_1(g) = \sigma_2(g)$.*

Proof. Let $\sigma_1, \sigma_2 \in \mathbf{Stab}_n$ such that for all $x \in \mathrm{sp}(g)$ we have $\sigma_1(x) = \sigma_2(x)$. Then $\sigma_2^{-1}\sigma_1 \in \mathbf{Stab}(\mathrm{sp}(g))$, and so $\sigma_1(g) = \sigma_2(g)$. □

### 4.1 The Support Theorem

In this subsection we state and prove the *support theorem*. This essentially says that when the orbits in a circuit $C$ have size bounded by a polynomial in the order of the circuit, then every gate in $C$ has a support bounded in size by a constant. To get to a proof of the theorem, we use a general result about permutation groups. To understand this, suppose $G \leq \mathbf{Sym}_n$ is a group with index $[\mathbf{Sym}_n : G] \leq n^k$. This is the case, for instance, for $G = \mathbf{Stab}(g)$ when the orbit of $g$ has size at most $n^k$. One way to have such a group $G$ of small index is when $G$ contains a large symmetric group. That is $G$ contains *all* permutations on $n - k$ elements, or in other words $G$ has a support of size $k$ or less. These are clearly not the only subgroups of $\mathbf{Sym}_n$ of small index as witnessed by the alternating group $\mathbf{Alt}_n$. Theorem 4.6 below tells us that this is, essentially, the only counter-example. That is, if $G$ has small index in $\mathbf{Sym}_n$, then it must contain a large alternating group. We then show that, in the case of circuits, this counter-example does not arise and the stabiliser groups of gates always contain a large symmetric group.

Theorem 4.6 ([10], Theorem 5.2B). *Let $Y$ be a set such that $n := |Y| > 8$, and let $k$ be an integer with $1 \leq k \leq \frac{n}{4}$. Suppose that $G \leq \boldsymbol{Sym}_Y$ has index $[\boldsymbol{Sym}_Y : G] < \binom{n}{k}$ then for some $X \subseteq Y$ with $|X| < k$ we have $\boldsymbol{Stab}_{\boldsymbol{Alt}_Y}(X) \leq G$.*

The second result needed to prove the support theorem asserts that if $g$ is moved by some permutation $\sigma$ then there are two children of $g$ that *witness* this fact. In other words, there exist gates $h, h' \in H_g$ such that $\sigma$ either moves $h$ or $h'$ outside of $H_g$ or $\sigma$ moves $h$ and $h'$ so as to ensure that the structures on the children of $g$ and $\sigma g$ cannot be isomorphic. We now define this notion formally.

*Definition 4.7.* Let $C$ be an injective symmetric circuit with unique extensions of order $n$, let $g$ be a gate in $C$. Let $h, h' \in H_g$ and let $\sigma \in \mathbf{Sym}_n$. We say that $(h, h')$ *witnesses that $\sigma$ moves $g$* if at least one of the following hold:

- $\sigma h \notin H_g$,

- $\sigma h' \notin H_g$, or
- for every $\lambda \in \mathrm{NI}(g)$ either $\lambda L(g)^{-1}(h) \neq L(g)^{-1}(\sigma h)$ or $\lambda L(g)^{-1}(h') \neq L(g)^{-1}(\sigma h')$.

We now show that for any permutation $\sigma \in \mathbf{Sym}_n$, $\sigma$ moves an internal gate $g$ if, and only if, there exists a pair of children of $g$ that witness that $\sigma$ moves $g$.

Lemma 4.8. *Let $C$ be an injective symmetric circuit with unique extensions of order $n$, let $g$ be an internal gate in $C$, and let $\sigma \in \boldsymbol{Sym}_n$. Then $\sigma \in \boldsymbol{Stab}_n(g)$ if, and only if, for all $h, h' \in H_g$, $(h, h')$ does not witness that $\sigma$ moves $g$.*

Proof. $\Rightarrow$: Suppose $\sigma \in \mathbf{Stab}_n(g)$. Then $\sigma H_g = H_g$ and there exists $\lambda$ such that $\sigma L(g)(x) = L(g)(\lambda x)$ for all $x \in \mathrm{ind}(g)$. It follows that for all $h, h' \in H_g$ we have that $\sigma L(g)(L(g)^{-1}(h)) = L(g)(\lambda L(g)^{-1}(h))$ and so $L(g)^{-1}(\sigma h) = \lambda L(g)^{-1}(h)$ and similarly $L(g)^{-1}(\sigma h') = \lambda L(g)^{-1}(h')$.

$\Leftarrow$: Suppose for all $h, h' \in H_g$ we have that $(h, h')$ does not witness that $\sigma$ moves $g$. It follows that $\sigma H_g = H_g$. Let $\lambda : \mathrm{unv}(g) \to \mathrm{unv}(g)$ be defined by $\lambda(a) = (L(g)^{-1}(\sigma(L(g)(\vec{a}))))(i)$ for all $a \in \mathrm{unv}(g)$ where $(\vec{a}, R) \in \mathrm{ind}(g)$ and $i \in [|\vec{a}|]$ are such that $\vec{a}(i) = a$. We now show that $\lambda$ is well defined. Let $(\vec{a}_1, R), (\vec{a}_2, R) \in \mathrm{ind}(g)$ and let $i$ and $j$ be such that $a := \vec{a}_1(i) = \vec{a}_2(j)$. Since $(L(g)(\vec{a}_1), L(g)(\vec{a}_2))$ does not witness that $\sigma$ moves $g$, it follows that there exists $\lambda' \in \mathrm{NI}(g)$ such that $\lambda'(\vec{a}_1) = \lambda'(L(g)^{-1}(L(g)(\vec{a}_1))) = L(g)^{-1}(\sigma(L(g)(\vec{a}_1)))$ and similarly $\lambda'(\vec{a}_2)) = L(g)^{-1}(\sigma(L(g)(\vec{a}_2)))$. Then

$$\begin{aligned}\lambda(a) &= L(g)^{-1}(\sigma(L(g)(\vec{a}_1)))(i) = (\lambda'(\vec{a}_1)))(i) = (\lambda'(\vec{a}_2)))(j)\\ &= L(g)^{-1}(\sigma(L(g)(\vec{a}_2)))(j) = \lambda(a).\end{aligned}$$

It follows almost immediately from the definition of $\lambda$ that $\lambda \in \mathrm{NI}(g)$ and $L(g)\lambda = \sigma L(g)$, and so $\sigma \in \mathbf{Stab}_n(g)$. □

Corollary 4.9. *Let $C$ be an injective symmetric circuit with unique extensions of order $n$ and let $g$ be an internal gate in $C$. Let $h, h' \in H_g$, $\sigma \in \boldsymbol{Sym}_n$, and $\pi \in \boldsymbol{Stab}_n(h) \cap \boldsymbol{Stab}_n(h')$. If $(h, h')$ witnesses that $\sigma$ moves $g$ then $\sigma\pi \notin \boldsymbol{Stab}_n(g)$.*

Proof. Let $h, h' \in H_g$, $\sigma \in \mathbf{Sym}_n$, and $\pi \in \mathbf{Stab}_n(h) \cap \mathbf{Stab}_n(h')$. Suppose $(h, h')$ witnesses that $\sigma$ moves $g$. From Lemma 4.8 it follows that either (i) $\sigma\pi h = \sigma h \notin H_g$, (ii) $\sigma\pi h' = \sigma h' \notin H_g$, or (iii) for every $\lambda \in \mathrm{NI}(g)$ either $\lambda L(g)^{-1}(\pi h) = \lambda L(g)^{-1}(h) \neq L(g)^{-1}(\sigma h) = L(g)^{-1}(\sigma\pi h)$ or (similarly) $\lambda L(g)^{-1}(\pi h') \neq L(g)^{-1}(\sigma\pi h')$. Thus $(\pi h, \pi h')$ witnesses that $\sigma\pi$ moves $g$, and so from Lemma 4.8 $\sigma\pi \notin \mathbf{Stab}_n(g)$. □

Let $C$ be a symmetric circuit of order $n$. For each gate $g \in C$ let $\mathbf{supp\text{-}size}(g)$ be the minimal size of a support of $g$. Let $\mathbf{supp\text{-}size}(C) := \max\{\mathbf{supp\text{-}size}(g) : g \in C\}$ and $\mathbf{orbit\text{-}size}(C) := \max_{g \in C} |\mathbf{Orb}(g)|$.

We are now ready to prove the support theorem.

Theorem 4.10 (Support Theorem). *Let $C$ be an injective symmetric circuit with unique extensions of order $n > 8$. For every $1 \leq k \leq \frac{n}{4}$ if $\boldsymbol{orbit\text{-}size}(C) < \binom{n}{k}$ then $\boldsymbol{supp\text{-}size}(C) < k$.*

Proof. Let $n > 8$ and let $k \in [\frac{n}{4}]$ be such that $\mathbf{orbit\text{-}size}(C) < \binom{n}{k}$. We prove by induction on the structure of the circuit that each gate has a support of size less than $k$.

Suppose $g$ is an input gate in $C$. If $g$ is a constant gate then the empty set is a support. Suppose $g$ is a relational gate. Let $S_g$ be the set of distinct elements in $\Lambda(g)$. This is clearly a support of $g$ and also $\mathbf{Stab}_n(g) = \mathbf{Stab}_n(S_g)$. From the orbit-stabiliser theorem we have $\frac{n!}{(n-|S_g|)!} = [\mathbf{Sym}_n : \mathbf{Stab}_n(g)] = \mathbf{orbit\text{-}size}(g) < \binom{n}{k} = \frac{n!}{k!(n-k)!}$. It follows that $k!(n-k)! < (n-|S_g|)!$ and so $|S_g| < k$.

Suppose $g$ is an internal gate and for all $h \in H_g$ we have $\mathbf{supp\text{-}size}(h) < k$. From Theorem 4.6 it follows that there exists $X \subseteq [n]$ such that $|X| < k$ and $\mathbf{Stab}_{\mathbf{Alt}_n}(X) \leq \mathbf{Stab}_n(g)$. Suppose, for the sake of contradiction, that

$\mathbf{Stab}_n(X) \not\leq \mathbf{Stab}_n(g)$. Then for all $a, b \in [n] \setminus X$ we have $(a\ b) \notin \mathbf{Stab}_n(g)$, as if $(a\ b) \in \mathbf{Stab}_n(g)$ then $\mathbf{Stab}_n(X)$ is generated by $\mathbf{Stab}_{\mathbf{Alt}_n}(X) \cup \{(a\ b)\}$ and is thus a subgroup of $\mathbf{Stab}_n(g)$.

Let $a, b \in [n] \setminus X$ be distinct. Then $(a\ b) \notin \mathbf{Stab}_n(g)$. It follows from Lemma 4.8 that there exists $h_1, h_1' \in H_g$ such that $(h_1, h_1')$ witnesses that $(a\ b)$ moves $g$. Since $3k + 2 < n$ there exist distinct $c, d \in [n] \setminus (X \cup \mathrm{sp}(h_1) \cup \mathrm{sp}(h_1'))$. Then $(c\ d)$ moves $g$ and fixes both $h_1$ and $h_1'$. From Corollary 4.9 it follows $(a\ b)(c\ d) \notin \mathbf{Stab}_n(g)$. This yields the desired contradiction as $(a\ b)(c\ d) \in \mathbf{Stab}_{\mathbf{Alt}_n}(X) \leq \mathbf{Stab}_n(g)$. □

Remark 4.11. *As noted in Section 1, Otto [17] previously introduced and studied explicitly order-invariant circuits, a model very similar to the symmetric circuits discussed here. The main lemma of his paper is a result analogous to Theorem 4.10 (see [17, Lemma 9]). His proof is short and straight-forward, and has the considerable advantage of not relying on other results from the literature – as ours does. Otto's techniques can be adapted to prove a support theorem similar to Theorem 4.10. However, the resultant theorem places a lower upper bound on the value of k and so, while it would suffice to prove our main theorem, it is a weaker result.*

The following corollary establishes that if a family of injective symmetric circuits with unique extensions has a polynomial size bound on the sizes of the orbits then there is a constant bound on the sizes of the supports.

Corollary 4.12. *Let $(C_n)_{n\in\mathbb{N}}$ be a family of injective symmetric circuits with unique extensions and such that* ***orbit-size****$(C_n) = O(n^k)$ for some $k$. Then* ***supp-size****$(C_n) = O(1)$.*

Proof. Let $k \in \mathbb{N}$ be such that $\mathbf{orbit\text{-}size}(C_n) = O(n^k)$. Then $\mathbf{orbit\text{-}size}(C_n) = O(\binom{n}{k+1})$. It follows from Theorem 4.10 that for large enough $n$, $\mathbf{supp\text{-}size}(C_n)$ is bounded by some constant multiple of $k+1$, and so $\mathbf{supp\text{-}size}(C_n) = O(1)$. □

### 4.2 Supports and Indexes

We now extend our discussion of supports to include not only gates but elements of the universes of gates. We first define for each gate $g$ in a symmetric circuit a natural action of $\mathbf{Stab}(\mathrm{sp}(g))$ on the universe of $g$. This gives rise to a similar notion of a support for these elements. We show that the support theorem extends to this context.

Let $C$ be an injective symmetric circuit with unique extensions of order $n$, let $g$ be an internal gate in $C$ with a support of size less than $\frac{n}{2}$, and let $a \in \mathrm{unv}(g)$. We define an action of $\mathbf{Stab}(\mathrm{sp}(g))$ on the universe of $g$ by $\sigma \cdot a := (L(g)^{-1}\sigma L(g)(\vec{a}))(i)$, for $\sigma \in \mathbf{Stab}(\mathrm{sp}(g))$, $a \in \mathrm{unv}(g)$, and where $(\vec{a}, R) \in \mathrm{ind}(g)$ and $i \in [|\vec{a}|]$ is such that $a = \vec{a}(i)$.

We say that $S \subseteq [n]$ is a *support of $a$ in $g$* if $\mathbf{Stab}(S) \leq \mathbf{Stab}_{\mathbf{Stab}(\mathrm{sp}(g))}(a)$. It follows from Lemma 4.3 that if there is a support of $a$ in $g$ of size less than $\frac{n}{2}$ then there is a unique minimum-size support of $a$ in $g$. We call this minimum-size support the *canonical support of $a$ in $g$* and denote it by $\mathrm{sp}_g(a)$. We write $\mathbf{supp\text{-}size}_g(a)$ to denote the minimum size of a support of $a$ in $g$ and $\mathbf{unv\text{-}supp\text{-}size}(C)$ to denote the maximum of $\mathbf{supp\text{-}size}_g(a)$ for any gate $g$ and $a \in \mathrm{unv}(g)$.

We aim to extend the support theorem to elements of the universes of the gates in a circuit. We first show that the canonical support of an element of the universe of a gate $g$ is contained in the union of the canonical supports of $g$ and a child of $g$.

Lemma 4.13. *Let $C$ be an injective symmetric circuit with unique extensions of order $n$ such that* ***supp-size****$(C) \leq \frac{n}{2}$. Let $g$ be a gate in $C$ and let $a \in unv(g)$. Let $h \in H_g$ be such that there exists $(\vec{a}, R) \in \mathrm{ind}(g)$ where $a \in \vec{a}$ and $L(g)(\vec{a}) = h$. Then $sp(h) \cup sp(g)$ is a support of $a$ in $g$.*

Proof. Note that $\mathbf{Stab}_{\mathbf{Stab}(\mathrm{sp}(g))}(h) = \bigcap_{b \in L(g)^{-1}(h)} \mathbf{Stab}_{\mathbf{Stab}(\mathrm{sp}(g))}(b)$. Then

$$\begin{aligned}
\mathbf{Stab}(\mathrm{sp}(h) \cup \mathrm{sp}(g)) &\leq \mathbf{Stab}(\mathrm{sp}(h)) \cap \mathbf{Stab}(\mathrm{sp}(g)) \\
&\leq \mathbf{Stab}(h) \cap \mathbf{Stab}(\mathrm{sp}(g)) \\
&= \mathbf{Stab}_{\mathbf{Stab}(\mathrm{sp}(g))}(h) \\
&= \bigcap_{b \in L(g)^{-1}(h)} \mathbf{Stab}_{\mathbf{Stab}(\mathrm{sp}(g))}(b) \\
&\leq \mathbf{Stab}_{\mathbf{Stab}(\mathrm{sp}(g))}(a).
\end{aligned}$$

□

The following extension of the support theorem is an immediate corollary of Lemma 4.13 and Theorem 4.10.

Theorem 4.14. *Let $C$ be an injective symmetric circuit with unique extensions of order $n > 8$. For every $1 \leq k \leq \frac{n}{4}$ if $\textbf{\textit{orbit-size}}(C) < \binom{n}{k}$ then $\textbf{\textit{unv-supp-size}}(C) < 2k$ and $\textbf{\textit{supp-size}}(C) < k$.*

Proof. Let $g$ be a gate in $C$ and Let $h \in H_g$ be such that there exists $(\vec{a}, R) \in \mathrm{ind}(g)$ where $a \in \vec{a}$ and $L(g)(\vec{a}) = h$. It follows from Theorem 4.10 that $|\mathrm{sp}(h)| < k$ and $|\mathrm{sp}(g)| < k$. From Lemma 4.13 it follows that $\mathrm{sp}(h) \cup \mathrm{sp}(g)$ is a support of $a$ in $g$. □

We can derive from Theorem 4.14 a corollary analogous to Corollary 4.12.

Corollary 4.15. *Let $(C_n)_{n \in \mathbb{N}}$ be a family of injective symmetric circuits with unique extensions such that $n \mapsto \textbf{\textit{orbit-size}}(C_n) = O(n^k)$ for some $k$. Then $n \mapsto \textbf{\textit{unv-supp-size}}(C_n) = O(1)$.*

## 5 NORMAL FORMS AND DECIDABILITY

In this section we show that many important circuit parameters can be computed in polynomial time so long as we restrict our attention to transparent circuits. These results are used in Section 7 to establish a translation from P-uniform families of transparent symmetric rank circuits to FPR-formulas.

The first result we prove, Lemma 5.1, establishes that the syntactic equivalence relation for transparent circuits can be computed in polynomial time. We use this result to show in Proposition 5.2 that the transparency property itself is polynomial-time decidable.

We show in Lemma 5.5 that there is a polynomial-time algorithm that takes as input a transparent circuit $C$ and outputs a reduced injective circuit computing the same function as $C$. This result allows us to assume, without a loss of generality, that all P-uniform families of transparent symmetric circuits consist of reduced injective circuits.

In the remainder of this section we consider the problem of computing supports and orbits. We show in Lemma 5.6 that there is a polynomial-time algorithm that determines the image of a gate in a reduced circuit under the action of a given permutation. We use this algorithm to show in Lemma 5.8 that there exists a polynomial-time algorithm that determines if a reduced circuit is symmetric, and if so computes the canonical supports and orbits of the gates in that circuit. We conclude this section by showing that this result can be extended to the support and orbits of elements of the universe of a gate.

We now show that the syntactic equivalence relation can be computed in polynomial time for transparent circuits.

Lemma 5.1. *There is an algorithm that takes as input a transparent circuit $C$ and outputs the syntactic equivalence relation on the gates of $C$. The algorithm runs in time polynomial in the size of $C$.*

Proof. The syntactic equivalence relation is defined inductively, and this definition can be implemented as an algorithm. To show that this algorithm runs in time polynomial in the size of the circuit it suffices to show that there exists a polynomial time algorithm that takes as input a transparent circuit $C$, two gates $g$ and $h$, and the syntactic equivalence relation for all gates of depth less than either $g$ or $h$ and decides if there exists $\lambda \in \mathrm{NI}(h)$ such that for all $x \in \mathrm{ind}(h)$, $L(h)(\lambda(x)) \equiv L(g)(x)$.

This algorithm is given as follows. We first attempt to greedily construct a bijection $f : \mathrm{ind}(g) \to \mathrm{ind}(h)$ such that for all $x \in \mathrm{ind}(g)$, $L(g)(x) \equiv L(h)(f(x))$. If this construction fails halt and output that $g$ and $h$ are not syntactically equivalent. Since each gate in $C$ has unique children, if this succeeds it follows that $L$ is the unique map with this domain and codomain that pairs syntactically equivalent gates. We check if $f$ is an isomorphism from the structure associated with $g$ to the structure associated with $h$. If so we output that $g \equiv h$ and otherwise output that $g \not\equiv h$. □

We now show that transparency is polynomial-time decidable. The algorithm works by first checking that each gate that is not fully symmetric has injective labels and then iterating over the circuit layer by layer, at each step computing the syntactic equivalence relation for the gates visited so far using the same approach for testing syntactic equivalence as in the proof of Lemma 5.1, and then using this to check whether there exists a gate that is not fully symmetric and without unique labels in the next layer. This algorithm can be easily implemented and we omit a full proof.

Proposition 5.2. *There is an algorithm that takes as input a circuit and decides if that circuit is transparent. This algorithm runs in time polynomial in the size of the circuit.*

We now define what it means to take a quotient of a circuit by the syntactic equivalence relation. Intuitively, the quotient of a circuit is defined by "merging" each syntactic equivalence class into a single gate and including a wire between any two gates $[h]$ and $[g]$ in the quotient circuit if, and only if, there is a wire from $h$ to $g$.

*Definition 5.3.* Let $C := \langle G, \Omega, \Sigma, \Lambda, L\rangle$ be a $(\mathbb{B}, \rho)$-circuit. A *quotient of* $C$ is a $(\mathbb{B}, \rho)$-circuit $C_\equiv := \langle G_\equiv, \Omega_\equiv, \Sigma_\equiv, \Lambda_\equiv, L_\equiv\rangle$, where $G_\equiv = G/_\equiv$, $\Omega_\equiv = \Omega/_\equiv$, $\Sigma = \Sigma/_\equiv$, $(\Lambda_\equiv)_R = \Lambda_R/_\equiv$ for all $R \in \rho$, and for all $[g] \in G_\equiv$ there exists $g' \in [g]$ such that $L_\equiv([g]) = L(g')/_\equiv$.

Note that if $C$ and $C'$ are quotients of the same circuit then for each gate $g$, $L(g)$ and $L'(g)$ are isomorphic as labelled structures. It follows that $C$ and $C'$ are isomorphic in the precise sense alluded right after Definition 3.7.

We now show that taking the quotient of a circuit preserves important properties, including the function computed by the circuit, the symmetry of the circuit, and whether the circuit has unique labels. We also show that the quotient of a circuit is reduced.

Lemma 5.4. *Let $C := \langle G, \Omega, \Sigma, \Lambda, L\rangle$ be a $(\mathbb{B}, \rho)$-circuit and $C_\equiv = \langle G_\equiv, \Omega_\equiv, \Sigma_\equiv, \Lambda_\equiv, L_\equiv\rangle$ be a quotient of $C$. Then $C_\equiv$ is reduced and $C$ and $C_\equiv$ compute the same function. Moreover, if $C$ is symmetric then $C_\equiv$ is symmetric, and for all $g \in G$, $g$ has unique labels in $C$ if, and only if, $[g]$ has unique labels in $C_\equiv$. Indeed, for all $\sigma \in \boldsymbol{Sym}_n$ if there exists $\pi \in \boldsymbol{Aut}(C)$ extending $\sigma$ then $\pi/_\equiv$ is an automorphism of $C_\equiv$ extending $\sigma$.*

Proof. Let $n$ be the order of $C$. We first prove that $C_\equiv$ and $C$ compute the same function. Let $\mathcal{A}$ be a $\rho$-structure of size $n$ and let $\gamma$ be a bijection from $A$ to $[n]$. We now show that for all $g \in G$, $C_\equiv[\gamma\mathcal{A}]([g]) = C[\gamma\mathcal{A}](g)$. We do this by induction on the depth of a gate. Suppose $g \in G$ has depth 0. In this case $g$ is an input gate and the result follows trivially. Suppose $g$ is an internal gate and suppose for all $h$ of depth less than $g$ we have that $C_\equiv[\gamma\mathcal{A}]([h]) = C[\gamma\mathcal{A}](h)$. We have that there exists $g' \in [g]$ such that $L_\equiv([g]) = L(g')/_\equiv$ and $\Sigma_\equiv([g]) = \Sigma(g) = \Sigma(g')$. We have from Lemma 3.11 and the inductive hypothesis

that there exists $\lambda \in \mathrm{NI}(g)$ such that $L^{\gamma\mathcal{A}}_{\equiv}([g]) = L^{\gamma\mathcal{A}}(g') = L^{\gamma\mathcal{A}}(g)\lambda$. It follows from the fact that $\Sigma(g)$ is a structured function that $C_{\equiv}[\gamma\mathcal{A}]([g]) = \Sigma_{\equiv}([g])(L^{\gamma\mathcal{A}}_{\equiv}([g])) = \Sigma(g')(L^{\gamma\mathcal{A}}(g')) = \Sigma(g)(L^{\gamma\mathcal{A}}(g)\lambda) = \Sigma(g)(L^{\gamma\mathcal{A}}(g)) = C[\gamma\mathcal{A}](g)$.

We now show that $C_{\equiv}$ is reduced. Suppose $[g], [h] \in G_{\equiv}$ and suppose $[g] \equiv [h]$. If $[g]$ and $[h]$ are both input or output gates then $[g] = [h]$. Suppose $[g]$ and $[h]$ are internal gates that are not output gates. Then $\Sigma(g) = \Sigma_{\equiv}([g]) = \Sigma_{\equiv}([h]) = \Sigma(h)$. There exists $g', h' \in G$ such that $g' \equiv g$ and $h' \equiv h$, and $L_{\equiv}([g]) = L(g')/_{\equiv}$ and $L_{\equiv}([h]) = L(h')/_{\equiv}$. It follows from $[g] \equiv [h]$ that $L(g')/_{\equiv}$ is isomorphic to $L(h')/_{\equiv}$. From this it follows that $g' \equiv h'$, and so $[g] = [h]$.

Let $\sigma \in \mathbf{Sym}_n$ and suppose there exists $\pi \in \mathbf{Aut}(C)$ extending $\sigma$. Let $\pi_{\equiv} = \pi/_{\equiv}$. We now show that $\pi_{\equiv}$ is an automorphism of $C_{\equiv}$ extending $\sigma$. It is easy to see that $\pi_{\equiv}$ is a bijection from $G_{\equiv}$ to $G_{\equiv}$. Let $[g] \in G_{\equiv}$. We have that $\Sigma_{\equiv}(\pi_{\equiv}[g]) = \Sigma_{\equiv}([\pi g]) = \Sigma(\pi g) = \Sigma(g) = \Sigma_{\equiv}([g])$. It is easy to check the automorphism conditions for input gates. Suppose $[g]$ is an internal gate. Let $g' \in [g]$ be such that $L_{\equiv}([g]) = L(g')/_{\equiv}$ and $h \in [g]$ be such that $L_{\equiv}(\pi_{\equiv}[g]) = L_{\equiv}([\pi h]) = L(\pi h)/_{\equiv}$. We have that $\pi L(g')$ is isomorphic to $L(\pi g')$, and it follows that $(\pi L(g'))/_{\equiv}$ is isomorphic to $L(\pi g')/_{\equiv}$. We then have $(\pi L(g'))/_{\equiv} = \pi_{\equiv}(L(g')/_{\equiv}) = \pi_{\equiv}L_{\equiv}([g])$ and, since $g' \equiv h$ and so $\pi g' \equiv \pi h$, we have that $L(\pi g')/_{\equiv}$ is isomorphic to $L(\pi h)/_{\equiv} = L_{\equiv}(\pi_{\equiv}[g])$. It follows that $\pi_{\equiv}L_{\equiv}([g])$ is isomorphic to $L_{\equiv}(\pi_{\equiv}[g])/_{\equiv}$. Suppose $[g]$ is an output gate. Then for $\vec{a} \in \mathbf{Dom}(\Omega)$, $\pi_{\equiv}\Omega_{\equiv}(\vec{a}) = [\pi\Omega(\vec{a})] = [\Omega(\sigma\vec{a})] = \Omega_{\equiv}(\sigma\vec{a})$. It follows that if $C$ is symmetric then $C_{\equiv}$ is symmetric.

We now show that for $g \in G$, $g$ has unique labels in $C$ if, and only if, $[g]$ has unique labels in $C_{\equiv}$. Let $g \in G$ and let $h \in [g]$ such that $L_{\equiv}([g]) = L(h)/_{\equiv}$. We first prove the backward direction. Suppose $[g]$ has unique labels. Then, since $L_{\equiv}([g])$ is injective, $L(h)/_{\equiv}$ must be injective and so $L(h)$ must be injective and no two child gates of $h$ can be syntactically-equivalent. It follows that $h$ has unique labels. Since $h \equiv g$ and $h$ has unique labels, it follows that $g$ has unique labels. We now prove the forward direction. Suppose $g$ has unique labels. Then, since $h \equiv g$ and $g$ has unique labels, $h$ has unique labels and so $L_{\equiv}([g])$ has injective labels. Since each syntactic equivalence class in $C_{\equiv}$ is a singleton it follows that $[g]$ has unique labels. □

We next show that there is a polynomial-time algorithm that takes as input a transparent circuit and outputs a reduced injective circuit computing the same function. Roughly, this algorithm first computes the syntactic equivalence relation of the input circuit using Lemma 5.1 and then, by picking representatives, defines a quotient circuit. It follows from Lemma 5.4 that the quotient circuit is reduced and computes the same function as the input circuit, but it may have fully symmetric gates that are not injective. To remedy this, we iterate over the circuit, "copying" each child of each fully symmetric gate an appropriate number of times and appending to each copy a distinct gadget that ensures that no two copies can be syntactically equivalent. The complete proof of this result, and the specific construction of these gadgets, is laborious but straight forward. We omit the details here and refer to the reader to [18] for a complete proof.

Lemma 5.5. *There is an algorithm that takes as input a transparent $(\mathbb{B}, \rho)$-circuit $C$ and outputs a $(\mathbb{B} \cup \mathbb{B}_{\boldsymbol{std}}, \rho)$-circuit $C'$ such that $C$ and $C'$ compute the same function, $C'$ is reduced and injective, and if $C$ is symmetric then $C'$ is symmetric. Moreover, this algorithm runs in time polynomial in the size of the input circuit.*

We now show that there is a polynomial-time algorithm for computing the image of a gate under the action of a permutation

Lemma 5.6. *There is an algorithm that takes as input a reduced injective $(\mathbb{B}, \rho)$-circuit $C$ of order $n$ and a permutation $\sigma \in \boldsymbol{Sym}_n$ and outputs for each gate $g$ the image of $g$ under the action of the unique automorphism extending $\sigma$ (if it exists). This algorithm runs in time polynomial in the size of the input circuit.*

Proof. Let $\langle G, \Omega, \Sigma, \Lambda, L\rangle = C$. We attempt to define a function $\pi : G \to G$ inductively, beginning with the action of $\sigma$ on the input gates and extending this action layer by layer.

Suppose $h$ is an input gate. If $h$ is constant let $\pi(h) = h$. If $h$ is a relational gate then, since $C$ is reduced, there is at most one gate $h'$ such that (i) $\Sigma(h) = \Sigma(h')$, (ii) $\sigma\Lambda(h) = \Lambda(h')$, and (iii) if $h$ is an output gate then $h' = \Omega(\sigma(\Omega^{-1}(h)))$. If such an $h'$ exists let $\pi(h) = h'$, otherwise halt and output that no automorphism exists. Suppose $h$ is an internal gate and $\pi$ has been defined for every gate of depth less than $h$. Since $C$ is reduced there exists at most one gate $h'$ such that (i) $\Sigma(h) = \Sigma(h')$, (ii) $\pi L(h)$ is isomorphic to $L(h')$, and (iii) if $h$ is an output gate then $h' = \Omega(\sigma(\Omega^{-1}(h)))$. If such an $h'$ exists let $\pi(h) = h'$, otherwise halt and output that no automorphism exists.

For $d \in \mathbb{N}$ let $C_d$ be the subcircuit induced by the set of gates of depth at most $d$ (i.e. the circuit defined on the induced subgraph). It is easy to show by induction that for each $d \in \mathbb{N}$ the function $\pi$ restricted to the gates of $C_d$ is an automorphism of $C_d$ extending $\sigma$. It follows that $\pi$ is an automorphism of $C$ extending $\sigma$.

We thus run the above inductive algorithm and if it succeeds we output the image of each gate under $\pi$, and otherwise output that no automorphism exists. To see that this may be done in time polynomial in the size of the circuit it suffices to show that for any two gates $h$ and $h'$ in $C$, we can check if $\pi L(h)$ is isomorphic to $L(h')$ in polynomial time. Since this circuit is reduced and injective, $\pi L(h)$ is isomorphic to $L(h')$ if, and only if, $L^{-1}(h')\pi L(h)$ is an automorphism. It can easily be checked if $L^{-1}(h')\pi L(h)$ is an automorphism of the structure associated with $h$.

□

We now use Lemma 5.6 to define an algorithm that computes in polynomial time the image of a given element of the universe of a gate under the action of a given permutation.

Lemma 5.7. *There is an algorithm that takes as input an injective reduced $(\mathbb{B}, \rho)$-circuit $C$ of order $n$, $\sigma \in \boldsymbol{Sym}_n$, $g$ a gate in $C$, and $a \in unv(g)$ and, if there exists an automorphism of $C$ extending $\sigma$ such that $\sigma \in \boldsymbol{Stab}(g)$, outputs $\sigma(a)$. The algorithm runs in time polynomial in the size of $C$ and the encoding of $\sigma$.*

Proof. Let $C = \langle G, \Omega, \Sigma, \Lambda, L\rangle$. We use the algorithm from Lemma 5.6 to check if $\sigma$ extends to an automorphism on $C$. We also check if $\sigma \in \mathbf{Stab}(g)$. If either of these checks fail, halt and return that no such automorphism exists. Let $h \in H_g$ and $\vec{b} := L(g)^{-1}(h)$ be such that $a \in \vec{b}$, and let $i \in [|\vec{b}|]$ be the index of $a$ in $\vec{b}$. Halt and output $\sigma a = (L(g)^{-1}(\sigma h))(i)$. □

We now show that there is a polynomial-time algorithm that takes as input a reduced injective symmetric circuit $C$ and a gate $g$ and outputs the orbit and canonical support of $g$.

Lemma 5.8. *There is an algorithm that takes as input an injective reduced symmetric circuit $C$ and a gate $g$ with a support of size less than $\frac{n}{2}$ and if $C$ is symmetric outputs $\boldsymbol{Orb}(g)$ and $sp(g)$ and otherwise outputs that $C$ is not symmetric. This algorithm runs in time polynomial in the size of the circuit.*

Proof. Let $n$ be the order of $C$. Let $T \subseteq \mathbf{Sym}_n$ be the set of transpositions. Note that $T$ generates $\mathbf{Sym}_n$. We can thus determine if $C$ is symmetric by running the algorithm in Lemma 5.6 for each transposition in $T$. Since $|T| = \binom{n}{2}$ the total time taken to execute this is polynomial in $|C|$. If $C$ is not symmetric we halt and output that $C$ is not symmetric. We assume for the rest of this proof that $C$ is symmetric.

Let $\hat{T} = T \cap \mathbf{Stab}(g)$. Let $\sim$ be the equivalence relation on $[n]$ defined such that $a \sim b$ if, and only if, $(a\ b) \in \hat{T}$. To see that $\sim$ is an equivalence relation observe that if $a \sim b$ and $b \sim c$ then $(a\ b)$ and $(b\ c)$ are in $\hat{T}$ and so, since $(a\ c) = (a\ b)(b\ c)(a\ b)$, we have $a \sim c$. Note that if $a, b \in [n] \setminus \mathrm{sp}(g)$ then $a \sim b$. It follows that there exists $S \in [n]/\sim$ such that $[n] \setminus \mathrm{sp}(g) \subseteq S$. Since $|[n] \setminus \mathrm{sp}(g)| > \frac{n}{2}$ we have that $S$ is the unique largest equivalence class in $[n]$.

Suppose $S \neq [n] \setminus \mathrm{sp}(g)$. Since $[n] \setminus \mathrm{sp}(g) \subsetneq S$, there exists $c \in \mathrm{sp}(g)$ such that $[n] \setminus (\mathrm{sp}(g) \setminus \{c\}) \subseteq S$. Thus for every $a, b \in [n] \setminus (\mathrm{sp}(g) \setminus \{c\})$ we have $(a\ b) \in \hat{T}$, and so $(a\ b) \in \mathbf{Stab}(g)$. But the set of all these transpositions generates $\mathbf{Stab}(\mathrm{sp}(g) \setminus \{c\})$, and so $\mathbf{Stab}(\mathrm{sp}(g) \setminus \{c\}) \leq \mathbf{Stab}(g)$. This contradicts the minimality of $\mathrm{sp}(g)$, and so we conclude $S = [n] \setminus \mathrm{sp}(g)$, and so $\mathrm{sp}(g) = [n] \setminus S$.

The computation of $\mathbf{Orb}(g)$ requires more than just computing the image of $g$ under the set of all transpositions. Let $S_0 := \{g\}$ and $S_{i+1} := \{t(h) : h \in S_i, t \in T\}$. The least fixed point of this process is the orbit of $g$. □

We can use similar methods to establish an analogous result for elements of the universe of the gate. We omit the details of the proof but state the result formally below.

Lemma 5.9. *There exists a polynomial-time algorithm that takes as input a transparent symmetric circuit $C$, a gate $g$ which has a support of size less than $\frac{n}{2}$, and an element $a \in unv(g)$, and outputs $sp_g(a)$ and $\boldsymbol{Orb}_g(a)$.*

## 6 TRANSLATING FORMULAS INTO CIRCUITS

In order to prove one direction of our main result we need to establish a translation from FPR-formulas to P-uniform families of transparent symmetric rank circuits. There is a known translation from FO-formulas to P-uniform families of symmetric circuits [15]. This translation associates an FO-formula $\theta(\vec{x})$ with a family of circuits $(C_n)_{n\in\mathbb{N}}$ such that each gate in $C_n$ corresponds to a subformula $\psi(\vec{y})$ and an assignment $\vec{a}$ to $\vec{y}$ in $[n]$. Each gate in $C_n$ is labelled according to the symbol at the head of the associated subformula, with existential and universal quantifiers translating to large disjunctions and conjunctions, respectively.

This approach can be extended to formulas of fixed-point logic by first unfurling the fixed-point operator in order to define a family of FO-formulas (see [16] for details) with a constant bound on width, and then translating each of these formulas to a symmetric circuit. The bound on width is ensured by renaming variables at each stage when unfurling the operator. This approach can be extended to logics with counting, so long as we include threshold or majority gates in the circuit and extend first-order logic with counting quantifiers, yielding a translation from FPC-formulas to bounded width P-uniform families of symmetric majority circuits [1].

We should like to use an analogous approach to define a translation from FPR to bounded width P-uniform families of transparent rank circuits. The problem is that these translations do not, in general, yield families of *transparent* circuits. The essential reason is that a formula might be invariant under some non-trivial permutations of the bound variables, and the translation to circuits described above preserves these "syntactic symmetries", potentially resulting in gates that are not fully symmetric with two syntactically equivalent children. We now sketch a translation that does produce transparent circuits by adding some additional gadgetry to the formula in order to remove symmetries of this sort. The full proof is available here [18].

Theorem 6.1. *For each* FPR*-formula $\theta(\vec{x})$ there exists a* P*-uniform family of transparent symmetric rank circuits $(C_n)_{n\in\mathbb{N}}$ that defines the same query as $\theta(\vec{x})$.*

This theorem is proved in three steps. First, we introduce a logic FO+rk, defined by extending FO with *rank quantifiers*[1]. Second, we show that for each FPR-formula $\theta(\vec{x})$ there exists a P-uniform family of FO+rk-formulas $(\theta_n(\vec{x}))_{n\in\mathbb{N}}$ and

[1]Dawar et al. [5] define a different rank quantifier and a corresponding extension of FO. They establish a similar translation from a fixed-point logic with rank operators to families of first-order formulas with rank quantifiers. The translation is only defined for formulas in which the rank operator only binds element variables. We are interested in the case where both element and number variables may be bound by the operator, which requires introducing a quite different rank quantifier. The logic FO+rk and the extension of first-order logic with rank quantifiers in [5] are thus defined quite differently and are not known to have the same expressive power.

$k \in \mathbb{N}$, such that each $\theta_n(\vec{x})$ defines the same query as $\theta(\vec{x})$ on structures of size $n$ and has width bounded by $k$. Thirdly, we define an algorithm that takes as input a natural number $n$, computes $\theta_n$ and then, by recursion on the structure of the formula, outputs a corresponding transparent symmetric rank circuit $C_n$ in time polynomial in $n$.

The rank quantifiers in FO+rk determine if the rank of a matrix, defined using a family of formulas, with entries taken to be in some fixed finite field is at least equal to a given threshold. There are some subtleties involved in defining these quantifiers formally, and we refer the reader to [18] for a complete definition of FO+rk. For our purposes it suffices to note that FO+rk is strong enough to define formulas of the form

$$\theta := \mathbf{rk}_p^{\geq t}[\vec{x}, \vec{y}]\phi(\vec{x}, \vec{y}),$$

where $\theta$ holds for a given structure if the 0-1 matrix defined by $\phi$ with rows indexed by assignments to $\vec{x}$ and columns indexed by assignments to $\vec{y}$ and interpreted as having entries in $\mathbb{F}_p$ has rank at least $t$.

The translation from FPR-formulas to bounded width P-families of FO+rk-formulas can be completed using essentially the same approach as in [14, 16] for translating FPC formulas to uniform families of FO+C-formulas. This works by unfurling the fixed-point operators, i.e. recursively replacing each second-order variable bound by the operator with the corresponding formula, and replacing number terms, while renaming bound variables to keep the total number of variables bounded. When unfurling these formulas we encode each FO+rk-formula as a branching program so as to avoid exponential blow-ups. The details of this translation are available here [18].

The final step in this translation, the recursive construction of a transparent symmetric circuit from an FO+rk-formula, also follows a similar approach to the standard translation discussed above, but now with some additional gadgetry added in order to remove symmetries in the formula. To understand what sorts of symmetries are relevant and why the standard translation would result in potentially non-transparent circuits consider as an example the FO+rk-formula $\mathbf{rk}_p^{\geq t}[x, y](E(x, x) \wedge E(y, y))$. The standard translation would result in a rank gate with each child corresponding to a map from $\{x, y\}$ to $[n]$. It is easy to see that for each $a, b \in \mathbb{N}$ the child corresponding to the assignment $(a, b) \mapsto (x, y)$ will be syntactically equivalent to the child corresponding to the assignment $(b, a) \mapsto (x, y)$. To remove these sorts of symmetries we preprocess each formula $\theta_n$ and replace each subformula of the form $\mathbf{rk}_p^{\geq t}[\vec{x}, \vec{y}]\phi$ with one of the form $\mathbf{rk}_p^{\geq t}[\vec{x}, \vec{y}](\phi \wedge \psi)$, where $\psi$ is chosen so as to not be invariant under any permutation of $\vec{x} \cup \vec{y}$.

## 7 TRANSLATING CIRCUITS INTO FORMULAS

In this section we prove the following theorem establishing a translation from uniform families of symmetric rank circuits to FPR-formulas.

Theorem 7.1. *For each* P*-uniform family of transparent symmetric rank circuits* $(C_n)_{n\in\mathbb{N}}$ *there exists an* FPR*-formula expressing the same query.*

The proof is structured as follows. Let $(C_n)_{n\in\mathbb{N}}$ be a P-uniform family of transparent symmetric rank circuits. From the Immerman-Vardi theorem there exists a sequence of FP[$\leq$]-formulas $\Phi$ such that $\Phi$ defines $C_n$ when interpreted in the structure $([n], \leq)$. From Lemma 5.5 we may assume, without loss of generality, that each $C_n$ is reduced and injective. From Theorem 4.14 there is a constant $k \in \mathbb{N}$ bounding the size of the supports of each gate and each element of the universe of a gate in each $C_n$. We show in Lemma 7.2 that for a given $n$-size structure $\mathcal{A}$ and bijection $\gamma : A \to [n]$ the evaluation of a gate $g$ in $C_n$ depends only on how $\gamma$ maps elements of $\mathcal{A}$ to $\mathrm{sp}(g)$. It follows that each gate $g$ in some $C_n$ can be associated with a fixed arity relation $\mathrm{EV}_g$ consisting of all assignments to $\mathrm{sp}(g)$ for which $g$ evaluates to 1. We

use the various algorithms discussed in Section 5 and the Immerman-Vardi theorem to show that we can recursively define each $\mathrm{EV}_g$ in FPR, allowing us to evaluate the output gates in FPR and so prove the result.

This overall approach is similar in structure to the one used by Anderson and Dawar [1] for translating P-uniform families of symmetric majority circuits to FPC-formulas. However, the important results so far discussed, e.g. the support theorem and the existence of polynomial-time algorithms computing circuit properties, do not generalise obviously and have required quite different arguments. The work of this section draws these results together and establishes the final step in the translation, the inductive definition of $\mathrm{EV}_g$ for each gate $g$ in the circuit. Anderson and Dawar establish this by first exhibiting a bijection from the orbit of a gate to the set of assignments to its support and then, using this bijection, counting the number of children of $g$ that evaluate to 1. This suffices as the gates in the circuits of interest there are all fully symmetric, and so counting the number of inputs that evaluate to 1 suffices to evaluate the gate. However, in our more general setting a gate could be labelled by a rank function. The evaluation of such a gate depends not just on the *number* of children that evaluate to true, but on the *structure* defined at $g$.

The majority of the work of this section is in showing that this structure can be recursively defined in FPR. In Section 7.1 we give a recursive construction of a structure and show that it suffices. In Section 7.2 we show how this construction can be implemented in FPR and complete the proof of Theorem 7.1.

### 7.1 Recursively Defining a Matrix

We first show that for a gate $g$ in a symmetric circuit $C$ of order $n$ the evaluation of $g$ for an input structure $\mathcal{A}$ and bijection $\gamma \in [n]^{\underline{A}}$ depends only how $\gamma$ maps to $\mathrm{sp}(g)$. We say two injective functions $f$ and $g$ are *compatible* if we can define an injection on the union of their domains that agrees with each function on their respective domains. We write $f \sim g$ to denote that $f$ and $g$ are compatible and $f|g$ to denote the injection defined on the union of their domains that agrees with each function on their respective domains.

Lemma 7.2. *Let $C$ be a symmetric rank circuit with vocabulary $\rho$ of order $n$. Let $\mathcal{A} \in \mathrm{fin}[\rho, n]$. Let $g$ be an internal gate, $\eta \in A^{\underline{sp(g)}}$, and $\gamma_1, \gamma_2 \in [n]^{\underline{A}}$ such that $\gamma_1^{-1} \sim \eta$ and $\gamma_2^{-1} \sim \eta$. Then $L^{\gamma_1\mathcal{A}}(g)$ and $L^{\gamma_2\mathcal{A}}(g)$ are isomorphic and consequently $C[\gamma_1\mathcal{A}](g) = C[\gamma_2\mathcal{A}](g)$.*

Proof. Let $\pi \in \mathbf{Sym}_n$ be the unique permutation such that $\pi\gamma_1 = \gamma_2$. Since $\gamma_1^{-1}$ and $\gamma_2^{-1}$ are both compatible with $\eta$, it follows that $\pi$ must fix $\mathrm{sp}(g)$ pointwise. From the definition of a support, we have that $\pi(g) = g$, and so $L(g)$ is isomorphic to $\pi L(g)$. Therefore there exists $\lambda \in \mathrm{NI}(g)$ such that $\pi L(g) = L(g)\lambda$, and so for all $a \in \mathrm{ind}(g)$,

$$\begin{aligned} L^{\gamma_1\mathcal{A}}(g)(a) &= C[\gamma_1\mathcal{A}](L(g)(a)) \\ &= C[\pi\gamma_1\mathcal{A}][\pi L(g)(a)] \\ &= C[\gamma_2\mathcal{A}][L(g)(\lambda(a))] \\ &= L^{\gamma_2\mathcal{A}}(g)(\lambda(a)). \end{aligned}$$

It follows that $L^{\gamma_1\mathcal{A}}(g)$ and $L^{\gamma_2\mathcal{A}}(g)$ are isomorphic and so $C[\gamma_1\mathcal{A}](g) = \Sigma(g)(L^{\gamma_1\mathcal{A}}(g)) = \Sigma(g)(L^{\gamma_2\mathcal{A}}(g)) = C[\gamma_2\mathcal{A}](g)$. □

For the remainder of this subsection we fix a P-uniform family of transparent symmetric rank circuits $C = (C_n)_{n\in\mathbb{N}}$ over a vocabulary $\rho$. By Corollary 4.12 there are constants $n_0$ and $k$ such that for all $n > n_0$, we have $\mathbf{supp\text{-}size}(C_n) \leq k$. Fix some $n > \max(n_0, 3k)$ and $\mathcal{A} \in \mathrm{fin}[\rho, n]$. Let $C := \langle G, \Omega, \Sigma, \Lambda, L\rangle := C_n$.

For each $g \in G$ let $\Gamma_g := \{\gamma \in [n]^{\underline{A}} : C[\gamma\mathcal{A}](g) = 1\}$ and let $\mathrm{EV}_g := \{\eta \in A^{\underline{\mathrm{sp}(g)}} : \exists \gamma \in \Gamma_g, \eta \sim \gamma^{-1}\}$. In other words, $\Gamma_g$ is the set of bijections for which $g$ evaluates to 1 and $\mathrm{EV}_g$ is the set of assignments to the support of $g$ that can be extended to a bijection for which $g$ evaluates to 1. It follows from Lemma 7.2 that $\Gamma_g$ is entirely determined by $\mathrm{EV}_g$. From Theorem 4.14 the domain of each $\eta \in \mathrm{EV}_g$ has cardinality at most $k$. In this sense $\mathrm{EV}_g$ can be thought of as a succinct encoding of $\Gamma_g$.

We aim to define the set $\mathrm{EV}_g$ for each gate $g$ in $C$ by induction on the structure of the circuit. This definition depends on the basis function labelling $g$ and has already been given in [1] for gates labelled by AND, OR, NOT, or MAJ functions. We therefore restrict our attention to the case where $g$ is lablelled by a RANK function.

For the remainder of this subsection we fix a gate $g \in G$ labelled by a rank function $\mathrm{RANK}^t_p[X]$ with $X = \uplus_{s\in[3]} X_s$. We also fix some $\eta \in A^{\underline{\mathrm{sp}(g)}}$.

We now introduce some notation. For $\gamma \in [n]^{\underline{A}}$ we write $L^\gamma$ to abbreviate $L^{\gamma\mathcal{A}}(g)$. For each $x \in \mathrm{unv}(g)$ we abbreviate notation and let $\mathrm{sp}(x) := \mathrm{sp}_g(x)$, $\mathbf{Orb}(x) := \mathbf{Orb}_{\mathbf{Stab}(\mathrm{sp}(g))}(x)$, and $\mathbf{Stab}(x) := \mathbf{Stab}_{\mathbf{Stab}(\mathrm{sp}(g))}(x)$. For each $h \in H_g$ let $A^h := \{\alpha \in A^{\underline{\mathrm{sp}(h)}} : \eta \sim \alpha\}$ be the set of assignments to the support of $h$ that are compatible with $\eta$. For each $x \in \mathrm{unv}(g)$ let $A^x := \{\alpha \in A^{\underline{\mathrm{sp}(x)}} : \eta \sim \alpha\}$ be the set of assignments to the support of $x$ that are compatible with $\eta$.

In Section 7.2 we encode each circuit as a structure over a linearly ordered universe, which then induces an order on the universe of each gate. As such, in this section we suppose for each $s \in [3]$ that there is some linear order on $X_s$. This allows us to choose a representative from each orbit. So, for each $s \in [3]$ let $\mathrm{minorb}(s) := \{\min(\mathbf{Orb}(x)) : x \in X_s\}$ and let $I^{g,\eta}_s := \{(x, \alpha) : x \in \mathrm{minorb}(s), \alpha \in A^x\}$. Let $I^{g,\eta} := \uplus_{s\in[3]} I^{g,\eta}_s$. For each $s \in [3]$ we think of each $x \in \mathrm{minorb}(s)$ as denoting an orbit in $X_s$ and each assignment in $A^x$ as encoding the action of a permutation in $\mathbf{Stab}(g)$ on $\mathrm{sp}(x)$, which determines an element of that orbit. In this way we think of each element in $I^{g,\eta}_s$ as encoding an element in $X_s$.

We next define a three sorted matrix $M^{g,\eta} : \tau_{\mathrm{mat}}[I^{g,\eta}] \to \{0, 1\}$ and show that $\mathbf{rk}_p(M^{g,\eta}) = \mathbf{rk}_p(L^\gamma)$. We do so by first defining a mapping that takes $((x_1, \alpha_1), (x_2, \alpha_2), (x_3, \alpha_3)) \in \tau_{\mathrm{mat}}[I^{g,\eta}]$ to a triple $(\sigma_1, \sigma_2, \sigma_3)$ of permutations in $\mathbf{Stab}(g)$ with the property that the assignments defined by taking the action of each $\sigma_i$ on $\alpha_i$ are pairwise compatible. Let $\epsilon$ be the injection formed by combining these assignments and $h := L(g)(\sigma_1(x_1), \sigma_2(x_2), \sigma_3(x_3))$. We then define $M^{g,\eta}$ by letting $M^{g,\eta}((x_1, \alpha_1), (x_2, \alpha_2), (x_3, \alpha_3)) = 1$ if, and only if, $\epsilon \in \mathrm{EV}_h$. We now complete this definition formally.

As an intermediary step we first define a function $\bar{f}^{g,\eta}$ that maps each $((x_1, \alpha_1), (x_2, \alpha_2), (x_3, \alpha_3)) \in \tau_{\mathrm{mat}}[I^{g,\eta}]$ to a sequence of injections $(\bar\sigma_1, \bar\sigma_2, \bar\sigma_3)$ defined as follows. Let $B \subseteq [n] \setminus \mathrm{sp}(g)$ have size $3k$ and let $B := B_1 \cup B_2 \cup B_3$ be a partition of $B$ into equal size pieces. For each $i \in [3]$ let $f_i : \mathbf{Dom}(\alpha_i) \setminus \mathrm{sp}(g) \to B_i$ be an injection. In the following definition it is useful to recall that $\mathrm{sp}(g) \subseteq \mathrm{sp}_g(x_i)$ for each $i \in [3]$.

- Let $\bar\sigma_1 : \mathbf{Dom}(\alpha_1) \to [n]$ be defined for $u \in \mathbf{Dom}(\alpha_1)$ such that if $u \in \mathrm{sp}(g)$ then $\bar\sigma_1(u) = u$ and otherwise $\bar\sigma_1(u) = f_1(u)$.
- Let $\bar\sigma_2 : \mathbf{Dom}(\alpha_2) \to [n]$ be defined for $u \in \mathbf{Dom}(\alpha_2)$ such that if $u \in \mathbf{Dom}(\alpha_1)$ then $\bar\sigma_2(u) = \bar\sigma_1(u)$ and otherwise $\bar\sigma_2(u) = f_2(u)$.
- Let $\bar\sigma_3 : \mathbf{Dom}(\alpha_3) \to [n]$ be defined for $u \in \mathbf{Dom}(\alpha_3)$ such that if $u \in \mathbf{Dom}(\alpha_1)$ then $\bar\sigma_3(u) = \bar\sigma_1(u)$ and if $u \in \mathbf{Dom}(\alpha_2)$ then $\bar\sigma_3(u) = \bar\sigma_2(u)$ and otherwise $\bar\sigma_3(u) = f_3(u)$.

We now make a few observations. We first note that each $\bar\sigma_i$ is an injection. We abuse notation slightly and write $\alpha_i \circ \bar\sigma_i^{-1}$ to denote the injection $\alpha_i \circ \bar\sigma_i^{-1} : \mathbf{Img}(\bar\sigma_i) \to A$ defined by $\alpha_i \circ \bar\sigma_i^{-1}(u) = \alpha_i(\bar\sigma_i^{-1}(u))$ for all $u \in \mathbf{Img}(\bar\sigma_i)$. We think of each $\alpha_i \circ \bar\sigma_i^{-1}$ as being a version of $\alpha_i$ with the domain shifted by $\bar\sigma_i$. It can be shown that the three functions $\alpha_i \circ \bar\sigma_i^{-1}$ for $i \in [3]$ are pairwise compatible. Lastly, note that $\mathrm{sp}(g) \subseteq \mathbf{Dom}(\bar\sigma_i)$ and for each $u \in \mathrm{sp}(g)$ we have $\bar\sigma_i(u) = u$. We summarise these assertions in the following lemma.

Lemma 7.3. *Let $z := ((x_1, \alpha_1), (x_2, \alpha_2), (x_3, \alpha_3)) \in \tau_{mat}[I^{g,\nu}]$ and let $(\bar{\sigma}_1, \bar{\sigma}_2, \bar{\sigma}_3) := \bar{J}^{g,\eta}(z)$. Then for all $j \in [3]$*

(1) *for all $u \in sp(g)$, $\bar{\sigma}_j(u) = u$,*
(2) *$\bar{\sigma}_j$ is an injection, and*
(3) *for all $j' \in [3]$, $\alpha_j \circ \bar{\sigma}_j^{-1}$ and $\alpha_{j'} \circ \bar{\sigma}_{j'}^{-1}$ are compatible.*

Let $J^{g,\eta} : \tau_{\text{mat}}[I^{g,\eta}] \to \{(h, \epsilon) : h \in H_g, \epsilon \in A^h\}$ be defined for $z := ((x_1, \alpha_1), (x_2, \alpha_2), (x_3, \alpha_3)) \in \tau_{\text{mat}}[I^{g,\eta}]$ as follows. Let $(\bar{\sigma}_1, \bar{\sigma}_2, \bar{\sigma}_3) := \bar{J}^{g,\eta}(z)$. For each $i \in [3]$ let $\sigma_i \in \mathbf{Stab}(\text{sp}(g))$ be such that for all $u \in \text{sp}(x_i)$, $\sigma_i(u) = \bar{\sigma}_i(u)$. Let $h := L(g)((\sigma_1(x_1), \sigma_2(x_2), \sigma_3(x_3))$ and let $\epsilon := (\alpha_1 \circ \bar{\sigma}_1^{-1} | \alpha_2 \circ \bar{\sigma}_2^{-1} | \alpha_3 \circ \bar{\sigma}_3^{-1})\big|_{\text{sp}(h)}$. It follows from Lemma 7.3 that $\epsilon$ is well defined. Let $J^{g,\eta}(z) := (h, \epsilon)$.

Let $M^{g,\eta} : \tau_{\text{mat}}[I^{g,\eta}] \to \{0, 1\}$ be defined for $z := ((x_1, \alpha_1), (x_2, \alpha_2), (x_3, \alpha_3)) \in \tau[I^{g,\eta}]$ such that if $(h, \epsilon) := J^{g,\eta}(z)$ then $M^{g,\eta}(z) = 1$ if, and only if, $\epsilon \in \text{EV}_h$.

We stated previously that we can think of each assignment to the support of a child or element of the universe of $g$ as encoding an element in its orbit. We now formalise this and show that for each $\gamma \in [n]^{\underline{A}}$ with $\gamma^{-1} \sim \eta$ and each $h \in H_g$ there is a natural surjective map defined by $\gamma$ from $A^h$ to $\mathbf{Orb}(h)$ and for each $x \in \text{unv}(g)$ there is a similarly defined surjection from $A^x$ to $\mathbf{Orb}(x)$. Let $\gamma \in [n]^{\underline{A}}$ be such that $\gamma^{-1} \sim \eta$. Let $h \in H_g$. For each $\epsilon \in A^h$ let $\Pi^\gamma_\epsilon$ be any permutation in $\mathbf{Stab}(\text{sp}(g))$ such that $\Pi^\gamma_\epsilon(u) = \gamma(\epsilon(u))$ for all $u \in \text{sp}(h)$. The action of $\Pi^\gamma_\epsilon$ on $h$ is defined independently of the choice of permutation, and so the function $\epsilon \mapsto \Pi^\gamma_\epsilon(h)$ is well defined. Let $x \in X$. For each $\alpha \in A^x$ let $\Pi^\gamma_\alpha$ be any permutation in $\mathbf{Stab}(\text{sp}(g))$ such that $\Pi^\gamma_\alpha(u) = \gamma(\alpha(u))$ for all $u \in \text{sp}(x)$. It follows similarly that the action of $\Pi^\gamma_\alpha$ on $x$ is defined independently of the choice of permutation and so the function $\alpha \mapsto \Pi^\gamma_\alpha(x)$ is well defined. It is easy to show that both of these maps are surjective.

We now define a function from $M^{g,\eta}$ to $L^\gamma$ (as $\tau_{\text{mat}}$-structures). Let $\gamma \in [n]^{\underline{A}}$ be such that $\gamma^{-1} \sim \eta$. For each $s \in [3]$ let $P^\gamma_s : I^{g,\eta}_s \to X_s$ be defined by $P^\gamma_s(x, \alpha) := \Pi^\gamma_\alpha(x)$ for all $(x, \alpha) \in I^{g,\eta}_s$. Let $P^\gamma = \uplus_{s \in S} P^\gamma_s$. We aim to show that $P^\gamma$ defines an epimorphism from $M^{g,\eta}$ to $L^{\gamma\mathcal{A}}(g)$. We first establish a correspondence between $\text{EV}_h$ and those elements of the orbit of $h$ that evaluate to 1.

Lemma 7.4. *Let $\gamma \in [n]^{\underline{A}}$ be such that $\gamma^{-1} \sim \eta$ and $h \in H_g$. Then for all $\epsilon \in A^h$, $\epsilon \in EV_h$ if, and only if, $C[\gamma\mathcal{A}](\Pi^\gamma_\epsilon(h)) = 1$.*

Proof. From the definition of $\text{EV}_h$ it follows that $\epsilon \in \text{EV}_h$ if, and only if, there exists $\delta \in [n]^{\underline{A}}$ such that $\delta^{-1} \sim \eta$, $\delta^{-1} \sim \epsilon$, and $C[\delta\mathcal{A}](h) = 1$. Let $\pi := \gamma \circ \delta^{-1}$. Then $\pi \in \mathbf{Stab}(\text{sp}(g))$ and $\pi \circ \delta = \gamma$. For $a \in \text{sp}(h)$ we have $\gamma(\epsilon(a)) = \pi(\delta(\epsilon(a))) = \pi(a)$. The second equality follows from the fact that $\delta^{-1} \sim \epsilon$ and so $\delta(\epsilon(a)) = a$. Then for all $a \in \text{sp}(h)$ we have $\Pi^\gamma_\epsilon(a) = \gamma(\epsilon(a)) = \pi(a)$, and so $\Pi^\gamma_\epsilon(h) = \pi(h)$. Thus

$$\epsilon \in \text{EV}_h \iff C[\delta\mathcal{A}](h) = 1 \iff C[\pi \circ \delta\mathcal{A}](\pi(h)) = 1$$
$$\iff C[\gamma\mathcal{A}](\pi(h)) = 1 \iff C[\gamma\mathcal{A}](\Pi^\gamma_\epsilon(h)) = 1.$$

□

The function $J^{g,\eta}$ maps triples of the form $((x_1, \alpha_1), (x_2, \alpha_2), (x_3, \alpha_3))$ to pairs of the form $(h, \epsilon)$. We now show that $\Pi^\gamma_\epsilon(h)$ is exactly the gate defined by mapping each $x_i$ to $\Pi^\gamma_{\alpha_i}(x_i)$.

Lemma 7.5. *Let $\gamma \in [n]^{\underline{A}}$ be such that $\gamma^{-1} \sim \eta$. Let $((x_1, \alpha_1), (x_2, \alpha_2), (x_3, \alpha_3)) \in \text{ind}(M^{g,\eta})$ and $(h, \epsilon) := J^{g,\eta}(((x_1, \alpha_1), (x_2, \alpha_2), (x_3, \alpha_3)))$. Then $\Pi^\gamma_\epsilon(h) = L(g)(\Pi^\gamma_{\alpha_1}(x_1), \Pi^\gamma_{\alpha_2}(x_2), \Pi^\gamma_{\alpha_3}(x_3))$.*

Proof. Let $\sigma_1, \sigma_2, \sigma_3 \in \mathbf{Stab}(\mathrm{sp}(g))$ be the permutations introduced in the definition of $J^{g,\eta}$. Let $i \in [3]$ and let $u \in \mathrm{sp}(x_i)$. Then $\sigma_i(u) \in \mathrm{sp}(\sigma(x_i)) \subseteq \mathrm{sp}(h) \cup \mathrm{sp}(g)$. The first of these follows from Lemma 4.4. Suppose $\sigma_i(u) \in \mathrm{sp}(g)$. Then since $\sigma_i \in \mathbf{Stab}(\mathrm{sp}(g))$ we have $\Pi^\gamma_\epsilon(\sigma_i(u)) = u = \Pi^\gamma_{\alpha_i}(u)$. Suppose instead that $\sigma_i(u) \in \mathrm{sp}(h)$. Then $\Pi^\gamma_\epsilon(\sigma_i(u)) = \gamma(\epsilon(\sigma_i(u))) = \gamma(\alpha_i(\bar{\sigma}_i^{-1}(\sigma_i(u)))) = \gamma(\alpha_i(u)) = \Pi^\gamma_{\alpha_i}(u)$. In either case we can conclude that $\Pi^\gamma_\epsilon(\sigma_i(x_i)) = \Pi^\gamma_{\alpha_i}(x_i)$, and so

$$\begin{aligned}\Pi^\gamma_\epsilon(h) &= \Pi^\gamma_\epsilon(L(g)(\sigma_1(x_1), \sigma_2(x_2), \sigma_3(x_3)) \\ &= L(g)(\Pi^\gamma_\epsilon(\sigma_1(x_1)), \Pi^\gamma_\epsilon(\sigma_2(x_2)), \Pi^\gamma_\epsilon(\sigma_3(x_3))) \\ &= L(g)(\Pi^\gamma_{\alpha_1}(x_1), \Pi^\gamma_{\alpha_2}(x_2), \Pi^\gamma_{\alpha_3}(x_3))\end{aligned}$$

□

We now show that $P^\gamma$ is a homomorphism from $M^{g,\eta}$ to $L^\gamma$.

Lemma 7.6. *Let $((x_1, \alpha_1), (x_2, \alpha_2), (x_3, \alpha_3)) \in \tau_{mat}[I^{g,\eta}]$. Let $\gamma \in [n]^{\underline{A}}$ be such that $\gamma^{-1} \sim \eta$. Then $M^{g,\eta}(((x_1, \alpha_1), (x_2, \alpha_2), (x_3, \alpha_3))) = L^\gamma(P^\gamma(x_1, \alpha_1), P^\gamma(x_2, \alpha_2), P^\gamma(x_3, \alpha_3))$.*

Proof. Let $(h, \epsilon) := J^{g,\eta}(z)$. Then

$$\begin{aligned}M^{g,\eta}(z) = 1 &\iff \epsilon \in \mathrm{EV}_h \\ &\iff C[\gamma\mathcal{A}](\Pi^\gamma_\epsilon(h)) = 1 \\ &\iff C[\gamma\mathcal{A}](L(g)(\Pi^\gamma_{\alpha_1}(x_1), \Pi^\gamma_{\alpha_2}(x_2), \Pi^\gamma_{\alpha_3}(x_3)) = 1 \\ &\iff L^\gamma(P^\gamma(x_1, \alpha_1), P^\gamma(x_2, \alpha_2), P^\gamma(x_3, \alpha_3)) = 1.\end{aligned}$$

The second equivalence follows from Lemma 7.4 and the third equivalence follows from Lemma 7.5. □

Lemma 7.7. *If $\gamma \in [n]^{\underline{A}}$ is such that $\gamma^{-1} \sim \eta$ then $P^\gamma$ is surjective.*

Proof. Let $x \in X$. Let $y = \min(\mathbf{Orb}(x))$. There exists $\sigma \in \mathbf{Stab}(\mathrm{sp}(g))$ such that $\sigma(y) = x$. Let $\alpha := \gamma^{-1} \circ \sigma\big|_{\mathrm{sp}(y)}$. Then for each $u \in \mathrm{sp}(y) \cap \mathrm{sp}(g)$ we have $\alpha(u) = \gamma^{-1} \circ \sigma\big|_{\mathrm{sp}(y)}(u) = \gamma^{-1}(u) = \eta(u)$, and so $\alpha \in A^y$. For each $u \in \mathrm{sp}(y)$ we have $\Pi^\gamma_\alpha(u) = \gamma(\alpha(u)) = \gamma(\gamma^{-1}\sigma\big|_{\mathrm{sp}(y)}(u)) = \sigma(u)$. It follows from Lemma 4.5 that $P^\gamma(y, \alpha) = \Pi^\gamma_\alpha(y) = \sigma(y) = x$. □

We have from Lemmas 7.6 and 7.7 that $P^\gamma$ defines an epimorphism from $M^{g,\eta}$ to $L^\gamma$. But $P^\gamma$ is not in general an isomorphism. The following result provides a useful characterisation of preimages under $P^\gamma$.

Lemma 7.8. *For all $(x, \alpha), (y, \beta) \in I_3^{g,\eta}$, $P(x, \alpha) = P(y, \beta)$ if, and only if, $x = y$ and there exists $\pi \in \boldsymbol{Stab}(x)$ such that $\alpha(u) = \beta \circ \pi(u)$ for all $u \in sp(x)$.*

Proof. $\Rightarrow$: Suppose $P(x, \alpha) = P(y, \beta)$. Let $\sigma := (\Pi^\gamma_\beta)^{-1} \circ \Pi^\gamma_\alpha$. Then $\sigma(x) = (\Pi^\gamma_\beta)^{-1} \circ \Pi^\gamma_\alpha(x) = y$. But then $x$ and $y$ are in the same orbit and are both minimum elements of that orbit. It follows that $x = y$ and so $\sigma \in \mathbf{Stab}(x)$. Moreover, for all $u \in \mathrm{sp}(x)$, $\gamma(\alpha(u)) = \Pi^\gamma_\alpha(u) = \Pi^\gamma_\beta(\sigma(u)) = \gamma(\beta(\sigma(u)))$. Since $\gamma$ is a bijection it follows that $\alpha(u) = \beta \circ \sigma(u)$ for all $u \in \mathrm{sp}(x)$.

$\Leftarrow$: Let $\sigma \in \mathbf{Stab}(x)$ such that $\alpha(u) = \beta \circ \sigma(u)$ for all $u \in \mathrm{sp}(x)$. Then $\Pi^\gamma_\beta(\sigma(u)) = \gamma(\beta(\sigma(u))) = \gamma(\alpha(u)) = \Pi^\gamma_\alpha(u)$ and so $\Pi^\gamma_\beta(x) = \Pi^\gamma_\beta(\sigma(x)) = \Pi^\gamma_\alpha(x) = \Pi^\gamma_\alpha(y)$. □

We could informally think of $M^{g,\eta}$ as containing numerous copies of the entries in $L^\gamma$. On this basis it might be conjectured then that the ranks of the associated matrices are equal, as repeated rows and columns in $M^{g,\eta}$ would have

no effect on the rank. This is not the case as the matrix associated with $M^{g,\eta}$ is defined by summing over the third sort, and so repeated entries affect the values in the matrix. We now define a new matrix from $M^{g,\eta}$ that corrects for these duplicates.

Let $M^{g,\eta}_* : I^{g,\eta}_1 \times I^{g,\eta}_2 \to \mathbb{N}_0$ be defined for $(x_1, \alpha_1), (x_2, \alpha_2) \in I^{g,\eta}_1 \times I^{g,\eta}_2$ such that

$$M^{g,\eta}_*((x_1, \alpha_1), (x_2, \alpha_2)) = \sum_{(x_3,\alpha_3)\in I^{g,\eta}_3} \frac{M^{g,\eta}((x_1, \alpha_1), (x_2, \alpha_2), (x_3, \alpha_3))}{|\mathbf{Orb}_{\mathbf{Stab}(x_3)}(\alpha_3)|}. \tag{1}$$

Let $L^Y_* : X_1 \times X_2 \to \mathbb{N}_0$ be the matrix associated with $L^Y$. We now show that $P^Y$ is a homomorphic mapping from $M^{g,\eta}_*$ to $L^Y_*$.

Lemma 7.9. *Let $\gamma \in [n]^{\underline{A}}$ be such that $\gamma^{-1} \sim \eta$. Then for all $((x_1, \alpha_1), (x_2, \alpha_2)) \in I^{g,\eta}_1 \times I^{g,\eta}_2$ we have $M^{g,\eta}_*((x_1, \alpha_1), (x_2, \alpha_2)) = L^Y_*(P^Y(x_1, \alpha_1), P^Y(x_2, \alpha_2))$.*

Proof. Let $((x_1, \alpha_1), (x_2, \alpha_2)) \in I^{g,\eta}_1 \times I^{g,\eta}_2$. Then

$$\begin{aligned}
L^Y_*(P^Y(x_1, \alpha_1), P^Y(x_2, \alpha_2)) &= |\{w \in X_3 : L^Y(P^Y(x_1, \alpha_1), P^Y(x_2, \alpha_2), w) = 1\}| \\
&= \sum_{w\in X_3} L^Y(P^Y(x_1, \alpha_1), P^Y(x_2, \alpha_2), w) \\
&= \sum_{(x_3,\alpha_3)\in I^{g,\eta}_3} \frac{L^Y(P^Y(x_1, \alpha_1), P^Y(x_2, \alpha_2), P^Y(x_3, \alpha_3))}{|(P^Y)^{-1}[P^Y(x_3, \alpha_3)]|} \\
&= \sum_{(x_3,\alpha_3)\in I^{g,\eta}_3} \frac{M^{g,\eta}((x_1, \alpha_1), (x_2, \alpha_2), (x_3, \alpha_3))}{|(P^Y)^{-1}[P^Y(x_3, \alpha_3)]|}
\end{aligned}$$

The third equivalence follows from Lemma 7.7 and the fourth equivalence follows from Lemma 7.6. It follows from Lemma 7.8 that

$$\begin{aligned}
|(P^Y)^{-1}[P^Y(x_3, \alpha_3)]| &= |\{(z, \delta) : P^Y(z, \delta) = P^Y(x_3, \alpha_3)\}| \\
&= |\{(z, \delta) : z = x_3 \wedge \exists \sigma \in \mathbf{Stab}(x_3) \forall u \in \mathrm{sp}(x)\, \alpha_3(u) = \delta \circ \sigma(u)\}| \\
&= |\mathbf{Orb}_{\mathbf{Stab}(x_3)}(\alpha_3)|,
\end{aligned}$$

from which the result follows. □

We now show that for any prime $p$, $\mathbf{rk}_p(M^{g,\eta}_*) = \mathbf{rk}_p(L^Y_*)$. We recall that the rank of a matrix is equal to the maximal order of a non-zero minor. We prove this result by establishing a correspondence between non-zero minors of these two matrices.

Theorem 7.10. *Let $\gamma \in A^{\underline{n}}$ be such that $\gamma^{-1} \sim \eta$ and let $p \in \mathbb{N}$ be prime. Then $\boldsymbol{rk}_p(M^{g,\eta}_*) = \boldsymbol{rk}_p(L^Y_*)$.*

Proof. It follows from Lemma 7.9 that $L^Y_*$ appears as a submatrix of $M^{g,\eta}_*$. Then for each $r \in \mathbb{N}_0$ if $L^Y_*$ has a non-zero $r \times r$ minor then so does $M^{g,\eta}_*$ and so $\mathbf{rk}_p(L^Y_*) \leq \mathbf{rk}_p(M^{g,\eta}_*)$.

Let $r = \mathbf{rk}_p(M^{g,\eta}_*)$ and let $S$ be an $r \times r$ submatrix of $M^{g,\eta}_*$ with non-zero determinant. Let $R \subset I^{g,\eta}_1$ and $C \subset I^{g,\eta}_2$ be sets indexing the rows and columns of $S$, respectively. Suppose there exist distinct $(x, \alpha), (y, \beta) \in R$ such that $P^Y(x, \alpha) = P^Y(y, \beta)$. Then from Lemma 7.9 we have for all $(z, \delta) \in C$ that

$$S((x, \alpha), (z, \delta)) = L^Y_*(P^Y(x, \alpha), P^Y(z, \delta)) = L^Y_*(P^Y(y, \beta), P^Y(z, \delta)) = S((y, \beta), (z, \delta)).$$

Then $S$ has two identical rows and so $\det(S) = 0$, a contradiction. It follows from this and from Lemma 7.9 that $P^Y$ embeds $S$ as a submatrix in $L_*^Y$, and so $\mathbf{rk}_p(L_*^Y) \geq \mathbf{rk}_p(M_*^{g,\eta})$. □

### 7.2 Constructing an FPR-Formula

Let $C := (C_n)_{n\in\mathbb{N}}$ be a P-uniform family of transparent symmetric rank circuits. From Lemma 5.5 we may assume without a loss of generality that each $C_n$ is reduced and injective.

Let $T := \{\text{AND}, \text{OR}, \text{NOT}, \text{RANK}\} \cup \rho \cup \{0, 1\}$. It follows from the Immerman-Vardi theorem and the P-uniformity of $C$ that there is an $\text{FP}^{\mathbb{N}}$-interpretation

$$\Phi := (\phi_G, \phi_\Omega, (\phi_{\Sigma,s})_{s\in T}, \phi_{\text{RANK}}, (\phi_{\Lambda_R})_{R\in\rho}, \phi_W, \phi_L)$$

such that for each $n \in \mathbb{N}$ when $\Phi$ is interpreted in a $\rho$-structure $\mathcal{A}$ of size $n$ it defines $C_n$ in the number domain. We abuse notation and also refer to this equivalent circuit as $C_n$. Let $t$ be the width of this interpretation. Throughout this subsection we use $\mu$, $\nu$, $\epsilon$, $\eta$, and $\delta$ to denote $t$-length sequences of number variables and $\kappa$ and $\pi$ to denote individual number variables. We now describe the formulas in $\Phi$ by describing relation that each formula defines when interpreted in a $\rho$-structure $\mathcal{A}$ of size $n$.

- $\phi_G(\mu)$ defines the set of gates in the circuit $C_n$ as a subset of $[n]^t$. We identify this set with $G_n$, and hence write $G_n \subseteq [n]^t$.
- $\phi_\Omega(\kappa_1, \ldots, \kappa_q, \mu)$ is defined such that $\mathcal{A} \models \phi_\Omega[a_1, \ldots, a_q, g]$ if, and only if, $g$ is a gate, $(a_1, \ldots, a_q) \in [n]^q$, and $\Omega_n(a_1, \ldots, a_q) = g$.
- $\phi_{\Sigma,s}(\mu)$ is defined for $s \in T$ such that $\mathcal{A} \models \phi_{\Sigma,s}[g]$ if, and only if, $g$ is an input gate and $\Sigma_n(g) = s$ or $g$ is an internal gate and $\Sigma_n(g)$ is a function of type $s$.
- $\phi_{\text{RANK}}(\mu, \eta_1, \eta_2, \delta_1, \delta_2, \delta_3)$ is defined such that $\mathcal{A} \models \phi_{\text{RANK}}[g, p, t, d_1, d_2, d_3]$ if, and only if, $\Sigma(g) = \text{RANK}_p^t[d_1, d_2, d_3]$.
- Let $R \in \rho$. Then $\phi_{\Lambda_R}(\mu, \delta_1, \ldots, \delta_{r_R})$ is such that $\mathcal{A} \models \phi_{\Lambda_R}[g, a_1, \ldots, a_{r_R}]$ if, and only if, $(a_1, \ldots, a_{r_R}) \in [n]^{r_R}$ and $g$ is a relational gate such that $\Sigma(g) = R$ and $(\Lambda_n)_R(g) = (a_1, \ldots, a_{r_R})$.
- $\phi_W(\mu, \nu)$ is defined such that $\mathcal{A} \models \phi_W[g, h]$ if, and only if, $g$ and $h$ are gates and $h \in H_g$.
- $\phi_L(\mu, \nu, \delta_1, \delta_2, \delta_3)$ is defined such that $\mathcal{A} \models \phi_L[g, h, a_1, a_2, a_3]$ if, and only if, $g$ is a rank gate, $h \in H_g$, and $L_n(g)((a_1, a_2, a_3)) = h$.

The interpretation $\Phi$ does not define a circuit exactly how we might expect. In particular, it does not contain formulas corresponding to $\Sigma_n$ or $L_n$. The formula $\phi_L$ only defines $L_n$ for rank gates and $\phi_{\Sigma,s}$ does not define $\Sigma_n$ but rather determines to which family of functions each $\Sigma_n(g)$ belongs. These formulas together suffice to define the circuit.

By Corollary 4.12 there are constants $n_0$ and $k$ such that for all $n > n_0$, we have $\textbf{supp-size}(C_n) \leq k$. Notice that for each $n \leq \max(n_0, 3k)$, there are constantly many bijections from the universe of a $\rho$-structure $\mathcal{A}$ of size $n$ to $[n]$. It follows there exists an FPR-formula that evaluates $C_n$ for any $n \leq \max(n_0, 3k)$ by explicitly quantifying over all of these constantly many bijections, and then evaluating the circuit with respect to each bijection. For the rest of this section we fix an $n > \max(n_0, 3k)$ and let $\mathcal{A}$ denote a $\rho$-structure of size $n$.

The gates in a circuit and elements of the universe of a gate are encoded as $t$-length sequences of number variables. In the remainder of this section we use $\mu$ and $\nu$ to denote gates and $\delta$ and $\epsilon$ to denote elements of the universe of a gate. We use $\kappa$, $\pi$ and $\lambda$ to denote single number variables and $\vec{\kappa}$ to denote a $2k$-length tuple of number variables. We use $\vec{x}$ and $\vec{y}$ to denote $k$-length sequences of element variables and use $\vec{z}$ to denote $2k$-length sequences of element variables. We use $U$ and $V$ to denote second-order variables. If $S$ is a subset of an ordered set we write $\vec{S}$ to denote the $|S|$-tuple

given by listing the elements of $S$ in order. Let $X$ and $Y$ be sets, $\vec{a}$ be a sequence in $X$, and $\vec{u}$ be a sequence of distinct elements in $Y$ such that $|\vec{u}| \leq |\vec{a}|$. Let $\alpha^{\vec{u}}_{\vec{a}} : \mathbf{Img}(\vec{u}) \to X$ be such that $\alpha^{\vec{u}}_{\vec{a}}(b) := \vec{a} \circ \vec{u}^{-1}(b)$ for all $b \in \mathbf{Img}(u)$.

We aim to give a recursive definition of $\mathrm{EV}_g$ in FPR. We have from Theorem 4.14 that $\mathrm{sp}(g)$ has size at most $k$, but it may not be exactly equal to $k$. If $|\mathrm{sp}(g)| = \ell$, we define

$$\overline{\mathrm{EV}}_g := \{(a_1, \ldots, a_k) \in [n]^k : \alpha^{(a_1,\ldots,a_\ell)}_{\vec{\mathrm{sp}}(g)} \in \mathrm{EV}_g \text{ and } \forall i, j \in [k]\ (i \neq j \implies a_i \neq a_j)\}.$$

More formally then, we aim to define an FPR-formula $\theta(\mu, \vec{x})$ such that $\mathcal{A} \models \theta[g, \vec{a}]$ if, and only if, $\vec{a} \in \overline{\mathrm{EV}}_g$. We do so by defining for each $s \in T$ a formula $\theta_s$ that gives a recursive definition of $\theta$ for any gate associated with the symbol $s$. That is, for a second-order variable $V$ with the same type as $(\mu, \vec{x})$ we define for each $s \in T$ a formula $\theta_s$ so that for each gate $g$ with $\mathcal{A} \models \phi_{\Sigma,s}[g]$ and each $\vec{a} \in A^k$, if $V$ is mapped to a relation $\beta(V)$ such that for all $h \in H_g$, $(h, \vec{b}) \in \beta(V)$, if, and only if, $\vec{b} \in \overline{\mathrm{EV}}_h$, then $\mathcal{A} \models \theta_s[g, \vec{a}; \beta(V)]$ if, and only if, $\vec{a} \in \overline{\mathrm{EV}}_g$.

Anderson and Dawar [1] have already defined $\theta_s$ for each $s \in T \setminus \{\mathrm{RANK}\}$. In each of these cases the definition of $\theta_s$ is very straightforward, and so we reproduce these formulas below with minimal discussion and minor changes for the sake of presentation.

We first define a few auxiliary formulas. These appear in [1], and we reproduce these definitions with minor modification. We have from the Immerman-Vardi theorem and Lemma 5.8 that there exists a $\mathrm{FP}^{\mathbb{N}}$-formula $\textsc{supp}(\mu, \kappa)$ such that $\mathcal{A} \models \textsc{supp}[g, u]$ if, and only if, $g$ is a gate and $u \in \mathrm{sp}(g)$. We can define from $\textsc{supp}$ a formula $\textsc{supp}_i$ for each $i \in \mathbb{N}$ such that $\mathcal{A} \models \textsc{supp}_i[g, u]$ if, and only if, $u$ is the $i$th element of $\mathrm{sp}(g)$. We define these formulas by induction as follows

$$\begin{aligned}
\textsc{supp}_1(\mu, \kappa) &:\equiv \textsc{supp}(\mu, \kappa) \wedge (\forall \pi(\pi < \kappa) \implies \neg\textsc{supp}(\mu, \pi)) \\
\textsc{supp}_{i+1}(\mu, \kappa) &:\equiv \textsc{supp}(\mu, \kappa) \wedge \exists \pi_1(\pi_1 < \kappa \wedge \textsc{supp}_i(\mu, \pi_1) \\
&\qquad \wedge \forall \pi_2((\pi_1 < \pi_2 < \kappa) \implies \neg\textsc{supp}(\mu, \pi_2))).
\end{aligned}$$

We can define from these formulas a formula $\textsc{agree}(\mu, \nu, \vec{x}, \vec{y})$ such that $\mathcal{A} \models \textsc{agree}(g, h, \vec{a}, \vec{b})$ if, and only if, $g$ and $h$ are gates, $h \in H_g$, and $\alpha^{\vec{a}}_{\vec{\mathrm{sp}}(g)} \sim \alpha^{\vec{b}}_{\vec{\mathrm{sp}}(h)}$. This formula is defined as follows

$$\begin{aligned}
\textsc{agree}(\mu, \nu, \vec{x}, \vec{y}) :\equiv \phi_G(\mu) \wedge \phi_G(\nu) \wedge \bigwedge_{1 \leq e, d \leq [k]} &[\forall \delta\, (\textsc{supp}_e(\mu, \delta) \wedge \textsc{supp}_d(\nu, \delta)) \implies x_e = y_d) \wedge \\
&\forall \delta_1, \delta_2\, ((\textsc{supp}_e(\mu, \delta_1) \wedge \textsc{supp}_d(\mu, \delta_2) \wedge x_e = x_d) \implies \delta_1 = \delta_2)].
\end{aligned}$$

We define $\theta_s$ for each $s \in \backslash\{\text{RANK}\}$ as follows [1]

$$\theta_0(\mu, \vec{x}) :\equiv \exists y\,(y \neq y)$$

$$\theta_1(\mu, \vec{x}) :\equiv \bigwedge_{1 \leq i < j \leq k} x_i \neq x_j$$

$$\theta_R(\mu, \vec{x}) :\equiv (\bigwedge_{1 \leq i < j \leq k} x_i \neq x_j) \wedge \exists y_1, \ldots, y_r \exists \kappa_1, \ldots, \kappa_r\, R(y_1, \ldots, y_r) \wedge \phi_{\Lambda_R}(\mu, \kappa_1, \ldots, \kappa_r) \wedge \bigwedge_{i \in [r]} \bigwedge_{j \in [k]} (\textsc{supp}_j(\mu, \kappa_i) \implies y_i = x_j)$$

$$\theta_{\text{OR}}(\mu, \vec{x}) :\equiv (\bigwedge_{1 \leq i < j \leq k} x_i \neq x_j) \wedge \exists \nu \exists \vec{y}\,(\psi_W(\nu, \mu) \wedge \textsc{agree}(\mu, \nu, \vec{x}, \vec{y}) \wedge V(\nu, \vec{y}))$$

$$\theta_{\text{AND}}(\mu, \vec{x}) :\equiv (\bigwedge_{1 \leq i < j \leq k} x_i \neq x_j) \wedge \forall \nu \forall \vec{y}\,((\psi_W(\nu, \mu) \wedge \textsc{agree}(\mu, \nu, \vec{x}, \vec{y})) \implies V(\nu, \vec{y}))$$

$$\theta_{\text{NOT}}(\mu, \vec{x}) :\equiv (\bigwedge_{1 \leq i < j \leq k} x_i \neq x_j) \wedge \exists \nu \exists \vec{y}\,(\psi_W(\nu, \mu) \wedge \textsc{agree}(\mu, \nu, \vec{x}, \vec{y}) \wedge \neg V(\nu, \vec{y}))$$

To understand how these formulas are defined let us consider $\theta_{\text{OR}}(\mu, \vec{x})$ as an example. The sequence $\vec{x}$ is intended to denote an assignment to the support of the gate $\mu$, and so we first check that each element in this potential assignment is unique. We then check that there exists a gate $\nu$ and assignment to its support $\vec{y}$ that is compatible with the assignment $\vec{x}$ to the support of $\mu$, and such that the gate $\nu$ evaluates to true when $\vec{y}$ is assigned to its support. It follows from Lemma 7.4 that this holds if, and only if, there exists at least one child of $\mu$ which evaluates to 1.

The remainder of this section is dedicated to defining $\theta_{\text{RANK}}$. We begin by extending the auxiliary formulas SUPP and AGREE to supports of elements of the universe of a gate. Let $s \in [3]$. From the Immerman-Vardi theorem and Lemma 5.9 there is a formula $\textsc{supp}^s(\mu, \delta, \kappa)$ such that $\mathcal{A} \models \textsc{supp}^s[g, b, u]$ if, and only if, $g$ is a gate, $b$ is the $s$th element of the universe of $g$, and $u \in \text{sp}(b)$. We can define for each $j \in \mathbb{N}$, using a similar approach as for SUPP, a formula $\textsc{supp}^s_j(\mu, \delta, \kappa)$ such that $\mathcal{A} \models \textsc{supp}^s_j[g, u, m]$ if, and only if, $\mathcal{A} \models \textsc{supp}^s[g, u, m]$ and $m$ is the $j$th element of $\text{sp}(u)$. We can use a similar approach as in the definition AGREE to define for each $s \in [3]$ a formula $\textsc{agree}^s_L(\mu, \nu, \delta, \vec{x}, \vec{y}, \vec{z})$ such that $\mathcal{A} \models \textsc{agree}^s_L[g, h, u, \vec{a}, \vec{b}, \vec{c}]$ if, and only if, $g$ and $h$ are gates with $h \in H_g$, $u$ is an element of the $s$th sort of the universe of $g$, and $\alpha^{\vec{a}}_{\vec{\text{sp}}(g)}$, $\alpha^{\vec{b}}_{\vec{\text{sp}}(h)}$, and $\alpha^{\vec{c}}_{\vec{\text{sp}}(u)}$ are pairwise compatible. Let $\textsc{agree}^s_L(\mu, \delta, \vec{x}, \vec{z}) :\equiv \exists \nu \exists \vec{y}\, \textsc{agree}^s_L(\mu, \nu, \delta, \vec{x}, \vec{y}, \vec{z})$.

We have from Lemma 5.7 and the Immerman-Vardi theorem that for each $s \in [3]$ there is a formula $\textsc{move}^s(\mu, \delta_1, \delta_2, \vec{\kappa})$ such that $\mathcal{A} \models \textsc{move}^s[g, b_1, b_2, \vec{u}]$ if, and only if, $g$ is a gate, $b_1$ and $b_2$ are elements of the $s$th sort of the universe of $g$, for all $a, b \in [2k]$ if $a \neq b$ then $u_a \neq u_b$, and there exists $\sigma \in \mathbf{Stab}(\text{sp}(g))$ such that for all $a \in [|\text{sp}(b_1)|]$, $\sigma(\vec{\text{sp}}(b_1)(a)) = u_a$ and $\sigma(b_1) = b_2$. In other words, $\mathcal{A} \models \textsc{move}^s[g, b_1, b_2, \vec{u}]$ if, and only if, the function that maps the support of $b_1$ to $\vec{u}$ extends to a permutation in $\mathbf{Stab}(\text{sp}(g))$ that maps $b_1$ to $b_2$. Note that the algorithm given in Lemma 5.7 takes as input a permutation, while the formula $\textsc{move}^s$ takes as input the behaviour of the permutation on the support of $b_1$. We recall that the action of a permutation on an element is entirely determined by the action on its support (see Lemma 4.5).

For each $s \in [3]$ let $\textsc{orb}^s(\mu, \delta_1, \delta_2) :\equiv \exists \vec{\kappa}\, \textsc{move}^s(\mu, \delta_1, \delta_2, \vec{\kappa})$. It can be seen that $\mathcal{A} \models \textsc{orb}^s[g, b_1, b_2]$ if, and only if, $b_1$ and $b_2$ are elements of the $s$th sort of the universe of $g$, and $b_1 \in \mathbf{Orb}(b_2)$. Let

$$\textsc{min-orb}^s(\mu, \delta) :\equiv \forall \epsilon\,(\textsc{orb}^s(\mu, \delta, \epsilon) \implies \delta \leq \epsilon).$$

We next define a series of FPR-formulas that implement the definition of $M^{g,\alpha}$ given in Section 7.1 for a rank gate $g$ and assignment $\alpha \in A^{\underline{\text{sp}(g)}}$. We first need to define $\bar{J}^{g,\alpha}$ and $J^{g,\alpha}$ in the logic. The definition of $\bar{J}^{g,\alpha}$ given in Section 7.1

can be directly implemented in an obvious manner to define an $\mathrm{FP}^{\mathbb{N}}$-formula $\psi_{\bar{J}}(\mu, \vec{x}, \delta_1, \vec{z}_1, \delta_2, \vec{z}_2, \delta_3, \vec{z}_3, \vec{\kappa}_1, \vec{\kappa}_2, \vec{\kappa}_3)$ such that $\mathcal{A} \models \psi_{\bar{J}}[g, \vec{a}, u_1, \vec{c}_1, u_2, \vec{c}_2, u_3, \vec{c}_3, \vec{m}_1, \vec{m}_2, \vec{m}_3]$ if, and only if, for $\alpha := \alpha^{\vec{a}}_{\vec{\mathrm{sp}}(g)}$,

$$\bar{J}^{g,\alpha}((u_1, \alpha^{\vec{c}_1}_{\vec{\mathrm{sp}}(u_1)}), (u_2, \alpha^{\vec{c}_2}_{\vec{\mathrm{sp}}(u_2)}), (u_3, \alpha^{\vec{c}_3}_{\vec{\mathrm{sp}}(u_3)})) = (\bar{\sigma}_1, \bar{\sigma}_2, \bar{\sigma}_3),$$

where for each $s \in [3]$, $\bar{\sigma}$ maps $\vec{\mathrm{sp}}(u_s)$ to an initial segment of $\vec{\kappa}_s$.

We now use $\psi_{\bar{J}}$ to define an $\mathrm{FP}^{\mathbb{N}}$-formula $\psi_J(\mu, \vec{x}, \delta_1, \vec{z}_1, \delta_2, \vec{z}_2, \delta_3, \vec{z}_3, \nu, \vec{y})$ such that $\mathcal{A} \models \psi_J[g, \vec{a}, u_1, \vec{c}_1, u_2, \vec{c}_2, u_3, \vec{c}_3, h, \vec{b}\,]$ if, and only if, $g$ is an internal gate, $h \in H_g$, and for $\alpha := \alpha^{\vec{a}}_{\vec{\mathrm{sp}}(g)}$ we have $J^{g,\alpha}((u_1, \alpha^{\vec{c}_1}_{\vec{\mathrm{sp}}(u_1)}), (u_2, \alpha^{\vec{c}_2}_{\vec{\mathrm{sp}}(u_2)}), (u_3, \alpha^{\vec{c}_3}_{\vec{\mathrm{sp}}(u_3)})) = (h, \alpha^{\vec{b}}_{\vec{\mathrm{sp}}(h)})$. This formula is defined as follows

$$\begin{aligned}
\psi_J(\mu, \vec{x}, \delta_1, \vec{z}_1, \delta_2, \vec{z}_2, \delta_3, \vec{z}_3, \nu, \vec{y}) :\equiv\ & \exists \vec{\kappa}_1, \vec{\kappa}_2, \vec{\kappa}_3 [\psi_{\bar{J}}(\mu, \vec{x}, \delta_1, \vec{z}_1, \delta_2, \vec{z}_2, \delta_3, \vec{z}_3, \vec{\kappa}_1, \vec{\kappa}_2, \vec{\kappa}_3) \wedge \\
& \exists \delta'_1, \delta'_2, \delta'_3 (\phi_L(\mu, \nu, \delta'_1, \delta'_2, \delta'_3) \wedge \\
& \exists \vec{z}'_1, \vec{z}'_2, \vec{z}'_3 [ \bigwedge_{j \in [3]} [\textsc{move}^j(g, \delta_j, \delta'_j, \vec{\kappa}_j) \wedge \textsc{agree}^j_L(\mu, \nu, \delta'_j, \vec{x}, \vec{y}, \vec{z}'_j) \wedge \\
& [ \bigwedge_{1 \le a < b \le 2k} \vec{z}'_j(a) \neq \vec{z}'_j(b) \wedge \bigwedge_{a \in [2k]} (\forall \lambda\, (\neg \textsc{supp}^j(\mu, \delta'_j, \lambda))) \vee \\
& \bigvee_{b \in [2k]} \textsc{supp}^j_a(\mu, \delta'_j, \vec{\kappa}_j(b)) \wedge \vec{z}'_j(a) = \vec{z}_j(b))) ]]]
\end{aligned}$$

We now define for $s \in [3]$ a formula $\psi^D_s$ that defines $I^{g,\alpha}_s$, the $s$th sort in the domain of $M^{g,\alpha}$. For $s \in [3]$ let

$$\psi^D_s(\mu, \vec{x}, \delta, \vec{z}) :\equiv \textsc{min-orbit}^s(\mu, \delta) \wedge \textsc{agree}^s_L(\mu, \delta, \vec{x}, \vec{z}).$$

We now define a formula $\psi_M$ that inductively defines for a rank gate $g$ and assignment $\alpha \in A^{\underline{\mathrm{sp}}(g)}$ a function $M$ which is equal to $M^{g,\alpha}$ when restricted to the domain of $M^{g,\alpha}$ and is 0 everywhere else. These additional 0s have no effect on the rank. More precisely, the formula $\psi_M(\mu, \vec{x}, \delta_1, \vec{z}_1, \delta_2, \vec{z}_2, \delta_3, \vec{z}_3; V)$ is defined such that $\mathcal{A} \models \psi_M[g, \vec{a}, u_1, \vec{c}_1, u_2, \vec{c}_2, u_3, \vec{c}_3; \beta(V)]$ if, and only if, for $\alpha := \alpha^{\vec{a}}_{\vec{\mathrm{sp}}(g)}$ and all $s \in [3]$, $(u_s, \alpha^{\vec{c}_s}_{\vec{\mathrm{sp}}(u_s)}) \in I^{g,\alpha}_s$ if

$$J^{g,\alpha}((u_1, \alpha^{\vec{c}_1}_{\vec{\mathrm{sp}}(u_1)}), (u_2, \alpha^{\vec{c}_2}_{\vec{\mathrm{sp}}(u_2)}), (u_3, \alpha^{\vec{c}_3}_{\vec{\mathrm{sp}}(u_3)})) = (h, \eta)$$

and $\beta(V)$ is an assignment to $V$ such that for all $h \in H_g$ and $\vec{b} \in A^{\underline{k}}$, we have $(h, \vec{b}) \in \beta(V)$ if, and only if, $\vec{b} \in \overline{\mathrm{EV}}_h$, then $\eta \in \mathrm{EV}_h$. This formula is defined as follows

$$\begin{aligned}
\psi_M(\mu, \vec{x}, \delta_1, \vec{z}_1, \delta_2, \vec{z}_2, \delta_3, \vec{z}_3; V) :\equiv\ & \bigwedge_{s \in [3]} \psi^D_s(\mu, \vec{x}, \delta_s, \vec{z}_s) \wedge \\
& \exists \nu, \vec{y}\, [\psi_J(\mu, \vec{x}, \delta_1, \vec{z}_1, \delta_2, \vec{z}_2, \delta_3, \vec{z}_3, \nu, \vec{y}) \wedge V(\nu, \vec{y})].
\end{aligned}$$

We now aim to define a formula $\psi^M_*$ that defines a structure similar to $M^{g,\alpha}_*$ but with some additional all-zero rows and columns. This structure gives a matrix that has the same rank as $M^{g,\alpha}_*$ and so suffices for evaluating $g$. We first define a number term $\zeta_{\mathrm{oc}}(\mu, \delta)$ such that if $g$ is a rank gate and $u$ is an element of the third sort of $g$ then $\zeta_{\mathrm{oc}}[g, u]$ evaluates to $|\mathbf{Orb}_{\mathbf{Stab}(u)}(\mathrm{sp}(u))|$. This formula is defined as follows

$$\zeta_{\mathrm{oc}}(\mu, \delta) :\equiv \#_{\vec{\kappa}}[\textsc{move}^3(\mu, \delta, \delta, \vec{\kappa})]$$

This number term counts the number of assignments to the support of the element of the universe denoted by $\delta$ that fix both $\delta$ and the support of the gate denoted by $\mu$. Note that any permutation that stabilises $\delta$ also fixes its support setwise, as follows from an obvious analogue of Lemma 4.4 for universe elements.

The number term $\zeta_{\text{oc}}$ and the formula $\psi_M$ define the numerators and denominators in Equation 1. The divisions and additions can be trivially expressed in $\text{FP}^{\mathbb{N}}$, and so we can define a number term $\zeta_*^M(\mu, \vec{x}, \delta_1, \vec{z}_1, \delta_2, \vec{z}_2; V)$ such that if $g$ is a rank gate, $\vec{a} \in A^{\underline{k}}$, $u_1$ is an element of the first sort of $g$, $u_2$ is an element of the second sort of $g$, and $\vec{b}_1, \vec{b}_2 \in A^{\underline{2k}}$ are such that $\alpha^{\vec{b}_1}_{\vec{\text{sp}}(u_1)}$ and $\alpha^{\vec{b}_2}_{\vec{\text{sp}}(u_2)}$ are compatible with $\alpha^{\vec{a}}_{\vec{\text{sp}}(g)}$, then $\zeta_*^M[g, \vec{a}, u_1, \vec{b}_1, u_2, \vec{b}_2]$ evaluates to $M_*^{g, \alpha^{\vec{a}}_{\vec{\text{sp}}(g)}}((u_1, \alpha^{\vec{b}_1}_{\vec{\text{sp}}(u_2)}), (u_1, \alpha^{\vec{b}_2}_{\vec{\text{sp}}(u_2)}))$, and otherwise $\zeta_*^M[g, \vec{a}, u_1, \vec{b}_1, u_2, \vec{b}_2]$ evaluates to 0.

The number term $\zeta_*^M$ for a gate $g$ and assignment $\alpha \in A^{\underline{\text{sp}(g)}}$ (written as a $k$-tuple) defines a matrix $M$ such that $\mathbf{rk}_p(M) = \mathbf{rk}_p(M^{g,\alpha})$ for any prime $p$. It follows from Theorem 7.10 that $\mathbf{rk}_p(M) = \mathbf{rk}(L_*^Y)$.

We define $\theta_{\text{RANK}}(\mu, \vec{x}; V)$ as follows

$$\theta_{\text{RANK}}(\mu, \vec{x}; V) :\equiv (\bigwedge_{1 \le i < j \le k} x_i \neq x_j) \wedge \exists \epsilon_1, \epsilon_2, \eta_1, \eta_2, \eta_3 [\phi_{\text{RANK}}(\mu, \epsilon_1, \epsilon_2, \eta_1, \eta_2, \eta_3) \wedge [\mathbf{rk}(\vec{z}_1 \delta_1 \le \eta_1, \vec{z}_2 \delta_2 \le \eta_2, \epsilon_2).\zeta_*^M(\mu, \vec{x}, \delta_1, \vec{z}_1, \delta_2, \vec{z}_2)]$$

We now define $\theta(\mu, \vec{x})$ as follows

$$\theta(\mu, \vec{x}) :\equiv [\mathbf{ifp}_{V, v\vec{y}} \bigvee_{s \in T} (\phi_{\Sigma, s}(\mu) \wedge \theta_s(v, \vec{y}))](\mu, \vec{x}).$$

We finally define the FPR-formula $Q$ that defines the same $q$-ary query as $C$. The definition of this formula is similar to one given in [1]. Let

$$\begin{aligned} Q(y_1, \dots y_q) :\equiv \exists \vec{x}\, \exists \mu, \kappa_1, \dots, \kappa_q \pi_1, \dots, \pi_k \, [&\theta(\mu, \vec{x}) \wedge \phi_\Omega(\kappa_1, \dots, \kappa_q, \mu) \wedge \\ &\bigwedge_{1 \le i \le k} (\text{SUPP}_i(\mu, \pi_i) \vee \forall \pi \, (\neg \text{SUPP}_i(\mu, \pi))) \wedge \\ &\bigwedge_{1 \le i \le k} \bigwedge_{1 \le j \le q} ((\text{SUPP}_i(\mu, \pi_i) \wedge (x_i = y_j)) \implies \kappa_j = \pi_i) \wedge \\ &\bigwedge_{1 \le j \le q} \bigvee_{1 \le i \le k} (x_i = y_j \wedge \text{SUPP}_i(\mu, \pi_i))]. \end{aligned}$$

We could informally understand the formula $Q$ as inverting the assignment denoted by $(y_1, \dots, y_q)$, selecting the corresponding output gate, and then evaluating this output gate. This completes the proof of Theorem 7.1 which, combined with Theorem 6.1, completes the proof of Theorem 3.15.

## 8 CONCLUDING DISCUSSION

FPR is one of the most expressive logics we know that is still contained in P and understanding its expressive power is an important question. The main result of this paper establishes an equivalence between the expressive power of FPR and the computational power of uniform families of transparent symmetric rank circuits. Not only does this establish an interesting characterization of an important logic, it also deepens our understanding of the connection between logic and circuit complexity and sheds new light on foundational aspects of the circuit model.

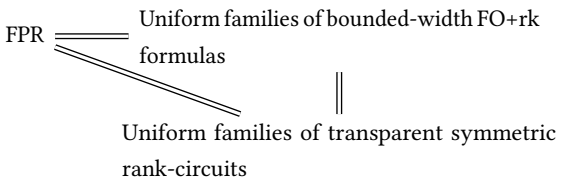
The circuit characterization helps emphasise certain important aspects of the logic. Given that P-uniform families of invariant circuits (without the restriction to symmetry) express all properties in P, we can understand the inability of

FPC (and, conjecturally, FPR) to express all such properties as essentially down to symmetry. As with other (machine) models of computation, the translation to circuits exposes the inherent combinatorial structure of an algorithm. In the case of logics, we find that a key property of this structure is its symmetry and the translation to circuits provides us with the tools to study it.

Still, the most significant contribution of this paper is not in the main result but in the techniques that are developed to establish it, and we highlight some of these now. The conclusion of [1] says that the support theorem is "largely agnostic to the particular [...] basis", suggesting that it could be easily adapted to include other gates. This turns out to have been a misjudgment. Attempting to prove the support theorem for a basis that includes rank threshold gates showed us the extent to which both the proof of the theorem and, more broadly, the definitions of circuit classes, rest heavily on the assumption that all functions computed by gates are symmetric. Thus, in order to define what the "symmetry" condition might mean for circuits that include rank threshold gates, we radically generalise the circuit framework to allow for gates that take structured inputs (rather than sets of 0s and 1s) and are invariant under isomorphisms. This leads to a refined notion of circuit automorphism, which allows us to formulate a notion of symmetry and prove a version of the support theorem. Again, in that proof, substantial new methods are required.

The condition of *transparency* makes the translation of uniform circuit families into formulas of logic (which is the difficult direction of our characterisation) possible, but it complicates the other direction. Indeed, the natural translation of formulas of FPR into uniform circuit families yields circuits which are symmetric, but not transparent. This problem is addressed by introducing gadgets in the translation—which for ease of exposition, we did in formulas of FO+rk which are then translated into circuits in the natural way. Thus, the restriction to transparent circuits is sufficient to get both directions of the characterisation.

In short, we can represent the proof of our characterisation through the three equivalences in this triangle.




This highlights another interesting aspect of our result. The first translation, of FPR to uniform families of FO+rk formulas was given in [5] and used there to establish arity lower bounds. However, this was for a weaker version of the rank logic rather than the strictly more expressive one defined by Grädel and Pakusa [11]. The fact that we can complete the cycle of equivalences with the more powerful logic demonstrates that the definition of Grädel and Pakusa is the "right" formulation of FPR.

### Future Work

There are many directions of work suggested by the methods and results developed in this paper. First of all, there is the question of transparency. We introduce it as a technical device that enables our characterisation to go through. Could it be dispensed with? Or are P-uniform families of transparent symmetric rank-circuits strictly weaker than families without the restriction of transparency?

The framework we have developed for working with circuits with structured inputs is very general and mostly not specific to rank gates. In his thesis Wilsenach [18] introduces *generalised operators*, which generalise Lindström quantifiers as well as counting and rank operators, and establishes a general correspondence analogous to Theorem 3.15

between fixed-point logics extended with generalised operators and symmetric circuits over an appropriate basis. It remains an open question how far this correspondence can be generalised. For example, can we use this framework to develop a circuit characterisation of CPTC?

At the moment, we have little by way of methods for proving inexpressibility results for FPR, whether we look at it as a logic or in the circuit model. The logical formulation lays emphasis on some parameters (the number of variables, the arity of the operators, etc.) which we can treat as resources against which to prove lower bounds. On the other hand, the circuit model brings to the fore other, more combinatorial, parameters. One such is the fan-in of gates and a promising and novel approach is to try and prove lower bounds for symmetric circuits with gates with bounded fan-in. We might ask if it is possible to compute AND[3] using a symmetric circuit with gates that have fan-in two. Perhaps we could also combine the circuit view with lower-bound methods from logic, such as pebble games. Dawar [4] has shown how the bijection games of Hella [13] can be used directly to prove lower bounds for symmetric circuits without reference to the logic. We also have pebble games for FPR [6], and it would be interesting to know if we can use these on circuits and how the combinatorial parameters of the circuit interact with the game.

Finally, we note that some of the interesting directions on the interplay between logic and symmetric circuits raised in [1] remain relevant. Can we relax the symmetry condition to something in between requiring invariance of the circuit under the full symmetric group (the case of symmetric circuits) and requiring no invariance condition at all? Can such restricted symmetries give rise to interesting logics in between FPR and P?

## ACKNOWLEDGMENTS

The work reported here has benefited enormously from discussions we have had with a number of fellow researchers. We would particularly like to thank Joanna Fawcett, who pointed us to Theorem 4.6 and Matt Anderson and Martin Otto with whom we had extensive discussions on symmetric circuits when we were all visitors at the Simons Institute for the Theory of Computing at Berkeley in 2016. In particular, Martin provided us with a note and an explanation of how his methods could be used to prove a version of the support theorem (see Remark 4.11 for details). He also served as an examiner for the second author's dissertation and his comments and suggestions given then have greatly improved this paper. For all this, we are extremely grateful.

## REFERENCES

[1] M. Anderson and A. Dawar. 2017. On Symmetric Circuits and Fixed-Point Logics. *Theory of Computing Systems* 60, 3 (2017), 521–551.

[2] A. Blass, Y. Gurevich, and S. Shelah. 1999. Choiceless polynomial time. *Annals of Pure and Applied Logic* 100, 1 (1999), 141 – 187. https://doi.org/10.1016/S0168-0072(99)00005-6

[3] A. Dawar. 2015. The Nature and Power of Fixed-Point Logic with Counting. *ACM SIGLOG News* 2, 1 (2015), 8–21.

[4] A. Dawar. 2016. On Symmetric and Choiceless Computation. In *Topics in Theoretical Computer Science*, Mohammad Taghi Hajiaghayi and Mohammad Reza Mousavi (Eds.). Springer International Publishing, Cham, 23–29.

[5] A. Dawar, M. Grohe, B. Holm, and B. Laubner. 2009. Logics with Rank Operators. In *2009 24th Annual IEEE Symposium on Logic In Computer Science (LICS)*. 113–122.

[6] A. Dawar and B. Holm. 2012. Pebble Games with Algebraic Rules. In *Automata, Languages, and Programming*, Artur Czumaj, Kurt Mehlhorn, Andrew Pitts, and Roger Wattenhofer (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 251–262.

[7] A. Dawar and G. Wilsenach. 2018. Symmetric Circuits for Rank Logic. In *27th EACSL Annual Conference on Computer Science Logic, CSL 2018, September 4-7, 2018, Birmingham, UK*. 20:1–20:16.

[8] A. Dawar and G. Wilsenach. 2020. Symmetric Arithmetic Circuits. In *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020*. 36:1–36:18. https://doi.org/10.4230/LIPIcs.ICALP.2020.36

[9] L. Denenberg, Y. Gurevich, and S. Shelah. 1986. Definability by constant-depth polynomial-size circuits. *Information and Control* 70, 2 (1986), 216–240.

[10] J.D. Dixon and B. Mortimer. 1996. *Permutation Groups*. Springer New York. https://books.google.co.uk/books?id=4QDpFN6k61EC

[11] E. Grädel and W. Pakusa. 2015. Rank Logic is Dead, Long Live Rank Logic!. In *2015 24th Annual Conference on Computer Science Logic, (CSL)*. 390–404.
[12] M. Grohe. 2017. *Descriptive Complexity, Canonisation, and Definable Graph Structure Theory.* Cambridge University Press.
[13] L. Hella. 1996. Logical Hierarchies in PTIME. *Information and Computation* 129, 1 (1996), 1 – 19.
[14] B. Holm. 2010. *Descriptive complexity of linear algebra.* University of Cambridge.
[15] N. Immerman. 1999. *Descriptive Complexity.* Springer New York.
[16] P. Kolaitis and M. Vardi. 1992. Infinitary logics and 0–1 laws. *Information and Computation* 98, 2 (1992), 258 – 294.
[17] M. Otto. 1997. The logic of explicitly presentation-invariant circuits. In *1996 10th International Workshop, Annual Conference on Computer Science Logic (CSL)*. Springer, Berlin, Heidelberg, 369–384.
[18] G. Wilsenach. 2019. *Symmetric Circuits and Model-Theoretic Logics.* Thesis. University of Cambridge. https://doi.org/10.17863/CAM.44848 Accepted: 2019-10-14T08:50:08Z.