3. Circuit Upper Bounds

The originality of mathematics consists in the fact that in mathematical science connections between things are exhibited which, apart from the agency of human reason, are extremely unobvious. *A. N. Whitehead [Whi25]*

3.1 Introduction

In Chapter 2, we investigated techniques for proving size lower bounds for restricted classes of circuits (monotonic or constant depth). Returning to the circuit synthesis problem of Chapter 1, recall that in Section 1.8.4, we showed an $O(n)$ upper bound for circuit size for *symmetric* boolean functions $f \in \mathcal{B}_n$. In this chapter, using methods from finite permutation group theory, we extend this result to "almost symmetric" boolean functions, and more generally study the notion of *invariance* or *automorphism* group of a boolean function. In [CK91], Clote and Kranakis defined the invariance group $\text{AUT}(f)$ of a function $f \in \mathcal{B}_n$ to be the set of permutations in S_n which leave f invariant under all inputs. Is there a relation between the algebraic structure and/or size of $\text{AUT}(f)$ and the circuit size $C(f)$? For how many boolean functions $f \in \mathcal{B}_n$ is $\text{AUT}(f)$ equal to a given subgroup G of the full symmetric group S_n? These and other questions are treated in the following pages.

The results of this chapter have a very distinct group-theoretic flavor in the methods used. After building intuition by presenting several examples which suggest relations between algebraic properties of groups and computational complexity of languages, we give sufficient conditions via the *Pólya cycle index* (i.e., the number of *orbits* of the group $G \leq S_n$ acting on 2^n) for an arbitrary finite permutation group to be of the form $\text{AUT}(f)$, for some $f \in \mathcal{B}_n$. We show that asymptotically "almost all" boolean functions have trivial invariance groups. For cyclic groups $G \leq S_n$, we give a logspace algorithm for determining whether the given group is of the form $\text{AUT}(f)$, for some $f \in \mathcal{B}_n$. Throughout this chapter we use standard terminology and notation from permutation group theory as found in Wielandt's classic treatise [Wie64].

Invariance groups demonstrate (for the first time) the applicability of group theoretic techniques in the study of upper bounds concerning the circuit size of languages. For any language L, let L_n be the characteristic function of the set of all strings in L of length exactly n, and let $\textsc{aut}_n(L)$ be the invariance group of L_n. We consider the *index* $|S_n : \textsc{aut}_n(L)| = n!/|\textsc{aut}_n(L)|$ as a function of n and study the class of languages whose index is polynomial in n. We use well-known lower bound results on the index of primitive permutation groups together with the O'Nan-Scott theorem, a deep result in the classification of finite simple groups, to show that any language with polynomial index is in (non-uniform) $\textsc{tc}^0$ and hence in (non-uniform) $\textsc{nc}^1$. Next, we present the beautiful result of Babai, Beals, and Takácsi-Nagy [BBTN92], which states that if a language $L \subseteq \{0,1\}^*$ has transitive invariance groups $\textsc{aut}(L_n)$ and only a polynomial number of orbits, then $L \in \textsc{tc}^0$ (this establishes a conjecture of [CK91]).

In Section 3.9, we explore several applications of the theory of invariance groups to the problem of computing boolean functions on anonymous, unlabeled networks. This leads to interesting efficient algorithms for computing boolean functions on rings [ASW88], tori [BB89], hypercubes [KK97] and Cayley networks [KK92].

3.2 Definitions and Elementary Properties

Given a function $f : \{0,\ldots,m-1\}^n \to \{0,\ldots,k-1\}$, the *invariance* or *automorphism group* of f, denoted by $\textsc{aut}(f)$, is the set of permutations on $\{1,\ldots,n\}$ which "respect" f, i.e., the set of $\sigma \in S_n$ such that for all $x_1,\ldots,x_n \in \{0,\ldots,m-1\}$,

$$f(x_1,\ldots,x_n) = f(x_{\sigma(1)},\ldots,x_{\sigma(n)}). \tag{3.1}$$

Definition 3.2.1. *For any permutation $\sigma \in S_n$, any n-tuple $x = (x_1,\ldots,x_n)$ of elements from the set $\{0,\ldots,m-1\}$, and any function $f : \{0,\ldots,m-1\} \to \{0,\ldots,k-1\}$, define*

$$x^\sigma = (x_{\sigma(1)},\ldots,x_{\sigma(n)})$$

and define $f^\sigma : \{0,\ldots,m-1\} \to \{0,\ldots,k-1\}$ by

$$f^\sigma(x) = f(x^\sigma), \text{ for all } x \in \{0,1\}^n.$$

The invariance group of f indicates how symmetric f is, in the sense that the larger the group $\textsc{aut}(f)$, the more symmetric the function f is. If for an input $x = (x_1,\ldots,x_n) \in \{0,1\}^n$ and a permutation σ, Equation (3.1) holds, then we also say that σ *fixes* f on input x. In what follows, it will be seen that there is a rich class of permutation groups which are representable as the invariance groups of boolean functions. For any language $L \subseteq \{0,1\}^*$ let L_n be the characteristic function of the set $L \cap \{0,1\}^n$ and let $\textsc{aut}_n(L)$ denote the invariance group of L_n.

A language L is said to *realize* a sequence $\mathbf{G} = \langle G_n : n \geq 1\rangle$ of permutation groups $G_n \leq S_n$, if it is true that $\textsc{aut}_n(L) = G_n$, for all n. To build intuition, as an example, we consider the following groups.

- **Identity.** I_n is generated by the identity permutation.
- **Reflection.** $R_n = \langle \rho \rangle$, where $\rho(i) = n+1-i$ is the reflection permutation

$$\begin{pmatrix} 1 & 2 & & \cdots & n \\ n & n-1 & \cdots & 1 \end{pmatrix}.$$

- **Cyclic.** $C_n = \langle (1, 2, \ldots, n) \rangle$.
- **Dihedral.** $D_n = C_n \times R_n$.
- **Hyperoctahedral.** $O_n = \langle (i, i+1) : i \text{ is even } \leq n \rangle$.

For the groups above we determine regular, as well as non-regular languages which realize them. We summarize the corresponding representability results in the following theorem. The details of the proof are left as Exercise 3.11.1.

Theorem 3.2.1 ([CK91]). *Each of the identity, reflection, cyclic (in the cyclic case only if $n \neq 3, 4, 5$), and hyperoctahedral groups can be realized by regular languages.*

Not every permutation group is representable as the invariance group of a boolean function.

Theorem 3.2.2 ([CK91]). *The alternating group A_n is not the invariance group of any boolean function $f \in \mathcal{B}_n$, provided that $n \geq 3$.*

Proof. Although this follows directly from our representability results given later, it is instructive to give a direct proof. Suppose that the invariance group of $f \in \mathcal{B}_n$ contains the alternating group A_n. Given $x \in 2^n$, for $3 \leq n$ there exist $1 \leq i < j \leq n$, such that $x_i = x_j$. It follows that the alternating group A_n, as well as the transposition (i, j) fix f on the input x. Consequently, every permutation in S_n must also fix f on x. As this holds for every $x \in 2^n$, it follows that $\text{AUT}(f) = S_n$.

Before we proceed with the general representability results, we will prove several simple observations that will be used frequently in the sequel. We begin with a few useful definitions.

Definition 3.2.2.

1. *For any $f \in \mathcal{B}_n$, define $\text{AUT}^-(f)$ to be the set*

$$\{\sigma \in S_n : (\forall x \in 2^n)(f(x) = 0 \Rightarrow f(x^\sigma) = 0)\}.$$

2. *For any $f \in \mathcal{B}_n$, define $\text{AUT}^+(f)$ to be the set*

$$\{\sigma \in S_n : (\forall x \in 2^n)(f(x) = 1 \Rightarrow f(x^\sigma) = 1)\}.$$

3. *For any permutation group $G \leq S_n$ and any $\Delta \subseteq \{1, 2, \ldots, n\}$, let G_Δ be the set of permutations $\sigma \in G$ such that $(\forall i \in \Delta)(\sigma(i) = i)$. The group G_Δ is called the pointwise stabilizer[1] of G on Δ (see [Wie64]).*

[1] We will not in general consider the *setwise* stabilizer of G with respect to Δ, defined as the set of permutations $\sigma \in G$ such that $(\forall i \in \Delta)(\sigma(i) \in \Delta)$.

4. *For any permutation σ and permutation group G, let $G^\sigma = \sigma^{-1}G\sigma$, also called the conjugate of G by σ.*
5. *For any $f \in \mathcal{B}_n$, let $1 \oplus f \in \mathcal{B}_n$ be defined by $(1 \oplus f)(x) = 1 \oplus f(x)$, for $x \in 2^n$.*
6. *If $f_1, \ldots, f_k \in \mathcal{B}_n$ and $f \in \mathcal{B}_k$, then $g = f(f_1, \ldots, f_k) \in \mathcal{B}_n$ is defined by $g(x) = f(f_1(x), \ldots, f_k(x))$.*

Define the natural isomorphism $\phi : S_n \to (S_{n+m})_{n+1,\ldots,n+m}$ by

$$\phi(\sigma)(i) = \begin{cases} \sigma(i) & \text{if } 1 \leq i \leq n \\ i & \text{if } n+1 \leq i \leq n+m. \end{cases}$$

For $X \subseteq S_n$, let $\phi(X)$ denote the image of ϕ on $X \subseteq S_n$. Now if $G \leq S_{n+m}$, and $H = G_{n+1,\ldots,n+m}$ is the pointwise stabilizer of G on $\{n+1, \ldots, n+m\}$, then we may at times identify $H \leq S_{n+m}$ with its isomorphic image $\phi^{-1}(H) \leq S_n$, and indeed write statements like $G_{n+1,\ldots,n+m} \leq S_n$. From the context, the meaning should be clear, and so cause no confusion.

Theorem 3.2.3 ([CK91]).

1. *If $f \in \mathcal{B}_n$ is symmetric, then* $\text{AUT}(f) = S_n$.
2. *Let $0 \leq m \leq n$. Given $f \in \mathcal{B}_n$, define flip(f, m) to be that $g \in \mathcal{B}_n$ satisfying*

$$g(x_1, \ldots, x_n) = \begin{cases} f(x_1, \ldots, x_n) & \textit{if the weight } |x_1 \cdots x_n|_1 \neq m \\ 1 - f(x_1, \ldots, x_n) & \textit{otherwise.} \end{cases}$$

   Then $\text{AUT}(g) = \text{AUT}(f)$. *This observation can be iterated, and so clearly* $\text{AUT}(f) = \text{AUT}(1 \oplus f)$, *for all $f \in \mathcal{B}_n$.*
3. *For any permutation σ,* $\text{AUT}(f^\sigma) = \text{AUT}(f)^\sigma$.
4. *For each $f \in \mathcal{B}_n$,* $\text{AUT}(f) = \text{AUT}^-(f) = \text{AUT}^+(f)$.
5. *If $f_1, \ldots, f_k \in \mathcal{B}_n$ and $f \in \mathcal{B}_k$ and $g = f(f_1, \ldots, f_k) \in \mathcal{B}_n$ then* $\text{AUT}(f_1) \cap \cdots \cap \text{AUT}(f_k) \subseteq \text{AUT}(g)$.
6. $(\forall k \leq n)(\exists f \in \mathcal{B}_n)(\text{AUT}(f) = S_k)$.

Proof. The proofs of (1) - (3), (5) are easy and are left as an exercise to the reader. We only prove the assertion of (4) for $\text{AUT}^+(f)$, since the proof for $\text{AUT}^-(f)$ is similar. Note that $\text{AUT}^+(f)$ is finite and closed under the group operation of composition, hence is a group. Trivially $\text{AUT}(f) \subseteq \text{AUT}^+(f)$. If $\sigma \in \text{AUT}^+(f)$, and $f(x) = 1$, then by hypothesis $f(x^\sigma) = 1$. If $f(x^\sigma) = 0$, then since $\sigma^{-1} \in \text{AUT}^+(f)$, we have that $f(x) = f((x^\sigma)^{\sigma^{-1}}) = 0$. It follows that $\text{AUT}^+(f) \subseteq \text{AUT}(f)$, as desired. To prove (6) we consider two cases. If $k + 2 \leq n$, then define f by

$$f(x) = \begin{cases} 1 \text{ if } x_{k+1} \leq x_{k+2} \leq \cdots \leq x_n \\ 0 \text{ otherwise.} \end{cases}$$

Let $\sigma \in \textsc{aut}(f)$. First notice that $(\forall i > k)(\sigma(i) > k)$. Next, it is easy to show that if σ is a nontrivial permutation, then there can be no $k \leq i < j \leq n$ such that $\sigma(j) < \sigma(i)$. This proves the desired result. If $k = n - 1$, then define the function f as follows.

$$f(x) = \begin{cases} 1 \text{ if } x_1, \ldots, x_{n-1} \leq x_n \\ 0 \text{ otherwise.} \end{cases}$$

A similar proof will show that $\textsc{aut}(f) = S_{n-1}$. This completes the proof of the theorem.

Representability will play an important role throughout the chapter.

Definition 3.2.3. *For $k \geq 2$, let $\mathcal{B}_{n,k}$ be the set of functions $f : \{0,1\}^n \to \{0, \ldots, k-1\}$. A permutation group $G \leq S_n$ is called k-representable if there exists a function $f \in \mathcal{B}_{n,k}$ such that $G = \textsc{aut}(f)$. A 2-representable group is also called strongly representable. $G \leq S_n$ is called representable if it is k-representable for some k.*

We will also consider a variant of the previous definition, by considering functions $f : \{0, \ldots, m-1\}^n \to \{0, \ldots, k-1\}$, in place of functions in $\mathcal{B}_{n,k}$.

Definition 3.2.4. *A permutation group $G \leq S_n$ is called weakly representable if there exists an integer $k \geq 2$, an integer $2 \leq m < n$ and a function $f : m^n \to k$, such that $G = \textsc{aut}(f)$.*

In our definition of representable and weakly representable, we required that an n-variable boolean function represent a subgroup $G \leq S_n$, where $m = n$. This is an important definitional point, as illustrated by the next result.

Theorem 3.2.4 (Isomorphism Theorem, [CK91]). *Every permutation group $G \leq S_n$ is isomorphic to the invariance group of a boolean function $f \in \mathcal{B}_{n(\lfloor \log n + 1 \rfloor)}$.*

Proof. First, some notation. Let $w = w_1 \cdots w_n$ be a word in $\{0,1\}^*$. Recall that the weight $|w|_1$ of w is the number of occurrences of 1 in w, and that $|w|$ denotes the length n of w. The word w is *monotone* if for all $1 \leq i < j \leq |w|$, $w_i = 1 \Rightarrow w_j = 1$. The *complement* of w, denoted by $\overline{w}$ is the word which is obtained from w by "flipping" each bit w_i, i.e., $|w| = |\overline{w}|$ and $\overline{w}_i = 1 \oplus w_i$, for all $1 \leq i \leq |w|$. Fix n and let $s = \lfloor \log n + 1 \rfloor$. View each word $w \in \{0,1\}^{ns}$ (of length ns) as consisting of n blocks, each of length s, and let $w(i) = w_{(i-1)s+1} \cdots w_{is}$ denote the i-th such block. For a given permutation group $G \leq S_n$, let L_G be the set of all words $w \in \{0,1\}^{ns}$ such that one of the following holds: either

1. $|w|_1 = s$ and if w is divided into n blocks

$$w(1), w(2), \ldots, \ w(n)$$

each of length s, then exactly one of these blocks consists entirely of 1s, while the other blocks consist entirely of 0s, or

2. $|w|_1 \leq s-1$ and for each $1 \leq i \leq n$, the complement $\overline{w}$ of the i-th block of w is monotone (thus each $w(i)$ consists of a sequence of 1s concatenated with a sequence of 0s), or
3. a) $|w|_1 \geq n$
   b) for each $1 \leq i \leq n$, the first bit of $w(i)$ is 0,
   c) the integers $bin(w,i)$, whose binary representations are given by the words $w(i)$ for $1 \leq i \leq n$, are mutually distinct
   d) $\sigma_w \in G$, where $\sigma_w : \{1,\ldots,n\} \rightarrow \{1,\ldots,n\}$ is the permutation defined by $\sigma_w(i) = bin(w,i)$.

The intuition for items (3a) and (3b) above is the following. The words with exactly s many 1s have all these 1s in exactly one block. This guarantees that any permutation respecting the language L_G must map blocks to blocks. By considering words with a single 1 (which by monotonicity must be located at the first position of a block), we guarantee that each permutation which respects L_G must map the first bit of a block to the first bit of some other block. Inductively, by considering the word with exactly $(r-1)$ many 1s, all located at the beginning of a single block, while all other bits of the word are 0s, we guarantee that each permutation which respects L_G must map the $(r-1)$-th bit of each block to the $(r-1)$-st bit of some other block. It follows that any permutation which respects L_G must respect blocks as well as the order of elements in the blocks; i.e., for every permutation $\tau \in \textsc{aut}_{ns}(L_G)$,

$$(\forall k \in \{0,\ldots,n-1\})(\exists m \in \{0,\ldots,n-1\})(\forall i \in \{1,\ldots,n\})(\tau(ks+i) = ms+i).$$

Call such a permutation *s-block invariant.* Given a permutation τ in the invariance group $\textsc{aut}_{ns}(L_G)$, let $\overline{\tau} \in S_n$ be the induced permutation defined by

$$\overline{\tau}(k) = m \Leftrightarrow (\forall 1 \leq i \leq n)\,[\tau(ks+i) = ms+i]\,.$$

Claim. $G = \{\overline{\tau} : \tau \in \textsc{aut}^{+}_{ns}(L_G)\}$.

Proof of Claim. ($\subseteq$) Notice that every element $\overline{\tau}$ of $G \leq S_n$ gives rise to a unique s-block invariant permutation $\tau \in S_{ns}$. If $w \in L_G \subseteq \{0,1\}^{ns}$, then considering separately the cases $|w|_1 \leq s$ and $|w|_1 \geq n$, by s-block invariance of τ, $w^\tau \in L_G$.

($\supseteq$) First, notice that if $w \in L_G \subseteq \{0,1\}^{ns}$ and the associated permutation $\sigma_w \in G \leq S_n$, then $\sigma_{(w^\tau)} = \overline{\tau} \circ \sigma_w \in G$. Now, let $w \in L_G$ be such that the associated σ_w is the identity on S_n. Then for any $\tau \in \textsc{aut}_{ns}(L_G)$, $w^\tau \in L_G$, so $\sigma_{(w^\tau)} = \overline{\tau} \circ \sigma_w = \overline{\tau} \in G$. This establishes the claim, which completes the proof of the theorem.

We conclude this section by comparing the different definitions of representability given above.

Theorem 3.2.5 ([CK91]). *For any permutation group $G \leq S_n$ the following statements are equivalent:*

1. *G is representable.*
2. *G is the intersection of a finite family of strongly representable permutation groups.*
3. *For some m, G is the pointwise stabilizer of a strongly representable group over S_{n+m}, i.e., $G = (\text{AUT}_{n+m}(f))_{\{n+1,\ldots,n+m\}}$, for some $f \in \mathcal{B}_{n+m}$ and $m \leq n$.*

Proof. First we prove that $1 \Rightarrow 2$. Indeed, let $f \in \mathcal{B}_{n,k}$ such that $G = \text{AUT}(f)$. For each $b < k$ define as follows a 2-valued function $f_b : 2^n \to \{b, k\}$:

$$f_b(x) = \begin{cases} b \text{ if } f(x) = b \\ k \text{ if } f(x) \neq b \end{cases}$$

It is straightforward to show that $\text{AUT}(f) = \text{AUT}(f_0) \cap \cdots \cap \text{AUT}(f_{k-1})$. But also conversely we can prove that $2 \Rightarrow 1$. Indeed, assume that $f_b \in \mathcal{B}_n$, $b < k$, is a given family of boolean valued functions such that G is the intersection of the strongly representable groups $\text{AUT}(f_b)$. Define $f \in \mathcal{B}_{n,2^k}$ as follows

$$f(x) = \langle f_0(x), \ldots, f_{k-1}(x) \rangle,$$

where for any integers $n_0, \ldots, n_{k-1}$, the symbol $\langle n_0, \ldots, n_{k-1} \rangle$ represents a standard encoding of the k-tuple $(n_0, \ldots, n_{k-1})$ as an integer. It is then clear that $\text{AUT}(f) = \text{AUT}(f_0) \cap \cdots \cap \text{AUT}(f_{k-1})$, as desired.

We now prove that $2 \Rightarrow 3$. Suppose that $G = \text{AUT}(f_0) \cap \cdots \cap \text{AUT}(f_k) \leq S_n$, where $f_0, \ldots, f_k \in \mathcal{B}_n$, and let $m = |k|$. Define $f \in \mathcal{B}_{n+m}$ by

$$f(x_1, \ldots, x_n, b_1, \ldots, b_m) = \begin{cases} f_r(x_1, \ldots, x_n) & \text{if } r = \sum_{i=1}^m b_i \cdot 2^{m-i} \leq k \\ 0 & \text{otherwise.} \end{cases}$$

Define the isomorphism $\phi : S_n \to (S_{n+m})_{n+1,\ldots,n+m}$ by

$$\phi(\sigma)(i) = \begin{cases} \sigma(i) & \text{if } 1 \leq i \leq n \\ i & \text{if } n+1 \leq i \leq n+m \end{cases}$$

and let $\psi : (S_{n+m})_{n+1,\ldots,n+m} \to S_n$ denote the inverse ϕ^{-1} of ϕ.

Claim. $\text{AUT}(f_0) \cap \cdots \cap \text{AUT}(f_k) = \psi(\text{AUT}(f)_{n+1,\ldots,n+m})$.

Proof of Claim. ($\subseteq$) Let $\sigma \in \text{AUT}(f_0) \cap \cdots \cap \text{AUT}(f_k) \subseteq S_n$, and let $\widetilde{\sigma} = \phi(\sigma) \in S_{n+m}$. Given $x \in \{0,1\}^n$ and $b \in \{0,1\}^m$, if $r = \sum_{i=1}^m b_i \cdot 2^{m-i} \leq k$, then $f_r(x) = f_r(x^\sigma)$ and so $f(x,b) = f((x,b)^{\widetilde{\sigma}}$. As well, if $r = \sum_{i=1}^m b_i \cdot 2^{m-i} > k$, then $f(x,b) = 0 = f((x,b)^{\widetilde{\sigma}}$. It follows that $\widetilde{\sigma} \in \text{AUT}(f)$, so $\sigma \in \psi(\text{AUT}(f)_{n+1,\ldots,n+m})$.

($\subseteq$) Let $\widetilde{\sigma} \in \text{AUT}(f)_{n+1,\ldots,n+m}$. Given $x \in \{0,1\}^n$ and $b \in \{0,1\}^m$, we have $f(x,b) = f((x,b)^{\widetilde{\sigma}}$. If $r = \sum_{i=1}^m b_i \cdot 2^{m-i} \leq k$, then

$$f_r(x) = f(x,b) = f((x,b)^{\widetilde{\sigma}}) = f(x^\sigma, b) = f_r(x^\sigma)$$

and so $\sigma \in \text{AUT}(f_r)$. Since this holds for all $r \le k$, $\sigma \in \text{AUT}(f_0) \cap \cdots \cap \text{AUT}(f_k)$.

Finally, we prove that $3 \Rightarrow 2$. Let $G \le S_n$ denote $\psi(\text{AUT}(f)_{n+1,\ldots,n+m})$, and let $\sigma \in G$, and $\widetilde{\sigma} = \phi(\sigma) \in \text{AUT}(f)_{n+1,\ldots,n+m}$. Then for any $x \in \{0,1\}^n$ and $b \in \{0,1\}^m$, if $r = \sum_{i=1}^m b_i \cdot 2^{m-i} \le k$, then $f(x,b) = f_r(x) = f_r(x^\sigma) = f(x^\sigma, b) = f((x,b)^{\widetilde{\sigma}})$, while if $r = \sum_{i=1}^m b_i \cdot 2^{m-i} > k$, then $f(x,b) = 0 = f(x^\sigma, b) = f((x,b)^{\widetilde{\sigma}})$. Thus $\widetilde{\sigma} \in \text{AUT}(f_0) \cap \cdots \cap \text{AUT}(f_k)$, so $\sigma \in \psi(\text{AUT}(f)_{n+1,\ldots,n+m})$. This concludes the proof of the theorem.

3.3 Pólya's Enumeration Theory

In the section, we present the rudiments of Pólya's enumeration theory. Our goal here is to emphasize the relevant elements of the theory without providing any complete proofs. The interested reader is advised to consult [Ber71] and [PR87] or better yet complete details of the proofs on her own.

Let G be a permutation group on n elements. Define an equivalence relation on integers as follows: $i \sim j \bmod G$ if and only if for some $\sigma \in G$, $\sigma(i) = j$. The equivalence classes under this equivalence relation are called *orbits*. Let $G_i = \{\sigma \in G : \sigma(i) = i\}$ be the *stabilizer* of i, and let i^G be the orbit of i. An elementary theorem [Wie64] asserts that $|G : G_i| = |i^G|$. Using this, we can obtain the well-known theorem of Burnside and Frobenius [Com70].

Theorem 3.3.1. *For any permutation group G on n elements, the number of orbits of G is equal to the average number of fixed points of a permutation $\sigma \in G$; i.e.,*

$$\omega_n(G) = \frac{1}{|G|} \sum_{\sigma \in G} |\{i : \sigma(i) = i\}|, \tag{3.2}$$

where $\omega_n(G)$ is the number of orbits of G.

A group G *acts* on a set X, if there is a map $\phi : G \times X \to X$, such that

1. $\phi(\sigma, x) = x$,
2. $\phi(\sigma \circ \tau, x) = \phi(\sigma, \phi(\tau, x))$

where e is the identity element of G, and $\circ$ is the group multiplication. The group G acts *transitively* on X if additionally

$$(\forall x, y \in X)(\exists \sigma \in G)(\phi(\sigma, x) = y).$$

Note that any group $G \le S_n$ acts on $\{0,1\}^n$ by the group action

$$\phi(\sigma, x) = \phi(\sigma, x_1 \cdots x_n) = (x_{\sigma(1)} \cdots x_{\sigma(n)}) = x^\sigma.$$

Moreover, any permutation $\sigma \in S_n$ can be identified with a permutation on 2^n defined as follows:

$$x = (x_1, \ldots, x_n) \to x^\sigma = (x_{\sigma(1)}, \ldots, x_{\sigma(n)}).$$

Hence, any permutation group G on n elements can also be thought of as a permutation group on the set 2^n. It follows from (3.2) that

$$|\{x^G : x \in 2^n\}| = \frac{1}{|G|} \sum_{\sigma \in G} |\{x \in 2^n : x^\sigma = x\}|,$$

where $x^G = \{x^\sigma : \sigma \in G\}$ is the orbit of x. We would like to find a more explicit formula for the right-hand side of the above equation. To do this notice that $x^\sigma = x$ if and only if x is invariant on the orbits of σ. It follows that $|\{x \in 2^n : x^\sigma = x\}| = 2^{o(\sigma)}$, where $o(\sigma)$ is the number of orbits of (the group generated by) σ acting on 2^n. Using the fact that $o(\sigma) = c_1(\sigma) + \cdots + c_n(\sigma)$, where $c_i(\sigma)$ is the number of i-cycles in σ (i.e., in the cycle decomposition of σ), we obtain Pólya's formula:

$$|\{x^G : x \in 2^n\}| = \frac{1}{|G|} \sum_{\sigma \in G} 2^{o(\sigma)} = \frac{1}{|G|} \sum_{\sigma \in G} 2^{c_1(\sigma)+\cdots+c_n(\sigma)}. \tag{3.3}$$

The number $|\{x^G : x \in 2^n\}|$ is called the *cycle index* of the permutation group G and will be denoted by $\Theta(G)$. If we want to stress the fact that G is a permutation group on n letters then we write $\Theta_n(G)$, instead of $\Theta(G)$. For more information on Pólya's enumeration theory, the reader should consult [Ber71] and [PR87].

Since the invariance group $\textsc{aut}(f)$ of a function $f \in \mathcal{B}_n$ contains G if and only if it is invariant on each of the different orbits x^G, $x \in 2^n$, we obtain that

$$|\{f \in \mathcal{B}_n : \textsc{aut}(f) \geq G\}| = 2^{\Theta(G)}.$$

It is also not difficult to compare the size of $\Theta(G)$ and $|S_n : G|$. Indeed, let $H \leq G \leq S_n$. If

$$Hg_1, Hg_2, \ldots, Hg_k$$

are the distinct right cosets of G modulo H then for any $x \in 2^n$ we have that

$$x^G = x^{Hg_1} \cup x^{Hg_2} \cup \cdots \cup x^{Hg_k}.$$

It follows that $\Theta_n(H) \leq \Theta_n(G) \cdot |G : H|$. Using the fact that $\Theta_n(S_n) = n+1$ we obtain as a special case that $\Theta_n(G) \leq (n+1)|S_n : G|$. In addition, using a simple argument concerning the size of the orbits of a permutation group we obtain that if $\Delta_1, \ldots, \Delta_\omega$ are different orbits of the group $G \leq S_n$ acting on $\{1, 2, \ldots, n\}$ then $(|\Delta_1| + 1) \cdots (|\Delta_\omega| + 1) \leq \Theta_n(G)$. We summarize these results in the following theorem.

Theorem 3.3.2. *For any permutation groups $H \leq G \leq S_n$, we have*

1. $\Theta_n(G) \leq \Theta_n(H) \leq \Theta_n(G) \cdot |G : H|$.
2. $\Theta_n(G) \leq (n+1) \cdot |S_n : G|$.
3. $n+1 \leq \Theta_n(G) \leq 2^n$.
4. *If* $\Delta_1, \ldots, \Delta_\omega$ *are different orbits of* G *acting on* $\{1, \ldots, n\}$ *then* $(|\Delta_1| + 1) \cdots (|\Delta_\omega| + 1) \leq \Theta_n(G)$.

3.4 Representability of Permutation Groups

Next we study the representability problem for permutation groups and give sufficient conditions via Pólya's cycle index for a permutation group to be representable. In addition we consider the effect on representability of several well-known group operations, like product, wreath product, etc.

A simple observation due to Kisielewicz [Kis99] relates representable groups with the automorphism groups of undirected graphs.

Theorem 3.4.1 ([Kis99]). *The automorphism group of an undirected graph is* 2*-representable.*

Proof. For each two-element set $e = \{i, j\}$, consider the n-tuple $x^e = (x_1^e, \ldots, x_n^e) \in \{0,1\}^n$ such that $x_i^e = x_j^e = 1$ and $x_k^e = 0$, for all $k \neq i, j$. Let the graph $G = (V, E)$ with vertex set $V = \{1, 2, \ldots, n\}$ and edge set E. Define the boolean function

$$f(x) = \begin{cases} 1 \text{ if } x = x^e, \text{ for some } e \in E \\ 0 \text{ otherwise.} \end{cases}$$

It is a simple exercise to show that $\text{AUT}(f)$ is precisely the automorphism group of the given graph.

In order to state the first general representation theorem we define for any $n+1 \leq \theta \leq 2^n$ and any permutation group $G \leq S_n$ the set $\mathbf{G}_\theta^{(n)} = \{M \leq G : \Theta_n(M) = \theta\}$. Also, for any $H \subseteq S_n$, and any $g \in S_n$, the notation $\langle H, g \rangle$ denotes the smallest subgroup of S_n containing the set $H \cup \{g\}$.

Theorem 3.4.2 (Representation Theorem, [CK91]). *For any permutation groups* $H < G \leq S_n$ *if* $H = G \cap K$*, for some representable permutation group* $K \leq S_n$*, then* $(\forall g \in G - H)(\Theta_n(\langle H, g \rangle) < \Theta_n(H))$*. Moreover, this last statement is equivalent to* H *being maximal in* $\mathbf{G}_\theta^{(n)}$*, where* $\Theta_n(H) = \theta$.

Proof. By Theorem 3.2.5, K is the intersection of a family strongly representable groups. Hence let $f_1, \ldots, f_k \in \mathcal{B}_n$ be such that $K = \cap_{i=1}^k \text{AUT}(f_i)$. Then

$$H = \bigcap_{i=1}^{k} \text{AUT}(f_i) \cap G.$$

Assume, to the contrary, that there exists a subgroup $K \leq G$ such that $H < K$ and $\Theta(K) = \Theta(H)$. This implies

$$\forall x \in 2^n (x^K = x^H).$$

We claim, however, that

$$K \subseteq \bigcap_{i=1}^{k} \text{AUT}(f_i) \cap G = H$$

which then contradicts the assumption that $H < K$. Indeed, let $\sigma \in K$ and $x \in \{0,1\}^n$. Then

$$x^K = (x^\sigma)^K = (x^\sigma)^H$$

so it follows that $x = (x^\sigma)^\tau$, for some $\tau \in H$. Consequently, $f_i(x) = f_i((x^\sigma)^\tau) = f_i(x^\sigma)$, for all $1 \leq i \leq k$, and so $K \subseteq \bigcap_{i=1}^{k} \text{AUT}(f_i) \cap G$, which establishes our claim.

It remains to prove the equivalence of the last statement in the theorem. Assume that H is a maximal element of $\mathbf{G}_\theta^{(n)}$, but that for some $g \in G - H$, we have that $\Theta_n(\langle H, g\rangle) = \Theta_n(H)$. But then $H < \langle H, g\rangle \leq G$, contradicting the maximality of H. To prove the other direction we argue as follows. Assume, to the contrary, that the hypothesis is true but that H is not maximal in $\mathbf{G}_\theta^{(n)}$. This means there exists $H < K \leq G$ such that $\Theta_n(K) = \Theta_n(H)$. Take any $g \in K - H$ and notice that

$$\Theta_n(\langle H, g\rangle) \geq \Theta_n(K) = \theta = \Theta_n(H) \geq \Theta_n(\langle H, g\rangle).$$

Hence, $\Theta_n(H) = \Theta_n(\langle H, g\rangle)$, contradicting our assumption.

Let $O(G)$ denote the set $\{x^G : x \in \{0,1\}^n\}$ of orbits of G acting on $\{0,1\}^n$. If the group G is of the form $\text{AUT}(f)$ for some function $f \in \mathcal{B}_{n,k}$, then (1) all n-tuples in every orbit of $O(G)$ must have the same value under f, and (2) for every permutation $\tau \notin G$, there must be an element x of $\{0,1\}^n$, such that x and x^τ belong to different orbits of $O(G)$, and these orbits have different values under f. Hence, in order to find a k-valued boolean function f whose invariance group is G, it is necessary and sufficient to find a function $F : O(G) \to \{0, \ldots, k-1\}$, such that for every permutation $\tau \notin G$, there exists $x \in \{0,1\}^n$, such that x and x^τ belong to different orbits of $O(G)$ and these orbits have different values under F.

Using deliberations issuing from the previous observation, we prove the following result concerning the representation of *maximal* permutation groups.

Theorem 3.4.3 (Maximality Theorem, [CK91, Kis99]).

1. *A permutation group $G \leq S_n$ is representable if and only if it is a maximal subgroup of S_n among those having the same number $\Theta(G)$ of orbits in $\{0,1\}^n$. In such a case G is representable by a function $f \in \mathcal{B}_{n,k}$ with $k \leq \binom{n}{\lfloor (n)/2 \rfloor}$.*

2. *All maximal subgroups of S_n are strongly representable, the only exceptions being: (a) the alternating group A_n, for all $n \geq 3$, and the conjugates of the following three types of groups: (b) the 1-dimensional, linear, affine group $AGL_1(5)$ over the field of 5 elements, for $n = 5$; (c) the group of linear transformations $PGL_2(5)$ of the projective line over the field of 5 elements, for $n = 6$; (d) the group of semi-linear transformations $P\Gamma L_2(8)$ of the projective line over the field of 8 elements, for $n = 9$.*

Proof. To prove 1, we argue as follows. If G is representable, then it must be maximal among the subgroups of S_n with the same set $O(G)$ of orbits of $\{0,1\}^n$. Indeed, if there were a permutation $\tau \notin G$ such that for every $x \in \{0,1\}^n$, x and x^τ always belong to the same orbit of $O(G)$, then the group $G' = \langle G, \tau \rangle$ has precisely the same orbits in $\{0,1\}^n$ as G itself. Therefore every function f on $\{0,1\}^n$ invariant under G is invariant under G'. Hence, G cannot be representable at all.

Conversely, suppose that $G \leq S_n$ is maximal among those having the same number $k = \Theta_n(G)$ of orbits in $\{0,1\}^n$. Define $f \in \mathcal{B}_{n,k}$ by $f(x) = i$ iff x is in the i-th orbit in some canonical listing of orbits.[2] Clearly $G = \text{AUT}(f)$. To achieve the upper bound $k \leq \binom{n}{\lfloor (n)/2 \rfloor}$, note that the orbits of $O(G)$ can be partitioned into $n+1$ natural levels, according to the weight (number of 1s) $|x|_1$ of an element $x \in \{0,1\}^n$ belonging to an orbit. It thus suffices to assign different values to those orbits of elements having median weight, and clearly there are at most $\binom{n}{\lfloor (n)/2 \rfloor}$ of these.

To prove 2, let M be a maximal subgroup of S_n. We distinguish two cases.

Case 1. $\Theta_n(M) > n+1$.
In this case, there is a level $O_i(M)$ consisting of more than one orbit. If $T \in O_i(M)$ then the boolean function f assigning 1 to all n-tuples in T, and 0 otherwise, strongly represents M.
Case 2. $\Theta_n(M) = n+1$.
In this case, M is not representable at all. Moreover, for any two subsets S, S' of $\{1, 2, \ldots, n\}$ of the same cardinality there is a permutation $\pi \in M$ such that $\pi(S) = S'$. We know from the main theorem of [BP55] that M is of one of the forms in the statement of the theorem.

Our previous study focused on representability results for maximal permutation groups. The following refinement appears to be very natural.

Definition 3.4.1. *Let $\mathcal{R}_k^n$ denote the class of k-representable permutation groups on n letters.*

[2] For $x, y \in \{0,1\}^n$, a possible canonical ordering is given by $x^G < y^G$ iff the lexicographic least element in the orbit of x is less the lexicographic least element in the orbit of y).

Clearly $\mathcal{R}_k^n \subseteq \mathcal{R}_{k+1}^n$. It is interesting to note that it is not known whether or not $\mathcal{R}_k^n$ forms a proper hierarchy. However, the following can be proved.

Theorem 3.4.4 ([Kis99]). *$\mathcal{R}_2^n \neq \mathcal{R}_3^n$, i.e., there exist 3-representable groups which are not 2-representable.*

Proof. The desired group D consists of the identity permutation, as well as the permutations

$$(1,2)(3,4), (1,3)(3,4), \text{ and } (1,4)(2,3).$$

It is easily checked that $\Theta(D) = 7$. Indeed, the orbits are the following:
weight 0: $\{0000\}$,
weight 1: $\{1000, 0100, 0010, 0001\}$,
weight 2: $\{1100, 0011\}, \{1010, 0101\}, \{1001, 0110\}$,
weight 3: $\{0111, 1011, 1101, 1110\}$,
weight 4: $\{1111\}$.

To show that D is 3-representable, we define a function $f : \{0,1\}^4 \to \{0,1,2\}$, which assigns different values to the weight 2 orbits. Inspection of these orbits shows that $\text{AUT}(f)$ cannot contain a transposition and it follows easily that $D = \text{AUT}(f)$.

However, D is not 2-representable. Assume, on the contrary, that there is a boolean function $g \in \mathcal{B}_2$, which represents (i.e., 2-represents) D. Two of the weight 2 orbits must be assigned the same value, say the first and the second one. It follows easily that the transposition $(2,3) \in \text{AUT}(g) = D$. However, this is a contradiction.

As noted above, all maximal permutation groups with the exception of A_n are of the form $\text{AUT}(f)$, provided that $n \geq 10$. Such maximal permutation groups include: the cartesian products $S_k \times S_{n-k}$ ($k \leq n/2$), the *wreath products* $S_k \wr S_l$ ($n = kl, k, l > 1$),[3] the affine groups $AGL_d(p)$, for $n = p^d$, etc. The interested reader will find a complete survey of classification results for maximal permutation groups in [KL88]. As well, it should be pointed out that there are many (nonmaximal) permutation groups which are not representable – for example wreath products $G \wr A_n$. For additional representation results, we refer the reader to Exercise 3.11.11.

Theorem 3.4.5. ([Kis99]) *If $G \leq S_n, H \leq S_m$ are k-representable for some $k \geq 2$ then $G \times H \leq S_{n+m}$ is r-representable for every r satisfying $r(r-1) \geq k$. In particular, $G \times H$ is k-representable.*

Proof. We follow the proof of Kisielewicz [Kis99]. Without loss of generality, assume $m \leq n$. Let $g, h \in \mathcal{B}_{n,k}$ be such that $G = \text{AUT}(g), H = \text{AUT}(h)$.

[3] The *wreath* product $G \wr H$ of $G \leq S_n$ with $H \leq S_m$ is a subgroup of $S_{n \cdot m}$, defined as $\{(\sigma_1, \ldots, \sigma_n; \tau) : \sigma_1, \ldots, \sigma_n \in G, \tau \in H\}$. Here, for $\sigma_1, \ldots, \sigma_m \in G \leq S_n, \tau \in H \leq S_m$, define $(\sigma_1, \ldots, \sigma_n; \tau)$ to be that permutation $\rho \in S_{n \cdot m}$ such that for $1 \leq i \leq n, 1 \leq j \leq m$, $\rho(i,j) = (\sigma_j(i), \tau(j))$.

Since $r(r-1) \geq k$, we may assume that g, h take values from the cartesian product $P = \{0, \ldots, r-1\} \times \{0, \ldots, r-2\}$. Let π_1, π_2 be the first and second projection operations on the set P. We define an r-valued boolean function $f : \{0,1\}^{m+n} \to \{0, \ldots, r-1\}$ as follows:

$$f(z) = \begin{cases} \pi_1(g(x)) & \text{if } z = x0^m, \text{ for some } x \in 2^n, x \neq 0^n, 1^n \\ \pi_2(g(x)) & \text{if } z = x1^m, \text{ for some } x \in 2^n, x \neq 0^n, 1^n \\ \pi_1(h(y)) & \text{if } z = 0^n y, \text{ for some } y \in 2^m, y \neq 0^m, 1^m \\ \pi_2(h(y)) & \text{if } z = 1^n y, \text{ for some } y \in 2^m, y \neq 0^m, 1^m \\ r-1 & \text{if } z = 1^n 0^m \\ 0 & \text{otherwise.} \end{cases}$$

Claim. $G \times H = \text{AUT}(f)$.
Proof of Claim. ($\subseteq$) Let $\sigma \in G, \tau \in H$ and let $z \in 2^{m+n}$ such that $z = xy$, with $x \in \{0,1\}^n, y \in \{0,1\}^m$. Then by the above definition,

$$f(x^\sigma y) = f(xy) = f(xy^\tau)$$

since $g(x^\sigma) = g(x)$ and $h(y^\tau) = h(y)$.
($\supseteq$) It is easily checked that by definition of f, for all $z \in \{0,1\}^{n+m}$, $|z|_1 = n$, and $z \neq 1^n 0^m$, we have $f(1^n 0^m) = r - 1 > f(z)$. Thus it easily follows that $G \times H \subseteq S_n \times S_m$. Now let $\rho = (\sigma, \tau) \in (S_n \times S_m - G \times H)$, and for specificity, assume that $\sigma \notin G$ (a similar argument works when $\tau \notin H$). There is an $x \in \{0,1\}^n$ such that $g(x) \neq g(x^\sigma)$ and $x \notin \{0^n, 1^n\}$. It follows that $\pi_i(g(x^\sigma)) \neq \pi_i(g(x))$ for $i = 1$ or $i = 2$. Consequently, $f(z) \neq f(z^\sigma)$ for either $z = x0^m$ or $z = x1^m$. This proves the desired assertion.

3.5 Algorithm for Representing Cyclic Groups

In this section we prove the following represention theorem for cyclic groups.

Theorem 3.5.1 ([CK91]). *There is a logspace algorithm, which, when given as input a cyclic group $G \leq S_n$, decides whether the group is 2-representable, in which case it outputs a function $f \in \mathcal{B}_n$ such that $G = \text{AUT}(f)$.*

Proof. We establish the correctness of the following algorithm:

```
Input
$G = \langle\sigma\rangle$ cyclic group.
Step 1
Decompose $\sigma = \sigma_1\sigma_2\cdots\sigma_k$, where $\sigma_1, \sigma_2, \ldots, \sigma_k$ are disjoint cycles
of lengths $l_1, l_2, \ldots, l_k \geq 2$, respectively.
Step 2
if for all $1 \leq i \leq k$,
        $l_i = 3 \Rightarrow (\exists j \neq i)(3|l_j)$ and
        $l_i = 4 \Rightarrow (\exists j \neq i)(\gcd(4, l_j) \neq 1)$ and
        $l_i = 5 \Rightarrow (\exists j \neq i)(5 | l_j)$
        **then** output $G$ is 2-representable.
**else output** $G$ is not 2-representable.
**end**
```

Before proceeding with the main proof we introduce some definitions.

Definition 3.5.1.

1. *A boolean function $f \in \mathcal{B}_n$ is called special if for all words w of length n, $|w|_1 = 1 \Rightarrow f(w) = 1$.*
2. *The support of a permutation σ, denoted by Supp(σ), is the set of i such that $\sigma(i) \neq i$. The support of a permutation group G, denoted Supp(G), is the union of the supports of the elements of G.*
3. *Let $\sigma_1, \ldots, \sigma_k$ be a collection of cycles. We say that the group $G = \langle \sigma_1, \ldots, \sigma_k \rangle$ generated by the permutations $\sigma_1, \ldots, \sigma_k$ is specially representable if there exists a special boolean function $f : \{0,1\}^{\Omega} \to \{0,1\}$ (where Ω is the union of the supports of the permutations $\sigma_1, \ldots, \sigma_k$), such that $G = \text{AUT}(f)$. Note that by definition every specially representable group is strongly representable.*

We now turn our attention to the proof of correctness of the above algorithm. The proof is in a series of lemmas.

Lemma 3.5.1. *Suppose that $\sigma_1, \ldots, \sigma_{n+1}$ is a collection of cycles such that both $\langle \sigma_1, \ldots, \sigma_n \rangle$ and $\langle \sigma_{n+1} \rangle$ are specially representable and have disjoint supports. Then $\langle \sigma_1, \ldots, \sigma_{n+1} \rangle$ is specially representable.*

Proof. Put $\Omega_0 = \cup_{i=1}^{n} Supp(\sigma_i), \Omega_1 = Supp(\sigma_{n+1})$ and let $|\Omega_0| = m$, $|\Omega_1| = k$. Suppose that $f_0 : 2^{\Omega_0} \to 2$ and $f_1 : 2^{\Omega_1} \to 2$ are special boolean functions representing the groups $\langle \sigma_1, \ldots, \sigma_n \rangle$ and $\langle \sigma_{n+1} \rangle$, respectively. By Theorem 3.2.3, without loss of generality we may assume that $1 = f_0(0^m) \neq f_1(0^k) = 0$, and for $u \in \{0,1\}^m$, $v \in \{0,1\}^k$, $|u|_1 = 1 = |v|_1$ we have $f_0(u) = 1 = f_1(v)$. Let $\Omega = \Omega_0 \cup \Omega_1$ and define $f : \{0,1\}^{\Omega} \to \{0,1\}$ by $f(w) = f_0(w \upharpoonright \Omega_0) \cdot f_1(w \upharpoonright \Omega_1)$.

Claim. $\langle \sigma_1, \ldots, \sigma_{n+1} \rangle = \text{AUT}_{\Omega}(f)$.

Proof of Claim. The containment from left to right is clear, so it remains to prove that $\text{AUT}_{\Omega}(f) \subseteq \langle \sigma_1, \ldots, \sigma_{n+1} \rangle$. Assume, on the contrary, that there is a permutation $\tau \in \text{AUT}_{\Omega}(f) - \langle \sigma_1, \ldots, \sigma_{n+1} \rangle$. We distinguish two cases.
Case 1. $(\exists i \in \Omega_0)(\exists j \in \Omega_1)(\tau(i) = j)$.
Let $w \in \{0,1\}^{\Omega}$ be defined by $w \upharpoonright \Omega_1 = 0^k$, and

$$(w \upharpoonright \Omega_0)(\ell) = \begin{cases} 0 \text{ if } \ell \neq i \\ 1 \text{ if } \ell = i \end{cases}$$

for $\ell \in \Omega_0$. Since f is a special boolean function, by using the fact that $1 = f_0(0^m) \neq f_1(0^k) = 0$, we obtain that $f(w) = 0 \neq f(w^\tau) = 1$, which is a contradiction.
Case 2. $(\forall i \in \Omega_0)(\tau(i) \in \Omega_0)$.
Put $\tau_0 = (\tau \restriction \Omega_0) \in \text{AUT}_{\Omega_0}$ and $\tau_1 = (\tau \restriction \Omega_1) \in \text{AUT}_{\Omega_1}$. By hypothesis, for all $w \in 2^\Omega$, we have that

$$f(w) = f_0(w \restriction \Omega_0) \cdot f_1(w \restriction \Omega_1) = f(w^\tau) = f_0((w \restriction \Omega_0)^{\tau_0}) \cdot f_1((w \restriction \Omega_1)^{\tau_1}),$$

which implies $\tau_0 \in \text{AUT}_{\Omega_0}(f_0)$ and $\tau_1 \in \text{AUT}_{\Omega_1}(f_1)$.

This completes the proof of the lemma.

An immediate consequence of the previous lemma is the following.

Lemma 3.5.2. *If G, H have disjoint support and are specially representable, then $G \times H$ is specially representable.*

In view of Theorem 3.2.1, we know that the cyclic group $\langle(1, 2, \ldots, n)\rangle$ is 2-representable exactly when $n \neq 3, 4, 5$. In particular, the groups $\langle(1, 2, 3)\rangle$, $\langle(1, 2, 3, 4)\rangle$, $\langle(1, 2, 3, 4, 5)\rangle$ are not representable. The following lemma may be somewhat surprising, since it implies that the group $\langle(1, 2, 3)(4, 5, 6)\rangle$, though isomorphic to $\langle(1, 2, 3)\rangle$, *is* strongly representable.

Lemma 3.5.3. *Let the cyclic group G be generated by a permutation σ, which is the product of two disjoint cycles of lengths ℓ_1, ℓ_2, respectively. Then G is specially representable exactly when the following conditions are satisfied:*
$(\ell_1 = 3 \Rightarrow 3|\ell_2)$ *and* $(\ell_2 = 3 \Rightarrow 3|\ell_1)$,$(\ell_1 = 4 \Rightarrow \gcd(4, \ell_2) \neq 1)$ *and*
$(\ell_2 = 4 \Rightarrow \gcd(4, \ell_1) \neq 1)$, $(\ell_1 = 5 \Rightarrow 5|\ell_2)$ *and* $(\ell_2 = 5 \Rightarrow 5|\ell_1)$.

Proof. It is clear that the assertion of the lemma will follow if we can prove that the three assertions below are true.

1. The groups $\langle(1, 2, \ldots, n)(n+1, n+2, \ldots, kn)\rangle$ are specially representable when $n = 3, 4, 5$.
2. The groups $\langle(1, 2, 3, 4)(5, \ldots, m + 4)\rangle$ are specially representable when gcd $(4, m) \neq 1$.
3. Let m, n be given integers, such that either $m = n = 2$, or $m = 2$ and $n \geq 6$, or $n = 2$ and $m \geq 6$, or $m, n \geq 6$. Then $\langle(1, 2, \ldots, m)(m + 1, m + 2, \ldots, m + n)\rangle$ is specially representable.

Proof of (1). We give the proof only for the case $n = 5$ and $k = 2$. The other cases $n = 3$, $n = 4$ and $k \geq 3$ are treated similarly. Let $\sigma = \sigma_0\sigma_1$, where $\sigma_0 = (1, 2, 3, 4, 5)$ and $\sigma_1 = (6, 7, 8, 9, 10)$. From the proof of Theorem 3.2.3, we know that

$$D_5 = \text{AUT}_5(L') = \text{AUT}_5(L''),$$

where $L' = 0^*1^*0^* \cup 1^*0^*1^*$ and $L'' = \{w \in L' : |w|_0 \geq 1\}$. Let L consist of all words w of length 10 such that

- either $|w|_1 = 1$,
- or $|w|_1 = 2$ and $(\exists 1 \leq i \leq 5)(w_i = w_{5+i}$ and $(\forall j \neq i, 5+i)(w_j = 0))$,
- or $|w|_1 = 3$ and $(\exists 0 \leq i \leq 4)(w = (1000011000)^{\sigma^i}$ or $w = (1100010000)^{\sigma^i})$,
- or $|w|_1 = 3$ and $w_1 \cdots w_5 \in L'$ and $w_6 ... w_{10} \in L''$.

Claim. $\langle(1,2,3,4,5)(6,7,8,9,10)\rangle = \textsc{aut}_{10}(L)$.

Proof of Claim. The containment from left to right is clear. For the containment from right to left, i.e., $\textsc{aut}_{10}(L) \subseteq \langle(1,2,3,4,5)(6,7,8,9,10)\rangle$, suppose that $\tau \in \textsc{aut}_{10}(L)$, but that, on the contrary, there exists an $1 \leq i \leq 5$ and a $6 \leq j \leq 10$ such that $\tau(i) = j$. Let the word w be defined such that $w_\ell = 0$, if $\ell = j$, and $= 1$ otherwise. From the fact that $0^5 \notin L''$, and the last clause in the definition of L, it follows that $w \notin L$ and $w^\tau \in L$, contradicting the assumption $\tau \in \textsc{aut}_{10}(L)$. Thus τ is the product of two disjoint permutations τ_0 and τ_1 acting on $1,2,\ldots,5$ and $6,7,\ldots,10$, respectively. Hence from the last clause in the definition of L we have that $\tau_0 \in D_5$ and $\tau_1 \in \pi^{-1}D_5\pi$, where $\pi(i) = 5+i$, for $i = 1,\ldots,5$. Let $\rho_0 = (1,5)(2,4)$ and $\rho_1 = (6,10)(7,9)$ be the reflection permutations on $1,2,\ldots,5$ and $6,7,\ldots,10$, respectively. To complete the proof of (1), it is enough to show that none of the permutations $\rho_0, \rho_1, \rho_0\rho_1, \rho_0\sigma_1^i, \sigma_0^i\rho_1, \sigma_0^i\sigma_1^j$, for $i \neq j$, belongs to $\textsc{aut}_{10}(L)$. To see this let $x = 1000011000 \in L$. Then if $\tau = \rho_0, \rho_1, \rho_0\rho_1, \rho_0\sigma_1^i$, for any $i = 1,2,3,5$ or $\tau = \sigma_0^i\rho_1$ for $i = 1,2,4,5$, then it is easily seen that $x^\tau \notin L$. Now, let $x = 110001000$. Then for $\tau = \rho_0\sigma_1^4$ and $\tau = \sigma_0^3\rho_1$ it is easy to check that $x^\tau \notin L$. Finally, for $x = 1000010000 \in L$ and $\sigma_0^i\sigma_1^j$, where $i \neq j$, we have that $x^\tau \notin L$. This completes the proof of part (1) of the lemma.

Proof of (2). Put $\sigma_0 = (1,2,3,4)$, $\sigma_1 = (5,6,\ldots,m+4)$, $\sigma = \sigma_0\sigma_1$. Let L be the set of words of length $m+4$ such that

- either $|w|_1 = 1$,
- or $|w|_1 = 2$ and $(\exists 0 \leq i \leq \mathrm{lcm}(4,m) - 1)(w = (100010^{m-1})^{\sigma^i})$,
- or $|w|_1 = 3$ and $(\exists 0 \leq i \leq \mathrm{lcm}(4,m) - 1)(w = (110010^{m-1})^{\sigma^i})$,
- or $|w|_1 > 3$ and $w_1 \cdots w_4 \in L'$ and $w_5 \cdots w_{m+5} \in L''$,

where $L' = 0^*1^*0^* \cup 1^*0^*1^*$ and L'', as given by Theorem 3.2.1, satisfies $\textsc{aut}_m(L'') = C_m$, and moreover, for all $i \geq 1$, $0^i \notin L''$. Clearly, $\langle(1,2,3,4)(5,6,\ldots,m+4)\rangle \subseteq \textsc{aut}_{m+4}(L)$. It remains to prove that

$$\textsc{aut}_{m+4}(L) \subseteq \langle(1,2,3,4)(5,6,\ldots,m+4)\rangle.$$

Let $\tau \in \langle(1,2,3,4)(5,6,\ldots,m+4)\rangle$. As before, τ can be decomposed into $\tau = \tau_0\tau_1$, where $\tau_0 \in D_4$, $\tau_1 \in \pi^{-1}D_m\pi$, and $\pi(i) = 4+i$ for $i = 1,2,\ldots,m$. Let $\rho = (1,4)(2,3)$ be the reflection on $1,2,3,4$. It suffices to show that none of the permutations $\rho\sigma_1^i, \sigma_0^i\sigma_1^j$, for $i \not\equiv \bmod 4$ are in $\textsc{aut}_{m+4}(L)$. Indeed, if $\tau = \sigma_0^i\sigma_1^j$, then let $x = 100010^{m-1}$. It is clear that $x \in L$, but $x^\tau \notin L$. Next assume that $\tau = \rho\sigma_1^i$. We distinguish the following two cases.

Case 1. $m = 4k$, i.e., a multiple of 4.

Let $x = 100010^{m-1}$. Then $x \in L$, but $x^\tau \notin L$ unless $x^\tau = x^{\sigma^j}$ for some j. In this case $j \equiv 3 \bmod 4$ and $j \equiv i \bmod 4k$. So it follows that $i = 3, 7, 11, \ldots, 4k-1$. Now let $y = 110010^{m-1}$. Then $y \in L$, but $y^\tau \notin L$ for the above values of i, unless $y^\tau = y^{\sigma^\ell}$ for some ℓ. In that case we have that $\ell \equiv 2 \bmod 4$ and $\ell \equiv i \bmod 4k$. So it follows that $i = 2, 6, 10, \ldots, 4k-2$. Consequently, $\tau \notin \text{AUT}_{m+4}(L)$.

Case 2. $\gcd(4, m) = 2$.
Let $x = 100010^{m-1}$. Then $x \in L$, but $x^\tau \notin L$ unless $x^\tau = x^{\sigma^j}$ for some j. In this case $j \equiv 3 \bmod 4$ and $j \equiv i \bmod 4k$. So it follows that for even values of i, $\tau \notin \text{AUT}_{m+4}(L)$. Let $y = 110010^{m-1}$. Then $y \in L$, but $y^\tau \notin L$ unless $y^\tau = y^{\sigma^\ell}$ for some ℓ. In that case we have that $\ell \equiv 2 \bmod 4$ and $\ell \equiv i \bmod m$. So it follows that for odd values of i, $\tau \notin \text{AUT}_{m+4}(L)$. This completes the proof of (2).
Proof of (3). A similar technique can be used to generalize the representability result to more general types of cycles.

A straightforward generalization of Lemma 3.5.3 is given without proof in the next lemma.

Lemma 3.5.4. *Let G be a permutation group generated by a permutation σ which can be decomposed into k-many disjoint cycles of lengths $\ell_1, \ell_2, \ldots, \ell_k$, respectively. The group G is specially representable exactly when the following conditions are satisfied for all $1 \leq i \leq k$,*
$\ell_i = 3 \Rightarrow (\exists j \neq i)(3 | \ell_j)$ **and**
$\ell_i = 4 \Rightarrow (\exists j \neq i)(\gcd(4, \ell_j) \neq 1)$ **and**
$\ell_i = 5 \Rightarrow (\exists j \neq i)(5 | \ell_j)$.

The correctness of the algorithm is an immediate consequence of the previous lemmas. This completes the proof of Theorem 3.5.1.

A slightly modified proof of Theorem 3.5.1 can also be found in [Kis99].

3.6 Asymptotics for Invariance Groups

Shannon's theorem from Section 2.2 states that almost all boolean functions require exponential size boolean circuits, and so are as difficult to compute as the hardest boolean function. Since any symmetric language $L \subseteq \{0,1\}^*$ (i.e., for which $\text{AUT}(L_n) = S_n$) can be computed with logdepth fan-in-2 boolean circuits, one might conjecture an inverse relationship between the size (or possibly algebraic structure) of the invariance group $\text{AUT}(f)$ of an n-ary boolean function f, and its boolean complexity. Indeed, we show below that almost all boolean functions have trivial invariance group (i.e., $\text{AUT}(f) = \{id_n\}$, where id_n is the identity permutation in S_n). This yields a type of 0-1 *law* , where for any sequence $\langle G_n \leq S_n n \geq 1 \rangle$ of permutation groups, we prove that the limit $\lim_{n\to\infty} |\{f \in \mathcal{B}_n : \text{AUT}(f) = G_n\}| 2^{-2^n}$ is either 0 or 1.

Theorem 3.6.1. *For any family $\langle G_n : n \geq 1\rangle$ of permutations groups such that each $G_n \leq S_n$*

$$\lim_{n\to\infty} \frac{|\{f \in \mathcal{B}_n : \text{AUT}(f) = \{id_n\}\}|}{2^{2^n}} = \lim_{n\to\infty} \frac{|\{f \in \mathcal{B}_n : \text{AUT}(f) \leq G_n\}|}{2^{2^n}} = 1.$$

Moreover, if $\liminf |G_n| > 1$ *then*

$$\lim_{n\to\infty} \frac{|\{f \in \mathcal{B}_n : \text{AUT}(f) \geq G_n\}|}{2^{2^n}} = \lim_{n\to\infty} \frac{|\{f \in \mathcal{B}_n : \text{AUT}(f) = G_n\}|}{2^{2^n}} = 0.$$

Proof. During the course of this proof we use the abbreviation $\Theta(m) := \Theta_m(\langle(1,2,\ldots,m)\rangle)$. First, we prove the second part of the theorem. By assumption, there exists an n_0, such that for all $n \geq n_0$, $|G_n| > 1$. Hence, for each $n \geq n_0$, G_n contains a permutation of order $k(n) \geq 2$, say σ_n. Without loss of generality we can assume that each $k(n)$ is a prime number. Since $k(n)$ is prime, σ_n is a product of $k(n)$-cycles. If $(i_1, \ldots, i_{k(n)})$ is the first $k(n)$-cycle in this product, then it is easy to see that

$$\Theta_n(\langle\sigma_n\rangle) \leq \Theta_n(\langle(i_1, \ldots, i_{k(n)})\rangle).$$

It follows that

$$\begin{aligned} |\{f \in \mathcal{B}_n : \text{AUT}(f) \geq G_n\}| &\leq |\{f \in \mathcal{B}_n : \sigma_n \in \text{AUT}(f)\}| \\ &= 2^{\Theta_n(\sigma_n)} \\ &\leq 2^{\Theta(k(n))\cdot 2^{n-k(n)}}. \end{aligned}$$

Pólya's cycle index formulas have been worked out for particular permutation groups, including the cyclic groups. In particular from [Ber71], we have the formula

$$\Theta(m) = \frac{1}{m} \cdot \sum_{k|m} \phi(k) \cdot 2^{m/k}$$

which gives the Pólya cycle index of the group $\langle(1,2,\ldots,m)\rangle$ acting on the set $\{1,2,\ldots,m\}$, where $\phi(k)$ is Euler's totient function.

However, it is easy to see that for k prime

$$\frac{\Theta(k)}{2^k} = \frac{1}{k} + \frac{2}{2^k} - \frac{2}{k2^k}.$$

In fact, the function on the right-hand side of the above equation is decreasing in k. Hence, for k prime,

$$\frac{\Theta(k)}{2^k} \leq \frac{\Theta(2)}{2^2} = \frac{3}{4}.$$

It follows that

$$\frac{|\{f \in \mathcal{B}_n : \text{AUT}(f) \geq G_n\}|}{2^{2^n}} \leq 2^{2^n\cdot[\Theta(k(n))\cdot 2^{-k(n)}-1]} \leq 2^{-2^{n-2}}.$$

Since the right-hand side of the above inequality converges to 0, the proof of the second part of the theorem is complete. To prove the first part notice, that

$$\{f \in \mathcal{B}_n : \text{AUT}(f) \neq id_n\} \subseteq \bigcup_{\sigma \neq id_n} \{f \in \mathcal{B}_n : \sigma \in \text{AUT}(f)\},$$

where σ ranges over cyclic permutations of order a prime number $\leq n$. Since there are at most $n!$ permutations on n letters we obtain from the last inequality that

$$\frac{|\{f \in \mathcal{B}_n : \text{AUT}(f) \neq \{id_n\}\}|}{2^{2^n}} \leq n! \cdot 2^{-2^{n-2}} = 2^{O(n \log n)} \cdot 2^{-2^{n-2}} \to 0,$$

as desired.

As a consequence of the above theorem we obtain that asymptotically almost all boolean functions have trivial invariance group.

An interesting generalization of Theorem 3.6.1 has been given by M. Clausen [Cla91]. Consider the group $GL(n,2)$ of invertible $n \times n$ matrices with entries $0, 1$. Let $K_n \leq GL(n,2)$ and for any boolean function $f \in \mathcal{B}_n$ define $K_n(f) = \{A \in K_n : (\forall x \in \{0,1\}^n) \left[f(A^{-1}x) = f(x)\right\}\right]$. We mention without proof (see Exercise 3.11.20) the following result.

Theorem 3.6.2 ([Cla91]). *Let $H_n \leq K_n \leq G_n(n,2)$ be such that $H_n > 1$ for all but a finite number of n. Then*

$$\lim_{n\to\infty} \frac{|\{f \in \mathcal{B}_n : K_n(f) = \{id_n\}\}|}{2^{2^n}} = \lim_{n\to\infty} \frac{|\{f \in \mathcal{B}_n : K_n(f) \leq H_n\}|}{2^{2^n}} = 1,$$

and

$$\lim_{n\to\infty} \frac{|\{f \in \mathcal{B}_n : K_n(f) \geq H_n\}|}{2^{2^n}} = \lim_{n\to\infty} \frac{|\{f \in \mathcal{B}_n : K_n(f) = H_n\}|}{2^{2^n}} = 0.$$

Note that 0-1 laws of the type described in Theorem 3.6.1 have been studied extensively in many branches of mathematical logic. For example, in Exercise 3.11.21, we state Fagin's 0-1 law for graphs.

3.7 Almost Symmetric Languages

In this section, we study the complexity of languages $L \in L(\mathbf{P})$. These are languages whose invariance groups have polynomial index; i.e., $|S_n : \text{AUT}_n(L)| = n^{O(1)}$. Using the classification results on finite simple groups, we will prove that languages in $L(\mathbf{P})$ are precisely the *almost symmetric* languages. The following result is proved by applying the intricate NC algorithm of [BLS87] for permutation group membership. By delving into a deep result in classification theory of finite simple groups, we later improve the conclusion to that of Theorem 3.7.3. For clarity however, we present the following theorem.

Theorem 3.7.1 ([CK91]). *For any language $L \subseteq \{0,1\}^*$, if $L \in L(\mathbf{P})$ then L is in non-uniform* NC.

Proof. As a first step in the proof we will need the following claim.
CLAIM. There is an NC^1 algorithm which, when given $x \in \{0,1\}^n$, outputs $\sigma \in S_n$ such that $x^\sigma = 1^m 0^{n-m}$, for some m.
Proof of Claim. We first illustrate the idea of proof by an example. Suppose that $x = 101100111$. By simultaneously going from left to right and from right to left, we swap an "out-of-place" 0 with an "out-of-place" 1, keeping track of the respective positions. (This is a well-known trick for improving the efficiency of the "partition" or "split" algorithm used in quick-sort.) This gives rise to the desired permutation σ. In the case at hand we find $\sigma = (2,9)(5,8)(6,7)$ and $x^\sigma = 1^6 0^3$.

Now we proceed with the proof of the claim. For $b \in \{0,1\}$, define the predicate $E_{k,b}(u)$, to hold when there are exactly k occurrences of b in the word u. The predicates $E_{k,b}$ are obviously computable in constant depth, polynomial size threshold circuits, i.e., in TC^0. By work of Ajtai, Komlós, and Szemerédi [AKS83], we have $\text{TC}^0 \subseteq \text{NC}^1$. For $k = 1, \ldots, \lfloor n/2 \rfloor$ and $1 \leq i < j \leq n$, let $\alpha_{i,j,k}$ be a log depth circuit which outputs 1 exactly when the k-th "out-of-place" 0 is in position i and the k-th "out-of-place" 1 is in position j. It follows that $\alpha_{i,j,k}(x) = 1$ if and only if "there exist $k-1$ zeros to the left of position i, the i-th bit of x is zero and there exist k ones to the right of position i" and "there exist $k-1$ ones to the right of position j, the j-th bit of x is one and there exist k zeros to the left of position j". This in turn is equivalent to

$$E_{k-1,0}(x_1, \ldots, x_{i-1}) \text{ and } x_i = 0 \text{ and } E_{k,1}(x_{i+1}, \ldots, x_n) \text{ and}$$

$$E_{k-1,1}(x_{j+1}, \ldots, x_n) \text{ and } x_j = 1 \text{ and } E_{k,0}(x_1 \ldots x_{j-1}).$$

This implies that the required permutation can be defined by

$$\sigma = \prod \left\{ (i,j) : i < j \text{ and } \bigvee_{k=1}^{\lfloor n/2 \rfloor} \alpha_{i,j.k} \right\}.$$

Converting the $\vee$-gate of fan-in $\lfloor n/2 \rfloor$ into a $\log(\lfloor n/2 \rfloor)$ depth tree of $\vee$-gates of fan-in 2, we obtain an NC^1 circuit to compute σ. This completes the proof of the claim.

Next we continue with the proof of the main theorem. Put $G_n = S_n(L)$ and let $R_n = \{h_1, \ldots, h_q\}$ be a complete set of representatives for the left cosets of G_n, where $q \leq p(n)$ and $p(n)$ is a polynomial such that $|S_n : G_n| \leq p(n)$. Fix $x \in \{0,1\}^n$. By the previous claim there is a permutation σ which is the product of disjoint transpositions and an integer $0 \leq k \leq n$ such that $x^\sigma = 1^k 0^{n-k}$. Since σ is its own inverse, $x = (1^k 0^{n-k})^\sigma$. In parallel for $i = 1, \ldots, q$ test whether $h_i^{-1}\sigma \in G_n$ by using the principal result of [BLS87], thus determining i such that $\sigma = h_i g$, for some $g \in G_n$. Then we obtain that

$$L_n(x) = L_n((1^k0^{n-k})^\sigma) = L_n((1^k0^{n-k})^{h_i g}) \ = \ L_n((1^k0^{n-k})^{h_i}).$$

By hardwiring the polynomially many values $L_n(1^k0^{n-k})^{h_i})$, for $0 \ \leq k \leq n$ and $1 \leq i \leq q$, we produce a family of polynomial size, polylogarithmic depth boolean circuits for L.

Theorem 3.7.1 involves a straightforward application of the beautiful NC algorithm of Babai, Luks and Seress [BLS87] for testing membership in a finite permutation group. By using the deep structure consequences of the O'Nan-Scott theorem below, together with Bochert's result on the size of the index of primitive permutation groups we can improve the NC algorithm of Theorem 3.7.1 to an optimal TC^0 (and hence NC^1) algorithm. First, we take the following discussion and statement of the O'Nan-Scott theorem from [KL88], page 376.

Let $I = \{1, 2, \ldots, n\}$ and let S_n act naturally on I. Consider all subgroups of the following five classes of subgroups of S_n.

α_1: $S_k \times S_{n-k}$, where $1 \leq k \leq n/2$,
α_2: $S_a \wr S_b$, where either ($n = ab$ and $a, b \geq 1$) or ($n = a^b$ and $a \geq 5, b \geq 2$),
α_3: the affine groups $AGL_d(p)$, where $n = p^d$,
α_4: $T^k \cdot (Out(T) \times S_k)$, where T is a non-abelian simple group, $k \geq 2$ and $n = |T|^{k-1}$, as well as all groups in the class
α_5: almost simple groups acting primitively on I.[4]

Theorem 3.7.2 (O'Nan-Scott). *Every subgroup of S_n not containing A_n is a member of $\alpha_1 \cup \cdots \cup \alpha_5$.*

Now we can improve the result of Theorem 3.7.1 in the following way.

Theorem 3.7.3 ([CK91]). *For any language $L \subseteq \{0,1\}^*$, if $L \in L(\mathbf{P})$ then $L \in \text{TC}^0$, hence $L \in \text{NC}^1$.*

Proof. The proof requires the following consequence of the O'Nan-Scott theorem.

Lemma 3.7.1 ([CK91]). *Suppose that $\langle G_n \leq S_n : n \geq 1 \rangle$ is a family of permutation groups, such that for all n, $|S_n : G_n| \leq n^k$, for some k. Then for sufficiently large N, there exists an $i_n \leq k$ for which $G_n = U_n \times V_n$ with the supports of U_n, V_n disjoint and $U_n \leq S_{i_n}, V_n = S_{n-i_n}$.*

Before proving the lemma, we complete the details of the proof of Theorem 3.7.3. Apply the lemma to $G_n = \text{AUT}_n(L)$ and notice that given $x \in \{0,1\}^n$, the question of whether x belongs to L is decided completely by the number

[4] Consider a permutation group G acting on a nonempty set X. A subset B of X is called a block if for all $g \in G$ the sets B and B^g are either equal or disjoint. The empty set, X itself, and all singletons of X are blocks (also called trivial blocks). A transitive permutation group G with no non-trivial blocks is called primitive.

of 1s in the support of $K_n = S_{n-i_n}$ together with information about the action of a finite group $H_n \leq S_{i_n}$, for $i_n \leq k$. Using the counting predicates as in the proof of Theorem 3.7.1, it is clear that appropriate TC^0 circuits can be built. This completes the proof of Theorem 3.7.3, assuming Lemma 3.7.1.

Proof. We have already observed that $G_n \neq A_n$. By the O'Nan-Scott theorem, G_n is a member of $\alpha_1 \cup \cdots \cup \alpha_5$. Using Bochert's theorem on the size of the index of primitive permutation groups (if a primitive permutation group $H \leq S_n$ does not contain the alternating group A_n, then $|S_n : H| \geq \lfloor (n+1)/2 \rfloor!$ [Wie64]), the observations of [LPS88] concerning the primitivity of the maximal groups in $\alpha_3 \cup \alpha_4 \cup \alpha_5$ and the fact that G_n has polynomial index with respect to S_n, we conclude that the subgroup G_n cannot be a member of the class $\alpha_3 \cup \alpha_4 \cup \alpha_5$. It follows that $G_n \in \alpha_1 \cup \alpha_2$. We show that in fact $G_n \notin \alpha_2$. Assume on the contrary that $G_n \leq H_n = S_a \wr S_b$. It follows that $|H_n| = a!(b!)^a$. We distinguish the following two cases.

Case 1. $n = ab$, for $a, b > 1$.
In this case it is easy to verify using Stirling's formula

$$(n/e)^n \sqrt{n} < n! < (n/e)^n 3\sqrt{n}$$

that

$$|S_n : H_n| = \frac{n!}{a!(b!)^a} \sim \frac{a^{n-a}}{3b^{a/2}(3/a)^a \sqrt{a}}.$$

Moreover, it is clear that the right-hand side of this last inequality cannot be asymptotically polynomial in n, since $a \leq n$ is a proper divisor of n, which is a contradiction.

Case 2. $n = a^b$, for $a \geq 5, b \geq 2$.
A similar calculation shows that asymptotically

$$|S_n : H_n| = \frac{n!}{a!(b!)^a} = \frac{n!}{a!(b'!)^a},$$

where $b' = a^{b-1}$. It follows from the argument of case 1 that this last quantity cannot be asymptotically polynomial in n, which is a contradiction. It follows that $G_n \in \alpha_1$. Let $G_n \leq S_{i_n} \times S_{n-i_n}$, for some $1 \leq i_n \leq n/2$.

We claim that there exists a constant k, for which $i_n \leq k$, for all but a finite number of ns. Indeed, notice that

$$|S_n : S_i \times S_{n-i}| = \frac{n!}{i!(n-i)!} = \Omega(n^i) \leq |S_n : G_n| \leq n^k,$$

which proves that $i_n \leq k$. It follows that $G_n = U_n \times V_n$, where $U_n \leq S_{i_n}$ and $V_n \leq S_{n-i_n}$. Since $i_n \leq k$ and $|S_n : G_n| \leq n^k$, we have that for n large enough, $V_n = S_{n-i_n}$. This completes the proof of the claim. Now let $L \subseteq \{0,1\}^*$ have polynomial index. Given a word $x \in \{0,1\}^n$, in TC^0, one can test whether the number of 1s occurring in the $n - i_n$ positions (where

$V_n = S_{n-i_n}$) is equal to a fixed value, hardwired into the n-th circuit. This, together with a finite look-up table corresponding to the U_n part, furnishes a TC^0 algorithm for testing membership in L.

3.8 Symmetry and Complexity

In [CK91], by adapting the counting argument of [Lup61a], it was shown that for any superpolynomial function f, there exist languages $L \subseteq \{0,1\}^*$ whose invariance groups G_n have at most $f(n)$ orbits when acting on $\{0,1\}^n$ and yet L is not computable in polynomial size circuits. Against this negative result it was there conjectured that if $L \subseteq \{0,1\}^*$ is a language whose invariance groups have polynomially many orbits ($\Theta_n(L_n) \leq n^{O(1)}$) then L is computable in non-uniform NC. Babai, Beals and Takácsi-Nagy [BBTN92] proved this conjecture by developing some very elegant structure theory for groups having polynomially many orbits. As an additional corollary, they obtained an NC solution of a specific case of the bounded valency graph isomorphism problem.

For group $G \leq S_n$ and words $x, y \in \{0,1\}^n$, recall the group action If $G \leq S_n$ is a permutation group, then recall the action of G on the collection of n-length words; namely, for $x, y \in \{0,1\}^n$, we write $x \sim y \bmod G$ to assert the existence of σ in G for which $x^\sigma = y$. The *orbit* of x is $\{y \in \{0,1\}^n : x \sim y\}$. We define the ORBIT PROBLEM for group $G \leq S_n$ as follows.

Input: $x, y \in \{0,1\}^n$
Output: Whether $x \sim y \bmod G$.

For families $\mathcal{G} = \langle G_n : G_n \leq S_n \rangle$ and $\mathcal{H} = \langle H_n : H_n \leq S_n \rangle$, we write $\mathcal{H} \leq \mathcal{G}$ to indicate $H_n \leq G_n$ for all $n \in \mathbf{N}$. Let $\Theta(G_n)$ be the number of orbits of G_n acting on $\{0,1\}^n$. For simplicity, we write G instead of $\mathcal{G}$ and suppress indices n in G_n. We also use the notation $Sym(\Omega)$ for the of permutations on the set Ω.

Proposition 3.8.1. *If $\mathcal{H} \leq \mathcal{G}$ and $\Theta(H) \leq n^{O(1)}$, then the orbit problem for $\mathcal{G}$ is AC^0 reducible to the orbit problem for $\mathcal{H}$.*

Proof. Since H_n is a subgroup of G_n, every H_n orbit is contained in a G_n orbit. There are at most $p(n)$ many orbits of H_n acting on 2^n, so

$$x \sim y \bmod G_n \iff \bigvee_{i=1}^{p(n)} x \sim y_i \bmod H_n$$

where $y_1, \ldots, y_{p(n)}$ are fixed representatives for those H_n orbits contained in the G_n orbit of y.

The following proposition lists some elementary facts about the number of orbits of a group with permutation domain Ω, when acting on the power set of Ω.

Proposition 3.8.2. *Let G, H be permutation groups.*

1. *If $H \leq G$ then $\Theta(H) \geq \Theta(G)$.*
2. *Assuming that G, H are have disjoint supports, $\Theta(G \times H) = \Theta(G) \cdot \Theta(H)$.*
3. *$\Theta(H \wr S_k) = \binom{\Theta(H)+k-1}{k} = \binom{\Theta(H)+k-1}{\Theta(H)-1}$*
4. *For $k \geq 3$, $\Theta(A_k) = \Theta(S_k)$ and $\Theta(H \wr A_k) = \Theta(H \wr S_k)$.*

Proof. of (1). Clear since every H orbit is contained in a G orbit.
Proof of (2). Straightforward.
Proof of (3). If the degree of H is m, then recall that the *wreath product* $H \wr S_k$ is given by the collection of permutations $\pi \in Sym(A \times B)$, where $|A| = m$, $|B| = k$ and $\pi = \langle \sigma_1, \ldots, \sigma_k; \tau \rangle$ for $\sigma_1, \ldots, \sigma_k$ independent permutations in H and τ in S_k. The action of π on the permutation domain $A \times B$ is given by $(i, j)^\pi = (i^{\sigma_j}, j^\tau)$.

Claim. There is a 1-1 correspondence between $\Theta(H \wr S_k)$ and the collection of all non-decreasing maps from $\{1, \ldots, k\}$ into $\{1, \ldots, \Theta(H)\}$.

Proof of Claim. Temporarily define a *canonical ordering* on $\{0,1\}^m$ as follows. For $x, y \in \{0,1\}^m$, let $x \prec y$ iff the weight $|x|_1$ of x is less than the weight $|y|_1$ of y or x, y have equal weights and x precedes y in the lexicographic ordering. Define $x \in \{0,1\}^m$ to be a *canonical representative* of an orbit of H if for all lexicographically smaller $y \in \{0,1\}^m$, $y \not\sim x \bmod H$. Let $\phi : \{0,1\}^m \to \{0,1\}^m$ by setting $\phi(u)$ to be that canonical representative lying in the same H-orbit as u. Let $\{x_1, \ldots, x_{\Theta(H)}\}_\prec$ be a listing of the canonical representatives of the orbits of H acting on $\{0,1\}^m$. Now given $u \in \{0,1\}^{mk}$, where $u = u_1 \cdots u_k$, and each $u_i \in \{0,1\}^m$, determine a permutation $\sigma \in S_k$ for which

$$\phi(u_{\sigma(1)}) \preceq \phi(u_{\sigma(2)}) \preceq \cdots \preceq \phi(u_{\sigma(k)}).$$

The claim now readily follows.

It is well-known (see for instance [Ber71]), that the number of non-decreasing maps from k into m is equal to the number of ways of choosing k objects from a collection of m objects, *allowing repetitions*, given by

$$\frac{(m+k-1)\cdots(m+1)(m)}{k!} = \binom{m+k-1}{k}.$$

Since we have established a 1-1 correspondence between $\Theta(H \wr S_k)$ and the collection of all non-decreasing maps from $\{1, \ldots, k\}$ into $\{1, \ldots, \Theta(H)\}$, it follows that $\Theta(H \wr S_k) = \binom{\Theta(H)+k-1}{k}$. Using the symmetry of the binomial coefficients, i.e., that $\binom{n}{k} = \binom{n}{n-k}$, the equality $\Theta(H \wr S_k) = \binom{\Theta(H)+k-1}{\Theta(H)-1}$ is immediate.

Proof of (4). Suppose that $x, y \in \{0,1\}^k$ and $x^\sigma = y$ for some $\sigma \in S_k$. If $\sigma \in A_k$, then let $\overline{\sigma} = \sigma$, otherwise, since $k \geq 3$, let τ be the transposition interchanging i, j, where $x_i = x_j$ and set $\overline{\sigma} = \sigma \circ \tau$. Then $\overline{\sigma} \in A_k$ and $x^{\overline{\sigma}} = y$. It follows that $x, y \in \{0,1\}^k$ are in the same S_k orbit iff they are in the same A_k orbit. The assertion for $H \wr A_k$ and $H \wr S_k$ is similarly proved. This concludes the proof of the Proposition.

Lemma 3.8.1 ([BBTN92]). *If $G \leq H \wr S_k$ and $\Theta(G) \leq n^c$, then*

$$\min(\Theta(H) - 1, k) \leq 2c.$$

Proof. Since $G \leq H \wr S_k$, Proposition 3.8.2 implies that $\Theta(G) \geq \Theta(H \wr S_k)$.
Case 1. $k \leq \Theta(H) - 1$.
Noting that for $a, b \geq 1$, and $i \geq 0$

$$\frac{a+b-i}{a-i} \geq \frac{a+b}{a}$$

so that

$$\binom{a+b}{a} \geq \left(\frac{a+b}{a}\right)^a$$

it follows that

$$\Theta(H \wr S_k) = \binom{\Theta(H)+k-1}{k} \geq \binom{2k}{k} \geq 2^k.$$

Thus $k \leq \log \Theta(G)$. For sufficiently large n, $n/(c \cdot \log(n))^2 \geq \sqrt{n}$, so

$$\begin{aligned} n^c &\geq \Theta(G) \\ &\geq \binom{\Theta(H)+k-1}{k} \\ &\geq \binom{\Theta(H)}{k} \\ &\geq \binom{n/k}{k} \\ &\geq \left(\frac{n}{k^2}\right)^k \\ &\geq n^{k/2} \end{aligned}$$

Hence $k \leq 2c$.
Case 2. $k > \Theta(H) - 1$.

$$\begin{aligned} \Theta(H \wr S_k) &= \binom{\Theta(H)+k-1}{\Theta(H)-1} \\ &\geq \binom{2 \cdot (\Theta(H)-1)}{\Theta(H)-1} \\ &\geq 2^{\Theta(H)-1} \end{aligned}$$

so $\Theta(H) - 1 \leq \log \Theta(G)$. Thus

$$\begin{aligned} n^c &\geq \Theta(G) \\ &\geq \binom{\Theta(H)+k-1}{\Theta(H)-1} \\ &\geq \binom{k}{\Theta(H)-1} \\ &\geq \binom{n/(\Theta(H)-1)}{\Theta(H)-1} \\ &\geq \left(\frac{n}{(\Theta(H)-1)^2}\right)^{\Theta(H)-1} \\ &\geq n^{(\Theta(H)-1)/2} \end{aligned}$$

Hence $\Theta(H) - 1 \leq 2c$.

We require some definitions in order to establish structure results for groups having polynomially many orbits.

Definition 3.8.1. *A subset $\Delta \subseteq \Omega$ is a block of imprimitivity of group $G \leq Sym(\Omega)$ if for every $\sigma \in G$, $\Delta^\sigma = \Delta$ or $\Delta^\sigma \cap \Delta = \emptyset$. The group G is primitive if the only blocks of imprimitivity of G are Ω and the singleton subsets of Ω. The group $G \leq Sym(\Omega)$ is transitive if for every $x, y \in \Omega$, there is $\sigma \in G$ such that $x^\sigma = y$.*

It is clear that if G is transitive and $\Delta_1, \ldots, \Delta_m$ is a system of blocks of imprimitivity, then all blocks have the same number of elements. Notice that for $G \leq S_n$, we distinguish between G acting on its permutation domain $\{1, \ldots, n\}$, G acting on the set 2^n of all n-length binary words, and G acting on the set 2^{2^n} of all boolean functions on n variables. A *structure forest* $\mathcal{F}$ for permutation group $G \leq Sym(\Omega)$ is a forest on which G acts as automorphisms such that the leaves form the permutation domain Ω and the roots correspond to orbits. Each node $v \in \mathcal{F}$ is identified with a block $B(v)$ of imprimitivity of G acting on Ω, where $B(v)$ consists of the leaves of $\mathcal{F}$ below v. Let

$$\mathcal{B}(v) = \{B(u) : u \text{ is a child of } v\}$$

Let $L(v) \leq Sym(B(v)$ denote the action of G_v on $B(v)$, and let $H(v) \leq Sym(\mathcal{B}(v))$ denote the action of G_v on $\mathcal{B}(v)$. A node $v \in \mathcal{F}$ is *primitive* if $H(v)$ is primitive, while v is a *giant* if $H(v)$ is the alternating or symmetric group. If G is transitive, then the structure forest is a tree and we write $k_i = |\mathcal{B}(v)|$ for $v \in \mathcal{L}_i$. In the general case where $\mathcal{F}$ is not a tree, we write $k_{i,j} = \mathcal{B}(v)$ where $v \in \mathcal{L}_i$ on tree T_j. The group K_i is the pointwise stabilizer of $\mathcal{L}_i$. Note that K_i is a normal subgroup of G, denoted by $K_i \lhd G$, since K_i is the kernel of the action of G on $\mathcal{L}_i$. If $v \in \mathcal{L}_i$ then $K_i \leq L(v)^{|\mathcal{L}_i|}$.

Theorem 3.8.1 ([Bab81]). *Suppose that $G \leq S_n$ is a primitive permutation group of degree n not containing A_n. Then*

$$|G| < \exp\{4\sqrt{n}\log^2(n)\}$$

The proof of this estimate will not be given, but we note that the proof does not use classification theory.

Theorem 3.8.2 (Babai–Pyber). *Suppose that $G \leq Sym(\Omega)$, $|\Omega| = n$, $\mathcal{F}$ is a primitive structure forest for G. For any $t > 1$, if $\mathcal{F}$ has no giant node of degree strictly greater than t, then*

$$\Theta(G) \geq 2^{n/c_1 t}$$

for some absolute constant c_1.

Proof. Let $\{\Delta_1, \dots, \Delta_m\}$ be the orbits of G acting on Ω. Then $G \leq \Pi_{i=1}^m G^{\Delta_i}$, so $\Theta(G) \geq \Pi_{i=1}^m \Theta(G^{\Delta_i})$. Thus it suffices to prove the theorem for transitive groups G. We may suppose t is sufficiently large to satisfy

$$t^{x-1} \geq \exp\{4\sqrt{x}\log^2(x)\}$$

for all $x \geq 2$. Set $c_2 \geq 8$ and $c_3 = 4c_2$. For t given, let $\Theta_t(n)$ be the minimum value of $\Theta(G)$ as G ranges over all transitive permutation groups of degree n having a primitive structure tree T with no giant node of degree strictly greater than t. For $1 \leq n \leq c_2 t$, it is clear that

$$\Theta_t(n) \geq n + 1 \geq 2 \geq 2^{c_2/c_3} \geq 2^{n/c_3 t}.$$

By induction on n, we show that following claim which immediately implies the statement of the theorem.

Claim. For $n \geq c_2 t$, $\Theta_t(n) \geq t2^{n/c_3 t}$.

Proof of Claim. Suppose that G is a transitive permutation group of degree $n \geq c_2 t$ and T is a primitive structure tree for G with no giant nodes of degree $> t$. Assume the claim holds for values less than n. Collapse all levels below $\mathcal{L}_1$ to a single level. Let $H = H(\text{root})$, $L = L(u)$ for some $u \in \mathcal{L}_1$.

Case 1. $k_1 \geq c_2 t$.

H is of degree k_0, so $|H_0| \leq k_0!$ and for $k_0 > t$, since H is primitive, by Theorem 3.8.1, $|H| \leq \exp\{4\sqrt{k_0}\log^2(k_0)\}$, so $|H| \leq t^{k_0 - 1}$. By the induction hypothesis, as L is of degree $k_1 < n$, $\Theta(L) \geq t \cdot 2^{k_1/c_3 t}$, so

$$\begin{aligned}\Theta(G) &\geq \Theta(K_1)/|H| \geq \Theta(L^{k_0})/|H| = \Theta(L)^{k_0}/|H| \\ &\geq (t2^{k_1/c_3 t})^{k_0}/t^{k_0-1} = t2^{k_1 k_0/c3t} = t2^{n/c3t}.\end{aligned}$$

Case 2. $k_1 < c_2 t \leq k_0$.
By Theorem 3.8.1,

$$\begin{aligned}|H| &\leq e^{4\sqrt{k_0}\log^2(k_0)} \leq 2^{4\log(e)\sqrt{k_0}\log^2(k_0)} \\ &\leq 2^{8\sqrt{k_0}\log^2(k_0)} \leq 2^{k_0/2}.\end{aligned}$$

Also,

$$2^{n/c_3 t} = 2^{k_0 k_1/c_3 t} < 2^{k_0 c_2 t/c_3 t} = 2^{k_0/4}.$$

Thus

$$\begin{aligned}\Theta(G) &\geq \Theta(s)^{k_0}/|H| \geq 2^{k_0}/|H| \geq 2^{k_2/2} \\ &\geq 2^{k_0/4} \cdot 2^{n/c_3 t} \geq t \cdot 2^{n/c_3 t}.\end{aligned}$$

Case 3. $k_0, k_1 < c_2 t$.
Then $G \leq S_{k_1} \wr S_{k_0}$ so

$$\Theta(G) \geq \Theta(S_{k_1} \wr S_{k_0}) = \binom{k_1 + k_0}{k_0} = \binom{k_0 + k_1}{k_1}.$$

By symmetry, we can assume that $k_0 \leq k_1$. As $n = k_0 k_1 \geq c_2 t$ and $k_0, k_1 < c_2 t$, it follows that $2 \leq k_0 \leq k_1$. Hence

$$\begin{aligned}\Theta(G) &\geq \binom{k_0 + k_1}{k_0} = (\frac{k_1 + k_0}{k_0})(\frac{k_1 + k_0 - 1}{k_0 - 1}) \cdots (\frac{k_1 + 1}{1}) \\ &\geq 2^{k_0 - 3} \cdot (\frac{k_1 + 3}{3})(\frac{k_1 + 2}{2})(\frac{k_1 + 1}{1}) \\ &\geq 2^{k_0 - 3} k_1^2 \geq 2^{k_0 - 3} k_1 k_0 \geq 2^{k_0 - 3} c_2 t \geq 2^{k_0} t \geq t 2^{(\frac{c_2}{c_3}) k_0} \\ &= t 2^{(\frac{c_2 t}{c_3 t}) k_0} \geq t 2^{n k_0 / c_3 t} \geq t 2^{n / c_3 t}.\end{aligned}$$

This completes the proof of the claim and hence of the theorem.

Corollary 3.8.1. *For transitive group $G \leq Sym(\Omega)$, there exists a depth 3 structure tree T, such that $k_0 k_2 \leq c_1 \log(\Theta(G))$, and the nodes on level 1 of T are giants.*

Proof. Let T' be the primitive structure tree for G and let t be the largest degree of giant nodes in T'. The level of these nodes is called the *explosion level.* Contract all levels above and below the explosion level to one level, keeping the root separate. This produces a depth 3 structure tree T. By Theorem 3.8.2, we have $k_1 \geq \frac{n}{c_1 \log(\Theta(G))}$. Now $n = k_0 k_1 k_2$, so

$$k_0 k_2 = n / k_1 \leq \frac{n}{n / c_1 \log(\Theta(G))} \leq c_1 \log(\Theta(G)).$$

We introduce some definitions. For subgroups H, K of group G, H is said to be a complement of K in G if $H \cap K = 1$ and $HK = G$. Let $B_1, \ldots, B_k$ be a system of blocks of imprimitivity for G. An element $\sigma \in G$ is clean if for all $1 \leq i \leq k$ either $B_i^\sigma \neq B_i$ or σ acts trivially on B_i ($\sigma(x) = x$ for all $x \in B_i$). A subgroup H is clean if it consists only of clean elements; H is a clean complement of K if it is clean and is a complement to K.

Lemma 3.8.2. *If $G \leq Sym(\Omega)$ is a transitive permutation group having a depth 2 structure tree T such that $H(root) = A_{k_0}$ and $k_0 \geq 4k_1$, then K_1 has a clean complement.*

Proof. Let $\mathcal{L}_1 = \{v_1, \ldots, v_{k_0}\}$ be the collection of nodes on the first level of T. For $\tau \in G$, let $\overline{\tau}$ denote the action of τ on $\mathcal{L}_1$. By Bertrand's postulate, there is a prime p satisfying $k_1 < p < k_0/2$. Take $\pi \in G$ such that $\overline{\pi}$ is a p-cycle. Since $k_1 < p$ and the order of an element divides the order of the group to which it belongs, there is an integer m not divisible by p for which $\overline{\pi}^m$ is the identity on $\mathcal{L}_1$, hence π is clean. Without loss of generality, suppose that $m = 1$ and that π permutes $v_1, \ldots, v_p$ cyclically and fixes each of $v_{p+1}, \ldots, v_{k_0}$ and their children. Similarly, there is an element $\pi' \in G$ such that π' permutes $v_p, \ldots, v_{2p-1}$ cyclically and fixes each v_i and its children for i different from $p, \ldots, 2p-1$. By abuse of language, we temporarily call a permutation $\sigma \in G$ a clean 3-cycle if $\overline{\sigma}$ is a 3-cycle permuting cyclically $v_{i_1}, v_{i_2}, v_{i_3}$ while fixing v_i and all its children for i different from i_1, i_2, i_3. It follows that the commutator $\sigma = [\pi, pi'] = \pi\pi'\pi^{-1}\pi'^{-1}$ is a clean 3- cycle and $\overline{\sigma} = (v_{p+1}, v_p, v_1)$. We leave it to the reader to verify that the conjugate $\theta\sigma\theta^{-1}$ of a clean 3-cycle is a clean 3- cycle and that a group generated by clean elements is a clean group. For $1 \leq i < k_0$, let $\sigma_i \in G$ be a clean 3- cycle with $\overline{\sigma_i} = (v_i, v_{i+1}, v_{k_0})$.
Case 1. k_0 is odd.
Then H is generated by $\sigma_1, \sigma_3, \sigma_5, \ldots, \sigma_{k_0-2}$.
Case 2. k_0 is even.
Let $A = \langle \sigma_1, \sigma_3 \rangle_{v_{k_0}}$, consisting of those $\sigma \in G$ generated by σ_1, σ_3 where σ fixes v_{k_0} and its children. Then A is clean. Let H be generated by $A, \sigma_4, \sigma_6, \ldots, \sigma_{k_0-2}$. It follows that H is a clean complement to K_1.

Theorem 3.8.3 ([BBTN92]). *Every language L with transitive automorphism groups* $\textsc{aut}(L_n)$ *and polynomially many cycles, i.e.,* $\Theta(\textsc{aut}(L_n)) \leq n^{O(1)}$*, is in* $\textsc{tc}^0$.

Proof. By Corollary 3.8.1, let T be a depth 3 structure tree where $H(u)$ is a giant for each $u \in \mathcal{L}_1$. Applying the clean complement Lemma 3.8.2 to each $B(u)$ for $u \in \mathcal{L}_1$, there is a clean complement $H_u = \langle 1_{B(u)} \rangle \wr A_{k_1}$ of K_2 with respect to $H(u)$. Thus $H_u K_2 = H(u)$ and $\Pi_{u \in \mathcal{L}_1} H_u \leq G$, so

$$\Theta(\Pi_{u \in \mathcal{L}_1} H_u) = \binom{k_1 + 2^{k_2} - 1}{2^{k_2} - 1}^{k_0} \geq \Theta(G).$$

By Lemma 3.8.1 $k_0, k_2 \leq 2c$ for an absolute constant c, so $\binom{k_1+2^{k_2}-1}{2^{k_2}-1}$ is polynomial in k_1 and hence polynomial in n. The orbit problem for $\Pi_{u \in \mathcal{L}_1} H_u$ is solved essentially by counting, and hence belongs to $\textsc{tc}^0$.

3.9 Applications to Anonymous Networks

The anonymous network was introduced in Section 1.11.8. In this section we concentrate on the study of the bit complexity of computing boolean functions on Rings and Hypercubes.

3.9.1 Rings

Recall that C_N is the cyclic group generated by the cycle $(1, 2, \ldots, N)$ and D_N is the dihedral group generated by the cycle $(1, 2, \ldots, N)$ and the reflection

$$\rho_N = \begin{pmatrix} 1 & 2 & \cdots N \\ N & N-1 & \cdots \ 1 \end{pmatrix}.$$

Let R_N denoted the ring of N processors.

Theorem 3.9.1 ([ASW88]). *Let f be a boolean function in $\mathcal{B}_N$. Then*

1. *f is computable in the oriented ring R_N if and only if* $\textsc{aut}(f) \geq C_N$.
2. *f is computable in the unoriented ring R_N if and only if* $\textsc{aut}(f) \geq D_N$.

Proof. The if part follows easily from the fact that if a boolean function is computable in the network then it must be invariant under its group of automorphisms. So we concentrate on the proof of the other direction.

For the case of oriented rings we have the following algorithm.

```
Algorithm for processor $p$:
send your bit left;
for $N$ steps do
     send the bit you receive from
     the right to the left;
     od
endfor
```

For the case of unoriented rings we have the following algorithm.

```
Algorithm for processor $p$:
send your bit both left and right;
for $\lfloor N/2 \rfloor$ steps do
     send the bit you receive in the direction
     opposite to the one you got it from;
     od
endfor
```

It is easy to see that these algorithms are correct.

3.9.2 Hypercubes

A natural labeling of the hypercube is the following, $\mathcal{L}$: the edge connecting nodes $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ is labeled by i if and only if $x_i \neq y_i$, i.e., $\mathcal{L}(x, y) = \mathcal{L}(y, x) = i$. In this subsection we will refer to a hypercube with this labeling as an canonically labeled hypercube and we will reserve the symbol $\mathcal{L}$ to denote this canonical labeling.

Of particular interest in the case of the canonically labeled hypercube are the *bit-complement* automorphisms that complement the bits of certain components, i.e., for any set $S \subseteq \{1, \ldots, n\}$ let $\phi_S(x_1, \ldots, x_n) = (y_1, \ldots, y_n)$, where $y_i = x_i + 1$, if $i \in S$, and $y_i = x_i$ otherwise (here addition is modulo 2). Let F_n denote the group of bit-complement automorphisms of Q_n.

Theorem 3.9.2. *The group of automorphisms of the canonically labeled hypercube $Q_n[\mathcal{L}]$ is exactly the group F_n of bit-complement automorphisms.*

Proof. Let $\phi \in Aut(Q_n[\mathcal{L}])$. We claim that for all $x, y \in Q_n$,

$$\phi(x) + \phi(y) = x + y. \tag{3.4}$$

Indeed, given x, y there is a path $x_0 := x, x_1, \ldots, x_k := y$ joining x to y. By definition, ϕ must preserve labels, i.e., for all $i < k$, $\phi(x_i) + \phi(x_{i+1}) = x_i + x_{i+1}$. Adding these congruences we obtain $\phi(x_0) + \phi(x_k) = x_0 + x_k$, which proves (3.4). Using (3.4) it is now easy to show that ϕ is uniquely determined from the value of $\phi(0^n)$, say $\phi(0^n) = (p_1, \ldots, p_n)$. It follows easily that $\phi = \phi_S$, where $S = \{1 \le i \le n : p_i \ne 0\}$.

The automorphism group of the unlabeled hypercube Q_n is larger than F_n. For any permutation $\sigma \in S_n$ let $\phi_\sigma(x_1, \ldots, x_n) = (x_{\sigma(1)}, \ldots, x_{\sigma(n)})$ and let P_n denote the group of these automorphisms. We mention without proof that it can be shown easily that P_n is a normal subgroup of $Aut(Q_n)$ and in fact $Aut(Q_n) = F_n \cdot P_n$.

First we characterize the class of boolean functions which are computable in the canonically labeled hypercube in terms of its group of automorphisms and provide an algorithm with bit complexity $O(N^2)$ for computing all such functions.

Theorem 3.9.3. *On the canonically labeled hypercube Q_n of degree n and for any boolean function $f \in \mathcal{B}_N$, $N = 2^n$, f is computable on the hypercube Q_n if and only if f is invariant under the bit-complement automorphisms of Q_n. Moreover, the bit complexity of any such computable function is $O(N^2)$.*

Proof. The "if" part is straightforward so we need only prove the "only if" part. Let $f \in \mathcal{B}_N$ be invariant under all bit-complement automorphisms of the hypercube. The algorithm proceeds by induction on the dimension n of the hypercube. Intuitively, it splits the hypercube into two $n-1$ dimensional hypercubes. The first hypercube consists of all nodes with $x_n = 0$ and the second of all nodes with $x_n = 1$. By the induction hypothesis the nodes of these hypercubes know the entire input configuration of their corresponding hypercubes. Every node in the hypercube with $x_n = 0$ is adjacent to unique node in the hypercube with $x_n = 1$. By exchanging their information all processors will know the entire input configuration and hence they can all compute the value of f on the given input.

More formally, the algorithm is as follows. For any sequences of bits I, J let IJ denote the concatenation of I and J. Let I_p^i denote the input to processor

p at the i-th step of the computation. Initially I_p^0 is the input bit to processor p.

```
**Algorithm for processor** $p$:
**initialize:** $I_p^0$ is the input bit to processor $p$;
**for** $i := 0, \ldots, n-1$ **do**
    **send** message $I_p^i$ to $p$s neighbor $q$ along the $i$-th link
    **let** $I_q^i$ be the message received by $p$ from $p$'s neighbor
    $q$ along the $i$-th link and **put** $I_p^{i+1} := I_p^i I_q^i$;
**od**;
**output** $f(I_p^n)$
```

The algorithm is depicted in Figure 3.3 for a given input. To prove the correctness of the algorithm it must be shown that all processors output the same correct bit, i.e., for all processors p, q, $f(I_p^n) = f(I_q^n)$. Let $I_p = I_p^n$ be the sequence obtained by processor p at the n-th stage of the above algorithm. We call I_p the view of processor p on input I. Let p, q be any two processors of the hypercube. Clearly, there is a unique bit-complement automorphism ϕ satisfying $\phi(p) = q$, namely $\phi = \phi_S$, where $i \in S$ if and only if $p_i \neq q_i$. Now it can be shown that this automorphism will map processor p's view, namely I_p, to the view of processor q, namely I_q. For any sequence $b_x b_{x'} \cdots$ of bits indexed by elements $x, x', \ldots \in Q_n$ define $\phi(b_x b_{x'} \cdots) = b_{\phi(x)} b_{\phi(x')} \cdots$. The proof of correctness is based on the identity

$$\phi(I_p) = I_{\phi(p)}. \tag{3.5}$$

To prove (3.5) it is sufficient to show that for all $i \leq n = \log N$, $\phi(I_p^i) = I_{\phi(p)}^i$. The proof is by induction on $i \leq n$. The result is clear for $i = 0$. Assume the result true for i. Let p', q' be p's and q's neighbors along the i-th edge, respectively. Then by definition we have

$$I_p^{i+1} = I_p^i I_{p'}^i \text{ and } I_q^{i+1} = I_q^i I_{q'}^i.$$

Since ϕ is a bit-complement automorphism and p, p' are connected via the i-th edge it follows that $\phi(p) = q$ and $\phi(p') = q'$. Using the induction hypothesis $\phi(I_p^i) = I_{\phi(p)}^i$ we obtain

$$\phi(I_p^{i+1}) = \phi(I_p^i)\phi(I_{p'}^i) = I_q^i I_{\phi(p')}^i = I_q^{i+1} = I_{\phi(p)}^{i+1}$$

This completes the inductive proof. It follows now that $\phi(I_p) = I_q$ which implies that $f(I_p) = f(I_q)$, since f is invariant under the bit-complement automorphisms of Q_n.

To study the bit complexity of the above algorithm, let $T(N)$ be the number of bits transmitted in order that at the end of the computation all the processors in the hypercube know the input of the entire hypercube. By performing a computation on each of the two $n-1$-dimensional hypercubes we obtain that their nodes will know the entire input corresponding to their nodes in $T(N/2)$ bits. The total number of bits transmitted in this case is $2 \cdot T(N/2)$. The final exchange transmission consists of $N/2$ bits being transmitted by $N/2$ nodes to their $N/2$ corresponding other nodes, for a total of $2 \cdot N/2 \cdot N/2 = N^2/2$. Hence we have proved that $T(N) \leq 2 \cdot T(N/2) + N^2/2$. It follows that $T(N) \leq N^2$, as desired.

Next we make several alterations to the previous algorithm and show how to improve the complexity bound to $O(N \cdot \log^4 N)$, for each boolean function $f \in \mathcal{B}_N$ which is computable in the hypercube. In all our subsequent discussions we use the notation and terminology established in the previous discussion. As before the new algorithm is also executed in $n = \log N$ steps, one step per dimension. However, now we take advantage of the information provided to p about the hypercube from its i-th view I_p^i. The main ingredients of the new algorithm are the following.

- We introduce a leader election mechanism which for each $i \leq \log N$ elects leaders among the processors with lexicographically maximal view at the i-th step of the algorithm.
- We use elementary results from the theory of finite permutation groups [Wie64] in order to introduce a coding mechanism of the views; leaders at the $(i-1)$st step exchange the encoded versions of their views I_p^{i-1}; upon receipt of the encoded view they recover the original view sent and elect new leaders for the i-th step.
- The leader election and coding mechanisms help keep low the number of bits transmitted during the i-th step of the algorithm to $O(N \cdot i^3)$ bits.

The technical details of the above description will appear in the sequel. We begin with some preliminary lemmas that will be essential in the proof of the main theorem.

Lemma 3.9.1. *If $I_p = I_q$ then the hypercube as viewed from p is identical to the hypercube as viewed from q. More formally, for each p let $I_p = \langle b_x : x \in N \rangle$. If $I_p = I_q$ and $\phi = \phi_S$, where $S = \{i \leq n : p_i \neq q_i\}$, then $\forall x \in Q_n (b_x = b_{\phi(x)})$.*

Proof. Indeed, notice that since $q = \phi(p)$

$$I_p = I_q = I_{\phi(p)} = \phi(I_p),$$

where the right-most equality follows from (3.5). This proves the lemma.

Lemma 3.9.2. *Let I be a fixed sequence of bits of length 2^n. Then the set of processors p such that $I_p = I$ can be identified in a natural way with a group of bit-complement automorphisms. Moreover, the number of processors p such that $I_p = I$ is either 0 or a power of 2.*

Proof. Let $\mathcal{G}$ be the following set of automorphisms

$$\mathcal{G} = \{\phi \in F_n : \forall p \in Q_n (I_p = I \Rightarrow I_{\phi(p)} = I)\}. \tag{3.6}$$

The identity element is in $\mathcal{G}$. In addition the identity

$$I_{\phi(\psi(p))} = \phi(\psi(I_p))$$

implies that $\mathcal{G}$ is also closed under multiplication. Since $\mathcal{G}$ is finite it is a group.

Next consider the set $\mathcal{J}$ of processors q satisfying $I_q = I$ and assume that $\mathcal{J} \neq \emptyset$. Let p_0 be an arbitrary but fixed element of $\mathcal{J}$. Without loss of generality we may assume that $p_0 = 0^n$. We claim that the sets $\mathcal{J}$ and $\mathcal{G}$ are equipotent. First we prove $|\mathcal{J}| \leq |\mathcal{G}|$. Indeed, for each $p \in \mathcal{J}$ there is a unique bit-complement automorphism, say ϕ_p, such that $\phi_p(0^n) = p$. We show that in fact $\phi_p \in \mathcal{G}$. To see this let q be an arbitrary element of $\mathcal{J}$. By assumption we have

$$I_{0^n} = I_p = I_q = I.$$

Thus using identity (3.5) we obtain

$$I = I_{\phi_p(0^n)} = \phi_p(I_{0^n}) = \phi_p(I_q) = I_{\phi_p(q)}.$$

In turn, this implies the desired inequality $|\mathcal{J}| \leq |\mathcal{G}|$. To complete the proof of the claim it remains to prove that $|\mathcal{G}| \leq |\mathcal{J}|$. But this is obvious since the mapping $\phi \rightarrow \phi(0^n)$ is 1-1. The above considerations complete the proof of the first part of the lemma.

To prove the second assertion we note that F_n can be identified with an n-dimensional vector space over the finite field $Z_2 = \{0, 1\}$ of two elements. The standard basis of this vector space consists of the bit-complement automorphisms

$$\phi_{\{1\}}, \phi_{\{2\}}, \ldots, \phi_{\{n\}}.$$

Any other bit-complement automorphism ϕ_S can be written as the sum (which in this case is the regular composition of functions) of the automorphisms $\phi_{\{i\}}$, where $i \in S$. As a vector subspace $\mathcal{G}$ has a basis consisting of a fixed number of bit-complement automorphisms. Moreover, $|\mathcal{G}|$ is a power of 2. It follows that if $|\mathcal{J}|$ is nonempty it must be a power of 2.

The group $\mathcal{G}$ defined in Lemma 3.9.2 is called the automorphism group of the string I. Clearly, it depends on the string I. However we do not mention it explicitly in $\mathcal{G}$ in order to avoid unnecessary notational complications.

Lemma 3.9.3. *Let $\mathcal{G}$ be the automorphism group of the string I. If $|\mathcal{G}| = 2^l$ then I can be coded with a string of length 2^{n-l} and l bit-complement automorphisms.*

Proof. We continue using the notation of Lemma 3.9.2. The group $\mathcal{G}$ defined above has a natural action on the hypercube Q_n. For each $x \in Q_n$ let $x^{\mathcal{G}}$ be the orbit of x under $\mathcal{G}$, i.e.,

$$x^{\mathcal{G}} = \{\phi(x) : \phi \in \mathcal{G}\}.$$

For each x the stabilizer $\mathcal{G}_x$ of $\mathcal{G}$ under x is the identity group, where the stabilizer group [Wie64] is defined by

$$\mathcal{G}_x = \{\phi \in \mathcal{G} : \phi(x) = x\}.$$

By the well-known stabilizer theorem [Wie64]

$$|\mathcal{G}_x| \cdot |x^{\mathcal{G}}| = |\mathcal{G}|.$$

Since $|\mathcal{G}_x| = 1$ we obtain that all the orbits of $\mathcal{G}$ have exactly the same size, namely $|\mathcal{G}| = 2^l$, and since $|Q_n| = 2^n$, there are exactly

$$\frac{2^n}{|\mathcal{G}|} = 2^{n-l}$$

pairwise disjoint orbits.

The above discussion gives rise to the following "coding" algorithm which can be applied by the processors concerned in order to code the given configuration I with a new (generally shorter) string. Each processor that knows I can execute the following "coding algorithm" (i.e., processor p applies this algorithm to the string $I = I_p^n$).

Coding Algorithm:
Input: $I = \langle b_x : x \in Q_n \rangle$ is the given configuration, where b_x is the bit corresponding to processor x.

1. Compute the group $\mathcal{G}$ of bit-complement automorphisms ϕ such that
$$\forall p \in Q_n (I_p = I \Rightarrow I_{\phi(p)} = I).$$
Assume that l is such that $|\mathcal{G}| = 2^l$.
2. Compute a set of l generators, i.e., a set $\phi_1, \ldots, \phi_l$ of bit-complement automorphisms which generate the group $\mathcal{G}$.
3. Compute the set of orbits of $\mathcal{G}$ in its natural action on Q_n. There are 2^{n-l} such orbits. For each orbit the processors choose a representative of the orbit in some canonical way, say lexicographically minimal; let $x(1), x(2), \ldots, x(2^{n-l})$ be the representatives chosen. Next the processor arranges them in increasing order according to the lexicographic order $\prec$, i.e., $x(1) \prec x(2) \prec \ldots \prec x(2^{n-l})$.
4. The code of I is defined to be the sequence $\langle I'; \phi_1, \phi_2, \ldots, \phi_l \rangle$, where I' is the sequence of bits of length 2^l given by
$$I' := b_{x(1)} b_{x(2)} \cdots b_{x(2^{n-l})}$$
and
$$\phi_1, \phi_2, \ldots, \phi_l$$
is a sequence of bit-complement automorphisms generating the group $\mathcal{G}$.

Output: $\langle I'; \phi_1, \phi_2, \ldots, \phi_l \rangle$.

It remains to prove that a processor can reconstruct I from its encoding. To do this it executes the following decoding algorithm.

Decoding Algorithm:
Input: $\langle I'; \phi_1, \phi_2, \ldots, \phi_l \rangle$, where I' is a string of length 2^{n-l} and $\phi_1, \phi_2, \ldots, \phi_l$ are bit-complement automorphisms.

1. Let $\mathcal{G}$ be the group generated by these automorphisms. Compute the set of orbits of $\mathcal{G}$ in its natural action on Q_n. There are 2^{n-l} such orbits. For each orbit choose as representative of the orbit the lexicographically minimal string in the orbit. Let $x(1), x(2), \ldots, x(2^{n-l})$ be the representatives chosen. Next the processor arranges them in increasing order according to the lexicographic order $\prec$, i.e., $x(1) \prec x(2) \prec \ldots \prec x(2^{n-l})$.
2. The coding algorithm guarantees that $I' = b_{x(1)} b_{x(2)} \cdots b_{x(2^{n-l})}$. Hence we can "fill-in" the remaining bits to form the string I since $b_x = b_y$ for x, y in the same orbit.

Output: I.

Indeed, by definition of the group $\mathcal{G}$ we have that for all $\phi \in \mathcal{G}$, $\phi(I) = I$. Hence by Lemma 3.9.1

$$\forall x \in Q_n \forall \phi \in \mathcal{G}(b_x = b_{\phi(x)}),$$

where $I = \langle b_x : x \in Q_n \rangle$. This explains why the decoding algorithm works.

Now we can prove the following theorem which significantly improves the upper bound of Theorem 3.9.3.

Theorem 3.9.4 ([KK97]). *There is an algorithm computing every boolean function $f \in \mathcal{B}_N$ (which is invariant under all bit-complement automorphisms) on the canonically labeled hypercube Q_n, $N = 2^n$, with bit complexity $O(N \cdot \log^4 N)$.*

Proof. For each fixed string $x = x_{i+1} \cdots x_n$ of bits of length $n - i$ let

$$Q_i(x) = \{u_1 \cdots u_i x : u_1, \ldots, u_i \in \{0, 1\}\}.$$

For each processor p represented by the sequence $p_1 \cdots p_n$ of bits the i-th hypercube of p is defined to be $Q_i(p_{i+1} \cdots p_n)$. Clearly we have that

$$Q_i(x) = Q_{i-1}(0x) \cup Q_{i-1}(1x).$$

Initially, $I_p^0 =$ "input bit to processor p" and each processor declares itself leader of its 0-dimension hypercube $Q_0(p) = \{p\}$. The leaders at the i-th step of the algorithm are among those processors whose "view" I_p^i of their i-th hypercube is lexicographically maximal among the set of strings I_q^i. Assume by induction that we have elected leaders for the $(i-1)$-th stage of the algorithm and that each processor has a path to such a leader along its

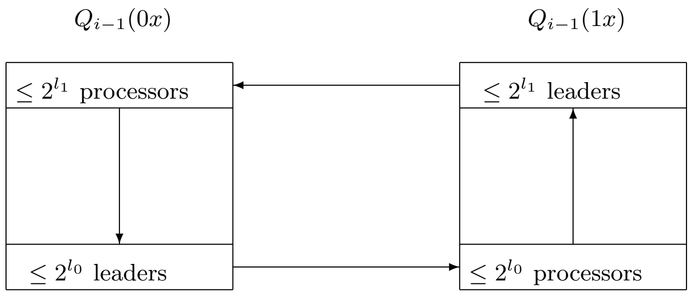

Fig. 3.1. Exchange of views among leaders in hypercube $Q_i(x)$

hypercube with edges $\leq i-1$. We show how to extend these assumptions to the i-th stage of the algorithm. The i-th stage of the new algorithm consists of the following steps.

1. The leader processors (elected at the $(i-1)$st stage) send their encoded views of their hypercube to their neighbors along the i-th dimension.
2. The processors of the opposite hypercube receiving the views route them to their leaders. (By induction hypothesis, all the processors know routes to their leaders along their hypercube; hence they can transmit the view received along such a route, for example the lexicographically minimal one.) Leaders that receive such encoded views decode the messages received as in Lemma 3.9.3, compute the corresponding views of their neighbors along their i-th edge and append it to their own view thus forming views at step i. To compute the view of their neighbors along their i-th edge each leader ℓ executes the following algorithm.
   a) Let ℓ's neighbor along the i-th edge be p and let $1 \leq k_1, \ldots, k_r \leq i-1$ be a path along p's subcube leading to a leader ℓ' in this subcube (by the induction hypothesis we can assume that such a path is known to p). By the previous argument the view $I_{\ell'}^{i-1}$ of ℓ' is known to ℓ (see Figure 3.2). Now ℓ requests this path from its neighbor p.
   b) Since $\phi_{\{k_1,\ldots,k_r\}}(\ell') = p$ it is clear that ℓ can compute p's view via the identity
$$\phi_{\{k_1,\ldots,k_r\}}(I_{\ell'}^{i-1}) = I_p^{i-1}.$$
3. If I_0 is the leader view in hypercube $Q_{i-1}(0x)$ and I_1 is the leader view in hypercube $Q_{i-1}(1x)$ then the leader view in hypercube $Q_i(x)$ will be
$$\begin{cases} I_0 I & \text{if } I_0 \succ I_1 \\ I_1 I & \text{if } I_1 \succ I_0 \\ I_0 I_1 & \text{if } I_0 = I_1 \end{cases}$$
for some string of length 2^{i-1} ($\succ$ denotes the lexicographic ordering). If L_0 is the set of leaders in hypercube $Q_{i-1}(0x)$ and L_1 is the set of

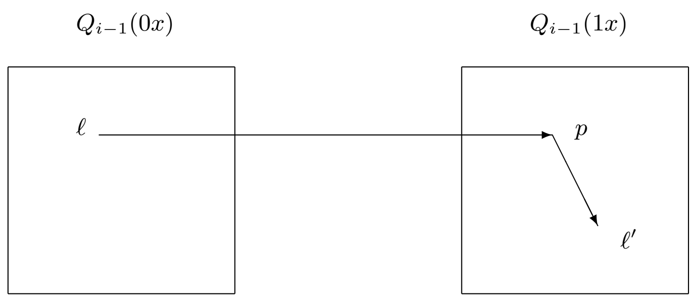

Fig. 3.2. In subcase 2(a), ℓ sends its view to p which routes it to ℓ'.

leaders in hypercube $Q_{i-1}(1x)$ then the set of leaders of the i-th stage will be among either L_0 or L_1 or $L_0 \cup L_1$, depending on the lexicographic comparison of I_0 and I_1. It follows that all the processors of $Q_i(x)$ will know paths to these new leaders. (Indeed, if $p \in Q_i(x)$ then either $p \in Q_{i-1}(0x)$ or $p \in Q_{i-1}(1x)$. Say, $p \in Q_{i-1}(0x)$. By induction p knows a path to a leader at the $i-1$st stage. But by the previous argument this leader knows a path to a leader at the i-th stage.)

4. Return to 1 and iterate, for $i = 1, 2, \dots, \log N$.

The mechanism for exchanging views at the i-th iteration of the above algorithm is depicted in Figure 3.1.

Now we estimate the bit complexity of the algorithm. The coding and decoding algorithms are "internal" and do not contribute anything to the total bit complexity. Suppose there are $\leq 2^l$ leaders elected at the i-th step of the algorithm. There exists a message w of length 2^{i-l} and a sequence of $l \leq i$ bit-complement automorphisms of the hypercube Q_i which "code" the view I_p^i. Since only the leaders transmit messages at the i-th step while the rest of the processors are "routing" messages to the leaders (processors are always at a distance $\leq i$ from a leader, since the diameter of the i-th hypercube is i), the total bit complexity at the i-th step of the algorithm is $O(2^i \cdot i^3)$ (since each encoded view consists of at most i bit-complement automorphisms and each bit-complement automorphism can be coded with i bits). Clearly, for each $i \leq \log N$ this algorithm is applied to 2^{n-i} subcubes simultaneously. Since the algorithm is iterated $\log N$ times it follows that the bit complexity of the new algorithm is

$$\sum_{i=1}^{\log N} 2^{n-i} \cdot O(2^i \cdot i^3) = O(N \cdot \log^4 N).$$

3.10 Historical and Bibliographical Remarks

The most important application of invariance groups is in providing a precise upper bound on the circuit complexity of boolean functions based on their degree of symmetry (see Theorem 3.8.3). There are several interesting open problems. Two such problems concern improving the $2^{O(\log n)}$ algorithm for testing the representability of an arbitrary group, as well as extending the logspace algorithm for testing the representability of cyclic groups to a larger class of permutation groups (Theorem 3.5.1). The work of Furst, Hopcroft and Luks [FHL80] should play a major role in such an endeavor. For additional information and results on the representation problem for boolean functions the reader should consult the papers [Kis99] and [Xia00].

The computability problems studied in Section 3.9 are a special case of the problem of collecting input data in a deterministic, distributed environment. There are several papers on trade-offs for input collection on anonymous networks as well as studies for randomized evaluation of boolean functions on anonymous rings [AAHK88]. For more details and references on anonymous networks the reader is referred to the survey article [Kra97b].

3.11 Exercises

Exercise 3.11.1 ([CK91]). ($\star$) Prove Theorem 3.2.1.
Hint: For the identity group take $L = 0^*1^*$, for the dihedral group $L = 0^*1^*0^* \cup 1^*0^*1^*$, for the reflection group $L = 0^*1^*0^*$. For the cyclic groups, if $n = 2$ take $L = (01 \cup 10)0^*1^*$, and if $n \geq 6$ then take $L = (L_0^1 \cup L_1^1) \cap L^2$, where

$$L_0^1 = 1^*0^*1^* \cup 0^*1^*0^* \cup 101000^*1 \cup 0^*1101000^*$$

$$L_1^1 = 0^*011010 \cup 0^*001101 \cup 10^*00110 \cup 010^*0011$$

and L^2 is the language $\overline{10^*00101}$. Notice that for $3 \leq n \leq 5$, if $C_n \subseteq \text{AUT}(f) \subseteq D_n$ then $\text{AUT}(f) = D_n$. For the hyperoctahedral group let L consist of the set of all finite strings $x = (x_1, \dots, x_n)$ such that for some $i \leq n/2$, $x_{2i-1} = x_{2i}$.

Exercise 3.11.2. Use the fact that for any permutation group G not containing A_n, $|S_n : G| \geq n$ [Wie64] to conclude that A_n is not isomorphic to the invariance group $\text{AUT}(f)$ of any $f \in \mathcal{B}_n$. However, A_n is isomorphic to the invariance group $\text{AUT}(f)$ for some boolean function $f \in \mathcal{B}_{n(\log n+1)}$ (see Theorem 3.2.4).

Exercise 3.11.3. One can generalize the notion of invariance group for any language $L \subseteq (k+1)^*$ by setting $L_n = L \cap \{0, \dots, k\}^n$ and $\text{AUT}(L_n)$ to be the set of permutations $\sigma \in S_n$ such that

$$\forall x_1, \dots, x_n \in \{0, 1, \dots, k\}(x_1, \dots, x_k \in L_n \iff x_{\sigma(1)}, \dots, x_{\sigma(n)} \in L_n).$$

Show that for all n, there exist groups $G_n \leq S_n$ which are strongly representable as $G_n = \text{AUT}(L_n)$ for some $L \subseteq \{0, 1, \ldots, n-1\}^n$ but which are not so representable for any language $L' \subseteq \{0, 1, \ldots, n-2\}^n$.
HINT: The alternating group $A_n = \text{AUT}(L_n)$, where $L_n = \{w \in \{0, \ldots, n-1\}^n : \sigma_w \in A_n\}$, where $\sigma_w : i \mapsto w(i-1)+1$. By a variant of the previous argument, A_n is not so representable by any language $L' \subseteq \{0, 1, \ldots, n-2\}^n$.

Exercise 3.11.4. Compared to the difficulties regarding the question of representing permutation groups $G \leq S_n$ in the form $G = \text{AUT}(f)$, for some $f \in \mathcal{B}_n$, it is interesting to note that a similar representation theorem for the groups $S(x) = \{\sigma \in S_n : x^\sigma = x\}$, where $x \in 2^n$, is relatively easy. It turns out that these last groups are exactly the permutation groups which are isomorphic to $S_k \times S_{n-k}$ for some k.
HINT: Given $x \in 2^n$ let

$$X = \{i : 1 \leq i \leq n \text{ and } x_i = 0\}, Y = \{i : 1 \leq i \leq n \text{ and } x_i = 1\}.$$

It is then easy to see that $S(x)$ is isomorphic to $S_X \times S_Y$. In fact, $\sigma \in S(x)$ if and only if $X^\sigma = X$ and $Y^\sigma = Y$.

Exercise 3.11.5. Notice the importance of assuming $m < n$ in the definition of weak representability. If $m = n$ were allowed then every permutation group would be weakly representable.
HINT: Given any permutation group $G \leq S_n$ define the function f as follows:

$$f(x_1, \ldots, x_n) = \begin{cases} 0 \text{ if } (x_1, \ldots, x_n) \in G \\ 1 \text{ otherwise} \end{cases}$$

(here, we think of $(x_1, \ldots, x_n)$ as the function $i \to x_i$ in n^n) and notice that for all $\sigma \in S_n$, $\sigma \in \text{AUT}(f)$ if and only if $\forall \tau \in S_n (\tau \in G \Leftrightarrow \tau\sigma \in G)$. Hence $G = \text{AUT}(f)$, as desired.

Exercise 3.11.6. Incidentally, it is not known if the $n(1+\log n)$ upper bound of Theorem 3.2.4 can be improved. However the idea of the proof of Theorem 3.2.4 can also be used to show that for any alphabet Σ, if $L \subseteq \Sigma^n$ then $\text{AUT}_n(L)$ (the set of permutations in S_n "respecting" the language L) is isomorphic to $\text{AUT}_{ns}(L')$, for some $L' \subseteq \{0,1\}^{ns}$, where $s = 1 + \log |\Sigma|$.

Exercise 3.11.7. The well-known graph non-isomorphism problem (NGIP) is related to the above group representation problem. Indeed, let

$$G = (\{v_1, \ldots, v_n\}, E_G), H = (\{u_1, \ldots, u_n\}, E_H)$$

be two graphs each on vertices and let $\text{ISO}(G, H) \leq S_{n+3}$ have generators σ satisfying:

$$\forall 1 \leq i, j \leq n (E_G(v_i, v_j) \Leftrightarrow E_H(u_{\sigma(i)}, u_{\sigma(j)})),$$

and in addition the permutation $n+i \to \sigma(n+i)$, $i = 1,2,3$, belongs to the group $C_3 = (n+1, n+2, n+3)$. It is easy to show that if G, H are isomorphic then there exists a group $K \leq S_n$ such that $\mathrm{ISO}(G,H) = K \times C_3$. On the other hand, if G, H are not isomorphic then $\mathrm{ISO}(G,\ H) = \langle id_{n+3} \rangle$. As a consequence of the non-representability of C_3, and the representability theorem of direct products, it follows that G, H are not isomorphic if and only if $\mathrm{ISO}(G,H) = \langle id_{n+3} \rangle$.

Exercise 3.11.8. An idea similar to that used in the proof of the representation theorem can also be used to show that for any representable permutation groups $G < H \leq S_n$,

$$2 \cdot |\{h \in \mathcal{B}_n : H = \textsc{aut}(h)\}| \leq |\{g \in \mathcal{B}_n : G = \textsc{aut}(g)\}|.$$

Hint: Assume that G, H are as above. Without loss of generality we may assume that there is no representable group K such that $G < K < H$. As in the proof of the representation theorem there exist $x, y \in 2^n$ such that $x = y \bmod H, x \neq y \bmod G$. Define two boolean functions $h_b \in \mathcal{B}_n$, $b = 0, 1$, as follows for $w \in 2^n$,

$$h_b(w) = \begin{cases} h(w) & \text{if } w \neq x \bmod G, w \neq y \bmod G \\ b & \text{if } w = x \bmod G \\ \bar{b} & \text{if } w = y \bmod G \end{cases}$$

Since $G \leq \textsc{aut}(h_b) < S(h)$, it follows from the above definition that each $h \in \mathcal{B}_n$ with $H = \textsc{aut}(h)$ gives rise to two distinct $h_b \in \mathcal{B}_n$, $b = 0, 1$, such that $G = \textsc{aut}(h_b)$. Moreover it is not difficult to check that the mapping $h \to \{h_0, h_1\}$, where $H = \textsc{aut}(h)$, is 1-1. It is now easy to complete the proof of the assertion.

Exercise 3.11.9. $(\star)$ Prove all the assertions made in Section 3.3.

Exercise 3.11.10. The automorphism group of a directed graph may not be 2-representable.
Hint: Look at the cyclic groups C_3, C_4, C_5 from Section 3.3.

Exercise 3.11.11. In this exercise we develop representability theorems for wreath products of permutation groups. For details on proofs the reader may consult [Kis99], [CK91]. Let $G \leq S_m, H \leq S_n$. Then

1. $(\star)$ G and H are k-representable $\Rightarrow G \wr H$ is k-representable.
2. $G \wr H$ is 2-representable $\Rightarrow H$ is representable.
3. $G \wr H$ is 2-representable and $2^n < m \Rightarrow G$ is weakly representable.
4. For p prime, a p-Sylow subgroup P of S_n is representable $\Leftrightarrow p \neq 3, 4, 5$.

Exercise 3.11.12. It is easy to see that in general $|S_n : G|$ and $\Theta_n(G)$ can diverge widely. For example, let $f(n) = n - \log n$ and let G be the group $\{\sigma \in S_n : \forall i > f(n)(\sigma(i) = i)\}$. It is then clear that $\Theta_n(G) = (f(n) + 1) \cdot 2^{\log n}$ is of order n^2, while $|S_n : G|$ is of order $n^{\log n}$. Another simpler example is obtained when G is the identity subgroup of S_n.

Exercise 3.11.13. The converse of part (1) of the Theorem 3.11.11 is not necessarily true. This is easy to see from the wreath product $A_3 \wr S_2$ which is representable, but that A_3 is not.
HINT: Consider the language

$$L = \{001101,\ 010011,\ 110100,\ 001110,\ 100011,\ 111000\} \subseteq 2^6.$$

We already proved that A_3 is not representable. We claim that $A_3 \wr S_2 = \text{AUT}_6(L)$. Consider the 3-cycle $\tau = (\{1,2\},\ \{3,4\},\ \{5,6\})$. It is easy to see $A_3 \wr S_2$ consists of the 24 permutations σ in S_6 which permute the two-element sets $\{1,2\}$, $\{3,4\}$, $\{5,6\}$ like in the three-cycles $\tau,\ \tau^2,\ \tau^3$. A straightforward (but tedious) computation shows that $\text{AUT}_6(L)$ also consists of exactly the above 24 permutations.

Exercise 3.11.14. Another class of examples of nonrepresentable groups is given by the direct products of the form $A_m \times G$, $G \times A_m$, where G is any permutation group acting on a set which is disjoint from $\{1, 2, \ldots, m\}$, $m \geq 3$.

Exercise 3.11.15 (Open Problem). At present, we do not know how to efficiently test the representability of arbitrary abelian groups (or other natural classes of groups such as solvable, nilpotent, etc.)

Exercise 3.11.16. If a given abelian group K can be decomposed into disjoint cyclic factors, then we have the following NC algorithm for testing representability: (1) use an NC algorithm [BLS87], [MC85], [Mul86] to "factor" K into its cyclic factors and then (2) apply the "cyclic-group" algorithm to each of the cyclic factors of K. In view of the result below the group K is representable exactly when each of its disjoint, cyclic factors is.

1. Let $G \leq S_m, H \leq S_n$ be permutation groups. Then $G \times H$ is representable $\Leftrightarrow$ both $G,\ H$ are representable.

Exercise 3.11.17. ($\star$) Show that

1. there is no regular language L such that for all but a finite number of n we have that $\text{AUT}_{2n}(L) = (S_{2n})_{\{1,2,\ldots,n\}}$.
2. there is a regular language L such that for all n we have that $\text{AUT}_{2n}(L) = (S_{2n})_{\{2i: i \leq n/2\}}$.

Exercise 3.11.18. The group S_n is generated by the cyclic permutation $c_n = (1, 2, \ldots, n)$ and the transposition $\tau = (1, 2)$ (in fact any transposition will do) [Wie64].

Exercise 3.11.19 ([CK91]). ($\star$) Consider a term $t(x, y)$ built up from the variables x, y by concatenation The number of occurrences of x and y in the term $t(x, y)$ is called the length of t and is denoted by $|t|$. For any permutations σ, τ let the permutation $t(\sigma, \tau)$ be obtained from the term $t(x, y)$ by substituting each occurrence of x, y by σ, τ, respectively, and interpreting concatenation as product of permutations. A sequence $\sigma = \langle \sigma_n : n \geq 1 \rangle$ of permutations is term-generated by the permutations c_n, τ if there is a term $t(x, y)$ such that for all $n \geq 2$, $\sigma_n = t(c_n, \tau)$. Show that

1. Let $\sigma = \langle \sigma_n : n \geq 1 \rangle$ be a sequence of permutations which is term-generated by the permutations $c_n = (1, 2, \ldots, n), \tau = (1, 2)$. Then for any regular language L, L^σ is also regular.
2. For any term t of length $|t|$ the problem of testing whether for a regular language L, $L = L^\sigma$, where $\sigma = \langle \sigma_n : n \geq 1 \rangle$ is a sequence of permutations generated by the term t via the permutations $c_n = (1, 2, \ldots, n), \tau = (1, 2)$, is decidable; in fact it has complexity $O(2^{|t|})$.

Exercise 3.11.20 ([Cla91]). Prove Theorem 3.6.2.
HINT: Show that if A is not the identity matrix then the number of orbits of A in $GL(n, 2)$ is at most $2^{n-1} + 2^{n-2} = 3 \cdot 2^{n-2}$.

Exercise 3.11.21 ([FKPS85]). ($\star$) Besides equality, the language of graph theory has a single binary relation I. The axioms of the theory of loopless, undirected graphs LUG are: $\forall x \neg I(x, x)$ and $\forall x, y (I(x, y) \leftrightarrow I(y, x))$. For arbitrary but fixed $0 < p < 1$, let $G_n = (V, E)$ run over random graphs of n nodes such that $\Pr[(i, j) \in E] = p$. Let $\phi_{r,s}$ denote the sentence: for any distinct $x_1, \ldots, x_r, y_1, \ldots, y_s$ there is an x adjacent to all the x_is and none of the y_is. Show that

1. any two models of LUG satisfying all sentences $\phi_{r,s}$ are isomorphic,
2. the set $\{\phi_{r,s}\}$ is complete,
3. $\lim_{n\to\infty} \Pr[G_n \models \phi_{r,s}] = 1$,
4. for any sentence ϕ of LUG, $\lim_{n\to\infty} \Pr[G_n \models \phi = 0$ or 1.

Exercise 3.11.22. For any language L and any sequence $\sigma = \langle \sigma_n : n \geq 1 \rangle$ of permutations such that each $\sigma_n \in S_n$ we define the language $L_n^\sigma = \{x \in 2^n : x^{\sigma_n} \in L_n\}$. For each n let $G_n \leq S_n$ and put $\mathbf{G} = \langle G_n : n \geq 1 \rangle$. Define $L^{\mathbf{G}} = \bigcup_{\sigma_n \in G_n} L_n^{\sigma_n}$. For each $1 \leq k \leq \infty$ let $\mathbf{F}_k$ be the class of functions $n^{c \log^{(k)} n}$, $c > 0$, where $\log^{(1)} n = \log n$, $\log^{(k+1)} n = \log \log^{(k)} n$, and $\log^{(\infty)} n = 1$. Clearly, $\mathbf{F}_\infty$ is the class $\mathbf{P}$ of polynomial functions. We also define $\mathbf{F}_0$ as the class of functions 2^{cn}, $c > 0$. Let $\mathbf{L}(\mathbf{F}_k)$ be the set languages $L \subseteq \{0, 1\}^*$ such that there exists a function $f \in \mathbf{F}_k$ satisfying $\forall n (|S_n : \mathrm{AUT}_n(L)| \leq f(n))$. We will also use the notation $L(\mathbf{EXP})$ and $L(\mathbf{P})$ for the classes $\mathbf{L}(\mathbf{F}_0)$ and $\mathbf{L}(\mathbf{F}_\infty)$, respectively. Show that

1. for any $0 \leq k \leq \infty$ and any language $L \in \mathbf{L}(\mathbf{F}_k)$,
2. $\mathbf{L}(\mathbf{F}_k)$ is closed under boolean operations and homomorphisms,

3. $(L \cdot \Sigma) \in \mathbf{L}(\mathbf{F}_k)$,
4. $L^\sigma \in \mathbf{L}(\mathbf{F}_k)$, where $\sigma = \langle \sigma_n : n \geq 1 \rangle$, with each $\sigma_n \in S_n$,
5. if $|S_n : N_{S_n}(G_n)| \leq f(n)$ and $f \in \mathbf{F}_k$ then $L^{\mathbf{G}} \in \mathbf{L}(\mathbf{F}_k)$, where $\mathbf{G} = \langle G_n : n \geq 1 \rangle$.
6. $L \in L(\mathbf{P})$ and $p \in \mathbf{P} \Rightarrow |S_{p(n)} : S_{p(n)}(L)| = n^{O(1)}$.
7. $L^1, L^2 \in L(\mathbf{EXP}) \Rightarrow L = \{xy : x \in L^1, y \in L^2, l(x) = l(y)\} \in L(\mathbf{EXP})$.
8. $\mathbf{L}(\mathbf{F}_\infty) = L(\mathbf{P}) \subset \ldots \subset \mathbf{L}(\mathbf{F}_{k+1}) \subset \mathbf{L}(\mathbf{F}_k) \subset \cdots \subset L(\mathbf{EXP}) = \mathbf{L}(\mathbf{F}_0)$,
9. $\mathbf{REG} \cap L(\mathbf{P}) \neq \emptyset$, $\mathbf{REG} - L(\mathbf{EXP}) \neq \emptyset$, $L(\mathbf{P}) - \mathbf{REG} \neq \emptyset$.

Exercise 3.11.23. A family $\langle p_n : n \geq 1 \rangle$ of multivariate polynomials in the ring $Z_2[x_1, \ldots, x_n]$ is of polynomial index if $|S_n : \text{AUT}(p_n)| = n^{O(1)}$. Show that for such a family of multivariate polynomials there is a family $\langle q_n : n \geq 1 \rangle$ of multivariate polynomials (in $Z_2[x_1, \ldots, x_n]$) of polynomial length such that $p_n = q_n$.

Exercise 3.11.24. Because of the limitations of families of groups of polynomial index proved in the claim above, we obtain a generalization of the principal results of [FKPS85]. Namely, for $L \subseteq \{0,1\}^*$ let $\mu_L(n)$ be the least number of input bits which must be set to a constant in order for the resulting language $L_n = L \cap \{0,1\}^n$ to be constant (see [FKPS85] for more details). Then we can prove the following result. If $L \in L(\mathbf{P})$ then $\mu_L(n) \leq (\log n)^{O(1)} \iff L \in \text{AC}^0$.

Exercise 3.11.25. Our characterization of permutation groups of polynomial index given during the proof of Theorem 3.7.3 can also be used to determine the parallel complexity of the following problem concerning "weight-swapping". Let $\mathbf{G} = \langle G_n : n \in \mathbf{N} \rangle$ denote a sequence of permutation groups such that $G_n \leq S_n$, for all n. By SWAP($\mathbf{G}$) we understand the following problem:

Input. $n \in \mathbf{N}$, $a_1, \ldots, a_n$ positive rationals, each of whose (binary) representations is of length at most n.
Output. A permutation $\sigma \in G_n$ such that for all $1 \leq i \leq n$, $a_{\sigma(i)} + a_{\sigma(i+1)} \leq 2$, if such a permutation exists, and the response "NO" otherwise.

Show that for any sequence $\mathbf{G}$ of permutation groups of polynomial index, the problem SWAP($\mathbf{G}$) is in non-uniform NC^1.

Exercise 3.11.26. Deduce from the proof of Theorem 2.2.1 that the number of boolean functions $f \in \mathcal{B}_n$ which can be computed by a circuit of size s with n input gates is $O(s^{2s})$.

Exercise 3.11.27. For any sequence $\mathbf{G} = \langle G_n : n \geq 1 \rangle$ of permutation groups $G_n \leq S_n$ it is possible to find a language L such that

$$L \notin SIZE(\sqrt{\Theta(G_n)}), \text{ and } \forall n(\text{AUT}(L_n) \supseteq G_n).$$

HINT: By Exercise 3.11.26 $|\{f \in \mathcal{B}_n : c(f) \leq q\}| = O(q^{2q}) = 2^{O(q \log q)}$, where $c(f)$ is the size of a circuit with minimal number of gates computing

f. Hence, if $q_n \to \infty$ then $|\{f \in \mathcal{B}_n : c(f) \leq q_n\}| < 2^{q_n^2}$. In particular, setting $q_n = \sqrt{\Theta(G_n)}$ we obtain

$$|\{f \in \mathcal{B}_n : c(f) \leq \sqrt{\Theta(G_n)}\}| < 2^{\Theta(G_n)} = |\{f \in \mathcal{B}_n : \text{AUT}(f) \supseteq G_n\}|.$$

It follows that for n big enough there exists an $f_n \in \mathcal{B}_n$ such that $\text{AUT}(f_n) \supseteq G_n$ and $c(f_n) > \sqrt{\Theta(G_n)}$.

Exercise 3.11.28. In this exercise we develop the notion of structure forest used extensively in Section 3.7

1. A structure tree for a transitive permutation group G acting on Ω can be constructed as follows. Take a strictly decreasing sequence

$$B_0 := \Omega \supset B_1 \supset \cdots \supset B_{r-1} \supset B_r = \{x\},$$

   of blocks of G with $B_0 := \Omega$ and B_r a singleton. Then the blocks $\{B_i^\sigma : i = 0, \ldots, r, \sigma \in G\}$ form a tree with respect to inclusion whose root is Ω and leaves are the singletons $\{x\}$, where $x \in \Omega$. Each element of the i-th level, denoted by $\mathcal{L}_i$, of this tree, can be written as the disjoint union of elements of the $i+1$st level. The number of elements of this union is a constant k_i which is independent of the level i. Moreover, $|\{\mathcal{L}_i\}| = k_0 k_1 \cdots k_{i-1}$. In particular, $n = |\{\mathcal{L}_{r+1}\}| = k_0 k_1 \cdots k_r$.
2. For each block B let $G_{\{B\}} = \{\sigma \in G : B^\sigma = B\}$ be the stabilizer of B. Let $L(B)$ denote the action of $G_{\{B\}}$ on B. Let $\mathcal{B}$ be the set of blocks which are sons of B in the above structure tree. Denote by $H(B)$ the action of $G_{\{B\}}$ on $\mathcal{B}$. Then we have that $L(B) \leq Sym(B), H(B) \leq Sym(\mathcal{B})$. Show that the groups $G_{\{B\}}$ of each level are conjugate of each other.
3. For each i let K_i stand for the pointwise stabilizer of level i, i.e.,

$$K_i = \{\sigma \in G : \forall B \in \mathcal{L}_i (B^\sigma = B)\}$$

   Show that K_i is a normal subgroup of G and $\frac{G}{K_i} \leq Sym(\mathcal{L}_i)$.
   Hint: For any groups $N \leq M$ consider the set $\mathcal{C}$ of left cosets of N with respect to M. Show that the kernel of the homomorphism $m \to mN$ is $\bigcap_{m \in M} m^{-1} N m$ which is also the largest normal subgroup of M contained in N.
4. Show that for $B \in \mathcal{L}_i$, $K_i \leq L(B)^{|\mathcal{L}_i|}, G \leq L(B) \wr (G/K_i)$.
5. If the group is not transitive break-up Ω into disjoint orbits. The action of the group on each of these orbits gives rise to a structure tree. The totality of these trees is called a structure forest.

Exercise 3.11.29 (Open Problem). For every permutation group $G \leq S_n$ let $k_n(G)$ denote the smallest integer k such that G is isomorphic to the invariance group of f. By Theorem 3.2.4, $k(G)n(1+\log n)$. Determine tighter bounds.

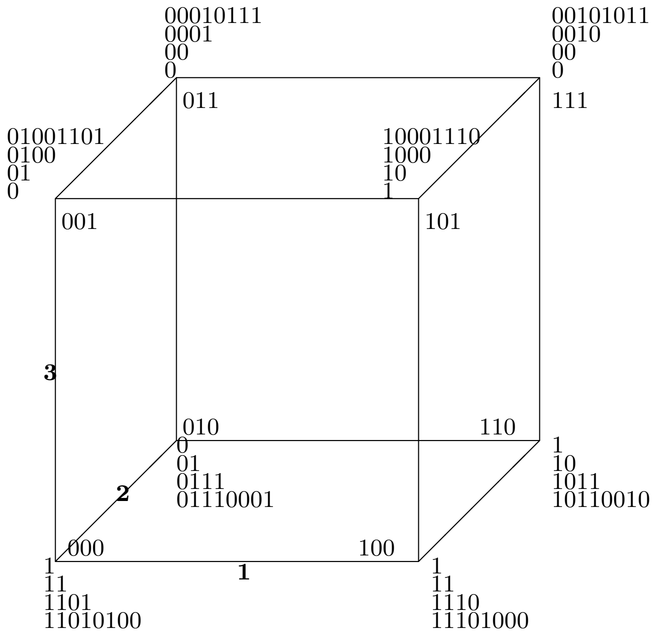

Fig. 3.3. Executing the $O(N^2)$ algorithm on a three-dimensional hypercube

Exercise 3.11.30 (Open Problem). $\mathcal{R}_k^n$ is the class of k-representable permutation groups on n letters. It is clear that $\mathcal{R}_k^n \subseteq \mathcal{R}_{k+1}^n$. Is $\mathcal{R}_k^n$ a proper hierarchy?

Exercise 3.11.31.
(1) Consider the three-dimensional hypercube depicted in Figure 3.3 with the input indicated. Let us trace the behavior of the algorithm on the given input for the bottom-left processor, say $p = 000$. Let $p_1 = 100, p_2 = 010, p_3 = 001$ be the neighbors of p along dimensions $1, 2, 3$, respectively. Following the algorithm the successive views of processor p are

$$\begin{aligned} I_p^0 &= 1 \\ I_p^1 &= I_p^0 I_{p_1}^0 = 11 \\ I_p^2 &= I_p^1 I_{p_2}^1 = 1101 \\ I_p^3 &= I_p^2 I_{p_3}^2 = 11010100. \end{aligned}$$

(2) Let b_p denote the input bit to processor p. A similar reasoning shows that $I_{111}^3 = 00101011$. We can show that I_{000}^3, I_{111}^3 are identical up to automorphism. Indeed, take the unique automorphism which maps 000 into 111,

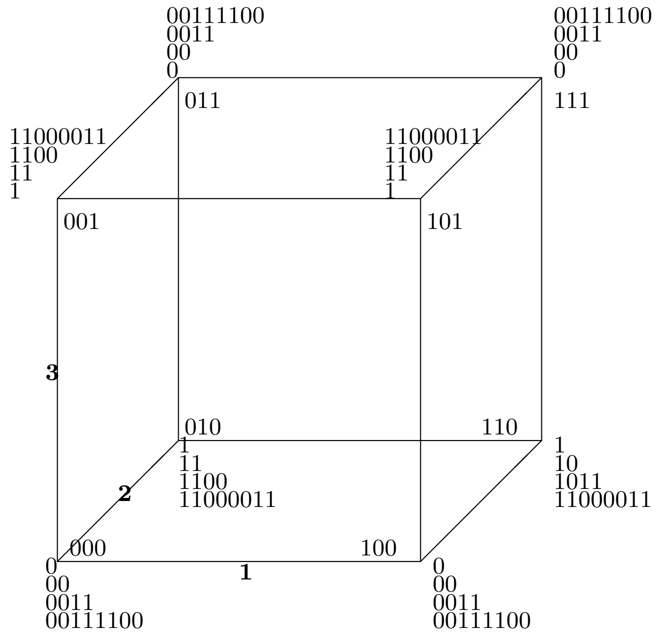

Fig. 3.4. Example illustrating the coding of views

namely $\phi = \phi_{\{1,2,3\}}$. Then we have

$$\begin{aligned} I^3_{000} &= b_{000}b_{100}b_{010}b_{110}b_{001}b_{101}b_{011}b_{111} \\ &= 11010100 \end{aligned}$$

and

$$\begin{aligned} I^3_{111} &= b_{\phi(000)}b_{\phi(100)}b_{\phi(010)}b_{\phi(110)}b_{\phi(001)}b_{\phi(101)}b_{\phi(011)}b_{\phi(111)} \\ &= b_{111}b_{011}b_{101}b_{001}b_{110}b_{010}b_{100}b_{000} \\ &= 00101011. \end{aligned}$$

Exercise 3.11.32. (1) Consider a three-dimensional hypercube with input as depicted in Figure 3.3. After the third iteration of the algorithm the view of processor 000 is $I^3_{000} = 11010100$. From its view I^3_{000} the processor 000 can reconstruct the views of all other processors. 000 is the only processor with this view. The group $\mathcal{G} = \{\phi : I^3_{000} = I^3_{\phi(000)}\}$ defined by Equation (3.6) is easily seen to be the trivial identity group generated by the identity automorphism e. The group has a natural action on the set of processors and gives rise to eight orbits:

$$\{000\}, \{100\}, \{011\}, \{111\}, \{010\}, \{110\}, \{101\}, \{100\}$$

Now the code for processor 000 is $\langle 11010100, e\rangle$.
(2) Consider a three-dimensional hypercube with input as depicted in Figure 3.4. After the third iteration of the algorithm the view of processor 000 is $I^3_{000} = 00111100$. From its view I^3_{000} the processor 000 can reconstruct the views of all other processors. There are four processors with this view, namely 000, 100, 011, 111. The group $\mathcal{G} = \{\phi : I^3_{000} = I^3_{\phi(000)}\}$ defined by Equation (3.6) is easily seen to be generated by the automorphisms $\phi_{\{1\}}, \phi_{\{2,3\}}$ and has size exactly $4 = 2^2$. The group has a natural action on the set of processors and gives rise to two orbits: $\{000, 100, 011, 111\}, \{001, 101, 010, 110\}$. Now the code for processor 000 is $\langle 01, \phi_{\{1\}}, \phi_{\{2,3\}}\rangle$, where 0 is the input bit of processor 000 and 1 is the input bit of processor 010.
(3) Here is how the decoding algorithm works. Suppose that a processor receives the code $\langle 01, \phi_{\{1\}}, \phi_{\{2,3\}}\rangle$. The processor constructs the orbit of the lexicographically minimal processor, namely 000. This is the orbit $\{000, 100, 011, 111\}$. Since $b_{000} = 0$ we know that

$$b_{000} = b_{100} = b_{011} = b_{111} = 0.$$

The remaining processors also form an orbit and the lexicographically minimal processor among them is 001. Since $b_{001} = 1$ we know that

$$b_{001} = b_{101} = b_{010} = b_{110} = 1.$$

Hence the decoded view of the processor is 00111100, as desired.

Exercise 3.11.33. The input configuration depicted in Figure 3.3 has a single leader, namely processor 100 with view 11101000. The input configuration depicted in Figure 3.4 has four leaders, namely 100, 110, 011, 101 with view 11000011. Notice that all processors can check from their view who and where the leaders are with respect to themselves. Now assume that the configuration depicted in Figure 3.3 is in the left-most hypercube in Figure 3.1 and the configuration depicted in Figure 3.4 is in the right-most hypercube in Figure 3.1. It is now clear that if the leaders of the corresponding three-dimensional hypercubes transmit their encoded views along dimension 4 all the processors of the four dimensional hypercube will be able to form views of the entire four dimensional hypercube.

Exercise 3.11.34 ([KK97]). On the canonically labeled hypercube Q_n, every symmetric function can be computed in $O(N \cdot \log^2 N)$ bits. Moreover the threshold function Th_k can be computed in $O(N \cdot \log N \cdot \log k)$ bits, where $k \leq N$.

Exercise 3.11.35 ([KK92]). ($\star$) Theorem 3.9.4 generalizes to arbitrary anonymous Cayley networks.

1. Show that if G is a set of generators for a group $\mathcal{G}$ then a boolean function f is computable on the naturally labeled Cayley network $\mathcal{N}_G[\mathcal{L}_G]$ if and only if f is invariant under all automorphisms of the network.
2. The bit complexity of computing all boolean functions which are computable on $\mathcal{N}_G[\mathcal{L}_G]$ is $O(|\mathcal{G}| \cdot \log^2 |\mathcal{G}| \cdot \delta^2 \cdot \sum_{g \in G} |g|^2)$, where δ is the diameter of the network, and $|g|$ the order of g in $\mathcal{G}$.
3. For any group $\mathcal{G}$ there is a set G of generators of $\mathcal{G}$ such that the above bit-complexity is $O(|\mathcal{G}|^3 \cdot \log^4 |\mathcal{G}|)$.
4. Contrast the classes of boolean functions computable on labeled and unlabeled Cayley networks.

Exercise 3.11.36. In this exercise we refer to the anonymous ring on N processors.

1. Show that OR_N requires $\Omega(N^2)$ bits on the anonymous ring.
2. ($\star$) **([MW86])** Non-constant boolean functions on N variables which are computable on an anonymous ring (oriented or not) of size N require $\Omega(N \log N)$ bits.
   HINT: First consider the case of oriented rings. Consider an arbitrary algorithm $\mathcal{A}$ computing a given non-constant boolean function f on N variables. Let S be the set of inputs w accepted by $\mathcal{A}$, i.e., such that $f(w) = 1$. We prove the lower bound for the synchronous ring (in which case it will be valid for the asynchronous case as well). Show that
   (a) if algorithm $\mathcal{A}$ rejects 0^N but accepts $0^n w$, for some word w, then $\mathcal{A}$ requires at least $N\lfloor n/2 \rfloor$ messages in order to reject 0^N, and
   (b) the average length of k pairwise distinct words $w_1, \ldots, w_k$ on an alphabet of size r is $> \frac{\log_r(k/2)}{2}$.
   Without loss of generality assume that 0^N is rejected. Assume that all processors terminate before time t when the input to $\mathcal{A}$ is ω, where ω is a word in S. Let $h_i(s)$ denote the history of processor p_i, i.e., $h_i(s) = m_i(1)\$ \cdots \$m_i(k)$, where \$ is a special symbol and $m_i(1), \ldots, m_i(k)$ are the messages received by p_i before time s in this order. Then $H_i = h_i(t)$ is the total history of p_i (on this computation). Since the length of H_i is less than twice the number of bits received by p_i a lower bound on the sum of the lengths of the histories of the processors implies a lower bound on the bit complexity of algorithm $\mathcal{A}$. Now to obtain the desired lower bound $\Omega(N \log N)$ we construct either an input with $\log N$ consecutive 0s (in which case part (a) applies) or else an input under whose execution the algorithm gives rise to at least $N \log N$ processors with distinct histories (in which case part (b) applies).
3. ($\star$) [**[ASW88]**] Assume N is odd and $N = 2n + 1$. Define $f_N(x) = 1$ if x is either $0(01)^n$ or a cyclical shift of it, and 0 otherwise. Show that f_N can be computed in $O(N)$ messages.
4. ($\star$) [**[MW86]**] There is a family $\{f_N\}$ of boolean functions computable with message complexity $O(N \log^* N)$. Use this to construct a family

of boolean functions computable with bit complexity $O(N \log N)$. This shows that the lower bound of part (2) is optimal.